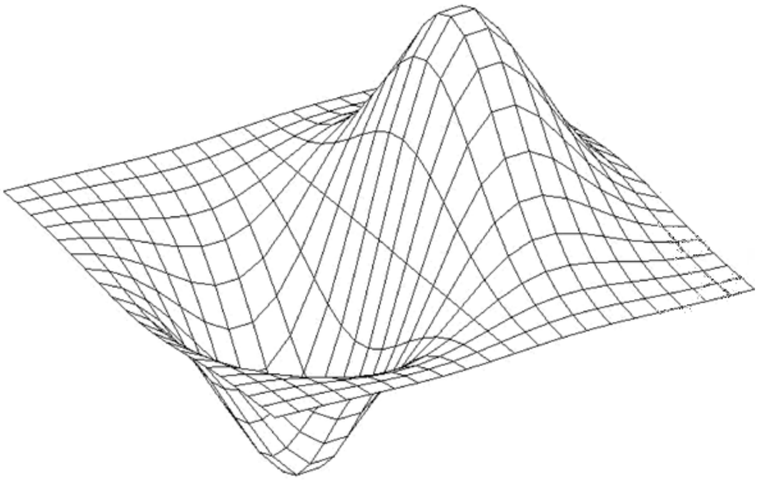
Institute of Geonics of the CAS, Ostrava

SNA'17

Seminar on Numerical Analysis

Modelling and Simulation
of Challenging Engineering Problems



Winter School

Methods of Numerical Mathematics and Modelling,
High-Performance Computing, Numerical Linear Algebra

Ostrava, January 30 – February 3, 2017

Programme committee:

Radim Blaheta	Institute of Geonics of the CAS, Ostrava
Zdeněk Dostál	VŠB - Technical University of Ostrava
Ivo Marek	Czech Technical University, Prague
Miroslav Rozložník	Institute of Computer Science of the CAS, Prague
Zdeněk Strakoš	Charles University, Prague

Organizing committee:

Radim Blaheta	Institute of Geonics of the CAS, Ostrava
Jiří Starý	Institute of Geonics of the CAS, Ostrava
Stanislav Sysala	Institute of Geonics of the CAS, Ostrava
Dagmar Sysalová	Institute of Geonics of the CAS, Ostrava
Hana Bílková	Institute of Computer Science of the CAS, Prague

Conference secretary:

Dagmar Sysalová	Institute of Geonics of the CAS, Ostrava

Institute of Geonics of the CAS
ISBN 978-80-86407-64-7

Preface

Seminar on Numerical Analysis 2017 (SNA'17) is a continuation in a series of SNA events held in different places in the Czech Republic and organized alternatively by Ostrava and Prague institutions. The SNA'17 is organized by the Institute of Geonics of the CAS in collaboration with VŠB - Technical University of Ostrava and IT4Innovations National Supercomputing Centre. Conference location is one of the lecture rooms in New Aula of the VŠB-TU Ostrava.

Let us note that SNA 2016 was reshaped to EMS School in Applied Mathematics (ESSAM) devoted to mathematical modelling, numerical analysis and scientific computing. The SNA'17 is turning back to more traditional winter event.

It provides opportunity for meeting and mutual information of the community working in computational mathematics and computer science, but an important part of SNA is devoted to the Winter School with tutorial lectures focused on selected important topics within the scope of numerical methods and modelling.

This year, a part of the Winter School will be the course Parallel Linear Algebra (PLA) organized by the European research infrastructure PRACE, particularly by the French PRACE Advanced Training Centre – Maison de la Simulation. Winter school lectures will cover the ongoing topics related to domain decomposition methods, interval computations and numerical verification. The PLA course will provide lectures from the area of direct and iterative parallel solvers, as well as practical training with selected programs.

We believe that the participants will enjoy the Winter School including the PLA course, as well as programme of contributed presentations, posters and complementary social events.

On behalf of the Programme and Organizing Committee of SNA'17,

Radim Blaheta and Jiří Starý

Contents

Winter school lectures

R. Blaheta:
Overlapping domain decomposition preconditioners for elliptic and parabolic problems in primal and mixed form

T. Brzobohatý, Z. Dostál, D. Horák et al.:
FETI methods for large problems – algorithms, scalability, and software implementation

M. Hladík:
Introduction to interval computation and numerical verification

Parallel linear algebra

M. Faverge:
Dense linear algebra - From BLAS to Chameleon
Sparse linear algebra
PASTIX: A sparse direct solver based on super-nodal method

G. Marait:
MaPHyS: a Massively Parallel Hybrid Solver

Z. Strakoš:
Krylov subspace methods from the historical, analytic, application, and high performance computing perspective

IgA based modelling of runner wheel flow

B. Bastl, M. Brandner, J. Egermaier, H. Horníková, K. Michálková, E. Turnerová

New Technologies for the Information Society, University of West Bohemia in Pilsen
Department of Mathematics, University of West Bohemia in Pilsen

1 Introduction

We focus on numerical solving of Navier-Stokes equations in 3D. We present a solver which is based on a recently proposed approach called isogeometric analysis, which uses isoparametric approach, i.e., the same basis functions are used for description of a geometry of a computational domain and also for representation of a solution. As isogeometric analysis is based on NURBS objects, any real application requires to handle the so-called multipatch domains, where a computational domain is composed of more parts and each part is represented by one NURBS object. The solver is used for the simulation of the flow through the runner wheel of the water turbine.

2 NURBS objects

NURBS surface of degree p, q is determined by a control net $\mathbf{P}$ (of control points $P_{i,j}$, $i = 0, \ldots, n$, $j = 0, \ldots, m$), weights $w_{i,j}$ of these control points and two knot vectors $U = (u_0, \ldots, u_{n+p+1})$, $V = (v_0, \ldots, v_{m+q+1})$ and is given by a parametrization

$$S(u,v) = \frac{\sum\limits_{i=0}^{n}\sum\limits_{j=0}^{m} w_{i,j} P_{i,j} N_{i,p}(u) M_{j,q}(v)}{\sum\limits_{i=0}^{n}\sum\limits_{j=0}^{m} w_{i,j} N_{i,p}(u) M_{j,q}(v)} = \sum_{i=0}^{n}\sum_{j=0}^{m} P_{i,j} R_{i,j}(u,v). \tag{1}$$

B-spline basis functions $N_{i,p}(u)$ and $M_{j,q}(v)$ of degree p are C^{p-1}-continuous in general. See e.g. [2] for details.

3 Navier-Stokes equations

The model of viscous flow of an incompressible Newtonian fluid in rotating domain can be described by the Navier-Stokes equations in the rotating frame of reference (we mention the stationary form here)

$$\begin{aligned} \nabla p + \boldsymbol{u}_R \cdot \nabla \boldsymbol{u}_A + \boldsymbol{\omega} \times \boldsymbol{u}_A - \nu \Delta \boldsymbol{u}_A &= \boldsymbol{f}, \qquad \text{in } \Omega, \\ \nabla \cdot \boldsymbol{u}_A &= 0, \qquad \text{in } \Omega, \end{aligned} \tag{2}$$

where $\Omega \subset \mathbb{R}^3$ is the computational domain, $\boldsymbol{u}_A = \boldsymbol{u}_A(\boldsymbol{x})$ is the vector function describing absolute flow velocity and $\boldsymbol{u}_R = \boldsymbol{u}_R(\boldsymbol{x})$ is the vector function describing relative flow velocity such that

$$\boldsymbol{u}_A = \boldsymbol{u}_R + \boldsymbol{\omega} \times \boldsymbol{r}, \tag{3}$$

where $\boldsymbol{r}$ is the position vector and $\boldsymbol{\omega} = (\omega, 0, 0)$ angular velocity vector (x-axis is assumed as the axis of rotation). $p = p(\boldsymbol{x})$ is the kinematic pressure function, ν describes kinematic viscosity and $\boldsymbol{f}$ additional body forces acting on the fluid.

The boundary value problem is considered as the system (2) together with the following boundary conditions

$$\begin{array}{rcll} \boldsymbol{u} & = & \boldsymbol{w} & \text{on } \partial\Omega_D \text{ (Dirichlet b. c.)}, \\ \nu\dfrac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} - \boldsymbol{n}p & = & \boldsymbol{0} & \text{on } \partial\Omega_N \text{ (Neumann b. c.)}, \\ \boldsymbol{u}_{c_1} & = & \boldsymbol{R}_o \boldsymbol{u}_{c_2}, & \\ p_{c_1} & = & p_{c_2} & \text{on } \partial\Omega_C \text{ (cyclic b. c.)}, \end{array} \tag{4}$$

where $\boldsymbol{u}$ stands for both $\boldsymbol{u}_A$ and $\boldsymbol{u}_R$, $\boldsymbol{R}_o$ is rotation matrix and c_1 and c_2 are corresponding boundaries. If the velocity is specified everywhere on the boundary, then the pressure solution is only unique up to a hydrostatic constant.

3.1 Galerkin approach and nonlinear iteration

Because of non-linearity of Navier-Stokes equations, it is necessary to solve the problem iteratively with linear problem in every step. One of the possibilities is to use the so-called Picard's method [1].

Let V be a velocity solution space and V_0 be the corresponding space of test functions, i.e.,

$$\begin{array}{rcl} V & = & \{\boldsymbol{u} \in H^1(\Omega)^d | \boldsymbol{u} = \boldsymbol{w} \text{ on } \partial\Omega_D\}, \\ V_0 & = & \{\boldsymbol{v} \in H^1(\Omega)^d | \boldsymbol{v} = \boldsymbol{0} \text{ on } \partial\Omega_D\}. \end{array} \tag{5}$$

We use Galerkin method and define finite dimensional spaces $V^h \subset V, V_0^h \subset V_0$, $W^h \subset L_2(\Omega)$ and their basis functions. Find $\boldsymbol{u}_h \in V^h$ and $p_h \in W^h$ so that all functions $\boldsymbol{v}_h \in V_0^h$ and $q_h \in W^h$ satisfy

$$\begin{array}{rcl} \nu \int\limits_\Omega \nabla \boldsymbol{u}_{Ah}^{k+1} : \nabla \boldsymbol{v}_h + \int\limits_\Omega (\boldsymbol{u}_{Rh}^k \cdot \nabla \boldsymbol{u}_{Ah}^{k+1})\boldsymbol{v}_h + \int\limits_\Omega (\boldsymbol{\omega} \times \boldsymbol{u}_{Ah}^{k+1})\boldsymbol{v}_h - \int\limits_\Omega p_h^{k+1} \nabla \cdot \boldsymbol{v}_h & = & \int\limits_\Omega \boldsymbol{f} \cdot \boldsymbol{v}_h, \\ \int\limits_\Omega q_h \nabla \cdot \boldsymbol{u}_{Ah}^{k+1} & = & 0. \end{array} \tag{6}$$

Isogeometric approach consists in taking the solution $\boldsymbol{u}_h$ as a linear combination of basis functions $R_i^u \in V^h$ and the solution p_h as a linear combination of basis functions $R_i^p \in W^h$, where R_i^u and R_i^p are NURBS basis function obtained from a NURBS description of a computational domain. In 3D, the solution has the form

$$\boldsymbol{u}_h = \sum_{i=1}^{n_d^u} (u_{1i}, u_{2i}, u_{3i})^T R_i^u + \sum_{i=n_d^u+1}^{n_v^u} (u_{1i}^*, u_{2i}^*, u_{3i}^*)^T R_i^u, \qquad p_h = \sum_{i=1}^{n^p} p_i R_i^p, \tag{7}$$

where n_d^u is the number of points where the Dirichlet boundary condition is not defined. Further, we assume that $\boldsymbol{f}$ is written as a linear combination of velocity basis function, i.e.,

$$\boldsymbol{f}_h = \sum_{i=1}^{n_v^u} (f_{1i}, f_{2i}, f_{3i})^T R_i^u.$$

For general $\boldsymbol{f}$, $\boldsymbol{f}_h$ can be obtained with the help of L_2 projection to a linear space spanned by basis functions $\{R_i^u\}_{1 \le i \le n_v^u}$.

4 Navier-Stokes equations on multipatch domains

Theory of NURBS objects directly implies that it is not possible to describe an object of arbitrary topology by one NURBS object. Thus, when isogeometric analysis is used for numerical solving of partial differential equations it is usually necessary to decompose a computational domain into subdomains, which are suitable for description by one NURBS object.

4.1 Conforming meshes

The simplest case of a multipatch domain is represented by the case where joining control nets of different patches coincide and the corresponding knot vectors are the same. Then, exactly the same meshes are obtained and we talk about a *conforming mesh* of the computational domain. The easiest approach for joining such NURBS patches in the followup computation with the help of isogeometric analysis is to identify the corresponding control points in the common control nets of joining NURBS patches and reduce the number of degrees of freedom in the computation.

4.2 Non-conforming meshes

More complicated multipatch domains composed of patches with nested meshes or even more general non-conforming meshes cannot be handled by identifying the corresponding control points. In these cases, one can use discontinuous Galerkin method to join such a configuration of patches into one computational domain. The main approach is to add several new terms into the weak formulation of the problem which are considered on the common interfaces of the patches. More details can be found in [3].

5 Test example

The simulation of the flow through the runner wheel region of the water turbine is presented. It is the solution of the Navier-Stokes equations (2) in the cyclic periodic domain (part of multipatch conforming mesh of this domain is depicted at the Fig. 1) with the following boundary conditions.

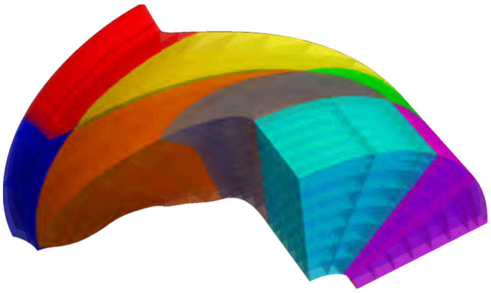


Figure 1: Part of multipatch conforming mesh of flow region in the runner wheel domain.

Inflow of the domain is set on left surface of blue and red patches, where velocity field is prescribed by the flow through the guide vanes region with the flow rate $Q = 5.54\text{m}^3/\text{s}$. Outflow of the domain is set on the opposite side i.e., homogeneous Neumann boundary condition is prescribed on this surface. The solid boundary surfaces of the domain are considered to be solid walls, i.e., Dirichlet boundary condition with $\boldsymbol{u} = \mathbf{0}\text{m/s}$ is prescribed. Cyclic boundary conditions are prescribed on the front and back surfaces of the patches. For pressure, homogeneous Neumann boundary condition is prescribed on the whole boundary.

The kinematic viscosity $\nu = 0.015\mathrm{m}^2/\mathrm{s}$ and angular speed $\omega = 56.3\mathrm{rad/s}$. Fig. 2 shows streamlines of velocity.

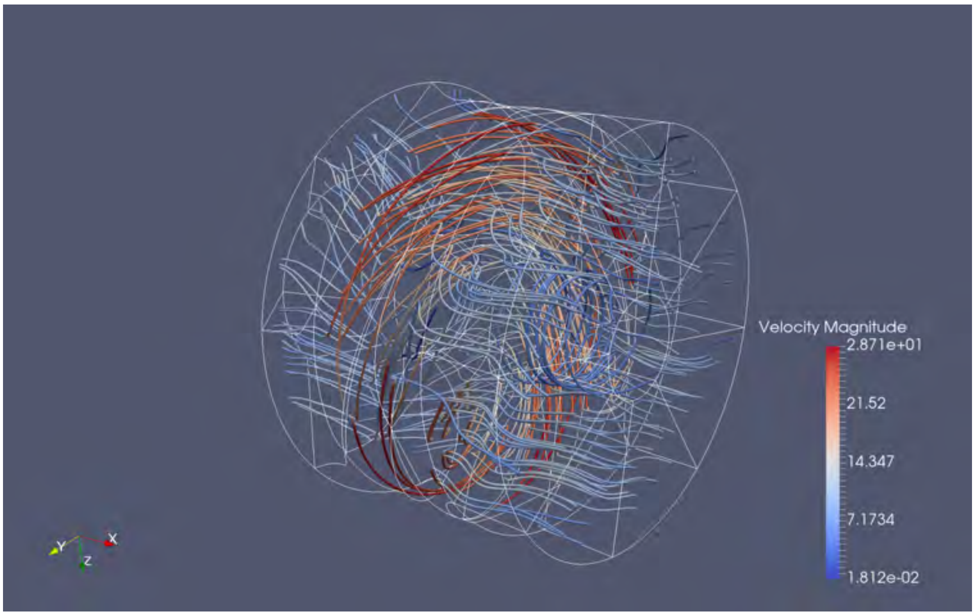


Figure 2: Streamlines of velocity.

6 Conclusion

This paper was devoted to the numerical simulation of the runner wheel flow. The solver based on isogeometric analysis was successfully used to the solution of Navier-Stokes equations with the rotation term. Complex region of the water turbine had to be decribed by the multipatch domain.

Acknowledgement: This work was supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports.

References

[1] M. Augustin, A. Caiazzo, A. Fiebach, J. Fuhrmann, V. John, A. Linke, R. Umla: *An assessment of discretizations for convection–dominated convection–diffusion equations.* Computer Methods in Applied Mechanics and Engineering, 2011, pp. 3395–3409.

[2] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs: *Isogeometric Analysis.* John Wiley & Sons, Ltd, 2009.

[3] A. Montlaur: *High-order discontinuous Galerkin methods for incompressible flows.* Escola Politécnica Superior de Castelldefels, Barcelona, Spain, 2009.

Gaussian random fields decomposition and sampling

M. Béreš, *S. Domesová*, *R. Blaheta*

Institute of Geonics of the CAS, Ostrava

1 Introduction

In mathematical modelling, we can encounter the need to analyze processes in some physical domain $\mathcal{D} \subset \mathbb{R}^n$ with only stochastic knowledge about the material. We shall say that the material properties are described as a random field. By the random field on $\mathcal{D} \subset \mathbb{R}^d$ we understand a real valued function $X\left(x,\omega\right)$, which for every fixed $x \in \mathcal{D}$ results in a random variable and for every fixed ω from the sample space Ω results in a function defined on $\mathcal{D}$, e.g. a function from $L^2(\mathcal{D})$. A common and natural type of random field is a Gaussian random field (GRF). For GRF $\forall x \in \mathcal{D}:\ X\left(x,\omega\right) \sim N\left(\mu\left(x\right);\sigma\left(x\right)\right)$.

GRF can be fully described by its mean value $\mu\left(x\right)$ and auto-covariance function $c\left(x,y\right) = \mathbb{E}\left(\left(X\left(x,\omega\right)-\mu\left(x\right)\right)\cdot\left(X\left(y,\omega\right)-\mu\left(y\right)\right)\right)$. Here we focus on isotropic GRF, which are specified by an auto-covariance function that takes only physical distance of x and y as a parameter. For numerical examples we use $\mathcal{D} = \langle 0,1\rangle^2$, $\mu\left(x\right) = 0$ and $c\left(x,y\right) = \sigma^2 \cdot \exp\left(-\frac{\|x-y\|}{\lambda}\right)$ with parameters $\lambda = 0.3$, $\sigma = 2$.

In typical applications, it is usually required

- to generate samples of the random field $X\left(x,\omega\right)$ (a sample is understood as a realization of $X\left(x,\omega\right)$ for some $\omega \in \Omega$) or
- to calculate the decomposition of the random field in the form of

$$X\left(x,\omega\right) \backsimeq \mu\left(x\right) + \sum_{i=1}^{N} \psi_i\left(x\right)\cdot\xi_i\left(\omega\right), \tag{1}$$

  where $\|\psi_i\left(x\right)\|$ should be rapidly decreasing with increasing value of i.

A random field is an infinite-dimensional object, therefore we first need to perform some discretization. Basically there are two ways of random field discretization:

- Point discretization, which is used, if we are interested only in some finite set of domain points $\{x_1,\ldots,x_N\} \subset \mathcal{D}$. It leads to a random vector representation of the studied random field.
- Discretization by the truncated Karhunen-Loève decomposition (KLD), which leads straightforwardly to the aforementioned decomposition form (1) of the random filed.

In this article we mainly focus on KLD and techniques for its efficient calculation and broad field of application, for other approaches we only present a review of suitable methods with references.

2 Point discretization

The point discretization of a random field is the most straightforward way to obtain some information about it. In the case of GRF it leads to the multi-variate Gaussian distribution with mean vector $\overline{\mu}: \left(\overline{\mu}\right)_i = \mu\left(x_i\right)$ and covariance matrix $\mathbb{C}: C_{i,j} = c\left(x_i, x_j\right)$. The covariance matrix is generally symmetric and positively semidefinite, but in the case of the discretized random field it can be assumed as symmetric positively definite (SPD).

First we examine sampling from the resulting multi-variate Gaussian distribution. Assume, that we can simply generate samples of the random vector $\overline{Y}$ of independent and identically distributed (i.i.d.) standard normal variables (zero mean, unit variance). Let $\overline{X}$ denote the desired multi-variate Gaussian distribution given by the mean vector $\overline{\mu}$ and the covariance matrix $\mathbb{C}$, than $\overline{X}$ can be expressed as

$$\overline{X} = \overline{\mu} + \mathbb{A} \cdot \overline{Y},$$

where $\mathbb{A}$ satisfies $\mathbb{C} = \mathbb{A} \cdot \mathbb{A}^T$. This formula specifies a direct approach to generate a sample x of the random vector $\overline{X}$ as $x = \overline{\mu} + \mathbb{A} \cdot y$, where y is sample of $\overline{Y}$. There are several ways to obtain $\mathbb{A} \cdot y$, that lead to different sampling methods.

In this section we mention four methods of sampling $\overline{Y}$, which can be divided in the following way:

- methods, that construct the matrix $\mathbb{A}$
  - construction using eigenvalue decomposition of $\mathbb{C}$
  - construction using Cholesky decomposition of $\mathbb{C}$
- methods, that don't require the matrix $\mathbb{A}$ to be explicitly constructed
  - Krylov subspace sampling method
  - Circular embedding method

Eigenvalue decomposition of $\mathbb{C}$. The covariance matrix is decomposed as $\mathbb{C} = \mathbb{Q} \cdot \mathbf{\Lambda} \cdot \mathbb{Q}^T$ (note $\mathbb{C}$ is SPD). Than $\mathbb{A}$ is constructed as $\mathbb{A} = \mathbb{Q} \cdot \sqrt{\mathbf{\Lambda}}$, where $\mathbf{\Lambda}$ is diagonal and $\sqrt{\mathbf{\Lambda}}$ is element-wise. This approach also offers the aforementioned decomposition form (1) of $\overline{X}$ (or $X\left(x, \omega\right)$) as

$$\overline{X} = \overline{\mu} + \sum_{i=1}^{N} \boldsymbol{q}_i \cdot \sqrt{\lambda_i} \cdot y_i,$$

where $\boldsymbol{q}_i$ are eigenvectors of $\mathbb{C}$ (or columns of $\mathbb{Q}$), λ_i are eigenvalues and y_i are i.i.d. standard random variables.

Cholesky decomposition of $\mathbb{C}$. The Cholesky decomposition factor can be used as $\mathbb{A}$, because $\mathbb{C}$ is SPD. In comparison to the previous approach, the decomposition form (1) has bad properties (low decrease in norm of summands in the decomposition form).

Krylov subspace sampling method. This iterative method is based on finding an approximation of $\sqrt{\mathbb{C}} \cdot y$ by projecting to Krylov space $\mathcal{K}(y, \mathbb{C})$ and using the Lanczos basis for approximation of the matrix square root. For the basic method see [2], for the preconditioned approach see [3].

Circular embedding method. This method is based on fast Fourier transform and requires a point discretization on a regular grid. It generates samples of random vector $\mathbb{A} \cdot \overline{Y}$, but cannot be interpreted as an operator on y in the aforementioned way, see [1].

3 Karhunen-Loève decomposition

KLD is an alternative to the point discretization, but now we don't constrain on a finite set of domain points. KLD is based on a fact, that the $L^2\left(\Omega, L^2\left(\mathcal{D}\right)\right)$ space (where the studied GRF belong) is identically isomorphic with the tensor product of spaces $L^2\left(\Omega\right)\otimes L^2\left(\mathcal{D}\right)$. The existence and construction of KLD is given by the Karhunen-Loève theorem, see [1, thm. 7.52]. It states, that KLD takes form

$$X\left(x,\omega\right)=\mu\left(x\right)+\sum_{j=1}^{\infty}\sqrt{\lambda_j}\cdot\psi_j\left(x\right)\cdot\xi_j\left(\omega\right), \tag{2}$$

where, in case of GRM, $\xi_j\left(\omega\right)$ are i.i.d. standard normal variables and $\left\{\lambda_j,\psi_j\right\}$ denote the eigenvalues and eigenfunctions of the covariance operator $\left(\mathcal{C}f\right)\left(x\right):=\int_{\mathcal{D}}c\left(x,y\right)\cdot f\left(y\right)\mathrm{d}y$.

For the construction of KLD we only need to calculate the solution of the following eigenvalue problem

$$\int\limits_{D}c\left(x,y\right)\cdot\psi_i\left(y\right)\mathrm{d}y=\lambda_i\cdot\psi_i\left(x\right),\forall i\in\mathbb{N},$$

which can be done using the Galerkin method.

For the Galerkin method consider a basis $\left\langle\phi_1\left(x\right),\ldots,\phi_n\left(x\right)\right\rangle=V_n\subset L^2\left(\mathcal{D}\right)$, than the approximation of eigenvectors takes form $\psi_i\left(x\right)\simeq\sum_{j=1}^{n}\overline{\psi_{ij}}\cdot\phi_j\left(x\right)$. We solve the following problem

$$\begin{cases}\text{Find } \overline{\psi_i}\in\mathbb{R}^n,\widetilde{\lambda_i}\in\mathbb{R}^+:\forall\phi_j\left(x\right): \\ \int\limits_{\mathcal{D}}\phi_j\left(x\right)\cdot\int\limits_{\mathcal{D}}c\left(x,y\right)\cdot\left(\sum_{j=1}^{n}\overline{\psi_{ij}}\cdot\phi_j\left(x\right)\right)\mathrm{d}y\mathrm{d}x=\widetilde{\lambda_i}\cdot\int\limits_{\mathcal{D}}\phi_j\left(x\right)\cdot\psi_i\left(x\right)\mathrm{d}x\end{cases}, \tag{3}$$

which can by formulated as a generalized eigenvalue problem

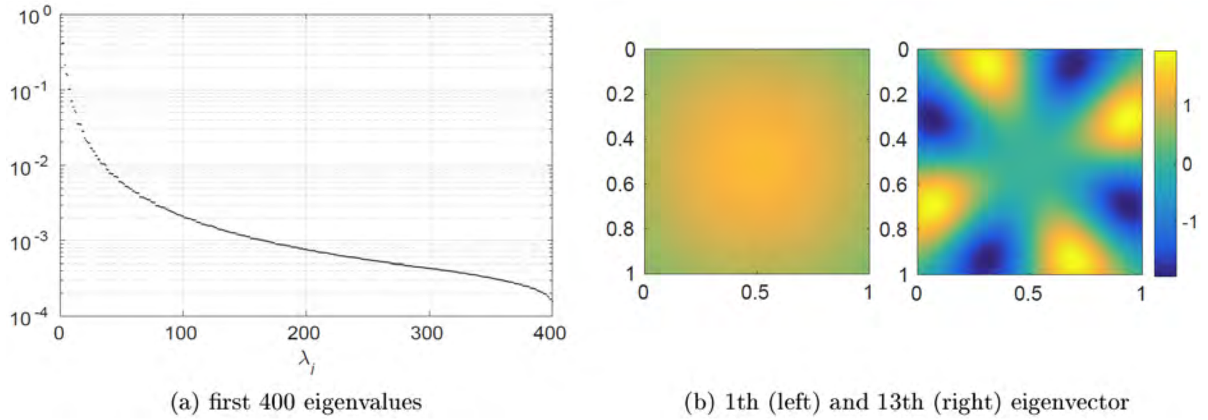
$$\mathbb{A}\overline{\psi_i}=\widetilde{\lambda_i}\cdot\mathbb{W}\cdot\overline{\psi_i},\ \mathbb{A}_{ij}=\int\limits_{\mathcal{D}}\int\limits_{\mathcal{D}}c\left(x,y\right)\cdot\phi_i\left(y\right)\cdot\phi_j\left(x\right)\mathrm{d}y\mathrm{d}x,\ \mathbb{W}_{ij}=\int\limits_{\mathcal{D}}\phi_i\left(x\right)\cdot\phi_j\left(x\right)\mathrm{d}x. \tag{4}$$

The difficult part is the choice of the basis V_n. Before we specify the basis functions, let state some general properties, which simplify the solution of the problem 4. One of such properties is the orthonormality of V_n, which leads to the standard eigenvalue problem (matrix $\mathbb{W}$ become identity matrix). Another useful property is that each of the basis functions is either "odd" or "even" with approximately equal number of "odd" and "even" functions (here parity is understood with respect to the middle of the domain $\mathcal{D}$). This property allows us to permute the matrix $\mathbb{A}$ into block diagonal form of 2^d blocks (d is the dimension of $\mathcal{D}$).

Specific bases tested in this article are three and were constructed as tensor product of 1d bases of piece-wise constant functions, normalized Legendre polynomials or goniometric basis of 1d solution of the equivalent problem (see [5]).

For experimental purposes, the calculation of $\mathbb{A}_{ij}$ is done numerically by the Gauss-Legendre quadrature of 100 points per dimension. Note, that this is very computationally expensive, we need to evaluate the integrand in 100^4 points for each non-zero entry of $\mathbb{A}$. The sample calculation using goniometric basis with 400 basis function (20 in one dimension) can be seen in the Figure 1. Note the very fast decay of eigenvalues, which assures low approximation error of the truncated KLD.

Next we compare the approximation error of the tested bases. We don't know the exact eigenfunctions, so we take the approximation obtained using the piece-wise constant basis of $500\times 500=$

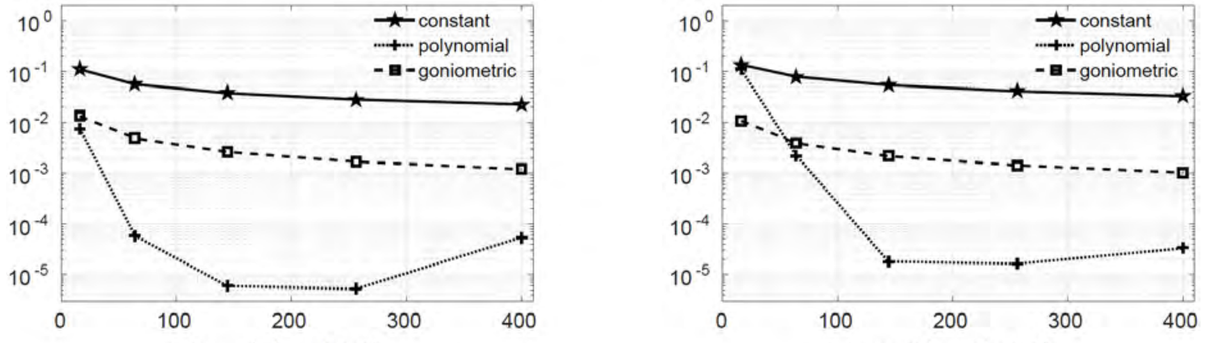


(a) first 400 eigenvalues

(b) 1th (left) and 13th (right) eigenvector

Figure 1: Results for goniometric basis with 400 basis function.

250000 functions as "precise" solution. We measure the error as $L^2(\mathcal{D})$ norm of the difference between the "precise" solution $\psi_i(\boldsymbol{x})$ and the Galerkin approximation $\psi_i^h(\boldsymbol{x})$, both multiplied by square roots of the corresponding eigenvalues, err $\left(\lambda_i^h \cdot \psi_i^h(\boldsymbol{x})\right) = \left\|\lambda_i \cdot \psi_i(\boldsymbol{x}) - \lambda_i^h \cdot \psi_i^h(\boldsymbol{x})\right\|_{L^2(\mathcal{D})}$. The resulting approximation errors for 1st and 13th eigenvector can be seen in the Figure 2. The results show, that the polynomial basis has the best approximation property. In the case of the polynomial basis, the convergence (in the Figure 2) stagnates for the number of basis functions over 100, which is caused by the choice of the "precise" solution and arithmetic instability of the evaluation of the higher order Legendre polynomials. Note, that the convergence of the Galerkin method can be sensitive to the precision of the numerical integration, which is more difficult for fast oscillating functions, such as polynomials or goniometric functions. On the other hand, the piecewise constant basis needs lower number of integration points with increasing number of basis functions.



(a) err $\left(\lambda_1^h \cdot \psi_1^h(\boldsymbol{x})\right)$

(b) err $\left(\lambda_{13}^h \cdot \psi_{13}^h(\boldsymbol{x})\right)$

Figure 2: Convergence of tested bases.

The sampling of GRF, when we have KLD (2), simply means to sample the i.i.d. standard normal variables $\xi_j(\omega)$. Note the similarity with the "eigenvalue decomposition of $\mathbb{C}$" method.

4 Sample usage of sampling and decomposition of GRF

Finally we list some applications of decomposition/sampling of GRF. Our research mostly concerns PDEs with unknown material parameters (e.g. permeability), for which only statistical behaviour is known. In the case of many natural materials, the statistical behaviour (of logarithm) of their properties has Gaussian random field distribution. Our current research topics, that involve GRF, are:

- Multilevel Monte Carlo approximation of the Darcy flow problem solution with random material field, where we only need to generate samples of GRF (see [4]).
- Stochastic Galerkin method for solving the same problem, where the decomposition form of GRF is used (see [6]).
- Bayesian inverse approach to a material field estimation using noised point measurements of the pore pressure and the Darcy's velocity (see [7]).
- Study of robustness of iterative solvers for problems with stochastic, oscillating coefficients.

Acknowledgement

This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602".

References

[1] G.J. Lord, C.E. Powell, T. Shardlow: *An introduction to computational stochastic PDEs.* Cambridge University Press, 2014.

[2] E. Aune, J. Eidsvik, Y. Pokern: *Iterative numerical methods for sampling from high dimensional Gaussian distributions.* Statistics and Computing, 2013.

[3] E. Chow, Y. Saad: *Preconditioned Krylov subspace methods for sampling multivariate Gaussian distributions.* SIAM Journal on Scientific Computing, 2014.

[4] R. Blaheta, M. Béreš, S. Domesová: *A study of stochastic FEM method for porous media flow problem.* Proceedings of the 1st ICAMER. CRC Press, 2016.

[5] D. Zhang, Z. Lu: *An efficient, high-order perturbation approach for flow in random porous media via Karhunen-Loève and polynomial expansions.* Journal of Computational Physics, 2004.

[6] M. Béreš: *Stochastic Galerkin Method for Random Material Problem.* WOFEX 2016. Ostrava, 2016.

[7] S. Domesová: *A Bayesian inverse approach to material parameters estimation in Darcy flow problem.* WOFEX 2016. Ostrava, 2016.

Řešení 3D úloh mechaniky tekutin metodou lattice Boltzmann

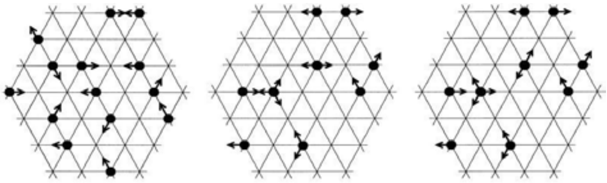
J. Blahoš, T. Karásek, T. Brzobohatý

IT4Innovations, VŠB - Technická univerzita Ostrava

1 Úvod

Metoda lattice Boltzmann (LBM) je jednou z numerických metod používaných pro řešení úloh proudění. Počátek jejího vývoje se datuje někdy do 70. až 80. let.

Předchůdcem LBM je metoda lattice gas. Ta přišla s jednoduchou myšlenkou simulace chování jednotlivých molekul tekutiny. Někdy se mluví o simulaci na mikroskopické úrovni. Výpočetní oblast se diskretizuje do pravidelné mřížky. V každém uzlu mřížky se nacházejí fiktivní částice, které se po mřížce přesouvají a navzájem interagují podle daných pravidel. Částice nikdy nemůže skončit jinde než na jednom z uzlů mřížky. Z toho plyne vedle prostorové diskretizace i diskretizace rychlosti.



Obrázek 1: Schéma kroku algoritmu metody lattice gas

Metoda lattice Boltzmann vychází ze stejných principů. Po pravidelné mřížce se pohybují částice s diskrétními rychlostmi do sousedních uzlů. Dále dochází ke kolizi (úpravě směrů přesunu), s omezením na zachování hmoty a hybnosti. Zároveň metoda LBM přináší některá vylepšení, které odstraňují nedostatky metody lattice gas. Klíčový je fakt, že metoda LBM na makroskopické úrovni řeší Navier-Stokesovy rovnice.

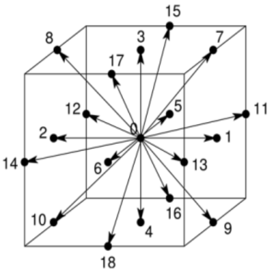
2 Algoritmus

Podstata metody LBM spočívá v manipulaci se sadou reálných čísel v každém uzlu mřížky. Tyto čísla bývají obvykle označovány za hodnoty distribuční funkce, přičemž každá hodnota popisuje množství částic mající jednu z množiny možných rychlostí. Makroskopické veličiny jako je hustota a rychlost proudění tekutiny lze pro daný uzel získat pomocí jednoduchých operací.

Samotný výpočet jednoho časového kroku sestává ze dvou částí - propagace a kolize. Propagace znamená prostý posun hodnot distribučních funkcí k sousedním uzlům ve směru odpovídající rychlosti. V kolizní části se využívá tzv. Bhatnagar-Gross-Krookův operátor, který z fyzikálního hlediska představuje relaxaci k rovnovážnému rozdělení částic. Toto rozdělení je definováno jako:

$$f_i^{eq} = \rho w_i \left[1 + 3\boldsymbol{u}\boldsymbol{v}_i + \frac{9}{2}(\boldsymbol{u}\boldsymbol{v}_i)^2 - \frac{3}{2}\boldsymbol{u}\boldsymbol{u} \right] \tag{1}$$

Jedná se o aproximaci Maxwell-Boltzmannovy distribuční funkce Taylorovým polynomem druhého řádu.



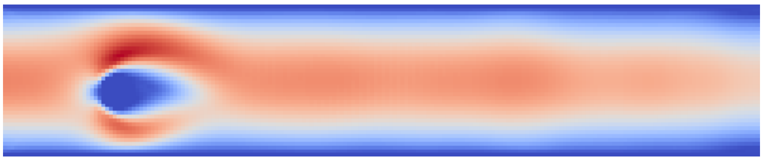
Obrázek 2: Mřížka D3Q19 - nejčastější mřížka používaná ve 3D, využívající 19 směrů přesunu

3 Paralelizace

Klíčovou otázkou v oblasti HPC je možnost a efektivita paralelizace. V tomto ohledu je LBM velice zajímavá, díky povaze použitého algoritmu. V propagační části dochází k prostému kopírování hodnot na sousední uzly. V kolizní části je pak výpočet v rámci uzlu zcela nezávislý. V dnešní době paralelního počítání na čím dál výkonnějších superpočítačích má proto tato metoda velký potenciál.

4 Závěr

V současné době je naimplementovaná základní funkční LBM v jazyce C++, využívající hybridní paralelizaci (MPI a OpenMP). Další výzkum se bude soustředit na optimalizaci implementace, a využitelnost pro řešení úloh s tzv. volnou hladinou a pro řešení rovnice "shallow water".



Obrázek 3: Vizualizace výstupu - řez obtékání válce ve 3D

Poděkování: Tato práce je podporována z projektu SGS - Numerické metody pro modelování environmentálních procesů SP2017/167

Literatura

[1] A.J. Wagner: *A Practical Introduction to the Lattice Boltzmann Method.* North Dakota State University, 2008.

[2] Ch. Peng: *The Lattice Boltzmann Method for Fluid Dynamics: Theory and Applications.* École Polytechnique Fédérale de Lausanne, 2011.

[3] K. Iglberger, N. Thürey, U. Rüde: *Simulation of moving particles in 3D with the Lattice Boltzmann method.* V: Computers and Mathematics with Applications 55 (2008) 1461 1468 Chair for System Simulation, Friedrich-Alexander University Erlangen/Germany

Balancing discretization and algebraic errors in goal-oriented error estimates for nonlinear problems

V. Dolejší, F. Roskovec

Faculty of Mathematics and Physics, Charles Universityin Prague

1 Introduction

Employing the Dual Weighted Residual method (DWR) for the goal-oriented error estimation, see e.g. [1], we propose an adaptive algorithm for solving nonlinear elliptic boundary value problems. Rather than measuring the error in a norm coming from the mathematical formulation of the solved problem, the error is estimated with respect to the so-called target functional, which represents some quantity of special interest. This is achieved by solving an additional (dual) problem having the target functional in its right-hand side. That enables us to adapt the computational mesh directly with respect to this goal, which may fasten the computation dramatically in many cases. On the other hand, the the necessity of solving the adjoint problem brings additional computational costs and also possible errors. Based on the design published in [2] and [3], we propose an adaptive algorithm, which keeps the linear and nonlinear algebraic errors under the level of the discretization error – all of these measured with respect to the target quantity.

2 DWR method for nonlinear problems

We consider the following abstract *primal problem:* Determine $J(u) = \int\limits_{\Omega} j_\Omega(u)\,\mathrm{d}x + \int\limits_{\partial\Omega} j_\Gamma(u)\,\mathrm{d}S$ given that

$$\mathcal{A}(u) = 0 \text{ in } \Omega, \qquad u = u_D \text{ on } \partial\Omega, \tag{1}$$

where $\Omega \subset \mathbb{R}^2$ is a bounded domain and $u_D, j_\Omega, j_\Gamma : \mathbb{R} \to \mathbb{R}$ are given functions.

We discretize the problem (1) by the discontinuous Galerkin method (dG). Then we say that $u_h \in V_h \subset V$ is the dG solution of the primal problem if it satisfies

$$a_h(u_h; \varphi_h) = 0 \qquad \forall \varphi_h \in V_h. \tag{2}$$

We assume that the discretization is consistent, i.e. $a_h(u; \varphi_h) = 0\ \forall \varphi_h \in V_h$. This property gives us the Galerkin orthogonality of the approximate solution.

2.1 Newton method

The problem (2) results in a nonlinear system of algebraic equations and it is further solved by an inexact Newton method. We introduce the damped Newton method for the problem (2). In every step of the method the next approximation is computed as $u_h^{(n+1)} = u_h^{(n)} + \lambda^n d^n$, where d^n is the exact solution of

$$a'_h(u_h^{(n)}; d^n, \psi) = -a_h(u_h^{(n)}; \psi), \forall \psi \in V_h. \tag{3}$$

where $a'_h(u_h^{(n)}; d^n, \cdot)$ denotes the Fréchet derivative of a_h with respect to its first variable at $u_h^{(n)}$ along the direction d^n and $\lambda^n \in (0,1]$ is a parameter which improves the convergence of the method in early iterations.

In each step of the Newton method, the linear system (3) is solved by some iterative method (e.g. GMRES). Due to the error of this iteration method we do not get the exact d^n, but only its approximation $d_A^{(n)}$ and hence in every step of the inexact Newton method we obtain

$$u_{h,A}^{(n+1)} = u_h^{(n)} + \lambda^n d_A^{(n)}.$$

2.2 Dual problem

The dual (adjoint) problem cannot be obtained directly in the nonlinear case. Therefore, according to [1] we employ the Euler-Lagrange equations. That gives us the linearized discrete *dual problem*

$$a'_h(u_h; \varphi_h, z_h) = J'(u_h; \varphi_h) \qquad \forall \varphi \in V_h, \tag{4}$$

where $J'(u_h; \varphi_h)$ denotes the Fréchet derivative of J at u_h along the direction φ_h and $a'_h(u_h; \varphi_h, \cdot)$ is the Fréchet derivative of a_h with respect to its first variable at u_h along the direction φ_h.

Exploiting the dual problem (4) we obtain the error representation of the target quantity

$$J(u) - J(u_{h,A}) \;=\; \frac{1}{2}\left[r_h(u_{h,A})(z - z_h^a) + r_h^*(u_{h,A}, z_h^a)(u - u_{h,A})\right] + r_h(u_{h,A})(z_h^a) + \mathcal{R}_h^{(3)}, \tag{5}$$

where $r_h(u_{h,A})(\varphi) = -a_h(u_{h,A}, \varphi)$, $r_h^*(u_{h,A}, z_h^a)(\psi) = J'(u_h)(\psi) - a'_h(u_h)(\psi, z_h)$ are the primal and dual residual, respectively, and $\mathcal{R}_h^{(3)}$ is a remainder, cubic in the error, which will be neglected.

Unlike the approach presented in [1] we take into account here also the inexact solution of the nonlinear algebraic problem. We note that, if u_h and z_h would be the exact solutions of the primal and dual discrete problems, respectively, then $r_h(u_h)(\varphi_h) = r_h^*(u_h, z_h)(\varphi_h) = 0, \quad \forall \varphi_h \in V_h$.

This property cannot be achieved in real computations. Moreover we do not even intent to obtain algebraically precise solutions, since it would increase the computational cost needlessly when the discretization error is still large. On contrary, we monitor the decrease of the algebraic error and stop iterating when the error is under the level of the discretization error up to a safety parameter $\kappa \in (0,1]$.

We solve both of the linear systems (3) and (4) by a Krylov iterative solver. Then for each inner iteration $i = 1, \ldots, i_{\max}$ we introduce

the *primal linear algebraic error identity*

$$t_{h,A}^n(\cdot) := a'_h(u_h^{(n)})(d_A^{n,i}, \cdot) + a_h(u_h^{(n)}; \cdot) \tag{6}$$

and the *dual linear algebraic error identity*

$$q_{h,A}^n(\cdot) := a'_h(u_h^{(n)})(\cdot, z_h^{n,i}) - J'(u_h^{(n)})(\cdot). \tag{7}$$

If $J : V \to \mathbb{R}$ is a linear functional and we denote $u_{h,A}^{(n+1)} = u_h^{(n)} + \lambda^n d_A^{(n)}$ and $u_h^{(n+1)} = u_h^{(n)} + \lambda^n d^{(n)}$, the inexact solution (due to errors of the linear solver) and the exact solution of one step of the Newton method, respectively, then we get the following error identity (see [3])

$$J(u_h^{(n+1)}) - J(u_{h,A}^{(n+1)}) = -\lambda^n \left(t_{h,A}^n(z_{h,A}^{(n)}) + q_{h,A}^n(d^{(n)} - d_A^{(n)}) \right). \tag{8}$$

2.3 Reconstruction

Unfortunately, the error identity (5) contains the exact solutions u and z, hence is not practically applicable. Therefore, these exact solutions have to be approximated. We can either compute those in some richer discrete space (on finer mesh and/or with higher polynomial degree), see e.g. [4], or we can reconstruct those from u_h and z_h. The method from [4] is very precise, but it increases the computation price significantly.

Therefore, we apply the weighted least-squares reconstruction, which was presented in [5]. For each element K of the triangulation $\mathcal{T}_h$ this method constructs a function $u_K^+ \in P^{p+1}(K)$, which is a solution of the weighted least-squares method on the patch of elements having at least a common vertex with the element K. In this way we construct $u_h^+ \approx u$ and $z_h^+ \approx z$.

3 Adaptive algorithm

Altogether, we define the estimate of the total error in the following way

$$J(u) - J(u_{h,A}^{(n)}) \approx \frac{1}{2}(\eta_{\mathrm{S}} + \eta_{\mathrm{S}}^*) + \eta_{\mathrm{N}} - \sum_{i=0}^{n-1} \lambda^i (\eta_{\mathrm{A}}^i + \eta_{\mathrm{A}}^{*,i}), \tag{9}$$

where we use the following error estimators:

- *primal discretization error estimate* $\eta_{\mathrm{S}} = r_h(u_{h,A}^{(n)})(z_h^+)$
- *dual discretization error estimate* $\eta_{\mathrm{S}}^* = r_h^*(z_{h,A}^{(n)})(u_h^+)$
- *non-linearity estimate* $\eta_{\mathrm{N}} = r_h(u_{h,A}^{(n)})(z_{h,A}^{(n)})$
- *primal algebraic estimate* $\eta_{\mathrm{A}}^i = t_{h,A}^i(z_{h,A}^{(i)})$
- *dual algebraic estimate* $\eta_{\mathrm{A}}^{*,i} = q_{h,A}^i(d_A^{(i)})$

Finally, with all of these error estimators in hands we introduce the adaptive algorithm. The constants $C_A^* \in (0, 1]$ should reduce the number of calculations of the particular estimates and suppress the inexactness of the estimates. We usually set those from $[10^{-3}, 10^{-1}]$.

```
REPEAT UNTIL $\frac{1}{2}(\eta_{\mathrm{S}} + \eta_{\mathrm{S}}^*) < \mathrm{TOL}$ :
    initialize $u_h^{(0)}, z_h^{(0)}$ and $\eta_{\mathrm{S}}$ from previous mesh , k:=0
    FOR $n = 0, \dots, n_{\max}$:
        r := k
        REPEAT UNTIL $\eta_{\mathrm{N}}(u_h^{(k)}, z_h^{(r)}) \le C_A^1 \eta_{\mathrm{S}}(u_h^{(r)}, z_h^{(r)})$ : %Newton iterations
            perform GMRES iterations for (3) with tol. $\eta_{\mathrm{A}}^i(u_h^{(k)}, z_h^{(r)}, d_{\mathrm{A}}^i) \le C_A^2 \eta_{\mathrm{N}}(u_h^{(k)}, z_h^{(r)})$
            find the optimal damping parameter $\lambda^k$, set $u_{h,A}^{(k+1)}$ and $k := k + 1$
        perform GMRES iterations for (4) with tol. $\eta_{\mathrm{A}}^{*,i}(u_h^{(k)}, z_h^{(k,i)}) \le C_A^3 \eta_{\mathrm{N}}(u_h^{(k)}, z_h^{(r)})$
        compute reconstructions $u_h^+, z_h^+$, update $\eta_{\mathrm{S}} = \eta_{\mathrm{S}}(k)$ and $\eta_{\mathrm{S}}^* = \eta_{\mathrm{S}}^*(k)$
        IF $\eta_{\mathrm{A}}(u_h^{(k)}, z_h^{(k)}) < C_A^1 \eta_{\mathrm{S}}$ : EXIT FOR loop
    set $u_{h,A} := u_{h,A}^k$ and $z_h^a = z_{h,A}^k$
    IF $\frac{1}{2}(\eta_{\mathrm{S}} + \eta_{\mathrm{S}}^*) > \mathrm{TOL}$ :
        refine elements with local error estimate $\eta_{\mathrm{S,K}} > \mathrm{TOL}_K = \frac{\mathrm{TOL}}{\#\mathrm{elements}}$
```

4 Conclusion

We presented a robust adaptive algorithm, which is designed to keep discretization and algebraic errors balanced and hence enable efficient computation of various nonlinear elliptic problem solved by dG method. Our experiments document stable and reliable performance of this algorithm. Nevertheless, we should not cover the fact that the algorithm is partly heuristic. We neglect the part of the error corresponding to $u-u_h^+$ and $z-z_h^+$ and also some terms coming from the linearization of the problem. This is a common problem in DWR methods. As far as we know there are some guaranteed estimates for linear problems using equilibrated flux reconstructions, but the extension to nonlinear problems is still unclear.

Acknowledgement: The research of V. Dolejší was supported by Grant No.: 13-00522S of the Czech Science Foundation. The research of F. Roskovec was supported by the Charles University, project GA UK No. 92315.

References

[1] W. Bangerth, R. Rannacher: *Adaptive Finite Element Methods for Differential Equations*, Lectures in Mathematics. ETH Z'urich, 2003.

[2] D. Meidner, R. Rannacher, J. Vihharev: *Goal-oriented error control of the iterative solution of finite element equations*, Journal of Numerical Mathematics **17** (2009), 143 pages.

[3] R. Rannacher, J. Vihharev: *Adaptive finite element analysis of nonlinear problems: balancing of discretization and iteration errors*, Journal of Numerical Mathematics **21** (2013), 23 pages.

[4] P. Šolín, L. Demkowicz: *Goal-oriented hp-adaptivity for elliptic problems*, Computer Methods in Applied Mechanics and Engineering, Vol. 193, No. 6–8 (2004), pp. 449–468.

[5] V. Dolejší, P. Šolín: *hp-discontinuous Galerkin method based on local higher order reconstruction*, Appl. Math. Comput. **279** (2016), pp. 219–235.

Investigating finite-precision Krylov subspace methods via rank-deficiency of the computed subspaces

T. Gergelits, I. Hnětynková, M. Kubínová, Z. Strakoš

Faculty of Mathematics and Physics, Charles University in Prague

1 Short-recurrence Krylov subspace methods

Krylov subspace methods represent a computationally attractive way of solving large and sparse linear algebraic problems of a general form

$$Ax = b, \qquad A \in \mathbb{R}^{n\times n},\ b \in \mathbb{R}^n. \tag{1}$$

Many of these methods rely mathematically on computation of an orthonormal basis of the Krylov subspaces

$$\mathcal{K}_k(A, r_0) \equiv \mathrm{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\}, \quad k = 1, 2, \ldots,$$

where $r_0 = b - Ax_0$. For a symmetric A, a sequence of orthonormal bases generating $\mathcal{K}_k(A, r_0)$, $k = 1, 2, \ldots$, can be computed by short recurrences, represented by the Lanczos tridiagonalization [5]. However, in finite precision computations, the use of short recurrences inevitably leads to the *loss of global orthogonality* and even the loss of linear independence among the generated vectors. Consequently, the computed Krylov subspaces become rank-deficient, which may cause a significant delay of convergence.

In this contribution, we investigate how the first k steps of the finite precision arithmetic computation can be related to the first l steps of the exact computation with the *same matrix and starting vector*.[1] Such pairing allows to compare not only the convergence curves, but also the computed approximations, corresponding residuals, or the generated subspaces.

2 A pairing strategy

We propose the following pairing based on the loss of linear independence in the kth computed Krylov subspace: For each iteration l in the exact computation, we aim at finding the corresponding iteration $k(l)$ in the finite precision computation as

$$k(l) \equiv \max\{j |\ \mathrm{num\_rank}(\bar{V}_j) = l\}, \tag{2}$$

where $\bar{V}_j$ is the matrix of the Lanczos vectors computed in finite precision arithmetic. The definition of numerical rank is generally a subtle issue. In this contribution we investigate several possible approaches.

[1] It should be pointed out that the analyses of Greenbaum and coauthors [3, 4] and Paige and coauthors [6] link the results of finite precision computations to exact computations for *larger problems*.

3 Observed phenomena

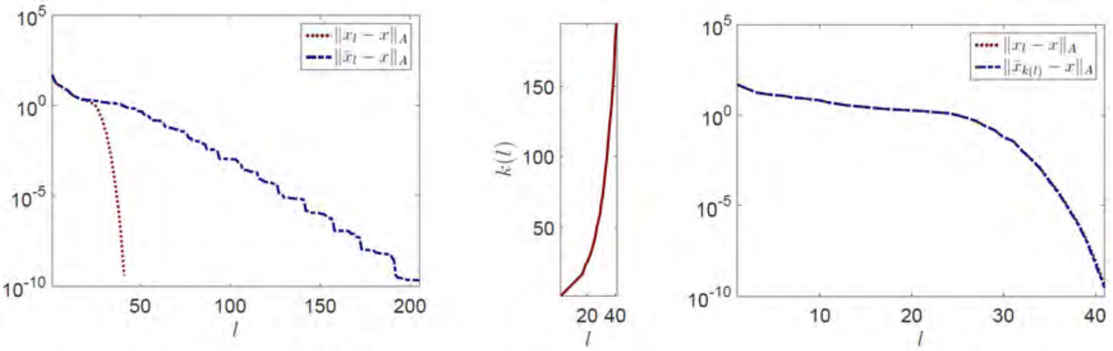
When applying the Lanczos method [5] to the system (1) with a symmetric positive definite matrix A, the quantity of interest is typically the energy norm (A-norm) of the error. Using the pairing (2), we have observed that

$$\|x_l - x\|_A \approx \|\bar{x}_{k(l)} - x\|_A, \tag{3}$$

see Figure 1. Moreover, typically we also have

$$\frac{\|x_l - \bar{x}_{k(l)}\|_A}{\|x_l - x\|_A} \ll 1, \quad \forall l, \tag{4}$$

meaning that the trajectory of the computed approximations $\bar{x}_{k(l)}$ is enclosed in a shrinking 'cone' around the trajectory of approximations x_l from exact arithmetic computations.



(a) convergence curves

(b) pairing

(c) after shift

Figure 1: The matrix `strakos` with $n = 100$, $\lambda_{\min} = 0.1$, $\lambda_{\max} = 1000$, and $\rho = 0.7$, see [7], and a random vector x. Convergence curves before (a) and after pairing (c).

While in the convergence curves of $\|\bar{x}_l - x\|_A$, the loss of orthogonality causes plateaus, see the typical staircase behavior in Figure 1a, in the convergence curves of $\|\bar{r}_k\| = \|b - A\bar{x}_k\|$, it reveals itself in oscillations, see Figure 1b. Contrary to expectations, $\|\bar{r}_{k(l)}\|$ and $\|r_l\|$ cannot be compared directly. We suggest to use the relation between residuals of norm-minimizing and Galerkin methods, see [1], and compare the exact and finite precision residual norms as

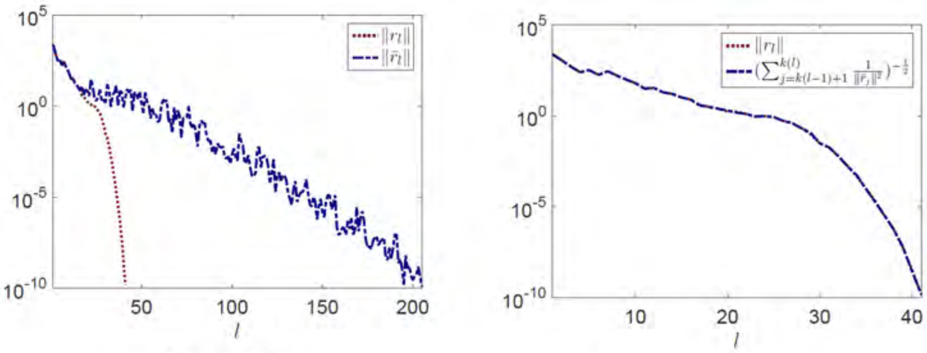
$$\frac{1}{\|r_l\|^2} \approx \sum_{j=k(l-1)+1}^{k(l)} \frac{1}{\|\bar{r}_j\|^2}. \tag{5}$$

The resulting match of the associated curves is shown in Figure 2.

Using the relationship between the Lanczos tridiagonalization and the Golub-Kahan bidiagonalization [2], similar pairings can be applied to bidiagonalization-based methods such as LSQR, Craig or LSMR. Here, however, the fact that solutions and residuals belong to different subspaces, both loosing global orthogonality, represents additional difficulty.

The poster will present some recent results, discussion and illustrative experiments of the studied topics.

Acknowledgment: This work was supported by Charles University, project GAUK 196216.



(a) residuals

(b) after summation + shift

Figure 2: The problem with matrix `strakos` from Figure 1. Residual convergence curves before (a) and after pairing (b).

References

[1] J. Cullum, A. Greenbaum: *Relations between Galerkin and norm-minimizing iterative methods for solving linear systems.* SIAM J. Matrix Anal. Appl. **17**(2), 1996, pp. 223–247.

[2] G. Golub, W. Kahan: *Calculating the singular values and pseudo-inverse of a matrix.* SIAM: Series B, Numerical Analysis 2, 1965, pp. 205–224.

[3] A. Greenbaum: *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences.* Linear Algebra Appl. 113, 1989, pp. 7–63.

[4] A. Greenbaum, Z. Strakoš: *Predicting the behavior of finite precision Lanczos and conjugate gradient computations.* SIAM J. Matrix Anal. Appl. **13**(1), 1992, pp. 121–137.

[5] C. Lanczos: *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators.* Journal of Research of the National Bureau of Standards 45, 1950, pp. 255–282.

[6] C.C. Paige: *An augmented stability result for the Lanczos Hermitian matrix tridiagonalization process.* SIAM Journal on Matrix Analysis and Applications **31**(5), 2010, pp. 2347–2359.

[7] Z. Strakoš: *On the real convergence rate of the conjugate gradient method.* Linear Algebra and its Applications, 154/156, 1991, pp. 535–549.

Stokes system with solution dependent threshold slip boundary conditions: approximation and numerical realization

J. Haslinger, R. Kučera, V. Šátek

Charles University in Prague & Institute of Geonics of the CAS, Ostrava
IT4Innovations, VŠB – Technical University of Ostrava
IT4Innovations, VŠB – Technical University of Ostrava & Brno University of Technology

This contribution deals with an approximation of the Stokes system involving the threshold slip boundary conditions of the Navier type. Unlike the classical Navier condition, this time a slip may occur only when the shear stress attains a threshold bound represented by a function g. We suppose that g is generally a nonlinear function of the tangential component of the flow velocity. Such boundary conditions occur in many practical problems (modeling of polymer melts flow, problems with multiple interfaces, flow of the fluid along hydrophobic surfaces). For the physical justification we refer to [1, 2]. The mathematical and numerical analysis of such a type of problems can be found in [3, 4, 5, 6], e.g. Using the fixed point approach we prove the existence and uniqueness of the solution to this problem for an appropriate class of threshold bounds g. The discretization will be done by P1-bubble/P1 elements which satisfy the LBB-condition. The mesh independent conditions under which the discrete problems have a (unique) solution will be presented and convergence results will be established. Finally, computational experiments will be shown.

References

[1] H. Hervet, L. Leger: *Flow with slip at the wall: from simple to complex fluids.* C.R. Physique **4** (2003), pp. 241–249.

[2] I. Rao, K. Rajagopal: *The effect of the slip boundary conditions on the flow of fluids in a channel.* Acta Mechanica **135** (1999), pp. 113–126.

[3] H. Fujita: *A coherent analysis of Stokes flows under boundary conditions of friction type.* J. of Comput. and Appl. Math. **149** (2002), pp. 57–69.

[4] C.L. Roux, A. Tani: *Steady solutions of the Navier-Stokes equations with threshold slip boundary conditions.* Math. Methods Appl. Sci. **30** (2007), pp. 595–624.

[5] M. Bulíček, J. Málek: *On unsteady internal flows of Bingham fluids subject to threshold slip on the impermeable boundary.* Recent Developments of Mathematical Fluid Mechanics, 2016, pp. 135–156.

[6] R. Kučera, J. Haslinger, V. Šátek, M. Jarošová: *Efficient methods for solving the Stokes problem with slip boundary conditions.* Math. Comput Simulation (in press 2016).

Řešení sdruženého transportu tepla vodou a horninou v kombinaci puklina-matrice

M. Hokr, P. Rálek

Technická univerzita v Liberci

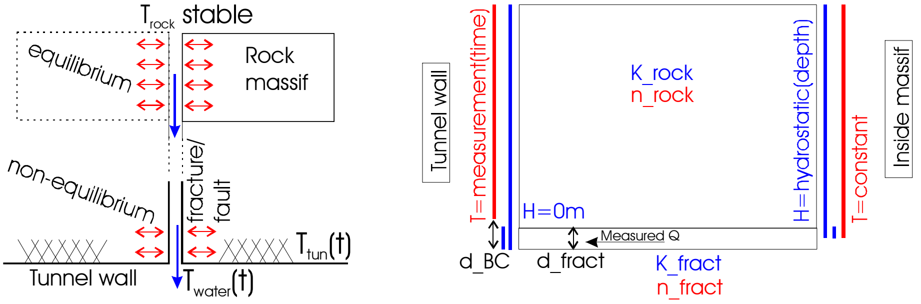
1 Úvod

Řešena je sdružená úloha proudění podzemní vody a transportu tepla v hornině. Modelová oblast se skládá ze dvou částí, málo propustné horniny, kde je dominantní transport tepla vedením a propustné zóny (pukliny), kde je dominantní transport tepla advekcí.

Proces je reprezentován standardními rovnicemi, které jsou numericky řešeny existujícími metodami v převzatém softwaru [1, 2]. Specifika úlohy vyplývají z geometrické konfigurace a propojení procesů mezi sebou. Úloha je postavena na datech reálné lokality a výsledky modelu jsou porovnány s terénním měření. Model tak přispívá k identifikaci vlastností horninového prostředí, i když samotné řešení inverzní úlohy není předmětem prezentované práce. Řešení navazuje na práci [3].

2 Úloha

Model reprezentuje situaci v okolí tunelu, do kterého přitéká voda svislou puklinou a zároveň vlivem využití tunelu v něm sezónně kolísá teplota vzduchu. Situaci je možné popsat buď ve větším měřítku, s celým objemem horniny mezi povrchem a tunelem, kdy model musí být 3D, nebo v menším měřítku okolí tunelu, kdy na základě osové symetrie je možné použít 2D model. Takový případ je znázorněn na Obr. 1 – výchozí koncept je uveden v levé části (předpokládáme ustálený stav daleko od tunelu a ovlivnění horniny v blízkosti tunelu) a modelová úloha s okrajovými podmínkami pro obě rovnice v pravé části. Zatímco měřený průběh teploty na stěně tunelu je zadán jako okrajová podmínka modelu, teplota vyvěrající vody je získána jako postprocesing výsledného pole teploty a je porovnávána s měřením.



Obrázek 1: Schéma řešené úlohy proudění a transportu tepla a zadaných okrajových podmínek: H je hydraulická výška, T je teplota, K je hydraulická vodivost, n je pórovitost, Q je průtok, d označuje rozdílnou šířku odpovídající puklině nebo umístění okrajové podmínky.

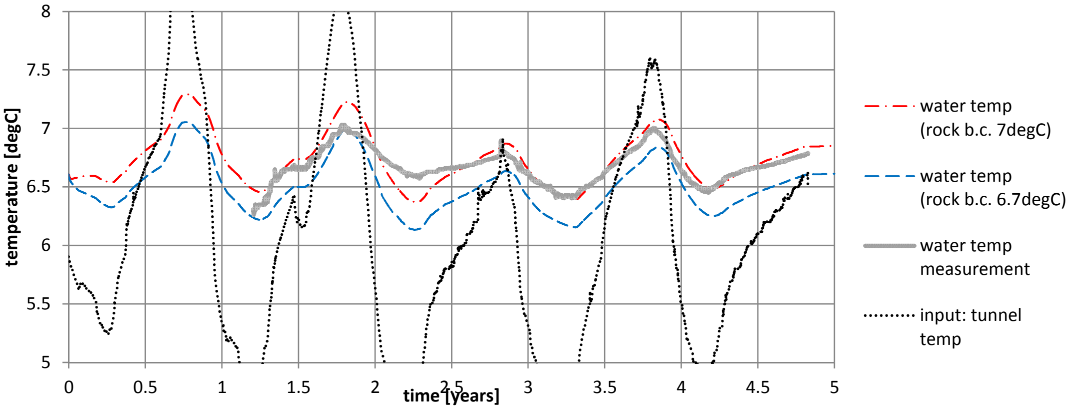
Zatímco běžně uvažované rovnice sdruženého procesu vycházejí z předpokladu lokální rovnováhy mezi teplotou vody a teplotou pevné fáze (zrna horniny), v řešené úloze vzniká nerovnováha ve větším měřítku (decimetry–metry) v důsledku nehomogenity rozložení toku vody, při uplatnění stejných předpokladů rovnováhy v lokálním měřítku.

3 Výsledky

Řešení úlohy ustáleného proudění je možné jednoduše kalibrovat volbou hydraulických vodivostí (K) pro blok horniny a puklinu, na základě měřených toků přes hranici (průsak to tunelu). Tepelné parametry jsou některé známé z literatury, jiné jsou odhadnuty až na základě kalibrace modelu na měřený průběh teploty vyvěrající vody. V úloze je nejistota vyplývající z abstrakce reálných podmínek – ostré rozhraní mezi hranicí se zadanou teplotou (stěna tunelu bez vývěru) a hranicí s vyhodnocovanou teplotou (vyvěrající vodou) (Obr. 1 vpravo).

Příklad porovnání pro částečně kalibrovaný model je na Obr. 2, který ukazuje identifikaci hodnoty ustálené teploty uvnitř masivu (rock b.c.). Model kvalitativně i kvantitativně vystihuje měřený průběh teplot, přičemž odchylky jsou přiměřené nejistotám ve vlastnostech horninového prostředí a nepřesnosti měření. Potvrzuje, že je možné teplotu vody využít jako indikátor nehomogenního toku prostorového rozložení toku vody v hornině.

V příspěvku bude rozebráno, které jednotlivé vlivy odpovídají různým fyzikálním podmínkám v hornině, a které jsou jen důsledek diskretizace nebo abstrakce modelu.



Obrázek 2: Porovnání měřeného časového průběhu teploty vyvěrající vody a dvou variant modelu s různou hodnotou neznámé okrajové podmínky.

Poděkování: Práce vznikla zčásti v rámci institucionální podpory CxI TUL a zčásti je výsledkem řešení projektu Správy úložišť radioaktivních odpadů “Výzkumná podpora pro bezpečnostní hodnocení hlubinného úložiště”.

Literatura

[1] M.G. Trefry, C. Muffels: *FEFLOW: a finite-element ground water flow and transport modeling tool*, Ground Water 45 (5), 2007, pp. 525–528.

[2] TUL: *FLOW123D version 2.0.0, Documentation of file formats and brief user manual*, NTI TUL, 2016, Online: http://flow123d.github.io/

[3] M. Hokr, A. Balvín, P. Rálek: *Estimation of rock fracture properties from thermal variations of the groundwater discharge into a tunnel*, In: FEFLOW 2015 Conference, DHI.

Numerical pricing of two-asset European-style arithmetic Asian options

J. Hozman, *T. Tichý*

Technical University of Liberec
VŠB – Technical University of Ostrava

1 Introduction

Options are the most interesting financial derivatives worldwide, both from the mathematical point of view and the range of financial applications. Therefore, an important part of activities at financial markets consists of using option pricing models, often formulated in the form of partial differential equations (PDE). These models have analytical solutions only under very strong simplifications and in the rest of cases an efficient, robust and accurate numerical approach is needed, cf. [1].

In this paper, we are concerned about pricing of one particular subclass of path-dependent options – continuous arithmetic Asian option contracts on two assets. The corresponding pricing model is formulated by a convection-diffusion-reaction equation, derived in a similar way as the multidimensional Black-Scholes equation for European basket options, see [2].

The numerical approach listed below arises from the concept of the discontinuous Galerkin (DG) method, for survey see [9]. This technique uses higher order piecewise polynomial discontinuous approximation on arbitrary meshes, without any requirement on inter-element continuity. Therefore, it is more suitable for numerical pricing of such exotic options than standard continuous treatment based on finite element methods, see [4].

We proceed as follows. First, we formulate the PDE system for Asian options on two assets and incorporate a dimensionality reduction. Secondly, we realize its discretization and finally, an illustrative numerical experiment is presented.

2 PDE models for two-asset Asian options

We recall the pricing model from [5]. Let V denote the value of two-asset European-style Asian option. This price function $V = V(S_1, S_2, A, t)$ depends on the actual time t, two asset prices $S_1(t)$, $S_2(t)$ and the continuous arithmetic average

$$A(t) = \frac{1}{t}\int_0^t \big(\alpha_1 S_1(u) + \alpha_2 S_2(u)\big)du \tag{1}$$

with positive weights α_1 and α_2 satisfying $\alpha_1 + \alpha_2 = 1$.

Using the standard market assumptions and following a common approach based on multidimensional Ito's lemma, construction of a risk-free portfolio and elimination of the randomness (cf. [5]), we solve the following deterministic parabolic partial differential equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma_1^2 S_1^2 \frac{\partial^2 V}{\partial S_1^2} + \rho\sigma_1\sigma_2 S_1 S_2 \frac{\partial^2 V}{\partial S_1 \partial S_2} + \frac{1}{2}\sigma_2^2 S_2^2 \frac{\partial^2 V}{\partial S_2^2}$$
$$+(r-q_1)S_1\frac{\partial V}{\partial S_1} + (r-q_2)S_2\frac{\partial V}{\partial S_2} + \frac{\alpha_1 S_1 + \alpha_2 S_2 - A}{t}\frac{\partial V}{\partial A} - rV = 0 \qquad (2)$$

for $t \in (0,T)$, $S_1 > 0$, $S_2 > 0$ and $A > 0$. The pricing equation (2) has the following (piecewise) constant parameters: $\rho \in (-1,1)$ – correlation, $r \geq 0$ – risk-free interest rate, $\sigma_i > 0$ – volatility of the i-th asset and $q_i \geq 0$ – dividend yield of the i-th asset, $i = 1,2$. The detailed description of these market parameters can be found in [2].

The specific feature of all Asian options is the way in which the average is incorporated into the payoff function. If we denote the strike price K and maturity T, then we distinguish four basic grouping introduced in Table 1.

payoffs	call	put
fixed strike	$\max(A-K,0)$	$\max(K-A,0)$
floating strike	$\max(\alpha_1 S_1 + \alpha_2 S_2 - A, 0)$	$\max(A - \alpha_1 S_1 - \alpha_2 S_2, 0)$

Table 1: Payoff functions for four basic types of Asian options.

Let us note that (2) with one of the terminal data from Table 1 represents a linear backward Cauchy problem with the parabolic operator degenerated in variable A. This undesirable feature of (2) can be overcome by a suitable dimensionality reduction possible for European-style options only.

In what follows, we consider only Asian options with floating strike. Inspired by approach from [7], we introduce new spatial variables $x = [x_1, x_2]$ as $x_1 = S_1/A$ and $x_2 = S_2/A$ together with the reversal time transformation $\hat{t} = T - t$ ($\hat{t}$ is time to maturity). Then easy calculation (cf. [5]) leads to the transformed forward pricing equation

$$\frac{\partial u}{\partial \hat{t}} - \sum_{i=1}^{2} \frac{\partial}{\partial x_i}\left(I\!D(x)\cdot\nabla u\right) + \sum_{i=1}^{2} b_i(x,\hat{t})\frac{\partial u}{\partial x_i} + \left(r - \frac{\alpha_1 x_1 + \alpha_2 x_2 - 1}{T-\hat{t}}\right) u = 0, \qquad (3)$$

where $u(x,\hat{t})$ is the new pricing function, $I\!D$ denotes the symmetric positive semi-definite matrix

$$I\!D(x) = \{D_{ij}\}_{i,j=1}^2 = \frac{1}{2}\begin{pmatrix} \sigma_1^2 x_1^2 & \rho\sigma_1\sigma_2 x_1 x_2 \\ \rho\sigma_1\sigma_2 x_1 x_2 & \sigma_2^2 x_2^2 \end{pmatrix}, \qquad (4)$$

and vector $(b_1, b_2)^T$ represents a field induced by physical fluxes, component-wisely written as

$$b_i(x,\hat{t}) = \left(\sigma_i^2 + \frac{1}{2}\rho\sigma_1\sigma_2 - r + q_i + \frac{\alpha_1 x_1 + \alpha_2 x_2 - 1}{T-\hat{t}}\right) x_i. \qquad (5)$$

Finally, the equation (3) is equipped with the initial condition u^0 defined as

$$u^0(x) := \begin{cases} \max(\alpha_1 x_1 + \alpha_2 x_2 - 1, 0), & \text{for call,} \\ \max(1 - \alpha_1 x_1 - \alpha_2 x_2, 0), & \text{for put.} \end{cases} \qquad (6)$$

In order to numerically solve Cauchy problem (3)–(5) with (6), it is necessary to localize it on bounded domain $\Omega := (0, x_1^{max}) \times (0, x_2^{max})$, where x_i^{max} stands, in fact, for the maximal price of the scaled i-th asset. We distinguish three parts of the rectangular boundary $\partial\Omega$ defined as $\Gamma_1 = \{0\} \times (0, x_2^{max})$, $\Gamma_2 = (0, x_1^{max}) \times \{0\}$ and $\Gamma_3 = \partial\Omega \cap \mathbb{R}_+^2$. According to [5], for put options, we prescribe the boundary conditions of a mixed type in the following sense

$$\Big(\mathbb{D} \cdot \nabla u(x, \hat{t})\Big) \cdot \vec{n} = 0 \text{ on } \Gamma_i,\ i = 1, 2, \qquad u(x, \hat{t}) = 0 \text{ on } \Gamma_3, \tag{7}$$

where $\vec{n}$ is the outer unit normal to Γ_i. Note that the generalization of all conditions for call options can be done straightforwardly with the aid of the so-called put-call parity, see [2].

Analogously to the afore-mentioned approach, the similar initial-boundary value problem can be formulated (with slight modifications) also for the case of two-asset Asian options with fixed strike. For both cases it is possible to derive the variational formulations and treat their solvability in weighted Sobolev spaces, see [8].

3 DG solver and numerical experiments

Since the governing equation (2) does not have a closed-form solution of Black-Scholes type, the exact solution of the problem (3)–(7) is difficult to find. Therefore, we use a numerical approach based on the DG framework, where the approximate solution is sought in the finite dimensional space S_h^p consisting from piecewise polynomial, generally discontinuous, functions of the p-th order defined on the domain Ω. Similarly as in [6], we introduce the semi-discrete solution $u_h = u_h(\hat{t})$ represented by the system of the ordinary differential equations

$$\frac{d}{d\hat{t}}(u_h, v_h) + \mathcal{A}_h(u_h, v_h) = l_h(v_h)(\hat{t}) \quad \forall\, v_h \in S_h^p,\ \forall\, \hat{t} \in (0, T), \tag{8}$$

where $u_h(0)$ is given by (6), $(\cdot, \cdot)$ denotes the inner product in $L^2(\Omega)$ and the bilinear form $\mathcal{A}_h(\cdot, \cdot)$ stands for the DG semi-discrete variant of diffusion, convection and reaction operator from (3) accompanied with penalties and stabilizations. Finally, the right-hand side form $l_h(\cdot)(\hat{t})$ contains terms arising from boundary conditions The detailed description of the afore-mentioned forms can be found in [9].

Further, to obtain the fully time-space discrete DG formulation, we discretize in temporal variable $\hat{t}$. Here, we consider Crank-Nicolson scheme, which is practically unconditionally stable and gives the second order convergence in time. It is equivalently written as the weighted average of forward Euler and backward Euler methods.

Let τ be the constant time step of partition $(0, T)$, then the DG approximate solution u_h^m of problem (8) at time level $\hat{t}_m$ is computed according to the following numerical scheme

$$\left(u_h^{m+1}, v_h\right) + \frac{\tau}{2}\mathcal{A}_h\left(u_h^{m+1}, v_h\right) = \left(u_h^m, v_h\right) - \frac{\tau}{2}\mathcal{A}_h\left(u_h^m, v_h\right) + \frac{\tau}{2}\Big(l_h(v_h)\left(\hat{t}_{m+1}\right) + l_h(v_h)\left(\hat{t}_m\right)\Big)$$
$$\forall\, v_h \in S_h^p,\ m = 0, 1, \ldots \tag{9}$$

with the starting data $u_h^0 \approx u^0$.

Moreover, one can easily identify that the discrete scheme (9) corresponds to the system of linear equations. More precisely, this system matrix is a composition of particular sparse matrices and has a matrix inverse, which implies the solvability of the discrete problem (9), i.e. the existence and uniqueness of the DG solution.

In conclusion, our aim is to demonstrate the usage of the DG method on the simplified problem of pricing of Asian put options on two underlying assets – exchange rates of EUR and

USD, both with respect to GBP (60% EUR and 40% USD), see Figure 1. All computations are carried out with an algorithm implemented in the solver Freefem++, from a mesh generation/adaptation, over the DG discretization and assembly of a linear algebraic problem to the basic post-processing. The detailed description can be found in [3].

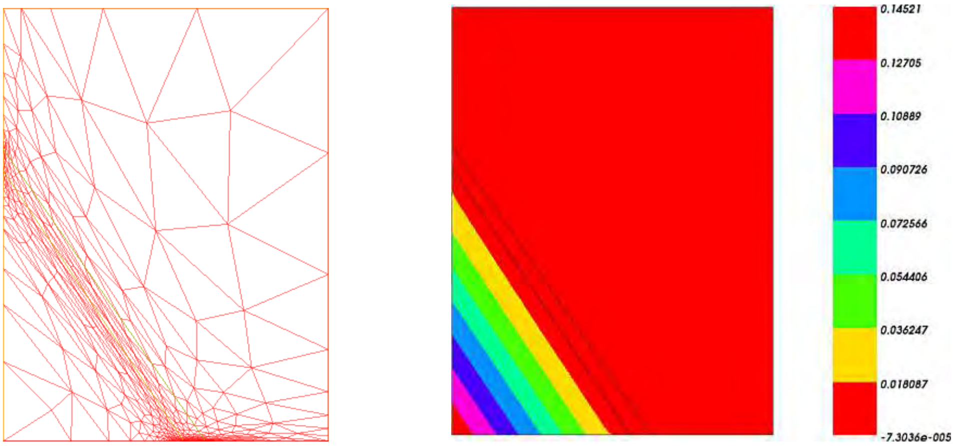


Figure 1: The adaptively refined domain (left) and the corresponding discrete solution at the month maturity (right), zoomed on $[0,3] \times [0,4]$. The model parameters: $\rho = 0.45$, $\sigma_1 = 0.1$, $\sigma_2 = 0.15$, $r = q_1 = q_2 = 0$.

Acknowledgement: Both authors were supported through the Czech Science Foundation under project No. 16-09541S. The support is greatly acknowledged.

References

[1] Y. Achdou, O. Pironneau: *Computational Methods for Option Pricing.* Society for Industrial and Applied Mathematics, Philadelphia, 2005.

[2] E.G. Haug: *The Complete Guide to Option Pricing Formulas.* McGraw-Hill, 2006.

[3] F. Hecht: *New development in FreeFem++.* Journal of Numerical Mathematics **20**, No. 3-4, 2012, pp. 251–265.

[4] J. Hozman, T. Tichý: *Numerical pricing of European basket options with discrete barrier via the discontinuous Galerkin method.* In: Financial Management of Firms and Financial Institutions, Ostrava: VSB-TUO, 2015, pp. 385-395.

[5] J. Hozman, T. Tichý: *DG method for the numerical pricing of multi-asset Asian options – a case of options with floating strike.* Applications of Mathematics, 24 pages, (submitted).

[6] J. Hozman, T. Tichý, D. Cvejnová: *A discontinuous Galerkin method for two-dimensional PDE models of Asian options.* AIP Conference Proceedings **1738**, article no. 080011, 2016.

[7] J.E. Ingersoll Jr.: *Theory of Financial Decision Making.* Rowman & Littlefield Publishers, Inc., New Jersey, 1987.

[8] A. Kufner: *Weighted Sobolev spaces.* Teubner-Texte zur Mathematik, 31, BSB B. G. Teubner Verlagsgesellschaft, Leipzig, 1980.

[9] B. Riviére: *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation.* Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, 2008.

Fuzzy set of functions defined with the help of an auxiliary membership function: A comparison of two approaches

J. Chleboun

Department of Mathematics, Faculty of Civil Engineering, Czech Technical University in Prague

1 Introduction

The contribution deals with integral membership functions defining fuzzy sets of functions.

It is quite common in the application oriented mathematical modeling that some input functions are not known exactly, but are burdened by uncertainty. Take, for instance, a differential equation and its coefficient-function or right-hand side function.

Although stochastic approaches are highly popular among the analysts dealing with uncertainty quantification, see [1, 4, 6, 8], for instance, not all problems involving uncertainty are suitable for the application of stochastic methods. If this is the case, a fuzzy set approach can be beneficial because weaker assumptions are necessary that in the case of stochastic methods.

Nevertheless, two different approaches are used to fuzzify functions. One option is to consider functions with fuzzy values and to generalize the notion of the derivative of such functions, see, for example, [2, 5]. In the other approach, the uncertainty in a function is represented by a set of crisp functions that, in the case of differential equations, can enter the modeled problem as coefficients or right-hand side functions. The set is made fuzzy by a membership function that determines the amount of uncertainty through an α-level value between zero and one. In the latter approach, the differential equation is not generalized, but all the input functions belonging to an α-dependent set (α-level set) has to be taken into account in the analysis of the uncertainty in the model output, see [7], for instance.

In the contribution, two types of membership functions are presented. Both are defined by means of a definite integral and an auxiliary function ρ, but differ in the properties of the auxiliary function. A general framework was sketched in [3] but without any application, algorithmization, and calculations. These are the subject of the current contribution.

2 Membership function

Let $\xi = x_0 < x_1 < \cdots < x_n = \zeta$ be mesh points in an interval $[\xi, \zeta]$ and let a_{low}, a_{upp}, and a_{unc} be three functions continuous on $[\xi, \zeta]$ and linear on $I_i = [x_i, x_{i+1}]$, $i = 0, 1, \dots, n-1$. Moreover, let

$$\forall x \in [\xi, \zeta] \quad a_{\mathrm{low}}(x) < a_{\mathrm{unc}}(x) < a_{\mathrm{upp}}(x).$$

Let us define

$$L_1 = \max_{i=0,1,\dots,n-1} \left\{ \max_{x \in (x_i, x_{i+1})} \left\{ |a'_{\mathrm{low}}(x)|, |a'_{\mathrm{unc}}(x)|, |a'_{\mathrm{upp}}(x)| \right\} \right\}$$

and let us choose $L_2 > 0$.

We are ready to introduce the set of admissible functions that will represent the uncertainty in the function a_{unc}:

$$\mathcal{U}_{\text{ad}} = \left\{ a \in C([\xi,\zeta]) |\ a|_{I_i} \text{ is linear, } \left| a'|_{I_i} \right| \le L_a,\ \left| (a' - a'_{\text{unc}})|_{I_i} \right| \le L_2, i = 0, \dots, n-1 \right\},$$

where $C([\xi,\zeta])$ stands for the continuous functions on $[\xi,\zeta]$ and $L_a \ge L_1$. In the modeling of the uncertain function a_{unc}, a set of functions a will be considered instead of a unique function a_{unc} that nevertheless remains included in $\mathcal{U}_{\text{ad}}$ as its "backbone." The functions from $\mathcal{U}_{\text{ad}}$ are continuous, piecewise linear, bounded from below and above by given functions a_{low} and a_{upp}, the derivative of a is bounded too and, moreover, it cannot deviate from the derivative of a_{unc} by more than L_2.

To fuzzify the set $\mathcal{U}_{\text{ad}}$, we first define $\Omega = \{(x,y) \in \mathbb{R}^2 |\ x \in [\xi,\zeta],\ y \in [a_{\text{low}}(x), a_{\text{upp}}(x)]\}$ and introduce a continuous auxiliary function $\rho : \Omega \to [0,1]$ such that $\rho(x, a_{\text{low}}(x)) = 0$, $\rho(x, a_{\text{unc}}(x)) = 1$, $\rho(x, a_{\text{upp}}(x)) = 0$, and $\rho(x,\cdot)$ is concave for each $x \in [\xi,\zeta]$. The function $\varphi_{\rho,x} = \rho(x,\cdot)$ can be interpreted as the membership function of the fuzzified quantity $a_{\text{unc}}(x)$.

Finally, a membership function associated with $\mathcal{U}_{\text{ad}}$ is defined

$$\mu_{\mathcal{U}_{\text{ad}}}(a) = \frac{1}{\zeta - \xi} \int_{\xi}^{\zeta} \rho(x, a(x))\, \mathrm{d}x, \tag{1}$$

where $a \in \mathcal{U}_{\text{ad}}$. It is obvious that $\mu_{\mathcal{U}_{\text{ad}}}(a_{\text{low}}) = 0 = \mu_{\mathcal{U}_{\text{ad}}}(a_{\text{upp}})$ and $\mu_{\mathcal{U}_{\text{ad}}}(a_{\text{unc}}) = 1$.

Then, for $\alpha \in [0,1]$, an α-level set is defined by

$${}^{\alpha}\mathcal{U}_{\text{ad}} = \{ a \in \mathcal{U}_{\text{ad}} |\ \mu_{\mathcal{U}_{\text{ad}}}(a) \ge \alpha \}. \tag{2}$$

It is common in applications that an a-dependent state problem is to be solved and its solution evaluated by $q(a)$, a scalar quantity of interest. As a consequence of the fuzziness of $\mathcal{U}_{\text{ad}}$, the quantity $q(a)$ is fuzzy and its membership function can be obtained from the Zadeh extension principle [9]. To this end, the extrema of q over ${}^{\alpha}\mathcal{U}_{\text{ad}}$ have to be identified for each $\alpha \in [0,1]$. Such a task can be solved approximately by numerical optimization over ${}^{\alpha}\mathcal{U}_{\text{ad}}$ for a finite set of α-values.

3 Two variants of $\varphi_{\rho,x}$

Cubic polynomials
In general,

$$\varphi_{\rho,x}(y) = c_0(x) + c_1(x)y + c_2(x)y^2 + c_3(x)y^3,$$

where the values $c_j(x)$, $j = 0,1,2,3$, can easily be inferred from the properties of ρ listed above, especially if a computer algebra system (such as Mathematica or Maple, for instance) is employed. To get concave functions, however, $a_{\text{unc}}(x)$ must be rather close to the center of the segment $[a_{\text{low}}(x), a_{\text{upp}}(x)]$.

By using the linearity of a_{low}, a_{upp}, and a on I_i and by further utilization of computer algebra tools, the function $\rho(x, a(x)$ can be expressed as a function of nodal values and x. In detail,

$$\rho(x, a(x) = \gamma(v_{a,n}; x), \tag{3}$$

where $v_{a,n} = (a_0, a_1, \dots, a_n)$ and a_i stands for $a(x_i)$. The expression for the function γ also contains the nodal values of a_{low}, a_{unc}, and a_{upp}, but, unlike $v_{a,n}$, these are considered fixed in the subsequent exposition.

In view of (3) and (1), $\mu_{\mathcal{U}_{\mathrm{ad}}}$ can be identified with μ_n, a function of $n+1$ real variables;

$$\mu_n(v_{a,n}) = \frac{1}{\zeta - \xi} \int_{\xi}^{\zeta} \gamma(v_{a,n}; x)\,\mathrm{d}x. \tag{4}$$

In the already mentioned optimization calculations related to the applications of the Zadeh extension principle, the function μ_n appears in a nonlinear constraint, as seen from (1)-(4). As a consequence, the calculation of $\mu_n(v_{a,n})$ and $\partial\mu_n(v_{a,n})/\partial a_i$, $i = 0, \ldots, n$, is important.

It turns out that $\mu_n(v_{a,n})$ can hardly be directly expressed in terms of general input parameters even if a computer algebra system is used. By implementing computer algebra results combined with numerical integration, parametric expressions defining $\mu_n(v_{a,n})$ and $\partial\mu_n(v_{a,n})/\partial a_i$, $i = 0, \ldots, n$, are available. If generality is sacrificed, that is, numbers instead of parameters are employed in a_{low}, a_{unc}, and a_{upp}, then direct formulae for $\mu_n(v_{a,n})$ and $\partial\mu_n(v_{a,n})/\partial a_i$, $i = 0, \ldots, n$, can be inferred. In this case, however, a change in a_{low}, a_{unc}, or a_{upp} has to be followed by the implementation of a piece of computer code (Matlab code, for instance) determined by the new setting of a_{low}, a_{unc}, and a_{upp} and generated by a relevant computer algebra tool.

Piecewise linear functions
A hat function for each $x \in [\xi, \zeta]$ is the simplest option:

$$\varphi_{\rho,x}(y) = \begin{cases} \dfrac{(y - a_{\mathrm{low}}(x))a_{\mathrm{unc}}(x)}{a_{\mathrm{unc}}(x) - a_{\mathrm{low}}(x)}, & y \in [a_{\mathrm{low}}(x), a_{\mathrm{unc}}(x)], \\ \dfrac{(a_{\mathrm{upp}}(x) - y)a_{\mathrm{unc}}(x)}{a_{\mathrm{upp}}(x) - a_{\mathrm{unc}}(x)}, & y \in [a_{\mathrm{unc}}(x), a_{\mathrm{upp}}(x)]. \end{cases}$$

A parallel to (3) and (4) can be inferred and, unlike the cubic case, the integral corresponding to (4) can be expressed by a formula comprising $v_{a,n}$ and the nodal values of a_{low}, a_{unc}, and a_{upp}. The same is valid for the first partial derivatives of $\mu_n(v_{a,n})$. This benefit is, however, to some degree impaired by the loss of differentiability at the points $(x_i, a_{\mathrm{unc}}(x_i))$, which complicates the optimization related to the Zadeh extension principle. This difficulty can by avoided by decomposing the original optimization problem into a sequence of problems restricted to subdomains where μ_n is differentiable.

4 Conclusions

Both approaches have their advantages and disadvantages. The complexity of cubic (or even higher degree) polynomials prevents obtaining fully analytic formulae for the membership function μ_n and its partial derivatives, but there is no problem with the differentiability of μ_n. Piecewise linear functions allow for analytic formulae, but their piecewise features complicate the algorithmization and coding. In describing fuzziness, on the other hand, piecewise linear functions are more flexible than polynomials.

Acknowledgements

This research has been financially supported by the Czech Science Foundation under grant No. 15-07299S and the conference expenses were covered by the Czech Science Foundation under grant No. 15-12227S. The author also wishes to thank Ms. Lenka Dohnalová for her assistance and Dr. Richard (Dick) Haas for fruitful discussions.

References

[1] I. Babuška, F. Nobile, R. Tempone: *A stochastic collocation method for elliptic partial differential equations with random input data*, SIAM J. Numer. Anal. 45 (3) (2007), pp. 1005–1034.

[2] B. Bede, L. Stefanini: *Generalized differentiability of fuzzy-valued functions*, Fuzzy Sets and Systems 230 (2013), pp. 119–141.

[3] J. Chleboun: *On fuzzy input data and the worst scenario method*, Appl. Math. 48 (6) (2003), pp. 487–496, available at: http://dml.cz/dmlcz/134545, accessed July 4, 2016.

[4] R.G. Ghanem, P.D. Spanos: *Stochastic Finite Elements: A Spectral Approach*, Dover Publications, Mineola, N. Y., 2012, revised edition of the work originally published by Springer-Verlag New York, Inc., in 1991.

[5] L.T. Gomes, L.C. d. Barros, B. Bede: *Fuzzy Differential Equations in Various Approaches*, SpringerBriefs in Mathematics, Springer, Cham, 2015.

[6] H.G. Matthies, A. Keese: *Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations*, Comput. Methods Appl. Mech. Engrg. 194 (12-16) (2005), pp. 1295–1331.

[7] M. Oberguggenberger, S. Pittschmann: *Differential equations with fuzzy parameters*, Math. Comput. Model. Dyn. Syst. 5 (3) (1999), pp. 181–202.

[8] D. Xiu: *Numerical Methods for Stochastic Computations: A Spectral Method Approach*, Princeton University Press, Princeton, NJ, 2010.

[9] H.-J. Zimmermann: *Fuzzy Set Theory–and Its Applications* (with a foreword by L. A. Zadeh), 4th Edition, Kluwer Academic Publishers, Boston, MA, 2001.

CFD motivated applications of model order reduction

M. Isoz

University of Chemistry and Technology, Prague
Institute of Thermomechanics of the CAS, Prague

1 Introduction

The ongoing advances in numerical mathematics and available computing power combined with the industrial needs promote a development of more and more complex models. However, such models are, due to their complexity, expensive from the point of view of the data storage and the time necessary for their evaluation. The model order reduction (MOR) seeks to reduce the computational complexity of large scale models. We present an approach to MOR based on the proper orthogonal decomposition (POD) with Galerking projection, which is well described for example by [4] or [5]. The problems arising from the nonlinearities present in the original model are adressed within the framework of the discrete empirical interpolation method (DEIM) of [1].

The main contribution of this work consists in providing a link between the POD-DEIM based MOR and OpenFOAM [3]. OpenFOAM is an open-source CFD toolbox capable of solving even industrial scale processes. Hence, the availability of a link between OpenFOAM and POD-DEIM based MOR enables a direct order reduction for large scale systems originating in the industrial practice.

2 Model order reduction based on proper orthogonal decomposition and discrete empirical interpolation

The proper orthogonal decomposition is a projection method for reducing the dimensions of general large-scale ODE systems regardless of their origin [4]. However, within our work we will restrict our interest to the systems obtained from the semi-discretization of time dependent or parameter dependent partial differential equations (PDEs). Furthermore, given our interest in OpenFOAM, which is a finite volume method (FVM) based solver for the problems of the computational fluid dynamics (CFD), we will take a special interest in ROM of the large-scales ODE systems generated by the FV discretization of the Navier-Stokes equations.

A scalar nonlinear PDE for an unknown function $y : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}$ may be rewritten as

$$\dot{y} + \mathcal{L}(t, y) = 0\,, \tag{1}$$

where the operator $\mathcal{L}$ represents all the terms of the original PDE apart from the temporal derivative. After the FV semi-discretization of the equation (1) one obtains the system

$$\Delta\Omega^h \dot{y} + \mathcal{L}^h(t, y) = 0\,, \tag{2}$$

where $\mathcal{L}^h(t, y)$ is the FV spatial discretization operator corresponding to the operator $\mathcal{L}$ and $\Delta\Omega^h := \operatorname{diag}(\delta\Omega_i^h) \in \mathbb{R}^{m\times m}$ is a diagonal matrix in which the symbol $\delta\Omega_i^h$ represents a volume of one element of the computational domain discretization.

In the OpenFOAM software, the operator $\mathcal{L}^h(t,y)$ has the structure $\mathcal{L}^h(t,y) = -\tilde{A}(t)y - \tilde{b}(t,y)$, where the linear, implicitly discretized, members are lumped in the term $\tilde{A}(t)y$ and the explicitly discretized nonlinearities are used for the construction of the vector $\tilde{b}(t,y)$. The size of the matrix $\tilde{A}$ and of the vector $\tilde{b}$ is determined by the number of the cells in the FV discretization mesh, m. Because the matrix $\Delta\Omega^h$ is diagonal, its inversion is cheap and one can rewrite the equation (2) as a (large) system of ODEs,

$$\dot{y} = A(t)y + b(t,y), \quad y(0) = y_0, \quad A(t) = (\Delta\Omega^h)^{-1}\tilde{A}(t),\, b(t,y) = (\Delta\Omega^h)^{-1}\tilde{b}(t,y)\,. \tag{3}$$

As POD is a projection method, its main objective is to find a subspace approximating a given set of data in an optimal least-square sense. In our case, the data is generated by sampling the solution of the full order model (3) at given times, $\{y_j := y(t_j)\}_{j=1}^n$, $t_j \in (0,T]$. These samples are called snapshots. The details on the theory of POD may be found for example in [4]. We will restrict our description to a sketch of the process of the reduced order model construction.

Let us denote the space containing the solution of the system (3) and its orthogonal basis as $V = \operatorname{span}\{\psi_j\}_{j=1}^d$. Then it is possible to rewrite the solution of (3) as

$$y(t) = \sum_{j=1}^d \eta_j\,\psi_j,\, \forall t \in [0,T], \quad \eta_j(t) := \langle y(t), \psi_j \rangle_W, \quad d = \dim(V)\,, \tag{4}$$

where by $\langle\cdot,\cdot\rangle_W$ we denote a W-weighted inner product in the L^2 space. The Fourier coefficients η_j, $j = 1,\dots,d$, are functions that map $[0,T]$ into $\mathbb{R}$.

We arrange the members of the sum in (4) in descending order by the amount of information on the original system they carry, take the first $l \le d$ members of and introduce the ansatz

$$y^\ell(t) = \sum_{j=1}^\ell \eta_j^\ell\,\psi_j,\, \forall t \in [0,T], \quad \eta_j^\ell(t) := \langle y^\ell(t), \psi_j \rangle_W, \quad l \le d\,, \tag{5}$$

which is an approximation of $y(t)$ provided $\ell < d$. Inserting (5) into (3) and assuming that the equality holds after projection of V on the ℓ-dimensional subspace $V^\ell = \operatorname{span}\{\psi_j\}_{j=1}^\ell$ we obtain the following system,

$$\dot{\eta}^\ell = A^\ell\eta^\ell + f^\ell(t,\eta^\ell),\, \forall\, t \in (0,T], \quad \eta^\ell(0) = \eta_0^\ell\,, \tag{6}$$

where we defined the reduced system matrix

$$A^\ell := (a_{ij}^\ell) \in \mathbb{R}^{l\times l}, \quad a_{ij}^\ell = \langle A\psi_j, \psi_i \rangle_W\,, \tag{7}$$

the ROM nonlinearities $f^\ell = (f_i^\ell)^{\mathrm{T}} : [0,T] \to \mathbb{R}^\ell$, $f_i^\ell(t,\eta) = \left\langle f\left(t, \sum_{j=1}^\ell \eta_j\psi_j\right), \psi_i \right\rangle_W$, and the initial condition $\eta^\ell(0) = \eta_0^\ell = (\langle y_0, \psi_1\rangle_W, \dots, \langle y_0, \psi_l \rangle_W)^{\mathrm{T}}$. The dimension of the newly defined system (6) is $\ell \le d \le m$.

The quality of the approximation is largely dependent on the choice of basis functions $\{\psi_j\}_{j=1}^\ell$. For the sake of brevity, let us only state (the proof may be found in [5]) that the columns of the matrix $\Psi \in \mathbb{R}^{m\times\ell}$ calculated via the Algorithm 1 are a suitable basis for the discrete representation of the space V^ℓ.

Furthermore, to make ROM completely independent of the full system dimension, it is necessary to address two issues. The first issue is the time dependence of the matrix A, which would cause the need to recalculate the matrix A^ℓ for each ROM evaluation.

A way to resolve the time dependence of the matrix A is to sample the system matrices the same way as the full system solution and to interpolate between the full system matrix snapshots. If one uses the linear interpolation, it is possible to write the approximate system matrix as

$$\hat{A}(t) := \varpi(t)A_{i-1} + (1-\varpi(t))A_i, \quad \varpi(t) = \frac{t-t_{i-1}}{t_i - t_{i-1}}, \quad i = 1,\dots,n\,. \tag{8}$$

Substituting the approximation (8) of the matrix A into A^ℓ matrix definition (7), one may define an approximate time dependent matrix of the reduced system as

$$\hat{A}^\ell(t) := \Psi^{\mathrm{T}} W \hat{A}(t)\Psi = \varpi(t)\Psi^{\mathrm{T}} W A_{i-1}\Psi + (1-\varpi(t))\Psi^{\mathrm{T}} W A_i \Psi = \varpi(t)A^\ell_{i-1} + (1-\varpi(t))A^\ell_i \tag{9}$$

and the reduced order model, once it is created, stays fully independent on the full system dimension.

The second problem arises when you look closely at the nonlinearities in (6). One may notice that to evaluate the non-linearity in the reduced order model $f^\ell(t,\eta^\ell)$, it is necessary to evaluate the function f at (t, y^ℓ) and $y^\ell(t) = \sum_{j=1}^{\ell} \eta_j^\ell(t)\psi_j \in \mathbb{R}^m$. This significantly increases the cost of the evaluation of ROM. In this work, we address this problem via the discrete empirical interpolation method of [1].

Algorithm 1 POD basis of rank ℓ

```
Require: Snapshots $\{y_j\}_{j=1}^n$, POD rank $\ell \le d$, symmetric positive-definite matrix of weights $W \in \mathbb{R}^{m \times m}$
Set $Y = [y_1,\dots,y_n] \in \mathbb{R}^{m\times n}$;
Determine $\bar{Y} = W^{1/2}Y \in \mathbb{R}^{m\times n}$;
Compute SVD, $[\bar{\Psi},\Sigma,\bar{V}] = \mathrm{svd}(\bar{Y})$;
Set $\sigma = \mathrm{diag}(\Sigma)$;
Compute $\varepsilon(l) = \sum_{i=1}^{\ell}\sigma_i / \sum_{i=1}^{d}\sigma_i$;
Truncate $\bar{\Psi} \leftarrow [\bar{\psi}_1,\dots,\bar{\psi}_\ell] \in \mathbb{R}^{m\times\ell}$;
Compute $\Psi = W^{-1/2}\bar{\Psi} \in \mathbb{R}^{m\times\ell}$;
return POD basis, $\Psi$, and ratio $\varepsilon(\ell)$
```

Algorithm 2 DEIM

```
Require: $p$ and $F = [f_1,\dots,f_n] \in \mathbb{R}^{m\times n}$
Compute POD basis $\Phi = [\phi_1,\dots,\phi_p]$ for $F$
idx ← $\arg\max_{j=1,\dots,m} |(\phi_1)_{\{j\}}|$;
$U = [\phi_1]$ and $\vec{i} = $ idx;
for $i = 2$ to $p$ do
    $u \leftarrow \phi_i$;
    Solve $U_{\vec{i}}c = u_{\vec{i}}$;
    $r \leftarrow u - Uc$;
    idx ← $\arg\max_{j=1,\dots,m} |(r)_{\{j\}}|$;
    $U \leftarrow [U, u]$ and $\vec{i} \leftarrow [\vec{i}, \mathrm{idx}]$;
end for
return $\Phi \in \mathbb{R}^{m\times p}$ and index vector, $\vec{i} \in \mathbb{R}^p$
```

DEIM is a combination of the greedy algorithm and POD. The reduction of the computational cost of the system nonlinearity evaluation is achieved by reducing the size of the argument of the function f (assuming it is point-wise evaluable). The details of the procedure may be found for example in the aforementioned article by [1]. We give only the method algorithm summarized in the Algorithm 2. The outputs of the Algorithm 2 may be used to approximate the nonlinearity in ROM by

$$f^\ell(t,\eta^\ell) \approx \tilde{f}(t,\eta^\ell) := \Psi^{\mathrm{T}} W \Phi (P^{\mathrm{T}}\Phi)^{-1} f(t, P^{\mathrm{T}}\Psi\eta^\ell)\,, \tag{10}$$

where the nonlinearity argument $P^{\mathrm{T}}\Psi\eta^\ell$ is in $\mathbb{R}^p$, $p \le m$. We would like also to emphasize that using DEIM, the nonlinearity samples $\{f_j := f(t_j, y_j)\}_{j=1}^n$ need to be included in the solution snapshots.

3 Reduced order model construction for incompressible Navier-Stokes equations

The application of the POD-DEIM based model order reduction to the systems originating in the FV discretization of the incompressible Navier-Stokes equations is not completely straightforward. In the incompressible Navier-Stokes equations,

$$\begin{aligned} u_t + \nabla\cdot(u\otimes u) - \nabla\cdot(\nu\nabla u) + \nabla p &= f\,, \\ \nabla\cdot u &= 0\,, \end{aligned} \tag{11}$$

the continuity equation $\nabla \cdot u = 0$ is pressure free. Thus, their discretization ultimately leads to a system of linear algebraic equations of the form,

$$\begin{pmatrix} M & N^T \\ N & 0 \end{pmatrix} \begin{pmatrix} u^h \\ p^h \end{pmatrix} = \begin{pmatrix} f^h \\ 0 \end{pmatrix}, \tag{12}$$

where we denoted the discrete representations of the considered functions by the superscript h. The matrix N coincides with a discrete representation of the ∇ operator. The matrix M is slightly more difficult. The Navier-Stokes equations for an incompresible isothermal flow (11) are nonlinear. Hence, the nonlinear convective term $\nabla \cdot u \otimes u$, needs to be linearized during the construction of the matrix M. If we apply the Newton linearization to the nonlinear convective term,

$$\nabla \cdot u^j \otimes u^j \approx u^{j-1} \nabla u^j + u^j \nabla u^{j-1}, \quad j \ldots \text{current time step/iteration}\,, \tag{13}$$

we can define a linear operator

$$\mathcal{M}(u^{j-1}, u^j) := \dot{u}^j + \nabla \cdot (\nu \nabla u^j) + \mathcal{P}(u^{j-1}, u^j)\,, \tag{14}$$

where $\mathcal{P}$ represents the Newton linearization operator. Then, the matrix M is a discrete representation of the operator $\mathcal{M}$.

The matrices M and N are, as results of the FV discretization, large and sparse. The system (12) is a so called saddle point problem and as such, it cannot be directly solved by the available methods of numerical linear algebra.

However, if one assumes the matrix M to be regular, it is possible to explicitly express the velocity from the first row of the system (12) and to substitute for it in the second one. Doing so, the following system for one unknown $p^h \in \mathbb{R}^m$ is obtained

$$N M^{-1} N^{\mathrm{T}} p^h = N M^{-1} f, \quad u^h = M^{-1} \left(f - N^{\mathrm{T}} p^h \right). \tag{15}$$

Nevertheless, as M is a large sparse matrix, its inverse is usually not obtainable and the system (15) needs to be solved iteratively by alternatively updating the values of p^h and u^h (see e.g. [2] or any description of SIMPLE or PISO algorithms).

The natural variable for the solution techniques for the incompressible Navier-Stokes equations based on the solution of the system (15) is the pressure. Thus, it would seem reasonable to base the reduced order model directly on the equation (15). Unfortunately, the pressure time derivative is not explicitly present in the Navier-Stokes equations and as a consequence neither in the system (15). Hence, it is not possible to rewrite the equation (15) in the form of the equation (1) directly.

To define the base system for the construction of ROM for the pressure we propose to split the operator $\mathcal{M}$ as

$$\mathcal{M} = \mathcal{M}_t + \mathcal{M}_x, \quad \mathcal{M}_t(u^j) := \dot{u}^j, \quad \mathcal{M}_x(u^{j-1}, u^j) = \nabla \cdot (\nu \nabla u^j) + \mathcal{P}(u^{j-1}, u^j)\,. \tag{16}$$

Moreover, we will denote the implicitly discretized part of the operator $\mathcal{M}$ as $\mathcal{M}^{\text{fvm}}$ and the explicitly evaluated part of the operator $\mathcal{M}$ as $\mathcal{M}^{\text{fvc}}$. Finally, the base system for the construction of ROM for the pressure is defined as

$$\dot{p} \approx A(t)p + b(t,p), \quad A(t) := N^{\text{fvm}} \left(M_x^{\text{fvm}} \right)^{-1} \left(N^{\text{fvm}} \right)^{\mathrm{T}}, \, b(t,p) := N^{\text{fvc}} \left(M_x^{\text{fvc}} \right)^{-1} f\,. \tag{17}$$

Please note, that the pressure derivative is defined only approximately as, at the moment, we do not have the proof of equality. Also let us emphasize that because of the linearization, the coefficients of the matrix M_x^{fvm} are dependent on the current velocity field and as a consequence, the matrix A is time dependent.

4 Numerical examples

In order to validate the proposed method, we performed a series of numerical tests. The first presented test is a creation of the reduced order model for a transient laminar flow in the vicinity of a cylindrical obstacle. Such a flow develops an instability that leads to the formation of the famous von Kármán vortex street. A comparison of the results of the full model (FVM simulation of the full Navier-Stokes equations on approximately 18000 cells) and the created ROM (system of 12 ODEs) is depicted in the Figure 1.

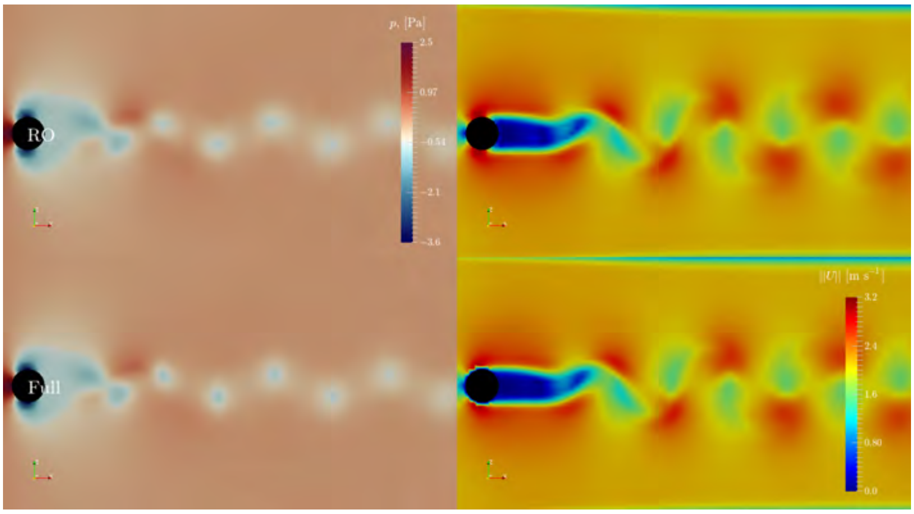


Figure 1: Qualitative comparison of the results of the CFD simulation and ROM results for the case of the flow around a cylindrical obstacle (von Kármán vortex street). Results of the ROM are depicted in the top part of the figure, results of the full model are depicted in the bottom. The left part of the image is colored according to the pressure field, the right part according to the velocity magnitude.

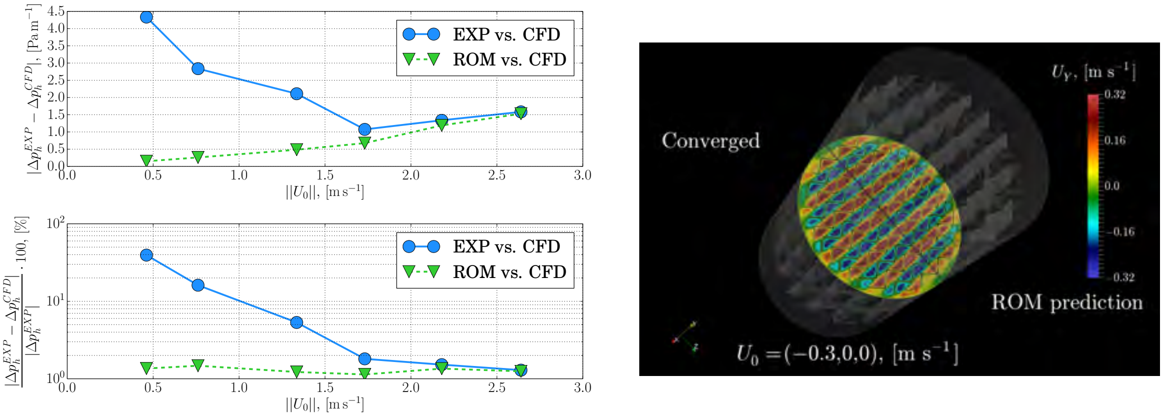


Figure 2: Comparison of the difference between the experimental data, CFD simulation and ROM estimate is depicted on the left side of the figure. On the right side, it is shown the qualitative comparison between the CFD result and ROM estimate for the velocity field.

The second selected test was a creation of ROM for a parametric study of the gas flow in a structured packing of the distillation columns. In this case, we assumed the flow to be at steady state and we studied the difference in pressure above and bellow the structured packing relative to the height of the packing, Δp_h in dependence on the gas inlet velocity.

The results of the test are depicted in Figure 2. The full model corresponds to a FVM simulation of Reynolds-averaged Navier-Stokes equations on approximately $5 \cdot 10^6$ cells. The created ROM consisted of 11 linear algebraic equations.

5 Conclusions

We proposed and validated an approach to use the proper orthogonal decomposition and the discrete empirical interpolation for the model order reduction of systems arising from the finite volume spatial discretization of the incompressible Navier-Stokes equations. The presented approach is specifically designed for the pressure-based Navier-Stokes equations solution methods (e.g. SIMPLE, SIMPLEC or PISO algorithms). We were able to link the proposed method with the OpenFOAM software whereby the method could have been tested even on systems with millions of cells and unstructured meshes. In the future, we plan to improve the mathematical background of the proposed approach to the model order reduction of the Navier-Stokes equations. Also, we would like to concentrate on the model order reduction for multiparametric systems.

Acknowledgments: The work of M. Isoz was supported by the Centre of Excellence for nonlinear dynamic behaviour of advanced materials in engineering CZ.02.1.01/0.0/0.0/15_003/0000493 (Excellent Research Teams) in the framework of Operational Programme Research, Development and Education. Moreover, the author thankfully acknowledges financial support from IGA of UCT Prague, grant numbers A2_FTOP_2016_024 and A1_FCHI_2016_004 and from the Czech Science Foundation, grant number GACR 13-01251S. Also, the author would like to express his deepest thanks to the Mass Transfer Laboratory of UCT Prague for providing theirs, yet unpublished, experimental data for the model validation. Last, but not least, the author would like to thank Michael Hinze from the University Hamburg for many helpful discussions on the proper orthogonal decomposition.

References

[1] S. Chaturantabut, D.C. Sorensen: *Nonlinear model reduction via discrete empirical interpolation.* SIAM J. Sci. Comp. 32, 2010, pp. 2737–2764.

[2] H.C. Elman, G.H. Golub: *Inexact and preconditioned Uzawa algorithms for saddle point problems.* SIAM J. Num. Anal. 31, 1994, pp. 1645–1661.

[3] OpenCFD. *OpenFOAM: The Open Source CFD Toolbox. User Guide.* OpenCFD Ltd, UK, 2016.

[4] R. Pinnau: *Model reduction via Proper Orthogonal Decomposition.* In W.H.A. Schilders, H.A. van der Vorst, J. Rommes (eds.): Model order reduction: theory, research aspects and applications, Springer Berlin Heidelberg, Berlin, Germany, 2008, pp. 95–109.

[5] S. Volkwein: *Proper orthogonal decomposition: theory and reduced-order modelling.* Universtity of Konstanz, Konstanz, Germany, 1. edition, 2013.

An adaptive recursive multilevel approximate inverse preconditioning: Computation of the Schur complement

J. Kopal, M. Rozložník, M. Tůma

Institute of Computer Science of the CAS, Prague
Institute of Mathematics of the CAS, Prague
Institute of Computer Science of the CAS, Prague & Charles University in Prague

1 Introduction

Many problems arising from mathematical modelling in science and engineering lead to solving system of linear algebraic equations in the form

$$\mathbb{A}x = b, \quad \mathbb{A} \in \mathbb{R}^{n\times n}, \quad x \in \mathbb{R}^n, \quad b \in \mathbb{R}^n, \tag{1}$$

where A is a large and sparse. Iterative methods based on Krylov subspaces such as the conjugate gradient method represent a natural choice for solving such problems. Most often, in order to be efficient, it is useful to employ an iterative method together with preconditioning.

The incomplete Cholesky factorization is often a preconditioner of choice. Application of this preconditioner needs at every iteration backward and forward substitutions and thus represents a bottleneck in parallel environment. We focus on a particular strategy to obtain the preconditioner via the generalized Gram–Schmidt process which may avoid these steps as it has been introduced in [1]. The approach originally uses a relative dropping strategy employing magnitudes of entries of the computed column vectors in the approximate inverse factor. Dropping strategy has been later extended in [2] to adaptive strategy. Later in [3] it has been summarized and accompanied by new theoretical results.

2 Approximate inverse preconditioning

In more detail we deal with approximate inverse factorizations for symmetric and positive definite matrices in the form

$$(\tilde{P}^T \mathbb{A} \tilde{P})^{-1} = (U^T U)^{-1} \approx \tilde{Z}\tilde{Z}^T,$$

which arise from the incomplete version of the generalized Gram–Schmidt proces with pivoting. The permutation matrix $\tilde{P}$ represents pivoting and forces the following inequalities for the entries of the Cholesky factor $U = [\alpha_{j,i}]$

$$\alpha_{1,1} \gtrapprox \alpha_{2,2} \gtrapprox \cdots \gtrapprox \alpha_{n,n} > 0, \tag{2}$$

$$\alpha_{k,k} \gtrapprox |\alpha_{k,j}|, \quad k = 1, \ldots, n, \quad j = k+1, \ldots, n, \tag{3}$$

which also implies

$$\alpha_{k,k}^2 \gtrapprox \sum_{i=k}^{j} \alpha_{i,j}^2, \quad j = k+1, \ldots, n. \tag{4}$$

It has been shown in [2, 3] that employing pivoting may significantly increase the robustness of preconditioning and also often reduces sparsity of the factor $\tilde{Z}$ in terms of nonzero entries $\mathrm{nnz}(\tilde{Z})$.

The goal of the incomplete algorithms is to deliver a sparse representation of the factorization, which is close to the original factorization in some sense. Incomplete algorithms very often use dropping rules that discard entries of small magnitudes. We deal with an extreme case of the inverse of the upper triangular matrix with entries satisfying the inequalities (2), (3), and (4). The parametrized matrix introduced by Kahan (1966)

$$U_n(\Theta) = \mathrm{diag}[1, s, \dots, s^{n-1}] \begin{bmatrix} 1 & -c & -c & \dots & -c \\ & 1 & -c & \dots & -c \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & -c \\ & & & & 1 \end{bmatrix},$$

where $s = \sin\Theta$ and $c = \cos\Theta$ is very useful in practice. For the entries of its inverse of $U_n^{-1}(\Theta) = [\beta_{i,j}]$ we can write

$$\beta_{i,j} = 0, \quad j < i, \qquad \beta_{i,j} = s^{1-j}, \quad i = j, \qquad \beta_{i,j} = s^{1-j}c(1+c)^{j-i-1}, \quad j > i. \tag{5}$$

It is easy to see that the entries in the strictly upper triangular part of $U_n^{-1}(\Theta)$ for $\Theta \in (0, \frac{\pi}{2})$ show an exponential growth when increasing their distance from the diagonal. For such problems it is very hard to find a sparse approximation even if the original matrix is sparse. We hope that this can be remedied by employing a multilevel framework that uses block densities to determine the levels of the scheme. The framework then enables to store dense portions of the column vectors in the preconditioner implicitly with a significantly less number of nonzeros. Let us also mention that a matrix in the new level is formed as an approximate Schur complement.

3 Multilevel scheme in approximate inverse preconditioning

Assume that $\mathbb{A}^{(1)} = \mathbb{A}$ is a symmetric and positive definite matrix. Consider the following sequence of symmetrically permuted symmetric and positive definite matrices for $\ell = 1, \dots, \ell_{max}$ using the notation

$$(\tilde{P}^{(\ell)})^T \mathbb{A}^{(\ell)} \tilde{P}^{(\ell)} = \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \\ (B^{(\ell)})^T & C^{(\ell)} \end{bmatrix}, \tag{6}$$

where $\tilde{P}^{(\ell)} \in \mathbb{R}^{n\times n}$ is a permutation matrix, $A^{(\ell)} \in \mathbb{R}^{n_\ell \times n_\ell}$, $B^{(\ell)} \in \mathbb{R}^{n_\ell \times \bar{n}_\ell}$, and $C^{(\ell)} \in \mathbb{R}^{\bar{n}_\ell \times \bar{n}_\ell}$, $\bar{n}_\ell = n - \sum_{k=1}^{\ell} n_k$. Let us note that n_ℓ plays the role of size of the problem at level ℓ.

In exact arithmetic the Schur complement of the block $A^{(\ell)}$ of the matrix (6) is equal to

$$\mathbb{A}^{(\ell+1)} = C^{(\ell)} - (B^{(\ell)})^T (A^{(\ell)})^{-1} B^{(\ell)}. \tag{7}$$

Naturally, there are several ways to obtain the approximation of $\mathbb{A}^{(\ell+1)}$ when approximate inverse factorization of $A^{(\ell)}$ is explicitly known. We studied three ways how to construct an approximation of $\mathbb{A}^{(\ell+1)}$ forming a matrix of the new level. The most simple of them is to compute

$$\mathbb{A}_{cgs}^{(\ell+1)} = C^{(\ell)} - \left((\tilde{Z}^{(\ell)})^T B^{(\ell)}\right)^T (\tilde{Z}^{(\ell)})^T B^{(\ell)}, \tag{8}$$

which in terms of quantities of the Gram–Schmidt process corresponds to its *classical* variant.

Alternatively we can approximate the block $U_{1,2}^{(\ell)} \approx \tilde{U}_{1,2}^{(\ell)}$ by the quantities of the *modified* Gram–Schmidt process, i.e., as

$$\mathbb{A}_{mgs}^{(\ell+1)} = C^{(\ell)} - (\tilde{U}_{1,2}^{(\ell)})^T \tilde{U}_{1,2}^{(\ell)}, \tag{9}$$

where

$$\tilde{U}_{1,2}^{(\ell)} = \begin{bmatrix} (\tilde{z}_1^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \\ (\tilde{z}_2^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \left(I - \begin{bmatrix} \tilde{z}_1^{(\ell)} (\tilde{z}_1^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \\ 0 \end{bmatrix} \right) \\ \vdots \\ (\tilde{z}_j^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \prod_{k=1}^{j-1} \left(I - \begin{bmatrix} \tilde{z}_k^{(\ell)} (\tilde{z}_k^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \\ 0 \end{bmatrix} \right) \\ \vdots \\ (\tilde{z}_{n_\ell}^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \prod_{k=1}^{n_\ell-1} \left(I - \begin{bmatrix} \tilde{z}_k^{(\ell)} (\tilde{z}_k^{(\ell)})^T \begin{bmatrix} A^{(\ell)} & B^{(\ell)} \end{bmatrix} \\ 0 \end{bmatrix} \right) \end{bmatrix} \begin{bmatrix} 0 \\ I_{\bar{n}_\ell} \end{bmatrix}.$$

Finally we can introduce a specific combination of the first approach with a correction

$$\mathbb{A}_{cgss}^{(\ell+1)} = \mathbb{A}_{cgs}^{(\ell+1)} + (B^{(\ell)})^T Z^{(\ell)} \left((\tilde{Z}^{(\ell)})^T A^{(\ell)} \tilde{Z}^{(\ell)} - I \right) (Z^{(\ell)})^T B^{(\ell)}. \tag{10}$$

These approaches have different numerical properties and also different computation costs.

4 Conclusion

The goal of this contribution is to present new results for multilevel approximate inverse preconditioning, discuss differences among possible ways to form the approximate Schur complement and present some relevant theory.

Acknowledgement: This work has been supported by the grant of the Grant Agency of the Czech Republic 13-06684S.

References

[1] M. Benzi, C.D. Meyer, M. Tůma: *A sparse approximate inverse preconditioner for the conjugate gradient method.* SIAM J. Sci. Comput. **17**(5), 1996, pp. 1135–1149.

[2] J. Kopal, M. Rozložník, M. Tůma: *Approximate inverse preconditioners with adaptive dropping.* Advances in Engineering Software **84**(1), 2015, pp. 13–20.

[3] J. Kopal, M. Rozložník, M. Tůma: *Factorized approximate inverses with adaptive dropping.* SIAM Journal on Scientific Computing **38**(3), 2016, pp. A1807–A1820.

Efficient time integration in hypoplasticity models for soils

T. Koudelka, J. Kruis

Department of Mechanics, Faculty of Civil Engineering, Czech Technical University in Prague

1 Introduction

Radioactive waste generated mainly in nuclear power plants, but also in health care, has to be safely dispose. For this purpose, deep geological repositories are designed and special attention has to be devoted to the integrity of such structures in order to achieve their very high reliability. High reliability requires multilevel barriers. The repositories are designed in stable rock host environment several hundred meters below the surface. They resemble tunnels with chambers where the radioactive waste is stored in special containers. The tunnels and chambers are equipped with concrete lining which is further improved by layers of expansive clays, especially bentonite is used. The bentonite has very large swelling or shrinkage capacity which is influenced by water content.

Behaviour of soils, rocks and concrete is usually described by elasto-plastic material models with hardening or softening. Sometimes, the models are combined with damage mechanics which reduces the stiffness with the help of the damage parameter. Unfortunately, bentonites behave strongly nonlinearly and the elasto-plastic models lead to unsatisfactory results. Therefore, material models based on the theory of hypoplasticity were derived. They are based on the rate form of stress-strain relationship which requires time integration.

2 Material model and integration methods

Hypoplasticity can take into account nonlinear behaviour of soils, influence of barotropy and pyknotropy and the history of deformation. Hypoplastic constitutive equation relates the stress rate $\dot{\boldsymbol{\sigma}}$ and the strain rate $\dot{\boldsymbol{\varepsilon}}$ in the form

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{M}\dot{\boldsymbol{\varepsilon}}, \tag{1}$$

where $\boldsymbol{M}$ is the fourth-order constitutive tensor which depends on the stress tensor $\boldsymbol{\sigma}$, the strain increments $\Delta\boldsymbol{\varepsilon}$ and the state variables $\boldsymbol{v}$. The stress has to be obtained by integration of equation (1). Details about the constitutive tensor can be found in references [1] and [2].

Many integration methods (e.g. forward Euler method, Crank-Nicolson method, Runge-Kutta-Fehlberg methods with substepping) were tested in connection with hypoplasticity material models. The performance of the low-order methods is generally unsatisfactory and it is suggested to use the Runge-Kutta-Fehlberg (RKF) method of suitable order with substepping.

With respect to RKF method, the constitutive relationship (1) is rewritten into the form

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{\Psi}(\boldsymbol{\sigma}, \Delta\boldsymbol{\varepsilon}, \boldsymbol{v}) \tag{2}$$

where $\boldsymbol{\Psi}$ is a tensor function. The RKF method has the form

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{\sigma}_k + \Delta t_k \sum_{i=1}^{s} b_i\, \boldsymbol{k}_i \left(\boldsymbol{\sigma}_k, \Delta\boldsymbol{\varepsilon}(t_{n+1}), \Delta t_k\right), \tag{3}$$

where $\boldsymbol{k}_i\left(\boldsymbol{\sigma}_k, \Delta\boldsymbol{\varepsilon}(t_{n+1}), \Delta t_k\right)$ represents the function $\boldsymbol{\Psi}$ evaluated for the given strain increment of the actual time step $\Delta\boldsymbol{\varepsilon}(t_{n+1}) = \boldsymbol{\varepsilon}(t_{n+1}) - \boldsymbol{\varepsilon}(t_n)$ and attained stress levels at the prescribed points of time interval. In (3), dimensionless step length $\Delta t_k \in (0; 1]$ has been introduced in the form

$$\Delta t_k = \frac{t_{k+1} - t_k}{t_{n+1} - t_n}. \tag{4}$$

In Runge-Kutta-Fehlberg method, the step length Δt_k is constructed according to the difference between solution of two embedded Runge-Kutta algorithms of different order of accuracy

$$\bar{\boldsymbol{\sigma}}_{k+1} = \boldsymbol{\sigma}_k + \Delta t_k \sum_{i=1}^{s} \bar{b}_i\, \boldsymbol{k}_i\left(\boldsymbol{\sigma}_k, \Delta\boldsymbol{\varepsilon}(t_{n+1}), \Delta t_k\right), \tag{5}$$

$$\tilde{\boldsymbol{\sigma}}_{k+1} = \boldsymbol{\sigma}_k + \Delta t_k \sum_{i=1}^{s+1} \tilde{b}_i\, \boldsymbol{k}_i\left(\boldsymbol{\sigma}_k, \Delta\boldsymbol{\varepsilon}(t_{n+1}), \Delta t_k\right), \tag{6}$$

where $\boldsymbol{k}_i$ represents values of function $\boldsymbol{\Psi}$ evaluated in selected times and corresponding stress values.

Generally, the substep $k+1$ is accepted if the relative error measure R_{k+1} of two solutions $\bar{\boldsymbol{\sigma}}_{k+1}$ and $\tilde{\boldsymbol{\sigma}}_{k+1}$ is less than a given tolerance ϑ, i.e.

$${}^{\sigma}R_{k+1} = \frac{\|\tilde{\boldsymbol{\sigma}}_{k+1} - \bar{\boldsymbol{\sigma}}_{k+1}\|}{\|\tilde{\boldsymbol{\sigma}}_{k+1}\|} \leq \vartheta\ . \tag{7}$$

Several time integration RKF schemes have been implemented for the time integration of equation (2). Their description in the form of Butcher's tables is given in Tables (1) and (2). It should be noted that the algorithm RKF-23bs is the Bogacki-Shampine coefficient pairs and the advantage of the method is that it provides better estimate of error with the minimum cost because the $\boldsymbol{k}_4$ can be used as $\boldsymbol{k}_1$ in the next step - First Same As Last (FSAL) concept.

0			
1/2	1/2		
1	-1	2	
	1/6	2/3	1/6
	0	1	0

0				
1/2	1/2			
3/4	0	3/4		
	2/9	1/3	4/9	
	7/24	1/4	1/3	1/8

Table 1: Butcher table for RKF-23 (left) and RKF-23bs Bogacki-Shampine (right)

0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	-7200/2197	7296/2197			
1	439/216	-8	3680/513	-845/4104		
1/2	-8/27	2	-3544/2565	1859/4104	-11/40	
	16/135	0	6656/12825	28561/56430	-9/50	2/55
	25/216	0	1408/2565	2197/4101	-1/5	

Table 2: Butcher table for RKF-45

3 Numerical experiments

The implemented hypoplasticity model was tested on simple benchmark examples with axisymmetric specimen 1x1 m subjected to various loading path:

1. Triaxial drained test with constant confining pressure and gradually increasing axial load, initial stress -200 kPa, constant suction -1.9 MPa.
2. Triaxial drained test with constant volume increment, initial stress -200 kPa, constant suction -1.9 MPa.

Resulting diagrams from these examples can be seen in Figures (1) and (2), where evolution of axial stresses (a) or degree of saturation (b) versus evolution of axial strains can be observed.

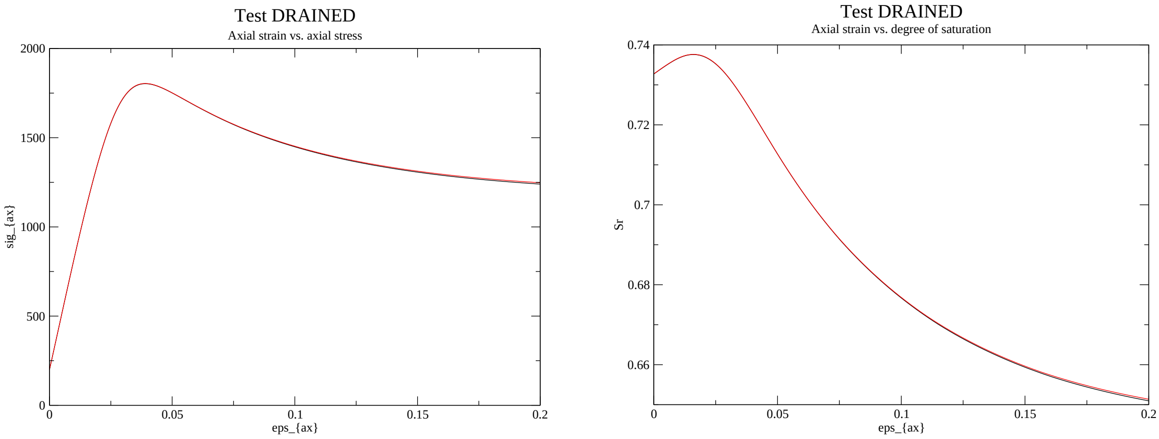


Figure 1: Triaxial drained test with constant confinig pressure - axial stress σ_{ax} vs. axial strain ε_{ax} (a), evolution of degree of saturation S_r according to axial strain ε_{ax} (b)

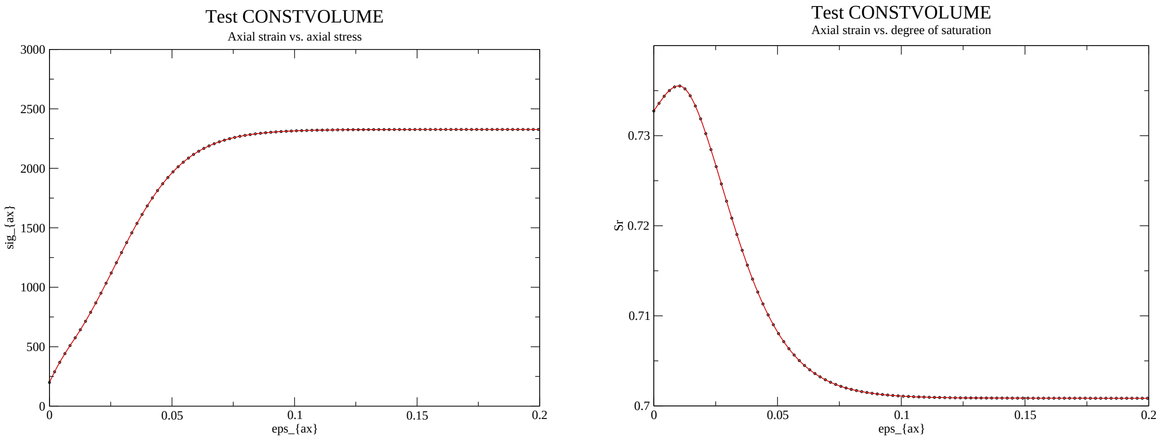


Figure 2: Triaxial drained test with constant volume increment - axial stress σ_{ax} vs. axial strain ε_{ax} (a), evolution of degree of saturation S_r according to axial strain ε_{ax} (b)

Comparison of integration schemes for various tolerances in benchmark example 1 (Triaxial drained test with constant confining pressure and gradually increasing axial load) is summarized in Table 3.

ϑ		RKF-23	RKF-23bs	RKF-45
10^{-7}	σ_{err}	1.57e-2	2.08e-3	0
	t [s]	34.6	21.1	84.7
10^{-6}	σ_{err}	1.85e-2	2.37e-3	1.89e-3
	t [s]	19.3	13.2	10.7
10^{-5}	σ_{err}	2.23e-2	5.78e-3	2.97e-3
	t [s]	11.95	8.81	8.41
10^{-4}	σ_{err}	3.67e-2	9.45e-3	6.07e-3
	t [s]	7.70	6.44	8.11
10^{-3}	σ_{err}	–	1.34e-2	8.92e-3
	t [s]	–	6.43	8.07

Table 3: Comparison of different integration schemes.

4 Conclusion

Suitable method for time integration in hypoplasticity models has to be selected with respect to highly nonlinear functions used in the fourth-order constitutive tensor. Numerical tests show that Runge-Kutta-Fehlberg method 23bs is the optimal choice.

Acknowledgement: Financial support for this work was provided by project number 15-05935S of Czech Science Foundation. The financial support is gratefully acknowledged.

References

[1] D. Mašín: *A hypoplastic constitutive model for clays.* International Journal for Numerical and Analytical Methods in Geomechanics, vol. 29, 2005, pp. 311–336.

[2] D. Mašín: *Double structure hydromechanical coupling formalism and a model for unsaturated expansive clays.* Engineering Geology, vol. 165, 2013, pp. 73–88.

Recent advances in PERMON

J. Kružík, M. Čermák, V. Hapla, D. Horák,
M. Pecha, R. Sojka, J. Tomčala, A. Vašatová

IT4Innovations National Supercomputing Center, VŠB - Technical University of Ostrava

1 Introduction

PERMON [5, 6] (Parallel, Efficient, Robust, Modular, Object–oriented, Numerical) is a scalable software toolbox for solution of quadratic programming (QP) problems.

QP problems arise in various disciplines including elasto-plasticity, contact problems with friction, shape optimization, vehicle routing problems, support vector machines, medical imaging, climate modeling, least-squares regression, data fitting, data mining, control systems and many others.

Our recent efforts have been mainly aimed at problems of mechanics. These problems may be described by partial differential equations (PDEs). To be solved with computers, they have to be discretized, e.g. with the popular Finite Element Method (FEM). We typically get large sparse linear systems of equations, but in case of constrained problems such as contact problems of mechanics, QP problems arise.

Large scale problems necessitates the use of parallelization to get a good time to solution and distribute the memory requirements. Domain decomposition methods (DDMs) solve the original problem by splitting it into smaller subdomain problems that are independent, allowing for a natural parallelization.

Finite Element Tearing and Interconnecting (FETI) methods [1, 2, 3, 4] form a successful subclass of DDMs. They are non-overlapping methods and combine iterative and direct solvers. FETI methods allow highly accurate computations scaling up to tens of thousands of cores.

Due to limitations of commercial packages, problems often have to be adapted to be solvable. This is an expensive process and results reflect less accurately physical phenomena. Moreover, it takes a long time before the most recent numerical methods needed for High Performance Computing (HPC) are implemented into such packages. These issues lead us to establish the PERMON toolbox.

2 PERMON toolbox

The PERMON toolbox makes use of theoretical results in QP algorithms, discretization techniques, and DDMs. It incorporates our own codes, and makes use of renowned open source libraries.

PERMON consists of several modules. PermonQP and PermonFLLOP form a solver layer. PermonCube and PermonMembrane are benchmark generators used for tests of the solver layer. PermonAIF provides a C interface. There are also several utility modules e.g., file interface.

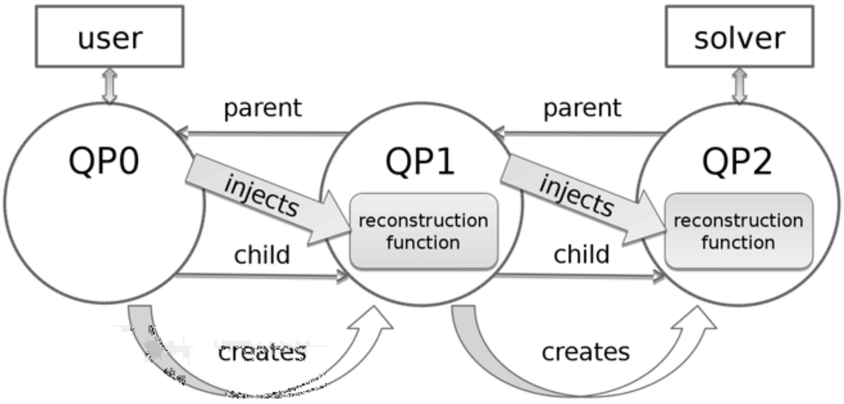


Figure 1: QP transformation chain: illustration of the PermonQP workflow

3 PERMON solver layer

The solver layer of PERMON depends on PETSc [7, 8, 9] and uses its time-proven coding style. It is formed by the PermonQP and optionally PermonFLLOP modules. PermonQP provides a base for solution of linear systems and QP problems. It includes data structures, transformations, algorithms, and supporting functions for QP. PermonFLLOP is an extension of the PermonQP package, adding support for DDM of the FETI type. This combination of DDM and QP algorithms is what makes PERMON unique.

3.1 PermonQP

PermonQP is a package providing a base for solution of quadratic programing (QP) problems. It includes data structures, transformations, algorithms, and supporting functions for QP. Its programming interface (API) is carefully designed to be easy-to-use, and at the same time efficient and suitable for High Performance Computing (HPC).

The solution process begins with QP problem specification. Then a series of QP transformation is applied. Each QP transformation creates a new, usually simplified, QP problem. Moreover, each new QP problem has a reconstruction function inserted by the QP transformation. This series of QP problems is represented as a bidirectional chain, see Figure 1. After a sufficiently easily solvable QP is obtained, an automatically or manually chosen solver is called. Then using the reconstruction functions we obtain the solution of the original QP problem.

QP problems can be either unconstrained or have a number of different constraints including e.g, equality, inequality, box, elliptical or conical constraints. PETSc KSP is usually used as a solver for unconstrained QP problems. It provides both iterative and direct solvers. For unconstrained or box constrained QP problems, PETSc TAO wrapper can be used. PermonQP also contains several other solvers e.g., a variant of augmented Lagrangian method called SMALXE or active-set methods like MPRGP [16, 17, 18].

PermonQP source code is freely available under BSD 2-Clause License.

3.2 PermonFLLOP

PermonFLLOP (shortly FLLOP, FETI Light Layer on Top of PETSc) is an extension of the PermonQP package, implementing the algebraic part of DDMs of the FETI type. The details of the FETI methods can be found in [1, 2, 3, 4].

Minimal input for FLLOP consists of l2g mapping, subdomains stiffness matrices and load vectors. These are generally obtained by volume-meshing the domain, that is then decomposed into subdomains using a partitioning software like METIS [10]. Then virtually any FEM implementation can be used to generate subdomains stiffness matrices and load vectors. L2g mapping assigns local degrees of freedom of each subdomain to the global degrees of freedom.

FLLOP is being prepared to be published under an open source license. Open source DDM codes are relatively rare. Let us mention the Multilevel BDDC solver library (BDDCML) by J. Šístek et al. [11], PETSc BDDC preconditioner implementation by S. Zampini [12], and codes by P. Jolivet et al. [13] on top of FreeFem++ [14, 15].

4 Recent advances

Throughout the PERMON toolbox there were several improvements in both performance and memory scalability. We implemented and thoroughly benchmarked Dirichlet and lumped FETI precondtitioners. New matrix type that condense a local part of matrix into a sequential matrix was developed and used to improve assembly process of FETI matrices. The support for multiple subdomains per core was improved and mainlined. Solution of FETI coarse problem was investigated. Ability to assemble FETI gluing matrix and kernel of stiffness matrix was added.

References

[1] C. Farhat, F.-X. Roux: *A method of finite element tearing and interconnecting and its parallel solution algorithm.* Int. J. Numer. Methods Eng. **32**, 1991, pp. 1205–1227.

[2] A. Klawonn, O.B. Widlund: *FETI and Neumann-Neumann Iterative Substructuring Methods: Connections and New Results.* Communications on Pure and Applied Mathematics **54**, 2001, pp. 57–90.

[3] Z. Dostál, D. Horák, R. Kučera: *Total FETI - an easier implementable variant of the FETI method for numerical solution of elliptic PDE.* Commun. Numer. Methods Eng. **22**(12), 2006, pp. 1155–1162.

[4] T. Kozubek et al.: *Total FETI domain decomposition method and its massively parallel implementation.* Advances in Engineering Software 6061, 2013, pp. 14–22. DOI 10.1016/j.advengsoft.2013.04.001

[5] PERMON web page, URL: `http://permon.it4i.cz/`

[6] V. Hapla et al.: *Solving Contact Mechanics Problems with PERMON.* Lecture Notes in Computer Science 9611, 2016, pp. 101–115. DOI 10.1007/978-3-319-40361-8_7

[7] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang: *PETSc users manual.* Tech. Rep. ANL-95/11 - Revision 3.4, Argonne National Laboratory, 2013.

[8] S. Balay, W.D. Gropp, L.C. McInnes, B.F. Smith: *Efficient management of parallelism in object oriented numerical software libraries.* Modern Software Tools in Scientific Computing, 1997, pp. 163–202.

[9] PETSc web page, URL: `http://www.mcs.anl.gov/petsc/`

[10] METIS web page, URL: `http://glaros.dtc.umn.edu/gkhome/metis/metis/overview`

[11] The Multilevel BDDC solver library (BDDCML) web page,
URL: `http://users.math.cas.cz/ sistek/software/bddcml.html`

[12] S. Zampini: *PCBDDC: A Class of Robust Dual-Primal Methods in PETSc.* SIAM Journal on Scientific Computing **38**(5), 2016, pp. 282–306. DOI 10.1137/15M1025785

[13] P. Jolivet et al.: *High performance domain decomposition methods on massively parallel architectures with freefem++.* Journal of Numerical Mathematics **20**(3-4), 2012, pp. 287–302. DOI 10.1515/jnum-2012-0015

[14] FreeFem++ web page, URL: `http://www.freefem.org/ff++/`

[15] F. Hecht: *New Depelopments in Freefem++.* J. Numer. Math. **20**(3-4), 2012, pp. 251–265. DOI 10.1515/jnum-2012-0013

[16] Z. Dostál *Optimal Quadratic Programming Algorithms, with Applications to Variational Inequalities.* SOIA 23. Springer US, New York, 2009.

[17] Z. Dostál, T. Kozubek, V. Vondrák, A. Markopoulos, T. Brzobohatý. *Scalable TFETI algorithm for the solution of multibody contact problems of elasticity.* International Journal for Numerical Methods in Engineering **82**(11), 2010, pp. 1384–1405. DOI 10.1002/nme.2807

[18] Z. Dostál, L. Pospíšil: *Optimal iterative QP and QPQC algorithms.* Annals of Operations Research, 2013, pp. 1–14. DOI 10.1007/s10479-013-1479-0

Circumradius condition: breakthrough or not ?

V. Kučera

Faculty of Mathematics and Physics, Charles University in Prague

1 Introduction

The finite element method (FEM) is perhaps the most popular numerical method for the solution of partial differential equations. Much work has been devoted to theoretical and practical aspects of the FEM. Possibly the most basic question is, when does the sequence of approximate solutions u_h converge to the exact solution u. Surprisingly this question is still far from being answered even for the simplest problems in 2D. This can be seen from the interest that the recently derived *circumradius condition* raised in the finite element community. This condition derived in [4], [5] is a generalization of the well known *maximum* and *minimum angle* conditions, which are sufficient for $O(h)$ convergence of piecewise linear finite elements. Here we comment on the circumradius condition and its relation to previous work and the maximum angle condition.

2 Angle conditions

We consider Poisson's problem in a 2D Lipschitz domain Ω:

$$-\Delta u = f \quad \text{in } \Omega \tag{1}$$

with (e.g.) homogenous Dirichlet boundary conditions. We consider a system of triangulations $\{\mathcal{T}_h\}_{h\in(0,h_0)}$ of $\Omega \subset \mathbb{R}^2$, which defines the piecewise linear finite element space $V_h = \{v_h \in C(\overline{\Omega}); v_h|_K \in P^1(K) \text{ for all } K \in \mathcal{T}_h\}$, where $P^1(K)$ is the space of linear functions on the triangular element $K \in \mathcal{T}_h$. The convergence is then usually measured with respect to the parameters $h_K = \text{diam}K$ and $h = \max_{K\in\mathcal{T}_h} h_K$.

For (1), estimates on the finite element error are usually obtained via Céa's lemma and estimates for Lagrange interpolation on triangles. For $u \in C(\overline{K})$, let $\Pi_K u \in P^1(K)$ be the Lagrange interpolation defined by the vertices of K. From this, we can construct the global interpolant $\Pi_h u \in V_h$ such that $(\Pi_h u)|_K = \Pi_K u$ for all $K \in \mathcal{T}_h$. The error of the finite element method is then estimated by the error of the interpolant.

The first condition for $O(h)$ convergence of $\Pi_h u$ to u in the $H^1(\Omega)$-seminorm is the so-called *minimum angle condition* derived independently in [10], [11].

Lemma 1. *Let $\gamma_0 > 0$ and let the minimal angle of $K \in \mathcal{T}_h$ satisfy $\gamma_K \geq \gamma_0$, then there exists a constant $C = C(\gamma_0)$ independent of u, h_K such that*

$$|u - \Pi_K u|_{1,2,K} \leq C h_K |u|_{2,2,K}. \tag{2}$$

If we assume that $\gamma_K \geq \gamma_0 > 0$ for all $K \in \mathcal{T}_h$ and all $h \in (0, h_0)$, we then obtain

$$|u - \Pi_h u|_{1,2,\Omega} \leq C h |u|_{2,2,\Omega}, \tag{3}$$

which is an $O(h)$ error estimate for the FEM in $H^1(\Omega)$.

Later, a generalization of Lemma 1 was proved independently by [1], [2] and [3] leading to the *maximum angle condition*:

Lemma 2. *Let $\alpha_0 < \pi$ and let the maximal angle of $K \in \mathcal{T}_h$ satisfy $\alpha_K \le \alpha_0$, then for all $p \in [1, \infty]$ there exists a constant $C_p(\alpha_0)$ independent of u, h_K such that*

$$|u - \Pi_K u|_{1,p,K} \le C_p(\alpha_0) h_K |u|_{2,p,K}. \tag{4}$$

Again, assuming that $\alpha_K \le \alpha_0 < \pi$ for all $K \in \mathcal{T}_h$ and all $h \in (0, h_0)$, we obtain a $W^{1,p}(\Omega)$ version of (3).

3 Circumradius condition

Recently, a generalization of Lemmas 1 and 2 was given in [4] and [5].

Lemma 3 (Circumradius estimate)**.** *Let $R_K \le 1$ be the circumradius of K. Then for all $p \in [1, \infty]$ there exists a constant C_p independent of u, K such that*

$$|u - \Pi_K u|_{1,p,K} \le C_p R_K |u|_{2,p,K}. \tag{5}$$

Assuming the *circumradius condition*

$$\lim_{h \to 0} \max_{K \in \mathcal{T}_h} R_K \to 0, \tag{6}$$

we obtain convergence (not $O(h)$ convergence) of the finite element method similarly as in (3). We shall refer to Lemma 3 as the *circumradius estimate*, although in [5] both (5) and (6) are ambiguously called the circumradius condition.

The circumradius condition raised a lot of interest in the finite element community, as it was the general opinion that such a well known and studied subject as the error of linear Lagrange interpolation on a triangle was well established and no breakthrough was expected. The matter of originality and correctness was also unclear due to the fact that the original paper [4] relied heavily on numerical computations and was written in Japanese and the rigorous proof in the subsequent paper [5] is rather technical. To clarify the matter, the author discussed these issues in the paper [6].

3.1 The $O(h)$-case

The first observation is the following. Since α_K is the largest angle in K, its opposing side has length h_K. By the law of sines,

$$2R_K = \frac{h_K}{\sin \alpha_K}. \tag{7}$$

If we substitute this expression into (5), we get an $O(h)$ estimate if and only if the denominator $\sin \alpha_K$ is uniformly bounded away from zero for all $K \in \mathcal{T}_h$, which is exactly the maximum angle condition. Therefore, as far as $O(h)$ convergence is concerned, the circumradius and maximum angle conditions are equivalent.

3.2 Relation to the maximum angle condition

The observation in Section 3.1 indicates that there might be a deeper connection of the circumradius and maximum angle conditions. This is indeed the case, as demonstrated in [8]. Here we only present the idea without going into technical details.

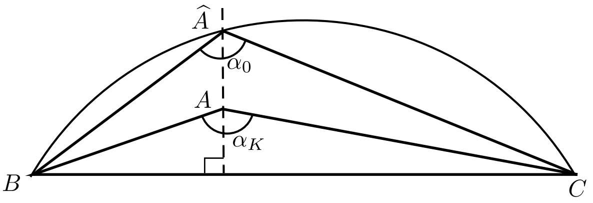


Figure 1: Dilation of a triangle $K \in \mathcal{T}_h$.

The basic idea is to take the triangle K, which can be arbitrary, and scale it appropriately so that the scaled triangle $\tilde{K}$ satisfies the maximum angle condition with a given angle α_0. Hence Lemma 2 can be applied on $\tilde{K}$ to obtain an $O(h_K)$ estimate of the interpolation error. By transforming the $O(h_K)$ error estimate on $\tilde{K}$ back to the original element K, one obtains an $O(R_K)$ estimate for the interpolation error on K.

The scaling (dilation) of K is defined as follows. Let K have vertices A, B, C, cf. Figure 1. Let the maximal angle α_K be at A, hence the side BC has length h_K. We choose $\alpha_0 < \pi$ as in the maximum angle condition. If $\alpha_K > \alpha_0$, we find the unique triangle $\widehat{K}$ with vertices $\widehat{A}, B, C$ such that $\alpha_{\widehat{K}} := \angle B\widehat{A}C = \alpha_0$ and $\widehat{A}, A$ have the same foot H of their altitudes to BC. This is possible, since the set of all vertices $\widehat{A}$ such that $\angle B\widehat{A}C = \alpha_0$ is a circular arc, cf. Figure 1. In other words, we have dilated the triangle K in the direction perpendicular to BC, such that the dilated triangle $\widehat{K}$ has maximum angle α_0.

The dilation factor can be easily evaluated and moreover, since the dilation is only in one direction, Sobolev (semi-)norms are transformed trivially from K to $\widehat{K}$ and back using powers of this factor. If one transforms the maximal angle estimate (4) using these relations, one obtains the result of Lemma 3 without the technical assumption $R_K \leq 1$. Moreover, once all the necessary quantities are evaluated, it is quite simple to apply the same technique to obtain similar estimates for higher order Lagrange interpolation, which is otherwise very technical, as in [6].

3.3 Prior work

The case of $p = 2$ in Lemma 1 was proved already by A. Rand in his Ph.D. thesis [9], however this result was not published in a journal. This case was then independently shown by K. Kobayashi in [4] where it is claimed that the constant in (5) can be taken as $C_2 = 1$. However, the proof relies heavily on numerical computations and is therefore hard to verify. Finally, the case of general p was proven by K. Kobayashi and T. Tsuchiya in [5] using the technique of [1].

As it turns out, Lemma 3 is already essentially proved in [7], although the final result is never formulated since the paper only deals with $O(h)$ convergence and the maximum angle condition. However, estimates using R_K are used throughout the paper and Lemma 5 could have been obtained in the following way. An intermediate step of (2.22) in [7] states

$$|\det B_K| = \frac{f_K g_K h_K}{2R_K}, \tag{8}$$

where $\det B_K$ is the Jacobian of the mapping from a reference triangle to K and f_K, g_K, h_K are the lengths of sides of K. An intermediate step of the chain of inequalities following (2.22) in [7] is

$$|v - \Pi_K v|_{1,p,K} \leq 32\hat{C}|\det B_K|^{-1} f_K g_K h_K |v|_{2,p,K}. \tag{9}$$

Substituting (8) into (9) immediately gives (5). The reason this is not done in the paper is that (8) is further estimated using the maximum angle condition, thus eliminating R_K from the final estimate in order to obtain $O(h)$ convergence.

Acknowledgement: This work was supported by the J. William Fulbright Commission in the Czech Republic.

References

[1] I. Babuška, A.K. Aziz: *On the angle condition in the finite element method.* SIAM J. Numer. Anal. **13** (1976), pp. 214–226.

[2] R.E. Barnhill, J.A. Gregory: *Sard kernel theorems on triangular domains with applications to finite element error bounds.* Numer. Math. **25** (1976), pp. 215–229.

[3] P. Jamet: *Estimations d'erreur pour des éléments finis droits presque dégénérés.* Rev. Franc. Automat. Inform. Rech. Operat., R 10 (1976), pp. 43–60.

[4] K. Kobayashi: *On the interpolation constants over triangular elements.* RIMS Kokyuroku, 1733 (2011), pp. 58–77.

[5] K. Kobayashi, T. Tsuchiya: *A Babuška-Aziz type proof of the circumradius condition.* Japan J. Ind. Appl. Math. **31** (2014), pp. 193–210.

[6] K. Kobayashi, T. Tsuchiya: *A priori error estimates for Lagrange interpolation on triangles.* Appl. Math. **60** (2015), pp. 485–499.

[7] M. Křížek: *On semiregular families of triangulations and linear interpolation.* Appl. Math., Praha **36** (1991), pp. 223–232.

[8] V. Kučera: *Several notes on the circumradius condition.* Appl. Math. **61** (2016), pp. 287–298.

[9] A. Rand: *Delaunay refinement algorithms for numerical methods.* Ph.D. thesis, Carnegie Mellon University, 2009, URL: `www.math.cmu.edu/~arand/papers/arand_thesis.pdf`.

[10] A. Ženíšek: *The convergence of the finite element method for boundary value problems of a system of elliptic equations.* Appl. Mat. **14** (1969), pp. 355–377.

[11] M. Zlámal: *On the finite element method.* Numer. Math. **12** (1968), pp. 394–409.

The use of sequential quadratic programming for solving reachability problems

J. Kuřátko

Institute of Computer Science of the CAS, Prague
Faculty of Mathematics and Physics, Charles University in Prague

1 Introduction

Complex technical systems that are modelled by a system of ordinary differential equations may feature states that are considered unsafe. Any solution that originates from an initial state and reaches an unsafe state represents a flaw, a dangerous scenario, which is to be avoided in the design of the system. We address computing these solutions and our approach is based on solving an equality constrained nonlinear programming problem [4].

To this end we use the Sequential Quadratic Programming method (SQP) [5]. The underlying saddle-point matrix is sparse and may be solved either iteratively or directly [2]. In the poster we show that when one applies LDL^T factorization [3, Alg. 4. 1. 2] on the resulting saddle-point matrix then the lower-triangular factor is again sparse and well-structured.

2 Problem formulation

The reachability problem we try to solve is described in [4] and we formulate it in this section. Consider a dynamical system whose dynamics is modelled by a system of ordinary differential equations of the following form

$$\dot{x} = f(x(t)), \quad x(0) = x_0\,, \tag{1}$$

where $x : \mathbb{R} \to \mathbb{R}^n$ is an unknown function of t, and the right hand side $f : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable. Denote the *flow* by $\Phi : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ for which when an initial state $x_0 \in \mathbb{R}^n$ is fixed then $\Phi(t) : \mathbb{R}^n \to \mathbb{R}^n$ represents the solution $x(t)$ of (1).

The problem we try to solve can be formulated in the following way [4]: *Consider a dynamical system* (1) *and let* Init *and* Unsafe *be two sets of states in* $\mathbb{R}^n$. *Find a solution of* (1) *such that* $x_0 \in \mathrm{Init}$ *and* $\Phi(t, x_0) \in \mathrm{Unsafe}$ *for some* $t \geq 0$.

3 Solution approach

Our approach to finding solutions of dynamical system (1) from Init to Unsafe is formulated as an equality constrained nonlinear programming problem

$$\min F(\chi) \qquad \text{subject to} \quad c(\chi) = 0\,. \tag{2}$$

Here the unknown vector χ has the form

$$\chi = \left[x_0^1, t_1, x_0^2, t_2, \ldots, x_0^N, t_N\right]^T \in \mathbb{R}^{N(n+1)}\ . \tag{3}$$

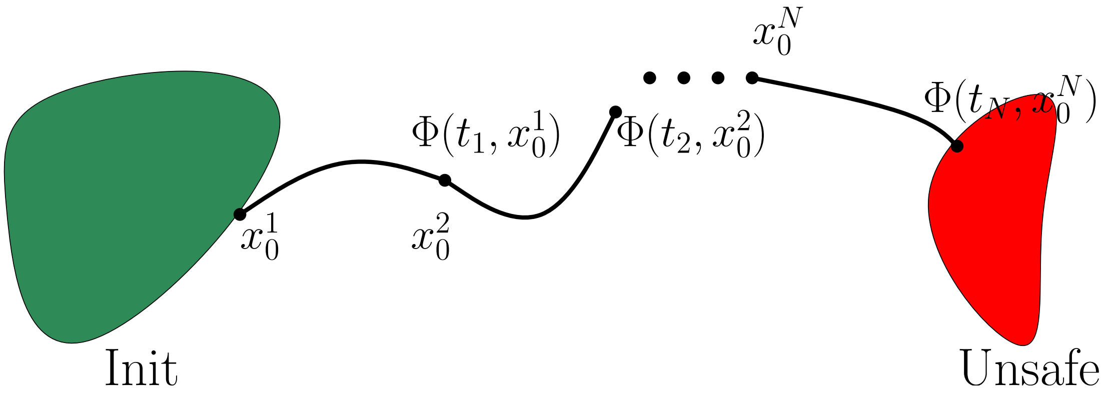


Figure 1: N connected solution segments

We use the idea of multiple shooting [1], that is, we put together shorter solution segments of the dynamical system (1) in order to compute one solution. Parameters (3) are initial states $x_0^i \in \mathbb{R}^n$, $1 \le i \le N$, of solution segments and t_i, $1 \le i \le N$, are their lengths. Here n denotes the state space dimension and N the number of solution segments. One such solution is illustrated in Fig. 1.

This equality constrained nonlinear problem (2) with the choice of the objective function (4) and the vector of constraints (5) is described and analyzed in [4]. The objective function and the vector of constraints we consider are of the form

$$F(\chi) = \sum_{i=1}^{N} t_i^2 \,, \tag{4}$$

$$c(\chi) = [g_{\mathrm{I}}, g_1, \dots, g_{N-1}, g_{\mathrm{U}}] \in \mathbb{R}^{(N-1)n+2} \,, \tag{5}$$

where $g_{\mathrm{I}} = \frac{1}{2}\left(\|x_0^1 - c_{\mathrm{I}}\|_{E_{\mathrm{I}}}^2 - 1\right) \in \mathbb{R}$, $g_{\mathrm{U}} = \frac{1}{2}\left(\|\Phi(t_N, x_0^N) - c_{\mathrm{U}}\|_{E_{\mathrm{U}}}^2 - 1\right) \in \mathbb{R}$, and constraints $g_i = x_0^{i+1} - \Phi(t_i, x_0^i) \in \mathbb{R}^n$ for $1 \le i \le N-1$. Norms $\|\cdot\|_{E_{\mathrm{I}}}$ and $\|\cdot\|_{E_{\mathrm{U}}}$ are energy norms. Since the sets Init, resp. Unsafe are ellipsoids their shape and size are given by symmetric positive definite matrices E_{I}, resp. E_{U}. Vectors $c_{\mathrm{I}} \in \mathbb{R}^n$ and $c_{\mathrm{U}} \in \mathbb{R}^n$ give centres of these ellipsoids.

4 Saddle-point matrix

We use the SQP method to solve the minimization problem (2). The underlying saddle-point matrix is of the form

$$K = \begin{bmatrix} H & B \\ B^T & 0 \end{bmatrix} ,$$

where $H \in \mathbb{R}^{N(n+1)\times N(n+1)}$ is a block diagonal symmetric positive definite matrix and $B \in \mathbb{R}^{N(n+1)\times(N-1)n+2}$ is a banded matrix with full column rank [4].

We expand on the results in [4] with application of direct solvers and making use of sparsity of the resulting saddle point matrix. Our main contribution thus lies in the computation of the LDL^T factorization of the saddle-point matrix K so that

$$\begin{aligned} \begin{bmatrix} H & B \\ B^T & 0 \end{bmatrix} &= LDL^T \,, \\ &= \begin{bmatrix} L_H & 0 \\ B^T L_H^{-T} D_H^{-1} & L_S \end{bmatrix} \begin{bmatrix} D_H & 0 \\ 0 & -D_S \end{bmatrix} \begin{bmatrix} L_H^T & D_H^{-1} L_H^{-1} B \\ 0 & L_S^T \end{bmatrix} , \end{aligned}$$

and in describing the exact structure of sparse matrices L_H, $B^T L_H^{-T} D_H^{-1}$ and L_S. Here we have $B^T H^{-1} B = L_S D_S L_S^T$ that is the LDL^T factorization of the Schur complement, and we put $H = L_H D_H L_H^T$. Note that the lower-triangular factors L_H and L_S have ones on the diagonal. The structures of the saddle-point matrix K and its lower-triangular factor L are illustrated in Fig. 2.

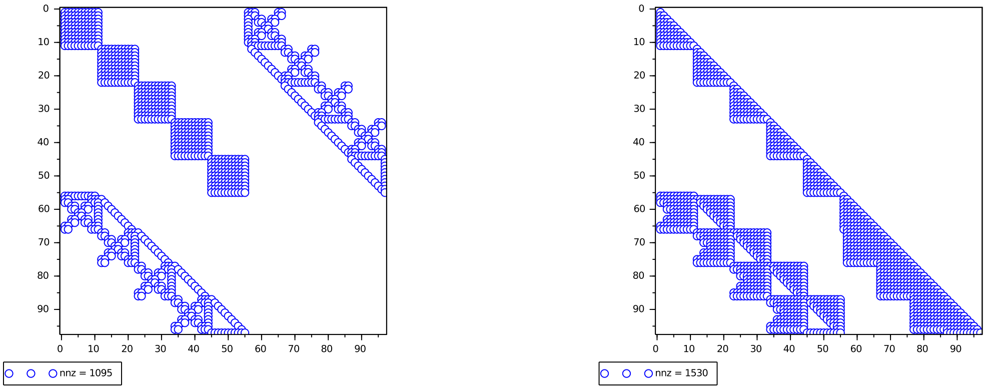


Figure 2: *On the left hand side*: the structure of matrix K with the block-diagonal matrix H and the banded matrix B; *On the right hand side*: the lower-triangular factor L of the saddle-point matrix K

Acknowledgement: This work was supported by the Czech Science Foundation (GACR) grant number 15-14484S with institutional support RVO:67985807.

References

[1] U.M. Ascher, L.R. Petzold: *Computer Methods for Ordinary Differential Equations and Differential-algebraic Equations*, volume 61. SIAM, 1998.

[2] M. Benzi, G.H. Golub, J. Liesen: *Numerical solution of saddle point problems.* Acta Numerica 14:1–137, 5 2005.

[3] G.H. Golub, C.F. Van Loan: *Matrix Computations (3rd Ed.).* Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[4] J. Kuřátko: *Solving reachability problems with singular and indefinite hessian by sequential quadratic programming.* Submitted and available at https://arxiv.org/abs/1611.01051, 2016.

[5] J. Nocedal, S.J. Wright: *Numerical Optimization.* Springer, 2nd edition edition, 2006.

Note on boundary conditions on artificial boundaries for Navier–Stokes fluid's flow

M. Lanzendörfer, *J. Hron*

Mathematical Institute, Faculty of Mathematics and Physics, Charles University in Prague

It seems natural to our minds to separate the "inner" and "outer", despite of them being inseparable in essence. In the (computational) fluid dynamics it is often practical to consider the flow problems in a bounded domain. In a majority of cases, certain parts of the domain's boundary are artificial: not related to any natural physical interface. The choice of boundary conditions to be imposed on artificial boundaries is then a modelling issue, where neither the physics alone nor the mathematical analysis alone give conclusive recommendations.

The talk will focus on the inflow and outflow boundary conditions for steady Navier–Stokes equations

$$\left.\begin{array}{rcl} \operatorname{div}\boldsymbol{v} &=& 0 \\ \operatorname{div}(\boldsymbol{v}\otimes\boldsymbol{v}) - \operatorname{div}\boldsymbol{T} &=& 0 \end{array}\right\} \text{ in } \Omega, \qquad \boldsymbol{T} = -p\boldsymbol{I} + \nu\left(\nabla\boldsymbol{v} + (\nabla\boldsymbol{v})^T\right).$$

Namely, we will be interested in the case that the flow rate is not known a priori, i.e. in the case that the (rather standard) choice

$$\mathbf{v} = \mathbf{v}_{\text{in}} \quad \text{on some } \Gamma_{\text{in}} \subset \partial\Omega$$

is not practical.

To pick one part of the issue, we will discuss a particular set of examples demonstrating the non-uniqueness and the (lack of) stability of steady solutions for the "do-nothing" b.c.'s and the b.c.'s based on prescribed constant traction

$$p\boldsymbol{n} - (\nabla\boldsymbol{v})\boldsymbol{n} = P_0\boldsymbol{n} \qquad \text{or} \qquad -\boldsymbol{T}\boldsymbol{n} = P_0\boldsymbol{n} \qquad \text{on some } \Gamma \subset \partial\Omega, \text{with } P_0 \in \mathbb{R}.$$

For instance, if such conditions are prescribed on two opposite parts of the domain's boundary, we will observe two different steady solutions. (For $P_0 = 0$ this gives us, among others, an exemplary non-trivial solution to steady Navier–Stokes problem with trivial data.) Such multiple solutions can be easily obtained in standard numerical simulations. We will also show that one of them is in certain sense asymptotically stable while the other one is unstable. In more complex geometries more steady solutions can be found.

References

[1] J.G. Heywood, R. Rannacher, S. Turek: *Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations*, Int. J. Numer. Meth. Fluids 22(5), 1996, pp. 325–352.

[2] M. Lanzendörfer, J. Málek, K.R. Rajagopal: *Numerical simulations of an incompressible piezoviscous fluid flowing in a plane slider bearing*, submitted for publication.

[3] M. Lanzendörfer, J. Hron: *On multiple solutions to the steady flow of incompressible fluids subject to constant traction or do-nothing boundary conditions on artificial boundaries*, in preparation.

Bifurcations in contact problems with Coulomb friction

T. Ligurský, Y. Renard

Institute of Geonics of the CAS, Ostrava
INSA, Lyon

1 Introduction

Let us consider static deformation of an elastic body whose reference configuration is the closure of a bounded domain $\Omega \subset \mathbb{R}^2$. Let the boundary $\partial\Omega$ be Lipschitz-continuous and split into three disjoint relatively open subsets Γ_D, Γ_N and Γ_c (see Figure 1). The body is fixed along Γ_D whereas an applied surface force of density $\boldsymbol{h}$ is prescribed on Γ_N. A flat rigid foundation supports the body along Γ_c, and the contact is modelled by unilateral conditions and the Coulomb friction law. We suppose that there is no initial gap between the body and the foundation. We consider that the surface force at a point $\boldsymbol{x} \in \Gamma_N$ depends on a real parameter γ and it may depend on $\boldsymbol{x}$ either, that is, $\boldsymbol{h} = \boldsymbol{h}(\gamma, \boldsymbol{x})$ in our model, γ being a loading parameter. We seek equilibrium states of the body for the values of γ from an interval I of our interest.

Discretisation of this problem is done by applying a conforming Lagrange finite-element method to a mixed variational formulation of the problem with Lagrange multipliers enforcing the Dirichlet and the contact boundary conditions. The contact conditions are approximated nodally and written in terms of projections. This gives the following discrete problem:

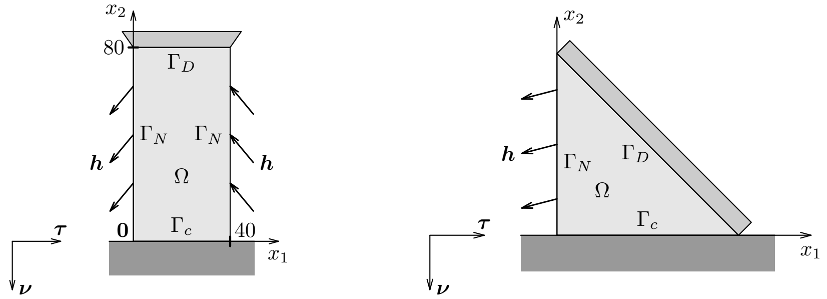
$$\left.\begin{array}{l} \textit{Find } \boldsymbol{y} := (\gamma, \boldsymbol{u}, \boldsymbol{\lambda}_D, \boldsymbol{\lambda}_\nu, \boldsymbol{\lambda}_\tau) \in I \times \mathbb{R}^{2(n_\Omega + n_D + n_c)} \textit{ such that} \\ \boldsymbol{H}(\boldsymbol{y}) = \boldsymbol{0}, \end{array}\right\}$$

where $\boldsymbol{H} \colon I \times \mathbb{R}^{2(n_\Omega + n_D + n_c)} \to \mathbb{R}^{2(n_\Omega + n_D + n_c)}$ is defined by

$$\boldsymbol{H}(\boldsymbol{y}) := \begin{pmatrix} \boldsymbol{A}(\boldsymbol{u}) - \boldsymbol{L}(\gamma) - \boldsymbol{B}_D^\top \boldsymbol{\lambda}_D - \boldsymbol{B}_\nu^\top \boldsymbol{\lambda}_\nu - \boldsymbol{B}_\tau^\top \boldsymbol{\lambda}_\tau \\ \boldsymbol{B}_D \boldsymbol{u} \\ -\frac{1}{r}\big(\lambda_{\nu,j} - ((\lambda_\nu - r\boldsymbol{B}_\nu \boldsymbol{u})_j)_-\big), \quad j = 1, \ldots, n_c \\ -\frac{1}{r}\big(\lambda_{\tau,j} - P_{[\mathscr{F}((\lambda_\nu - r\boldsymbol{B}_\nu\boldsymbol{u})_j)_-, -\mathscr{F}((\lambda_\nu - r\boldsymbol{B}_\nu\boldsymbol{u})_j)_-]}((\lambda_\tau - r\boldsymbol{B}_\tau \boldsymbol{u})_j)\big), \quad j = 1, \ldots, n_c \end{pmatrix}.$$

Here, $\boldsymbol{u} \in \mathbb{R}^{2n_\Omega}$ is the vector of nodal displacements, $\boldsymbol{\lambda}_D \in \mathbb{R}^{2n_D}$ is the Lagrange multiplier corresponding to the Dirichlet condition, and $\boldsymbol{\lambda}_\nu, \boldsymbol{\lambda}_\tau \in \mathbb{R}^{n_c}$ are the normal and tangential Lagrange multipliers on the contact zone, respectively. Further, $\boldsymbol{A}(\boldsymbol{u})$ and $\boldsymbol{L}(\gamma)$ are the vectors of internal elastic forces and external applied ones, respectively, and $\boldsymbol{B}_D$ is the kinematic transformation matrix linking $\boldsymbol{u}$ with the Lagrange multiplier $\boldsymbol{\lambda}_D$. The matrices $\boldsymbol{B}_\nu$ and $\boldsymbol{B}_\tau$ associate the vector of nodal displacements with the vectors of contact nodal displacements in the directions of the unit inward normal $\boldsymbol{\nu}$ and the unit tangent $\boldsymbol{\tau}$ to the foundation, respectively. The notation $(.)_-$ means the negative part (or equivalently, the projection onto $(-\infty, 0]$), and $P_{[a,b]}(.)$ stands for the projection onto an interval $[a, b]$, $a \leq b$. The friction coefficient is represented by a non-negative constant $\mathscr{F}$ here and $r > 0$ is a fixed augmentation parameter.

Following [2], we shall present the behaviour of bifurcations for different finite-element meshes in two model problems from [1] in our talk.



(a) Rectangular body.

(b) Triangular body.

Figure 1: Geometries of the model problems.

2 Problem with a rectangular body

Firstly, consider deformation of a rectangular block that is 40 mm wide and 80 mm high from Figure 1(a). A plane-strain approximation of the nonlinear Ciarlet-Geymonat constitutive law is used for the material of the block. Namely, the first Piola-Kirchhoff stress tensor $\hat{\boldsymbol{\sigma}}$ is given by

$$\hat{\boldsymbol{\sigma}}(\boldsymbol{x}, \boldsymbol{F}) = (\tilde{\boldsymbol{\sigma}}(\tilde{\boldsymbol{F}}))_{1\leq i,j\leq 2}, \quad \tilde{\boldsymbol{F}} = \begin{pmatrix} \boldsymbol{F} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix}, \quad \boldsymbol{F} \in \mathbb{R}^{2\times 2},$$

$$\tilde{\boldsymbol{\sigma}}(\tilde{\boldsymbol{F}}) = 2b\big(\mathrm{tr}(\tilde{\mathrm{F}}^\top\tilde{\mathrm{F}})\big)\mathrm{I} + 2(\mathrm{a} - \mathrm{b}\tilde{\mathrm{F}}\tilde{\mathrm{F}}^\top)\tilde{\mathrm{F}} + \big(2\mathrm{c}\det(\tilde{\mathrm{F}}^\top\tilde{\mathrm{F}}) - \mathrm{d}\big)\tilde{\mathrm{F}}^{-\top}, \quad \tilde{\mathrm{F}} \in \mathbb{R}^{3\times 3},$$

where

$$\lambda = 4000 \text{ N/mm}^2, \quad \mu = 120 \text{ N/mm}^2, \quad a = 30 \text{ N/mm}^2$$

and

$$b = \frac{\mu}{2} - a, \quad c = \frac{\lambda}{4} - \frac{\mu}{2} + a, \quad d = \frac{\lambda}{2} + \mu.$$

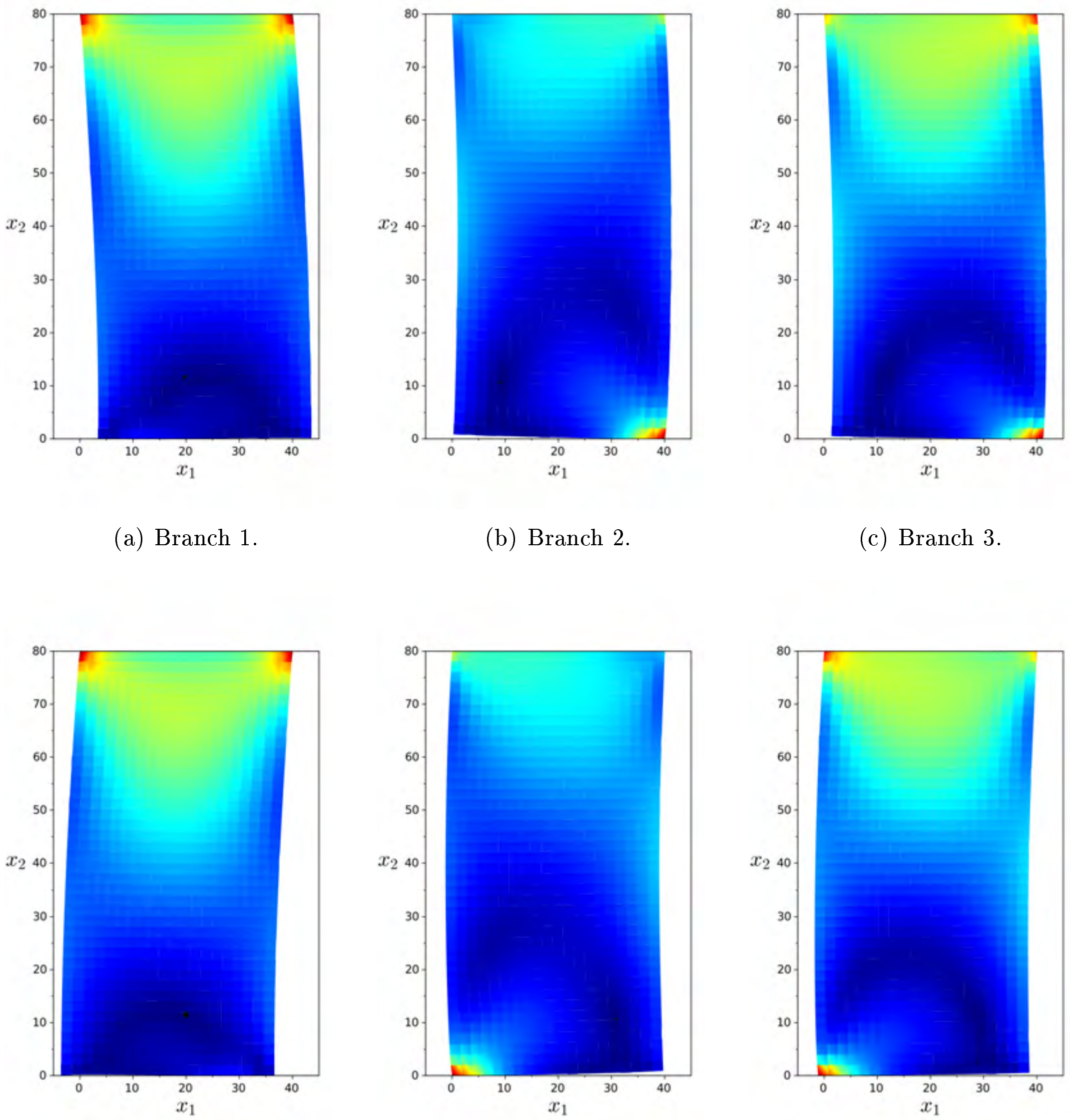
We have prescribed $\boldsymbol{h}(\gamma, \boldsymbol{x}) = \gamma(-2, 0.12(x_1 - 20))$ (in N/mm^2) on both lateral sides of the block, $r = 10$ and $\mathscr{F} = 1$. Discretisation is done by approximating the displacement and the Lagrange multiplier for the Dirichlet condition with continuous piecewise bilinear functions on rectangular meshes.

As described in [1], there are six solution branches intersecting at $\gamma = 0$ for a uniform mesh with 800 squares and 21 contact nodes, which are illustrated in Figure 2. Branches 1, 2 and 3 correspond to forcing the block to the right with no contact, contact-stick and contact-slip to the right of the lower right vertex of the block. Branches 4, 5 and 6, which are symmetric with respect to the axis $x_1 = 20$ of the block, correspond to forcing the block to the left with no contact, contact-stick and contact-slip to the left of the lower left vertex.

To explore the bifurcation phenomenon in the contact problem thoroughly, we have performed our method of piecewise-smooth numerical branching [2] for discretisations on different uniform meshes, namely with 200, 800, 3200 and 12800 squares. As we shall show in our talk, we have found six branches distinctly separated for all meshes. The branches seem to be stable and to converge with the meshes.

3 Problem with a triangular body

Secondly, let us consider deformation of an isosceles triangle with legs 1 m long shown in Figure 1(b). Let us restrict ourselves to the small-deformation framework with Hooke's law with



(a) Branch 1.

(b) Branch 2.

(c) Branch 3.

(d) Branch 4.

(e) Branch 5.

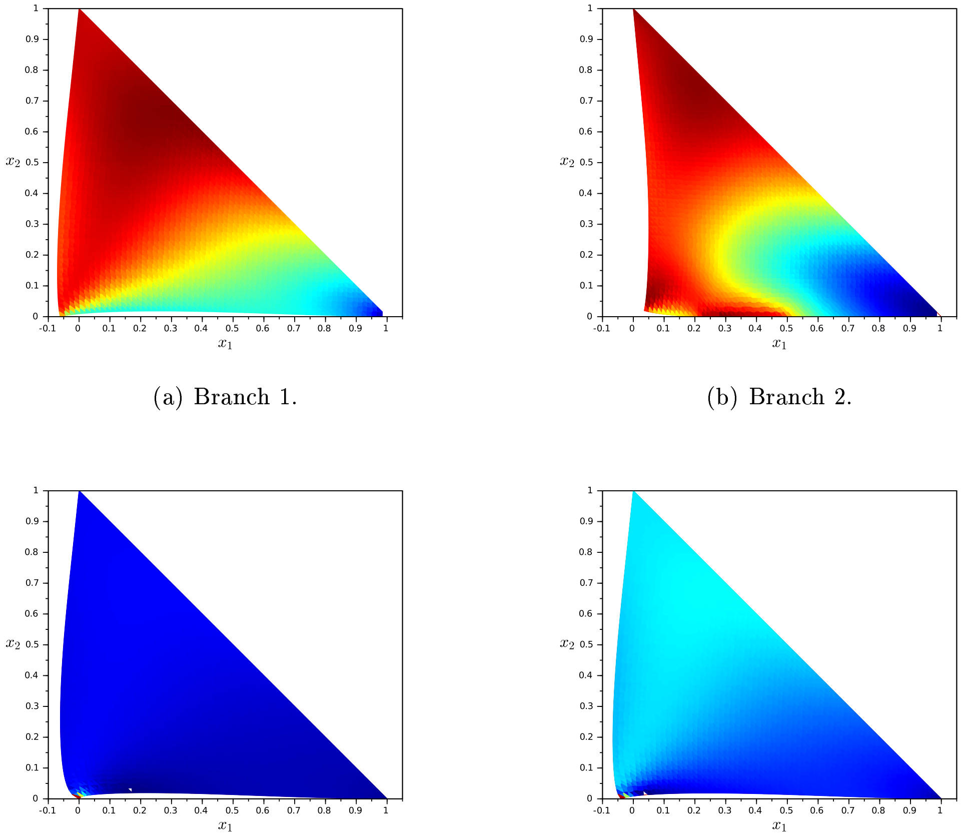
(f) Branch 6.

Figure 2: Deformed bodies corresponding to the solutions with $|\gamma| = 1$: (a), (b), (c) $\gamma = -1$; (d), (e), (f) $\gamma = 1$.

the Lamé constants $\lambda = 100$ GN/m^2 and $\mu = 82$ GN/m^2, and let $r = 10$ and $\mathscr{F} = 1.7$. Both the displacement and the Lagrange multiplier for the Dirichlet condition are approximated with continuous piecewise linear functions over triangular meshes.

There are four solution branches intersecting at $\gamma = 0$ for $\boldsymbol{h} = \boldsymbol{h}(\gamma) = \gamma(-26, -7.5)$ GN/m^2 and the discretisation with a uniform mesh with 4096 triangles and 64 contact nodes, see Figure 3 [1]. They correspond to a partial contact and slip of the triangle to the right (Branch 2), and to no contact, contact-stick and contact-slip to the left of the lower left vertex of the triangle with pulling the whole triangle to the left (Branches 1, 3 and 4).

To explore the bifurcation in this contact problem, we have taken uniform meshes with 4096, 16384, 65536 and 262144 triangles. We shall show that the bifurcation behaviour is more complex



(a) Branch 1.

(b) Branch 2.

(c) Branch 3.

(d) Branch 4.

Figure 3: Deformed bodies corresponding to the solutions with $|\gamma| = 1$: (b) $\gamma = -1$; (a), (c), (d) $\gamma = 1$.

here. Branches 1 and 4 approach one another for finer meshes, and they disappear both for the finest mesh. Nevertheless, regarding the branching of the corresponding contact problem with forces $\boldsymbol{h} = (h_1, h_2)$ over the plane h_1–h_2, one can find it stable and convergent, again.

Acknowledgement: T. Ligurský has been supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602".

References

[1] T. Ligurský, Y. Renard: *Bifurcations in piecewise-smooth steady-state problems: abstract study and application to plane contact problems with friction*. Comput. Mech. 2015, **56**(1), pp. 39–62.

[2] T. Ligurský, Y. Renard: *A Method of Piecewise-Smooth Numerical Branching*. Accepted to Z. Angew. Math. Mech.

Elastodynamics of thin-walled structures using 3D mixed finite elements

D. Lukáš, J. Schöberl

Department of Applied Mathematics, FEECS & IT4Innovations, VŠB-TU Ostrava
Institute for Analysis and Scientific Computing, TU Vienna

1 Introduction

This is a joint work with Honeywell International s.r.o. in Brno.

Ultrasonic waves are important means for structural health monitoring (SHM) of aircrafts. Development of the SHM technology can benefit from fast and reliable computer simulations of the elastic wave equation. There are several difficulties to deal with. First of all, due to energy reasons the actuating pulse is short (5 periods) so that the Fourier transform cannot be efficiently employed. The wave equation is approximated using an unconditionaly stable Newmark scheme in time and finite elements in space. Secondly, a typical frequency is 10^5 Hz and the measurement time is 10^{-3} seconds, which leads to large numbers (10^4) of time-steps. Finally, the structures under consideration are thin so that the standard 3d displacement finite elements suffer from the shear-locking effect. Namely, convergence of the method deteoriates when refining the discretization. The reason is a bad aspect ratio of the geometry under consideration. Therefore various plate and shell models were introduced, e.g. the Kirchhoff and Reissner-Mindlin plate. A main drawback of conventional shell models is that they include rotational quantities, which cause difficulties for nonsmooth domains or when connecting shells to a structure. As a remedy the Hellinger-Reissner mixed formulation can be employed.

2 Mixed TD-NNS finite elements

In the mixed finite element method we simultaneously search for both displacement and stress field. The weak mixed formulation of the elastodynamic problem reads to find stresses $\sigma(x,t)$ satisfying $\sigma_n(x,t) = T(x,t)$ on the boundary Γ, and the displacements $u(x,t)$ such that

$$\left\{ \begin{array}{rcll} \int_\Omega \sigma : \tau + \int_\Omega \operatorname{div} \tau \cdot u &=& 0 & \forall \tau, \\ \int_\Omega \operatorname{div} \sigma \cdot v - \rho \int_\Omega \frac{\partial^2 u}{\partial t^2} \cdot v &=& 0 & \forall v, \end{array} \right. \tag{1}$$

where : stands for the Frobenius inner product and $\tau(x), v(x)$ denote the stress and displacement test functions, respectively, where $\tau_n(x) = 0$ on Γ. The first equation in (1) represents the Hooke's law, the second equation describes the Newton (force equilibrium) law. The key point is now choosing proper regularity of the stresses and displacements. In the standard mixed setting one lefts the displacements piecewise discontinuous, $u \in L^2(\Omega)$, and the stresses are symmetric tensors with continuous normal components, $\sigma \in H_{\text{sym}}(\text{div};\Omega)$. However, a known stable finite element discretization over tetrahedra [1] leads to 162 DOFs per element.

Here we follow another recent idea of Joachim Schöberl and Astrid Pechstein (born Sinwel) [2, 3] who chose tangential-continuous displacements, $u \in H(\text{curl};\Omega)$, and normal-normal-continuous stresses, $\sigma \in H(\operatorname{div}\operatorname{div};\Omega)$, which leads to a tetrahedral finite element of only 36 DOFs. The

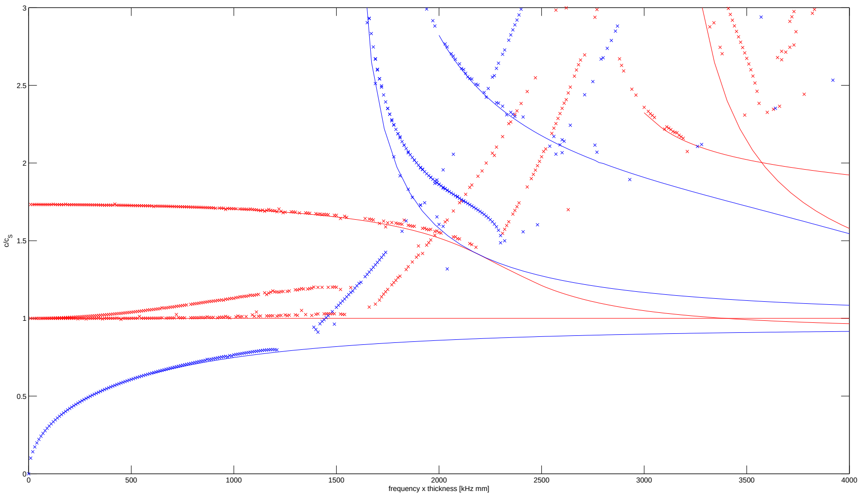


Figure 1: Theoretical (solid lines) and numerical (crosses) SH/Lamb symmetric (red) and anti-symmetric (blue) dispersion curves using TD-NNS anisotropic hexahedral elements.

method is referred to as TD-NNS. Note that in (1) the mixed terms, e.g., $\int_\Omega \operatorname{div} \sigma \cdot v$, has to be understood in the distributional sense.

In [3] prismatic triangular anisotropic elements allowing for discretizations of thin structures are proposed and analyzed. Here we additionaly propose prismatic hexahedral elements with 2 DOFs per edge, 4 DOFs per vertical face and 2 (bubble) DOFs per element as far as displacements are considered. Concerning stresses we have 9 DOFs per face and 70 (bubble) DOFs per element. We can get rid of the large amount of stress bubbles by hybridization so that the (face, normal-normal) continuity of stresses is broken and reinforced again by Lagrange multipliers, which act as normal displacements. Therefore we end up in the following purely displacement finite element:

- 2 DOFs per edge representing tangential displacements,
- 4 DOFs per vertical face representing tangential displacements,
- 9 DOFs per (both vertical and horizontal) face representing normal displacements,
- and 2 DOFs per element (cell) representing tangential displacements.

The hybridized mixed finite element discretization of elastodynamics admits the standard primal form, where both the mass and stiffness matrix are assembled elementwise, though, now they are more involved. In Fig. 1 we present a good correspondence of the theoretical dispersion diagram to the numerical simulations. The new TD-NNS elements are robust with respect to the thickness. In Fig. 2 we depict simulation of ultrasonic elastic waves in a real-world thin-walled structure. The system was numerically integrated in time by an unconditionally stable Newmark scheme.

Acknowledgement: This work has been supported by the Technology Agency of the Czech Republic under the grant TA03010140.

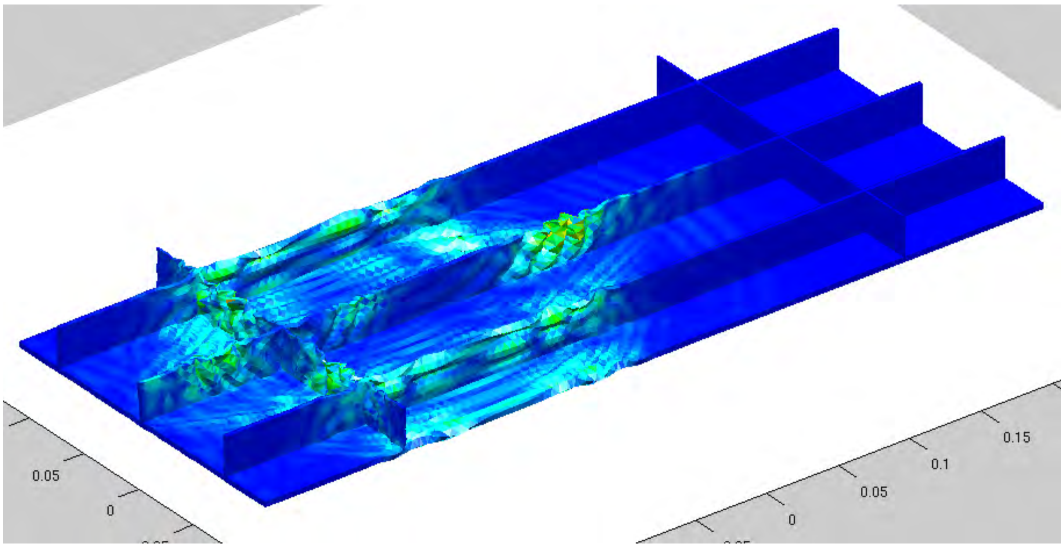


Figure 2: Simulation of ultrasonic elastic waves propagating in a thin-walled structure using TD-NNS tetrahedral elements.

References

[1] S. Adams, B. Cockburn: *A mixed finite element method for elasticity in three dimensions.* Journal of Scientific Computing **25**(3), 2005, pp. 515–521.

[2] A. Pechstein, J. Schöberl: *Tangential-displacement and normal-normal-stress continuous mixed finite elements for elasticity.* Mathematical Models and Methods in Applied Sciences **21**(8), 2011, 1761–1782.

[3] A. Pechstein, J. Schöberl: *Anisotropic mixed finite elements for elasticity.* International Journal for Numerical Methods in Engineering **90**(2), 2012, pp. 196–217.

HTFETI pro řešení nesymetrických systémů

A. Markopoulos, T. Brzobohatý, V. Ryška

IT4Innovations, VŠB - Technická univerzita Ostrava

1 Úvod

Metoda FETI (Finite Element Tearing and Interconnecting) [1] je často využívaná pro řešení problémů velkého rozsahu. Její algoritmus spočívá v eliminaci primárních neznámých způsobem, který umožňuje řešit duální systém lineárních rovnic pomocí projektovaných variant Krylovovských metod.

Prezentace se zaměřuje na případy, kdy rozdělení řešeného problému na příliš velké množství podoblastí negativně ovlivňuje výkon metody FETI. Pomocí metody Hybridní FETI (HFETI) navržené Klawonnem a spol. (viz např. [2]) můžeme zredukovat tento efekt slepením několika sousedních podoblastí do clusterů. Tento postup upravuje strukturu algoritmu tak, že $\tilde{n}$ clusterů ($\tilde{n} <$ počet podoblastí) se chová jako $\tilde{n}$ podoblastí ve standardní FETI metodě.

2 Skalární advekčně-difúzní rovnice

Mějme ohraničenou oblast $\Omega \subset \mathbb{R}^2$ s po částech hladkou hranicí $\partial\Omega$. Uvažujme následující okrajovou úlohu:

$$\left\{ \begin{array}{rcl} \underbrace{-\nabla \cdot (\mathbf{D}\nabla c)}_{\text{difúze}} + \underbrace{\mathbf{a} \cdot \nabla c}_{\text{advekce}} & = & \mathbf{f} \text{ v } \Omega \\ c & = & g \text{ na } \partial\Omega, \end{array} \right. \tag{1}$$

kde $c = c(x_1, x_2)$ reprezentuje neznámou skalární funkci, $\mathbf{D} = \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix}$ je tenzor difúze, $\mathbf{a} = (a_1, a_2)^T$ označuje vstupní vektorové pole rychlostí advekce, $\mathbf{f} = \mathbf{f}(x_1, x_2)$ je zdrojový člen a g je okrajová podmínka předepsaná na $\partial\Omega$.

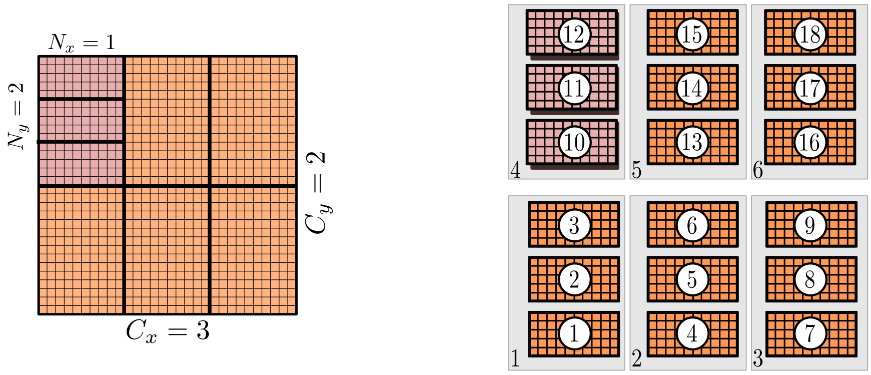
V našem případě advekce převládá nad difúzí, což způsobuje značnou nestabilitu numerického MKP řešení. Tuto nestabilitu řešíme pomocí CAU metody blíže popsané v článku [3]. Princip této metody zjednodušeně spočívá v přičtení tzv. „numerické difúze“ k matici tuhosti, která vyplyne z metody konečných prvků (MKP), jak ukazuje následující rovnice

$$\underbrace{(\mathbf{K}_{diff} + \mathbf{K}_{adv} + \mathbf{K}_{CAU})}_{\mathbf{K}} \mathbf{u} = \underbrace{\mathbf{f}_G + \mathbf{f}_{CAU}}_{\mathbf{f}}. \tag{2}$$

Zde $\mathbf{K}_{diff}$ a $\mathbf{K}_{adv}$ označují symetrickou a nesymetrickou matici korespondující k difúznímu respektive advekčnímu členu. Součet těchto dvou matic odpovídá matici tuhosti vzniklé z MKP stejně jako vektor $\mathbf{f}_G$. Matice $\mathbf{K}_{CAU}$ a vektor $\mathbf{f}_{CAU}$ reprezentují dříve zmíněnou „numerickou difúzi“. Vektor neznámých je označen jako $\mathbf{u}$.

3 Hybridní Total FETI metoda

Pro srozumitelnější popis HTFETI metody použijeme hierarchicky dekomponovanou doménu (viz obrázek 1).



Obrázek 1: Doménová dekompozice: $C_x = 3$, $C_y = 2$, $N_x = 1$, $N_y = 3$, $n_x = 10$, and $n_y = 5$.

Diskretizace a dekompozice sestává z 3 úrovní:

- 1. úroveň - dekompozice na clustery. Každý z nich při výpočtu zaměstná 1 výpočetní uzel. Parametry C_x a C_y určují počet clusterů v x-ovém a y-ovém směru.
- 2. úroveň - každý cluster je dekomponován na podoblasti. Parametry N_x a N_y udávají počet podoblastí ve směrech x a y.
- 3. úroveň - každá podoblast je diskretizována uniformně 2D konečnými prvky. Jejich počet v x-ovém a y-ovém směru udávají parametry n_x, n_y.

Implementace HTFETI se velice podobá klasické TFETI metodě. V obou algoritmech jsou Lagrangeovy multiplikátory (LM) použity jak pro lepení podoblastí dohromady, tak pro vynucení Dirichletových okrajových podmínek. Z toho důvodu jsou všechny podoblasti plovoucí, a proto matice tuhosti na všech podoblastech vykazují defekt a lze na ně použít stejný algoritmus, který počítá se singulární maticí.

Poděkování: Tato práce byla podpořena Ministerstvem školství mládeže a tělovýchovy z Národního programu udržitelnosti II (NPU II) v rámci projektu „IT4Innovations excellence in science - LQ1602“ a dále také Vysokou školou báňskou - Technickou univerzitou Ostrava z SGS projektu SP2016/150 „Modelování povodní a znečištění II“.

Literatura

[1] C. Farhat, F.-X. Roux: *A method of Finite element tearing and interconnecting and its parallel solution algorithm*, Int J Numer Meth Eng 32 (1991), pp. 1205–1227.

[2] A. Klawonn, R. Rheinbach: *Highly scalable parallel domain decomposition methods with an application to biomechanics*, ZAMM 1 (2010), pp. 5–32.

[3] E.D.C. Gomes, G.A. Benitez: *A new upwind function in stabilized Finite element formulations, using linear and quadratic elements for scalar convection-diffusion problems.* Computer Methods in Applied Mechanics and Engineering, Volume 193(2004), pp. 2383–2402, http://dx.doi.org/10.1016/j.cma.2004.01.015.

On the optimal initial conditions for an inverse problem of model parameter estimation

C. Matonoha, Š. Papáček

Institute of Computer Science of the CAS, Prague
Institute of Complex Systems, University of South Bohemia in České Budějovice, Nové Hrady

1 Introduction

The aim of this contribution is to establish the link between experimental conditions (protocol) and the accuracy of the results. The idea is presented in a simplified case study of FRAP (Fluorescence Recovery After Photobleaching) data processing [7, 3]. It serves as a paradigmatic example of the inverse problem of the diffusion parameter estimation from spatio-temporal measurements of fluorescent particle concentration. A natural question is how the experimental settings influence the accuracy of resulting parameter estimates. There are many rather empirical recommendations related to the design of a photobleaching experiment, e.g., the bleach spot shape and size, the region of interest (location and size), the total time of measurement, cf. [2]. However, we should have a more rigorous tool for the choice of experimental design factors. This goal can be achieved through a reliable process model, the Fickian diffusion equation, and performing the subsequent sensitivity analysis with respect to the model parameters. Thus, we can define an optimization problem as the maximization of the sensitivity measure. The special focus of this contribution concerns the search for the optimal bleaching pattern [2, 5], or, from the more mathematical viewpoint, we show how to find an optimal binary-valued initial conditions in a diffusion-parameter estimation problem.

2 Problem formulation

We consider the Fickian diffusion problem with a *constant* diffusion coefficient $\delta > 0$ and assume a spatially radially symmetric observation domain, i.e., the data are observed on a cylinder with the radius $R = 1$ and height $T = 1$. In FRAP, the simplest governing equation for the spatio-temporal distribution of fluorescent particle concentration $u(r, t)$ is the diffusion equation

$$\frac{\partial u}{\partial t} = \delta \left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} \right), \tag{1}$$

where $r \in [0, 1],\ t \in [0, 1]$, with the initial and Neumann boundary conditions

$$u(r, 0) = u_0(r), \quad \frac{\partial u}{\partial r}(1, t) = 0. \tag{2}$$

The main issue in FRAP and related estimation problems is to find the value of the diffusion coefficient δ from spatio-temporal measurements of the concentration $u(r, t)$, cf. [6, 7].

Obviously, the measured data are discrete and each data entry quantifies the variable u at a particular spatio-temporal point (r, t) in a finite domain, i.e.,

$$\begin{aligned} u(r_i, t_j), \qquad & i = 0 \dots n, \quad r_0 = 0, \quad r_n = 1, \\ & j = 0 \dots m, \quad t_0 = 0, \quad t_n = 1, \end{aligned}$$

where i is the spatial index uniquely identifying the pixel position where the value of fluorescence intensity u is measured and j is the time index (the initial condition corresponds to $j = 0$). Usually, the data points are uniformly distributed both in time (the time interval Δt between two consecutive measurements is constant) and space, i.e., on an equidistant mesh with the step-size Δr, cf. [4].

Given the data as above, the diffusion coefficient δ can be computed numerically by solving the inverse problem to (1)-(2). Because of unavoidable noise in the data, one obtains an estimated value $\overline{\delta}$ which reasonably well approximates the true δ. It can be shown [1, 4], that for our case of single scalar parameter estimation and white noise as data error assumed, the expected relative error in δ depends on the data noise and a factor, which we call the global semi-relative squared sensitivity S_{GRS}, as follows

$$\mathbb{E}\left(\left|\frac{\overline{\delta}-\delta}{\delta}\right|^2\right) \sim \frac{\sigma^2}{S_{GRS}}, \tag{3}$$

where σ^2 denotes the variance of the additive Gaussian noise. The sensitivity measure S_{GRS} is

$$S_{GRS} := \delta^2 \sum_{i=0}^{n} \sum_{j=1}^{m} \left[\frac{\partial}{\partial \delta} u(r_i, t_j)\right]^2, \tag{4}$$

where $\frac{\partial}{\partial \delta} u(r_i, t_j)$ is the usual sensitivity of the model output in the spatio-temporal point (r_i, t_j) with respect to the parameter δ. It is obvious from this estimate that if the noise level is fixed, the estimation of δ can only be improved by switching to an experimental design with a higher sensitivity.

The sensitivity measure (4) involves several design parameters. If all the above parameters $R, T, \Delta r, \Delta t$ are fixed, there is only one way to maximize the sensitivity measure S_{GRS}: to consider the *initial bleach* u_0 in (2) as the experimental design parameter. By optimizing the bleach design, we mean to select the initial conditions in such a way that S_{GRS} is maximized and hence the expected error in δ is minimized. In order to do so, we have to choose the class of designs from which we select the initial conditions. Without loss of generality, we assume $u_0(r)$ is a $\{1, 0\}$-function

$$u_0(r) = \begin{cases} 1, & r \in B, \\ 0, & \text{else}, \end{cases} \tag{5}$$

where B is an open subset of $[0, 1]$ (not necessarily continuous), which we call further as the bleaching pattern or the bleach shape. The set B determines the initial condition $u_0(r)$ and thus the value S_{GRS}. The goal is to find such a set B_{opt} determining the non-zero sub-vector of $u_0(r)$, and thus the optimal initial condition $u_0^{opt}(r)$, where S_{GRS} reaches its maximum:

$$B_{opt} = \arg \max_{B \subset [0,1]} S_{GRS}. \tag{6}$$

Depending on the different restrictions imposed on the initial bleach, we can study problem (6) not only with a fixed bleach depth ($u_0(r)$ is a $\{1, 0\}$-function) but also with a fixed energy, i.e. with an additional restriction

$$\int u_0(r) \mathrm{d}r = c, \tag{7}$$

where $c > 0$ is a given constant.

The function u_0 is discontinuous, so the problems with definition and existence of a solution to (1)-(2) occur. This drawback is partially restricted by the fact that B is an open subset. Thus, we keep a classical formulation of the initial boundary value problem (1)-(2) and use a finite difference Crank-Nicholson (CN) scheme to compute a numerical solution $u(r_i, t_j)$, $i = 0 \dots n-1$, $j = 1 \dots m$. Replacing the derivative with a finite difference, the sensitivity measure S_{GRS} can be approximated with a value S_{app} as follows

$$S_{GRS} \approx S_{app} = \sum_{j=1}^{m} j^2 \sum_{i=0}^{n} \left[u(r_i, t_j) - u(r_i, t_{j-1})\right]^2 . \tag{8}$$

The values $u(r_i, t_j)$ are computed from $u(r_i, t_{j-1})$ using the CN scheme, thus no extra work is necessary.

3 Numerical example

To demonstrate the optimal configurations of the initial condition let us choose $n = 30$, $m = 200$ and find such an initial condition $(u_0(r_0), \dots, u_0(r_n))^T \in \mathcal{R}^{n+1}$ in form of a $\{1, 0\}$-function, cf. (5), that maximizes the value S_{app} (8) for $1/\delta = 1, 2, \dots, 120$ (notice the inverse values of δ). For the sake of simplicity we consider four types of shapes (or patterns) of the initial condition[2] (the sets B in (5)):

$$\begin{aligned}
\text{disk}: &\quad u_0 = (1, \dots, 1\ [, 0, \dots, 0])^T \\
\text{annulus}: &\quad u_0 = (0, \dots, 0, 1, \dots, 1\ [, 0, \dots, 0])^T \\
\text{disk+annulus}: &\quad u_0 = (1, \dots, 1, 0, \dots, 0, 1, \dots, 1\ [, 0, \dots, 0])^T \\
\text{double annulus}: &\quad u_0 = (0, \dots, 0, 1, \dots, 1, 0, \dots, 0, 1, \dots, 1\ [, 0, \dots, 0])^T
\end{aligned}$$

Figure 1 shows the result of the optimization problem (6), i.e., the vertical lines visualize both the shape and size of the initial condition u_0 leading to the maximal value of S_{app}. Figure 2 shows the maximum values of S_{app} vs. $1/\delta$ for the four different classes of the initial shapes of bleaching patterns. There are intervals of δ where the value S_{app} is maximal for disk ($\delta \geq 1/17$), annulus ($\delta \geq 1/44$), disk+annulus ($1/82 \leq \delta \leq 1/18$), and double annulus ($1/120 \leq \delta \leq 1/45$). For $\delta \to 0$ we can find more complicated shapes as optimal initial conditions.

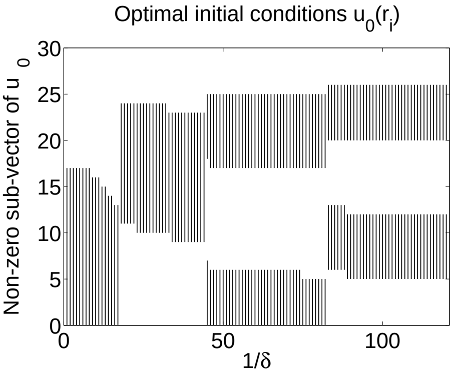


Figure 1: The result of optimization problem (6): each vertical line indicates the non-zero sub-vector of u_0 for which S_{app} is maximal.

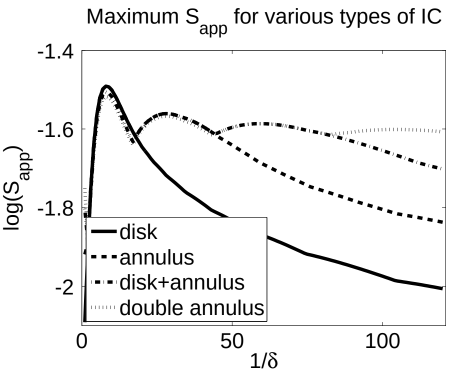


Figure 2: Each curve indicates the maximum value of S_{app} determined by initial conditions from the four groups listed above.

[2] The last zero sub-vector, marked in square brackets, can be empty in each u_0.

4 Conclusion

Although the experimental devices for FRAP measurements allow an arbitrary bleaching pattern, there is hardly any study concerning its influence on the accuracy of resulting parameter estimates. Thus, we formulated the problem of the optimal initial condition for the further identification of a constant diffusion coefficient. We set a sensitivity measure as the optimality criterion to be maximized in order to have the expected error minimal, cf. (4). Our preliminary numerical results indicate that there exist specific initial conditions u_0 that maximize the sensitivity measure S_{app} and therefore minimize the error in the model parameter estimate (diffusion coefficient δ). The optimal initial shapes or bleaching patterns are functions of δ. We found not only disks of various radii (the usual bleach shape used in the FRAP community) but also annuli and other more complicated radially symmetric patterns. These optimal initial conditions depend not only on δ but also on other parameters reflecting the experimental protocol. How exactly these parameters as well as the restriction on a fixed energy influence the solution of our problem of bleaching pattern optimization is the subject of ongoing research.

Acknowledgement: This work was supported by the long-term strategic development financing of the Institute of Computer Science (RVO:67985807), by the OeAD within the programme “Aktion Oesterreich-Tschechien (AOeCZ-Universitaetslehrerstipendien)", and by the Ministry of Education, Youth and Sport of the Czech Republic – projects CENAKVA (No. CZ.1.05/2.1.00/01.0024) and CENAKVA II (No. LO1205 under the NPU I program).

References

[1] D.M. Bates, D.G. Watts: *Nonlinear regression analysis: Its applications.* John Wiley & Sons, New York, 1988.

[2] D. Blumenthal, L. Goldstien, M. Edidin, L.A. Gheber: *Universal approach to FRAP analysis of arbitrary bleaching patterns.* Scientific Reports **5**, 11655 (2015); doi: 10.1038/srep 11655.

[3] R. Kaňa, E. Kotabová, M. Lukeš, Š. Papáček, C. Matonoha, L.N. Liu, O. Prášil, C.W. Mullineaux: *Phycobilisome mobility and its role in the regulation of light harvesting in red algae.* Plant Physiology **165(4)** (2014), pp. 1618–1631.

[4] S. Kindermann, Š. Papáček: *On data space selection and data processing for parameter identification in a reaction-diffusion model based on FRAP experiments.* Abstract and Applied Analysis, Article ID 859849 (2015).

[5] S. Kindermann, Š. Papáček: *Optimization of the shape (and topology) of the initial conditions for diffusion parameter identification*, submitted.

[6] C.W. Mullineaux, M.J. Tobin, G.R. Jones: *Mobility of photosynthetic complexes in thylakoid membranes.* Nature **390** (1997), 421–424.

[7] Š. Papáček, R. Kaňa, C. Matonoha: *Estimation of diffusivity of phycobilisomes on thylakoid membrane based on spatio-temporal FRAP images.* Mathematical and Computer Modelling **57** (2013), pp. 1907–1912.

Solution of contact problems for nonlinear beam and foundation

H. Netuka, J. Machalová

Department of Mathematical Analysis and Applications of Mathematics
Faculty of Science, Palacký University, Olomouc

1 Introduction

We will consider the following nonlinear beam model

$$EI\,w'''' - E\alpha\,(w')^2 w'' + P\mu\,w'' = f \qquad \text{in } (0,\mathrm{L}). \tag{1}$$

The beam has a constant stiffness given by the elastic modulus of material E and constant area moment of inertia I. Its length is L and thickness is 2h. The width of the beam we consider as a unit. The distributed transverse load is denoted by $q(x)$ and $w(x)$ describes the deflection of the beam at a position x. Furthermore, in (1) we denoted

$$I = \frac{2}{3}\mathrm{h}^3, \quad \alpha = 3\mathrm{h}(1-\nu^2), \quad \mu = (1+\nu)(1-\nu^2), \quad f = (1-\nu^2)q \tag{2}$$

with ν as the Poisson's ratio. Finally, P is an external axial force applied on $x = \mathrm{L}$ which causes compression for $P > 0$ or tension for $P < 0$. This mathematical model was developed in 1996 by D.Y. Gao (see [2]) and is successfully used since its publication.

The potential energy of this beam $\Pi_G \colon V \to \mathbb{R}$ is defined by

$$\Pi_G(v) = \frac{1}{2}\int_0^{\mathrm{L}} EI(v'')^2 \mathrm{d}x + \frac{1}{12}\int_0^{\mathrm{L}} E\alpha(v')^4 \mathrm{d}x - \frac{1}{2}\int_0^{\mathrm{L}} P\mu(v')^2 \mathrm{d}x - \int_0^{\mathrm{L}} fv\,\mathrm{d}x, \quad v \in V, \tag{3}$$

where V is the space of kinematically admissible deflections, so that essential boundary conditions should be entered into V. It can be proved that this functional is coercive on V and for $P \le P_{cr}$ strictly convex as well, where the critical load P_{cr} is determined by

$$P_{cr} = \min_{w \in V} \frac{\int_0^{\mathrm{L}} EI\,(w'')^2\,\mathrm{d}x}{\int_0^{\mathrm{L}} \mu\,(w')^2\,\mathrm{d}x}. \tag{4}$$

This implies that for any f and $P \colon P \le P_{cr}$ the variational problem

$$\left\{ \begin{array}{l} \text{Find } w \in V \text{ such that} \\ \Pi_G(w) = \min\limits_{v \in V} \Pi_G(v) \end{array} \right. \tag{5}$$

has exactly one solution and the stationary condition $\Pi_G'(w, v) = 0$ leads to the governing equation (1).

2 Normal compliance contact model

Now let us consider a foundation lying in the distance $\mathrm{g} \le 0$ below the Gao beam (see Fig. 1). We can distinguish two principal cases: deformable and undeformable (or rigid) foundation. Here we will study only the first case with $k_F > 0$ as the foundation modulus.

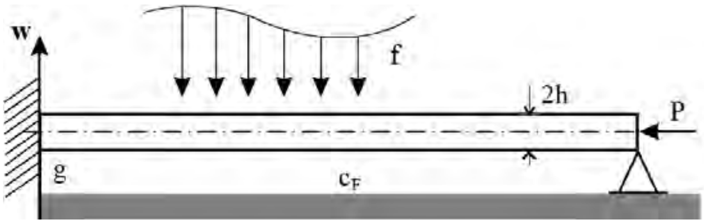


Figure 1: Gao beam and deformable foundation.

Simple considerations (see [5]) lead to the enhanced equation

$$EI\,w'''' \,-\, E\alpha\,(w')^2 w'' \,+\, P\mu\,w'' \,=\, f + T(w) \qquad \text{in } (0,\mathrm{L}), \tag{6}$$

where contact force T is given by

$$T(w) \,=\, c_F\,(\mathrm{g} - w)^+, \qquad\qquad c_F \,=\, (1-\nu^2)k_F,\ v^+(x) \,=\, \max\{0, v(x)\}. \tag{7}$$

Such relation is usually referred as the so-called *normal compliance condition.* The first formulation of contact problem using normal compliance condition was presented in [7].

Equation (6) is associated with the total potential energy Π_T of the whole system given by

$$\Pi_T(v) \,=\, \Pi_G(v) + \frac{1}{2}\int_0^{\mathrm{L}} c_F((\mathrm{g}-v)^+)^2 \mathrm{d}x, \qquad v \in V. \tag{8}$$

This functional has the same properties which have been mentioned above in connection with the functional Π_G. Hence the problem

$$\left\{ \begin{array}{l} \text{Find } w \in V \text{ such that} \\ \Pi_T(w) \,=\, \min\limits_{v \in V} \Pi_T(v) \end{array} \right. \tag{9}$$

has one solution if $P \le P_{cr}$. It is worth noting that we did not obtain variational inequality what is usual in the case of Signorini conditions, which are relevant for rigid foundations.

3 Solution using optimal control

Our idea is that we transform the contact problem (9) into an optimal control problem. For this purpose let us perform the variable transformation $z = v'$. Using the method of Lagrange multipliers, we construct the Lagrangian

$$\begin{aligned} \mathcal{L}(v,z,\lambda) \;=\; & \frac{1}{2}\int_0^{\mathrm{L}} EI\,(v'')^2 \mathrm{d}x + \frac{1}{12}\int_0^{\mathrm{L}} E\alpha z^4 \mathrm{d}x - \frac{1}{2}\int_0^{\mathrm{L}} P\mu z^2 \mathrm{d}x - \int_0^{\mathrm{L}} fv \mathrm{d}x + \\ & + \frac{1}{2}\int_0^{\mathrm{L}} c_F((\mathrm{g}-v)^+)^2 \mathrm{d}x + \int_0^{\mathrm{L}} \lambda(v'-z)\mathrm{d}x, \qquad v \in V, z \in Z, \lambda \in \Lambda, \end{aligned} \tag{10}$$

where V is the given function space, Z is the space of their derivatives and $\Lambda = L^2((0,\mathrm{L}))$. Problem (9) is then replaced by

$$\mathcal{L}(v,z,\lambda) \longrightarrow \underset{v,z,\lambda}{\mathrm{stat}}, \tag{11}$$

which investigates a stationary point of Lagrangian $\mathcal{L}$ (for more details see [3]). After some calculations we obtain (among other results)

$$\int_0^{\mathrm{L}} EI\,w''\varphi'' \mathrm{d}x - \int_0^{\mathrm{L}} f\varphi \mathrm{d}x - \int_0^{\mathrm{L}} u\varphi \mathrm{d}x \,=\, 0 \qquad \forall \varphi \in V, \tag{12}$$

where we have set

$$\int_0^{\mathrm{L}} u\varphi \mathrm{d}x = \int_0^{\mathrm{L}} c_F(\mathrm{g} - w)^+\varphi \mathrm{d}x - \int_0^{\mathrm{L}} \lambda\varphi' \mathrm{d}x \qquad \forall\varphi \in V. \tag{13}$$

From (12) we get the so-called *state problem*

$$\left\{ \begin{array}{l} \text{For given } u \text{ find } \ w := w(u) \in V \text{ such that} \\ \displaystyle\int_0^{\mathrm{L}} EI\, w''\varphi'' \mathrm{d}x = \int_0^{\mathrm{L}} (f + u)\varphi \mathrm{d}x \qquad \forall\varphi \in V. \end{array} \right. \tag{14}$$

Next, using this relation, we rearrange functional (8) to the following form

$$\begin{aligned} J(w, u) \;=\; & \frac{1}{2}\int_0^{\mathrm{L}} u(w - \widehat{w})\mathrm{d}x + \frac{1}{12}\int_0^{\mathrm{L}} E\alpha(w')^4\mathrm{d}x - \frac{1}{2}\int_0^{\mathrm{L}} P\mu(w')^2\mathrm{d}x + \\ & + \frac{1}{2}\int_0^{\mathrm{L}} c_F\big((\mathrm{g} - w)^+\big)^2\mathrm{d}x, \end{aligned} \tag{15}$$

where function $\widehat{w}$ solves the state equation for $u = 0$. Now we are able to define the optimal control problem

$$\left\{ \begin{array}{l} \text{Find a function } u^* \in U_{ad} \text{ such that} \\ J\left(w(u^*), u^*\right) = \displaystyle\min_{u \in U_{ad}} J\left(w(u), u\right), \\ \text{where } \ w(u) \in V \ \text{ solves the state problem (14)} \end{array} \right. \tag{16}$$

with admissible set of controls determined by

$$U_{ad} = \{u \in L^2((0, \mathrm{L})) : \; |u(x)| \leq C \;\; \text{a.e. in } (0, \mathrm{L})\} \tag{17}$$

for some positive constant C, because we do not want to break the beam. More information on the optimal control of differential equations can be find in monographs [4] and [10].

In our case it can be proved that under the assumption $P \leq P_{cr}$ problem (16) has just one optimal pair $(w^*, u^*) \in V \times U_{ad}$ and the function $w^* = w(u^*)$ solves variational problem (9).

4 Numerical realization

As we can see from (16), computational procedure will consist of two parts: evaluation of given state problem (14) simultaneously with minimization of the transformed functional J. The first task does not make much troubles, because the finite element solution for Euler–Bernoulli beam is well-known, see e.g. [8].

Hence for given control value u^k we compute state w^k and then for the pair (w^k, u^k) we perform one step of minimization process for functional $J(w, u)$. For this purpose we used the conditioned gradient method (see [10]). Descent directions was given by gradients of $J(w, u)$ which were evaluated by means of adjoint problem technique. Step lengths were determined by using algorithms described in [6].

For illustrative purposes a simple example is given here. We consider the beam from Fig. 1 with the following data:
$\mathrm{L} = 1\,\mathrm{m}$, $\mathrm{h} = 0.1\,\mathrm{m}$, $E = 21{\cdot}10^{10}\,\mathrm{Pa}$, $I = 0.666667{\cdot}10^{-3}\,\mathrm{m}^4$, $\nu = 0.3$, $q = -5{\cdot}10^8\,\mathrm{N}$ (constant).

From Fig. 2 it is seen that the tougher foundation on the right enables significantly smaller penetration into the foundation, hence the contact zone is smaller than the one on the left side. The blue stars show deflection of the Euler–Bernoulli model, the red stars correspond to the Gao beam and green crosses represent the foundation.

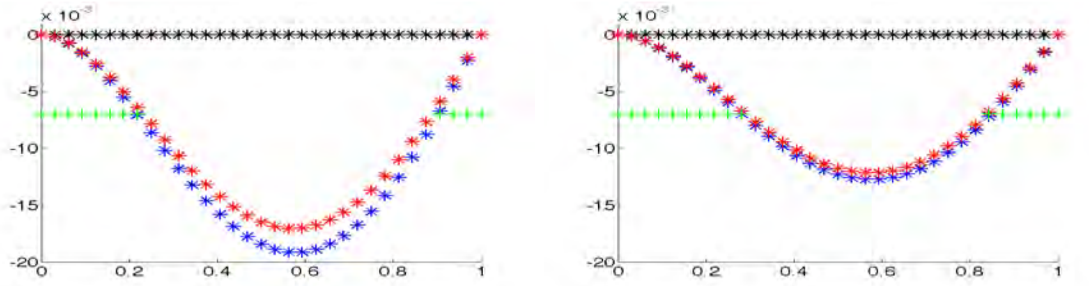


Figure 2: $k_F = 5 \cdot 10^8\,\mathrm{Nm}^{-3}$ $k_F = 5 \cdot 10^{10}\,\mathrm{Nm}^{-3}$

5 Conclusion

The idea used here is based on the original method applied in [9] and [1]. The authors call it the *control variational method.* The subject of the papers was linear cantilever beam (Euler–Bernoulli model) with foundation. Our research significantly generalized these works as we consider nonlinear Gao beam in addition subjected to axial load and with all kinds of possible boundary conditions. We have also expanded and refined numerical solution for this approach.

References

[1] M. Barboteu, M. Sofonea, D. Tiba: *The control variational method for beams in contact with deformable obstacles.* ZAMM 92 (1), pp. 25–40, 2012.

[2] D.Y. Gao: *Nonlinear elastic beam theory with application in contact problems and variational approaches.* Mechanics Research Communications, 23 (1), pp. 11–17, 1996.

[3] A.S. Kravchuk, P.J. Neittaanmäki: *Variational and Quasi-Variational Inequalities in Mechanics.* Solid Mechanics and Its Applications (Book 147). Springer, 2007.

[4] J.-L. Lions: *Optimal Control of Systems Governed by Partial Differential Equations.* Springer-Verlag, Berlin, 1971.

[5] J. Machalová, H. Netuka: *Solution of Contact Problems for Nonlinear Gao Beam and Obstacle.* Journal of Applied Mathematics, Vol. 2015, Article ID 420649, 12 pages, 2015. http://dx.doi.org/10.1155/2015/420649

[6] J. Nocedal, S.J. Wright: *Numerical optimization.* Second edition. Springer-Verlag, New York, 2006.

[7] J.T. Oden, J.A.C. Martins: *Models and computational methods for dynamic friction phenomena.* Computer Methods in Applied Mechanics and Engineering, 52, 527–634, 1985.

[8] J.N. Reddy: *An Introduction to the Finite Element Method.* Third edition. McGraw-Hill Book Co., New York, 2006.

[9] M. Sofonea, D. Tiba: *The control variational method for contact of Euler-Bernoulli beams.* Bull. Transilvania Univ. Brasov, Ser. III, Math. Inf. Phys. 2, pp. 127–136, 2009.

[10] F. Tröltzsch: *Optimal Control of Partial Differential Equations. Theory, Methods and Applications.* AMS, Providence, Rhode Island, 2010.

Multilevel methods for elliptic differential equations with log-normally distributed parameters

I. Pultarová

Mathematical Institute, Faculty of Mathematics and Physics, Charles University in Prague,
Department of Mathematics, Faculty of Civil Engineering, Czech Technical University in Prague

1 Introduction

Several classes of numerical methods can be used for the solution of elliptic differential equations with randomly distributed data: the Monte Carlo methods, collocation methods and the stochastic Galerkin method. In our contribution, we focus on the stochastic Galerkin method. It is applied to the variational form of the problem with respect to both physical and random variables. We assume that coefficients at the elliptic part of the equation depend on N random variables $y_1, \ldots, y_N$ which are independent and log-normally distributed with the joint probability density ρ. Then the problem is to find $u(x,y) \in H = H_0^1(D) \otimes L_\rho^2(\mathcal{R}^n)$ such that

$$\int_{\mathcal{R}^n} \int_D \exp\left(a(x,y)\right) \nabla u(x,y) \nabla v(x,y) \rho(y)\, dx\, dy = \int_{\mathcal{R}^n} \int_D b(x) v(x,y) \rho(y)\, dx\, dy, \tag{1}$$

where $D \subset \mathcal{R}^d$, $d = 1, 2$ or 3, $x \in D$ is a spatial variable and y is a finite dimensional random field, $y = (y_1, \ldots, y_N)$, see, for example, [2].

The solution $u(x,y)$ of (1) is assumed to be approximated by a generalized polynomial chaos expansion and by finite element (FE) functions of the physical variable x

$$u(x,y) = \sum_{r=1}^{F} \sum_{j=1}^{M} u_{jr} \Phi_j(y) \psi_r(x),$$

where $\{\Phi_j(y)\}_{j=1}^M$ is a set of N-variate tensor product polynomials orthogonal with respect to the scalar product of $L_\rho^2(\mathcal{R}^N)$. In particular, $\Phi_j(y) = \prod_{i=1}^N \varphi_{ji}(y_i)$, where $\varphi_k(y_i)$ is Hermite orthogonal polynomial of order k. For approximation we use $0 \leq k_i \leq P_i$, where P_i are some given constants. Assuming a linear expansion of $a(x,y)$, we obtain a system of $M \times F$ linear equations with $M \times F$ unknowns u_{jr},

$$Au = B,$$

where

$$\begin{aligned} A_{ir,js} &= \int_{\mathcal{R}^N} \int_D \exp\left(a_0(x) + \sum_{k=1}^N a_k(x) y_k\right) \nabla \psi_s(x) \nabla \psi_r(x) \Phi_i(y) \Phi_j(y) \rho(y)\, dx\, dy, \\ B_{ir} &= \int_{\mathcal{R}^N} \int_D b(x) \psi_r(x) \Phi_i(y) \rho(y)\, dx\, dy \end{aligned}$$

The matrix A is huge and usually poorly conditioned, and even if A is never constructed explicitly in practical computation, it is worth to search for efficient preconditioning methods. These methods are usually based on some natural block splitting of A [5] or use some hierarchy of approximation subspaces [6, 7, 8, 9].

2 Block and multilevel preconditioning

In our contribution, we present and study several block preconditioning methods and algebraic multilevel (AML) methods [1, 3], and prove guaranteed upper bounds for the resulting condition numbers of the preconditioned matrix A. We focus on the following methods

(a) block diagonal preconditioning with $P_N + 1$ blocks corresponding to approximation subspaces which contain the approximation polynomials $\varphi_{k_N}(y_N)$ of orders $0, 1, \ldots, P_N$, respectively

(b) AML preconditioning in a form of a V-cycle with a hierarchical splitting of the approximation spaces with respect to the degree of approximation polynomials $\varphi_{k_N}(y_N)$

(c) AML preconditioning in a form of a W-cycle for the same hierarchy as in (b).

We introduce a tool for proving upper bounds for the resulting condition numbers for all of these methods. Some of the proofs are based on estimating the strengthened CBS constants for the hierarchical approximation subspaces and for the energy scalar product defined by the problem. Our methodology is based on an "element-wise" approach, which is well known from the classical AML theory [1, 3, 4]. We mention, that the guaranteed bounds for the CBS constants can be employed for two-sided a posteriori error estimates and for adaptive algorithms as well.

Acknowledgement. This work was supported by ERC-CZ project LL1202.

References

[1] O. Axelsson: *Iterative Solution Methods.* Cambridge University Press, 1996.

[2] I. Babuška, F. Nobile, R. Tempone: *A stochastic collocation method for elliptic partial differential equation with random input data*, SIAM Review 2, 2010, pp. 317-355.

[3] V. Eijkhout, P. Vassilevski: *The role of the strengthened C.B.S.-inequality in multi-level methods*, SIAM Review 33, 1991, pp. 405-419.

[4] J. Kraus, S. Margenov: *Robust Algebraic Multilevel Methods and Algorithms*, Radon Series on Computational and Applied Mthematics 5, RICAM, Walter de Gruyter, 2009, Germany.

[5] C.E. Powell, H.C. Elman: *Block-diagonal preconditioning for spectral stochastic finite-element systems*, IMA Journal of Numerical Analysis 29, 2009, pp. 350-375

[6] I. Pultarová: *Adaptive Algorithm for stochastic Galerkin method*, Applications of Mathematics 60, 2015, pp. 551-571.

[7] I. Pultarová: *Hierarchical preconditioning for the stochastic Galerkin method: upper bounds to the strengthened CBS constants*, Computer and Mathematics with Applications 71, 2016, pp. 949-964.

[8] B. Sousedík, R.G. Ghanem: *Truncated hierarchical preconditioning for the stochastic Galerkin FEM*, International Journal for Uncertainty Quantification 4, 2014, pp. 333-348.

[9] B. Sousedík, R.G. Ghanem, E.T. Phipps: *Hierarchical Schur complement preconditioner for the stochastic Galerkin methods*, Numerical Linear Algebra with Applications 21, 2013, pp. 136-151.

Computable estimates of the distance to the set of divergence free fields with applications to Hencky's plasticity

S. Repin, S. Sysala, J. Haslinger

University of Jyväskylä & V.A. Steklov Institute of Mathematics of the RAS, St. Petersburg
Institute of Geonics of the CAS, Ostrava
Institute of Geonics of the CAS, Ostrava & Charles University in Prague

Abstract

Estimates of the distance of a function to the space of divergence free fields follow for example from the well-known inf-sup condition for incompressible media [1, 7, 9, 10, 11] and contain an LBB-type constant. This constant can be estimated by analytical methods only for specific shapes of domains and the Dirichlet boundary conditions [4, 8]. Their semi-analytic extensions to more general domains and mixed boundary conditions have been proposed in the recent papers [10, 11, 12, 13] concerning a posteriori error estimates for incompressible flow problems.

In this contribution, we use such estimates in Hencky's plasticity problem to analyze a limit factor λ^* for a prescribed load [3, 15]. Finding λ^* is one of the most important tasks in quantitative analysis of the Hencky problem since beyond this limit value no physically reasonable solution may exist. Within the kinematical approach to limit analysis, λ^* can be defined as the infimum of a nonsmooth and convex functional subject to the divergence free constraint imposed on displacement fields. This minimization problem is called the limit analysis problem [3, 15]. It can be solved, e.g. by the penalty method studied in [5, 6, 14] or by the augmented Lagrangian method [2].

The presented estimates of the distance to the set of divergence free fields enable us to find guaranteed and fully computable upper bounds of λ^* using functions which need not satisfy the divergence free constraint. To get bounds which are close to λ^* we utilize special functions, namely solutions to a discrete penalized version of the limit analysis problem. We construct a sequence of upper bounds that converges to λ^* as the discretization parameter tends to zero. This result is very useful since the sequence of the discrete limit load parameters need not converge to λ^* due to possible locking phenomena.

References

[1] I. Babuška, A.K. Aziz: *Survey lectures on the mathematical foundations of the finite element method.* In: *The mathematical Formulations of the Finite Elements Method with Applications to the Partial Differential Equations.* Academic Press, New York, 1972, pp. 5–359.

[2] A. Caboussat, R. Glowinski: *Numerical solution of a variational problem arising in stress analysis: The vector case.* Discret. Contin. Dyn. Syst. **27** (2010), pp. 1447–1472.

[3] E. Christiansen: *Limit analysis of colapse states.* In P.G. Ciarlet, J.L. Lions (eds): *Handbook of Numerical Analysis*, Vol IV, Part 2, North-Holland, 1996, pp. 195–312.

[4] M. Costabel, M. Dauge: *On the inequalities of Babuška-Aziz, Friedrichs and Horgana-Payne.* Arch. Ration. Mech. Anal. **217** (2015), pp. 873–898.

[5] J. Haslinger, S. Repin, S. Sysala: *A reliable incremental method of computing the limit load in deformation plasticity based on compliance: Continuous and discrete setting.* Journal of Computational and Applied Mathematics **303** (2016), pp. 156–170.

[6] J. Haslinger, S. Repin, S. Sysala: *Guaranteed and computable bounds of the limit load for variational problems with linear growth energy functionals.* Applications of Mathematics **61** (2016), pp. 527–564.

[7] O.A. Ladyzenskaya, V.A. Solonnikov: *Some problems of vector analysis, an generalized formulations of boundary value problems for the Navier-Stokes equation.* Zap. Nauchn. Semin. LOMI, **59**, 1976, pp. 81–116.

[8] L.E. Payne: *A bound for the optimal constant in an inequality of Ladyzhenskaya and Solonnikov.* IMA journal of applied mathematics **72** (2007), pp. 563–569.

[9] S. Repin: *A posteriori error estimates for the Stokes problem.* J. Math. Sci. **109**, No.5, 2002, pp. 1950–1964.

[10] S. Repin: *A Posteriori Estimates for Partial Differential Equations.* Walter de Gruyter, Berlin, 2008.

[11] S. Repin: *Estimates of the distance to the set of divergence free field.* Journal of Mathematical Sciences **210** (2015), pp. 822–834.

[12] S. Repin: *Estimates of the Distance to the Set of Solenoidal Vector Fields and Applications to A Posteriori Error Control.* Computational Methods in Applied Mathematics **15** (2015), pp. 515–530.

[13] S. Repin: *Localized forms of the LBB condition and a posteriori estimates for incompressible media problems. Mathematics and Computers in Simulation (2016).* In press.

[14] S. Sysala, J. Haslinger: *Truncation and indirect incremental methods in Hencky's perfect plasticity.* To appear in: Proceedings of ETAMM 2016: *Mathematical Modelling in Mechanics* (M. Sofonea, F. dell'Isola, D. Steigmann eds.), Springer series "Advanced Structured Materials".

[15] R. Temam. *Mathematical Problems in Plasticity.* Gauthier-Villars, Paris, 1985.

Integer programming for Vehicle Routing Problem

R. Sojka, V. Hapla, D. Horák

IT4Innovations, VŠB - Technical university of Ostrava
Department of applied mathematics, VŠB - Technical university of Ostrava

1 Introduction

The Vehicle Routing Problem (VRP) deals with a delivering goods from one or more depots to multiple customers by fleet of vehicles. The most studied type of VRP is the Capacited Vehicle Routing Problem (CVRP). In CVRP, each customer requires a certain amount of goods and each vehicle has its specified capacity. The aim is to minimize the total cost of the route. Determining the optimal solution is an NP-hard problem. The exact solution can be found only for a limited number of customers and vehicles. Therefore, research is mostly focused on heuristic and metaheuristic algorithms which can find the approximate solution in an acceptable time.

2 MILP formulation

VRP is most commonly formulated using the (mixed) Integer Linear Programming (ILP, MILP) model. VRP can be described by many ILP formulations. In this section, Vehicle Flow Model [1] is described.

Let us have the following parameters:

N: number of customers
V: number of available vehicles
d_i: demand of customer i
T_v: capacity of vehicle v
c_{ij}: distance between node i and node j

where the node with index 0 represents a depot and nodes with indices from 1 to $N+1$ are customers. Variables x and y are binary and variable u is real. Variable x_{ij}^v equals 1 iff the vehicle v travels on arc (i, j). Variable y_{ij} equals 1 iff the vehicle travels on arc (i, j). Our problem reads

$$\min \sum_{v=1}^{V} \sum_{i=0}^{N} \sum_{j=0}^{N} c_{ij} x_{ij}^v \tag{1}$$

$$\sum_{i=1}^{N} \sum_{j=0}^{N} d_i x_{ij}^v < T_v, \quad v = 1, 2..., V \tag{2}$$

$$\sum_{v=1}^{V} x_{ij}^v = y_{ij}, \quad i, j = 0, 1, ..., N \tag{3}$$

$$\sum_{j=0,j\neq i}^{N} y_{ij} = 1, \quad i = 1, 2, ..., N \tag{4}$$

$$\sum_{i=0,i\neq j}^{N} y_{ij} = 1, \quad j = 1, 2, ..., N \tag{5}$$

$$\sum_{j=1}^{N} y_{0j} \leq V \tag{6}$$

$$\sum_{i=1}^{N} y_{i0} \leq V \tag{7}$$

$$u_i - u_j + (N+1)y_{ij} \leq N, \quad 1 \leq i \neq j \leq N \tag{8}$$

$$\sum_{j=0,j\neq i}^{N} x_{ij}^{v} = \sum_{j=0,j\neq i}^{N} x_{ji}^{v}, \quad i = 1, 2, ..., N, \quad v = 1, 2, ..., N \tag{9}$$

Constraints (2) ensure that capacity of the vehicle is not exceeded. Constraints (3), (4) and (5) ensure exactly one visit for each arc and each customer by any vehicle. The number of available vehicles is specified by constraints (6), (7). Constraints (8) are subtour elimination constraints and constraints (9) ensure the continuity of the route. Other MILP or ILP formulations can be found in [2].

The exact solution of this problem can be obtained by using algorithms such as Branch-and-Bound, Column Generation, Branch-and-Cut or more recent Branch-and-Cut-and-Price algorithms.

3 Binary QP formulation

In [3], the Binary Quadratic Programming (BQP) formulation was introduced. There is three-index binary variable x_{ij}^{v} introduced in this formulation. Variable x_{ip}^{v} equals 1 iff vehicle v is located in node i at step p. This formulation leads to following minimization problem

$$\min \sum_{v=1}^{V} \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{p=1}^{N} c_{ij} x_{ip}^{v} x_{jp+1}^{v} \tag{10}$$

$$\sum_{v=1}^{V} \sum_{p=1}^{N} x_{ip}^{v} = 1, \quad i = 1, 2..., N \tag{11}$$

$$\sum_{i=0}^{N} x_{ip}^{v} = \sum_{i=0}^{N} x_{ip+1}^{v}, \quad p = 0, 1, ..., N, \quad v = 1, 2, ..., V \tag{12}$$

$$\sum_{i=1}^{N} d_i \sum_{p=1}^{N} x_{ip}^{v} \leq T_v, \quad v = 1, 2, ..., V. \tag{13}$$

Constraints (11) ensure that each customer is visited exactly once. The route continuity is ensured by (12) and constraints (13) ensure that the capacity of the vehicle is not be exceeded.

The main advantage of the BQP formulation is the reduction of number of constraints. Branch and Bound algorithms can be used for this problem. The bounds of the objective function can be obtained from the QP or SDP relaxation.

Acknowledgement: This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070); Project of major infrastructures for research, development and innovation of Ministry of Education, Youth and Sports with reg. num. LM2011033; by the internal student grant competition project SP2017/169 ”PERMON toolbox development III”.

References

[1] L. Bodin, B. Golden, A. Assad, M. Ball: *Routing and scheduling of vehicles and crews - the state of the art*, Vol. 10, Computers and Opertions Research, 1983, pp. 63–212.

[2] P. Toth, D. Vigo: *Vehicle Routing: Problems, Methods, and Applications*, Second Edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2014.

[3] P. Ji, H. Wu, Y. Wu: *Quadratic programming for the vehicle routing problem.* In: Proceedings of the 7th International Symposium on Operations Research and Its Applications (ISORA'08), 2008, pp. 82–90.

Computing zeros of analytic functions by integral contour method

E. Straková, P. Vodstrčil, D. Lukáš

Department of Applied Mathematics, FEECS & IT4Innovations, VŠB-TU Ostrava

1 Introduction

We present a numerical method for computing zeros of an analytic complex function. This abstract relies on Master thesis of the first author, which is a compilation of a series of papers of professor Sakurai and collaborators. We find the method particularly interesting for a forthcoming research on locating eigenvalues of large-scale matrices, which we mention in Conclusion.

Let Ω be a simply connected domain in $\mathbb{C}$, $f : \mathbb{C} \to \mathbb{C}$ be a function holomorphic in Ω and γ be a positively oriented curve in Ω that does not pass any zero of f. We consider the problem of finding all zeros of f in the interior of γ. We use an integral contour method which is referred to a concept relying on Cauchy's residue theorem. Denoting by $z_1, \dots, z_n$ the mutually distinct zeros and by $\alpha_1, \dots, \alpha_n \in \mathbb{N}$ their respective multiplicities, we can calculate following Newton sums by the residue theorem

$$s_k := \sum_{i=1}^{n} \alpha_i z_i^k = \frac{1}{2\pi i} \int_\gamma z^k \frac{f'(z)}{f(z)} \, \mathrm{d}z. \tag{1}$$

Note that $N := s_0$ is the total number of zeros. We translate the problem of finding zeros of f to searching for the same zeros of a polynomial

$$P_N(z) := z^N + \sigma_1 z^{N-1} + ... + \sigma_N,$$

the coefficients of which we get from Newton's identities

$$\begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ s_1 & 2 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ s_{N-2} & \cdots & s_1 & N-1 & 0 \\ s_{N-1} & \cdots & \cdots & s_1 & N \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_{N-1} \\ \sigma_N \end{pmatrix} = - \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_{N-1} \\ s_N \end{pmatrix}.$$

The method of Newton's identities is usually ill-conditioned due to bad conditioning of the polynomial, i.e., small changes in the polynomial coefficients generate large changes in the zeros. Moreover, the larger N the more accurate quadrature in (1) has to be employed.

2 Formal orthogonal polynomials

The method of formal orthogonal polynomials gives more accurate approximation of the zeros of analytic function. This method has been introduced in [1, 2].

Let $\mathcal{P}$ be the linear space of polynomials with complex coefficients. We introduce bilinear form $\langle \cdot, \cdot \rangle : \mathcal{P} \times \mathcal{P} \to \mathbb{C}$ by

$$\langle \phi, \psi \rangle := \frac{1}{2\pi i} \int_\gamma \phi(z) \psi(z) \frac{f'(z)}{f(z)} \, \mathrm{d}z = \sum_{i=1}^{n} \alpha_i \phi(z_i) \psi(z_i).$$

Polynomial φ_t of degree $t \geq 0$ is called a *formal orthogonal polynomial (FOP)* if it is monic and satisfies

$$\left\langle z^k, \varphi_t(z) \right\rangle = 0 \quad \text{for all } k \in \{0, 1, \cdots, t-1\}, \tag{2}$$

If φ_t is uniquely determined by (2), then it is referred to as a *regular FOP* and t is a *regular index.* The regular FOP of degree $t \geq 1$ exists if and only if matrix $H_t := (s_{p+q})_{p,q=0}^{t-1} = (\langle 1, z^{p+q} \rangle)_{p,q=0}^{t-1}$ is nonsingular. The zeros of a regular FOP are eigenvalues of matrix pencil $H_t^{(1)} - \lambda H_t$, where $H_t^{(1)} := (s_{p+q+1})_{p,q=0}^{t-1}$. The problem of finding zeros $z_1, \ldots, z_n$ of f is now equivalent to finding a regular FOP φ_n. The respective multiplicities $\alpha_1, \ldots, \alpha_n$ solves following Vandermonde system, for which there is a stable method relying on Newton polynomial interpolation,

$$\begin{pmatrix} 1 & \cdots & 1 \\ z_1 & \cdots & z_n \\ \vdots & & \vdots \\ z_1^{n-1} & \cdots & z_n^{n-1} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-1} \end{pmatrix}.$$

Unfortunately, computing eigenvalues of $H_n^{(1)} - \lambda H_n$ is still ill-conditioned. We replace it by a better conditioned problem of finding eigenvalues of pencil $M_n^{(1)} - \lambda M_n$, where $M_n := (\langle \varphi_p, \varphi_q \rangle)_{p,q=0}^{n-1}$ and $M_n^{(1)} := (\langle \varphi_p, \varphi_1 \varphi_q \rangle)_{p,q=0}^{n-1}$ with φ_k being suitable monic polynomials. In case of H_n strongly regular, i.e., all its leading principal submatrices are regular, then all of $\varphi_0, \varphi_1, ..., \varphi_n$ are regular FOPs. In this case M_n and $M_n^{(1)}$ are diagonal and tridiagonal, respectively. Otherwise, if H_n not being strongly regular, we establish a set of regular indices $\{i_k\}$, $k = 0, ..., K$, where K is the number of regular blocks in H_n. If $n \geq 1$, then $i_0 = 0, i_1 = 1$ and $i_K = n$. We define the sequence of $\{\varphi_t\}_{t=0}^{\infty}$ as follows: If t is a regular index, then φ_t is the regular FOP. Otherwise, $\varphi_t(z) := z^{t-r} \varphi_r(z)$, where r is the largest index less than t, and φ_t is called an *inner polynomial.* The polynomials can be grouped into blocks such that every block starts with the regular polynomial and the remaining polynomials in this block are inner.

In the practical algorithm there are two thresholds ε_{cond} and ε_{stop} with $\varepsilon_{stop} < \varepsilon_{cond}$. The value ε_{cond} determines the length of blocks and ε_{stop} decides whether the algorithm is being stopped, i.e. decides whether $r = n$ and $\varphi_r = \varphi_n$. If $|\langle \varphi_r, \varphi_r \rangle| \geq \varepsilon_{cond}$ for some regular index r, then φ_{r+1} is being generated as FOP (if r is regular index, then $|\langle \varphi_r, \varphi_r \rangle| = \det \frac{H_{r+1}}{H_r} = \det \frac{G_{r+1}}{G_r}\bigg)$. Else we search for the smallest t such that $t \leq N - 1 - r$ and $\left|\left\langle z^t \varphi_r, \varphi_r \right\rangle\right| > \varepsilon_{cond}$. Then $t + 1$ is being the length of block and t is being the number of inner polynomials in the block. If we fail to find such t and $\left|\left\langle z^t \varphi_r, \varphi_r \right\rangle\right| < \varepsilon_{stop}$ for all $t \in \{0, \cdots, N-1-r\}$, then $n = r$, compute the zeros of φ_n and algorithm will be stopped. If we fail to find such t and there exists some $t \in \{0, \cdots, N-1-r\}$ such that $\left|\left\langle z^t \varphi_r, \varphi_r \right\rangle\right| \geq \varepsilon_{stop}$, then the length of block will be m such that $|\langle z^m \varphi_r, \varphi_r \rangle| = \max\limits_{0 \leq t \leq N-1-r} \left|\left\langle z^t \varphi_r, \varphi_r \right\rangle\right|$ and algorithm continues until the last regular index r is less than N.

3 Numerical examples

We give two examples. First of all, we consider

$$f(z) := \prod_{j=1}^{10} \left(z - \frac{1}{2} j \right), \; \gamma(t) := \frac{11}{2} \mathrm{e}^{2\pi i t}.$$

By the method of Newton's identities we obtain the following approximation of zeros z_4 and z_9:

$$\begin{aligned} z_4 &= +2.203620388030727 + 0.4032628524238667i;\\ z_9 &= +4.328112677568831 + 0.0000315934022037i. \end{aligned}$$

After replacing Newton's identities by eigenvalue problem $H_{10}^{(1)} - \lambda H_{10}$ the approximations read

$$\begin{aligned} z_4 &= +2.375661289285481 - 1.945767526 \cdot 10^{-5}i;\\ z_9 &= +4.488679268520042 - 1.196308016 \cdot 10^{-6}ii. \end{aligned}$$

Finally, the method of FOPs gives the best approximation

$$\begin{aligned} z_4 &= +1.999999975589223 + 2.338471947 \cdot 10^{-7}i;\\ z_9 &= +4.499999938630227 - 4.909355538 \cdot 10^{-9}i. \end{aligned}$$

In the second example we consider

$$f(z) := \mathrm{e}^{3z} + 2z\cos(z) - 1,\ \gamma(t) := 2\mathrm{e}^{2\pi i t}.$$

If we set $\varepsilon_{cond} = 1$ and $\varepsilon_{stop} = 10^{-12}$, the algorithm decides that $n = 4$, generates φ_0, φ_1 as a regular FOP, φ_2 generates as an inner polynomial, φ_3 and φ_4 generates as a regular FOP. The approximation of the zeros is obtained as follows:

$$\begin{aligned} z_1 &= -\underline{1.84423395326221}6 - 3.189796250 \cdot 10^{-16}i;\\ z_2 &= +\underline{0.53089493029293}1 - \underline{1.33179187675112}3i;\\ z_3 &= +\underline{0.53089493029293}8 + \underline{1.33179187675112}8i;\\ z_4 &= -1.21 \cdot 10^{-14} - 5.681456752 \cdot 10^{-15}i. \end{aligned}$$

4 Conclusion

The presented method can be further extended towards finding eigenvalues of both linear and nonlinear operators, see the references. Our particular aim is to find eigenvalues λ_j of a large real and symmetric positive definite matrix $A \in \mathbb{R}^{M\times M}$ in a certain interval on the real axis, where we expect n eigenvalues, $n \ll M$. Given a (random) nonzero vector $v \in \mathbb{R}^M$ we search for poles of the function

$$f(z) := v(A - z\,I)^{-1}v = \sum_{j=1}^{M} \frac{\nu_j}{\lambda_j - z},$$

where ν_j are related to coordinates of v in an orthonormal basis of the eigenvectors of A. We can employ the integral contour method, see Fig. 1. However, each evaluation of f involves a solution to the large system with matrix A. Therefore, our further research shall address two essential problems:

- We shall find a suitable quadrature method to minimize the number of evaluations of f.
- We shall find a reasonable way to update the solver when perturbing the matrix diagonal.

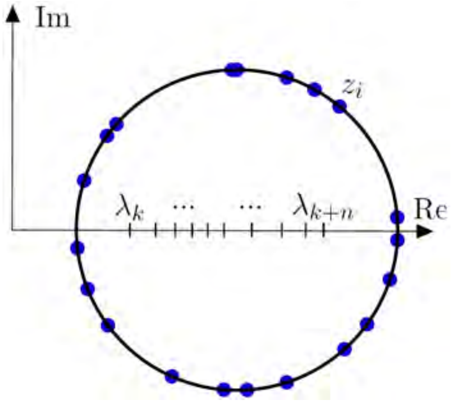


Figure 1: Integral contour method

References

[1] P. Kravanja, T. Sakurai, M. Van Barel: *On locating clusters of zeros of analytic functions.* BIT Numerical Mathematics 39 (1999), pp. 646–682.

[2] P. Kravanja M. Van Barel, A. Haegemans: *Computing zeros of analytic functions via modified moments based on formal orthogonal polynomials.* Department of Computer Science, K.U.Leuven, Leuven, Belgium,1996.

[3] T. Sakurai, H. Sugiura: *A projection method for generalized eigenvalueproblems using numerical integration.* J. Comp. Appl. Math. 159 (2003), pp. 119–128.

[4] T. Ikegami, T. Sakurai, U. Nagashima: *A filter diagonalization for generalized eigenvalue problems based on the Sakurai-Sugiura projection method.* J. Comp. Appl. Math. 33 (2009), pp. 1927–1936.

[5] T. Ikegami, T. Sakurai: *Contour integral eigensolver for non-Hermitian systems: a Rayleigh-Ritz-type approach.* Taiwan. J. Math. 14 (2010), pp. 825–837.

[6] J. Asakura, T. Sakurai, H. Tadano, T. Ikegami, K. Kimura: *A numericalmethod for nonlinear eigenvalue problems using contour integrals.* JSIAM Letters 1 (2009), pp. 52–55.

[7] T. Sakurai, Y. Futamura, H. Tadano: *Efficient parameter estimation and implementation of a contour integral-based eigensolver.* Journal of Algorithms and Computational Technology 7 (2013), pp. 249–269.

Incremental estimation of the largest and smallest Ritz values in the conjugate gradient method

P. Tichý

Faculty of Mathematics and Physics, Charles University in Prague

The (preconditioned) Conjugate Gradient (CG) method by Hestenes and Stiefel [2] is the iterative method of choice for solving linear systems $Ax = b$ with a real symmetric positive definite matrix A. During the process, it is often desirable to get some information about $\lambda_{\min}$ and $\lambda_{\max}$, the smallest and the largest eigenvalue of A. The information about the eigenvalues can then be used, e.g., to approximate norms of errors, to estimate the ultimate level of accuracy, or, to estimate the condition number of A. Note that since A is symmetric and positive definite, it holds that $\lambda_{\max} = \|A\|$ and $\lambda_{\min}^{-1} = \|A^{-1}\|$.

A natural way to approximate $\lambda_{\min}$ and $\lambda_{\max}$ during the CG computations is to use minimum and maximum Ritz values that can be determined from the CG coefficients. The straightforward approach would require storing the tridiagonal Jacobi matrices and computing their eigenvalues. Such an approach would be expensive with increasing iterations. In this contribution we generalize results of [4] and present a very simple way to approximate the maximum and the minimum Ritz value incrementally at a negligible cost.

The Lanczos and the CG algorithms. We briefly recall the Lanczos and Conjugate Gradient algorithms as well as their relationships; see, for instance [3].

Algorithm 3 Lanczos algorithm

```
input $A$, $v$
$\beta_0 = 0$, $v_0 = 0$
$v_1 = v/\|v\|$
for $k = 1, \ldots$ do
    $w = Av_k - \beta_{k-1}v_{k-1}$
    $\alpha_k = v_k^T w$
    $w = w - \alpha_k v_k$
    $\beta_k = \|w\|$
    $v_{k+1} = w/\beta_k$
end for
```

Given a starting vector v and a symmetric matrix $A \in \mathbb{R}^{N\times N}$, one can consider a sequence of nested subspaces

$$\mathcal{K}_k(A, v) \equiv \operatorname{span}\{v, Av, \ldots, A^{k-1}v\}, \quad k = 1, 2, \ldots,$$

called Krylov subspaces. The dimension of these subspaces is increasing up to an index n called the grade of v with respect to A, at which the maximum dimension is attained, and $\mathcal{K}_n(A, v)$ is invariant under multiplication with A. Assuming that $k < n$ the Lanczos algorithm computes an orthonormal basis $v_1, \ldots, v_{k+1}$ of the Krylov subspace $\mathcal{K}_{k+1}(A, v)$. The basis vectors v_j satisfy the matrix relation

$$AV_k = V_k T_k + \beta_{k+1} v_{k+1} e_k^T$$

where $V_k = [v_1 \cdots v_k]$ and T_k is the $k \times k$ symmetric tridiagonal matrix of the recurrence coefficients computed in Algorithm 3:

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & & \\ & & \ddots & \beta_{k-1} \\ & & \beta_{k-1} & \alpha_k \end{bmatrix}.$$

The coefficients β_j being positive, T_k is a Jacobi matrix. The Lanczos algorithm works for any symmetric matrix, but if A is positive definite, then T_k is positive definite as well.

Algorithm 4 Conjugate gradients

```
input $A$, $b$, $x_0$
$r_0 = b - Ax_0$
$p_0 = r_0$
for $k = 1, \ldots, n$ do
    $\gamma_{k-1} = \frac{r_{k-1}^T r_{k-1}}{p_{k-1}^T A p_{k-1}}$
    $x_k = x_{k-1} + \gamma_{k-1} p_{k-1}$
    $r_k = r_{k-1} - \gamma_{k-1} A p_{k-1}$
    $\delta_k = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}}$
    $p_k = r_k + \delta_k p_{k-1}$
end for
```

When solving a system of linear algebraic equations $Ax = b$ with symmetric and positive definite matrix A, the CG method (Algorithm 4) can be used. CG computes iterates x_k that are optimal since the A-norm of the error is minimized over the manifold $x_0 + \mathcal{K}_k(A, r_0)$. The residual vectors r_k are proportional to the Lanczos basis vectors v_j and hence mutually orthogonal. It is well-known (see, for instance [3]) that the recurrence coefficients computed in both algorithms are connected via formulas which can be written in the matrix form as $T_k = R_k^T R_k$, where

$$R_k = \begin{bmatrix} \frac{1}{\sqrt{\gamma_0}} & \sqrt{\frac{\delta_1}{\gamma_0}} & & \\ & \ddots & \ddots & \\ & & \ddots & \sqrt{\frac{\delta_{k-1}}{\gamma_{k-2}}} \\ & & & \frac{1}{\sqrt{\gamma_{k-1}}} \end{bmatrix}.$$

In other words, CG computes implicitly the Cholesky factorization of the Jacobi matrix T_k generated by the Lanczos algorithm. Therefore, it holds that

$$\|T_k\| = \|R_k\|^2, \qquad \|T_k^{-1}\| = \|R_k^{-1}\|^2,$$

and to approximate $\|T_k\|$ and $\|T_k^{-1}\|$, one can use algorithms that incrementally estimate the maximum singular values of the upper triangular matrices R_k and R_k^{-1}.

Incremental estimation. Consider the problem of incremental estimation of norms of upper triangular matrices extended in each update by one column; see, e.g., [1]. Let $R \in \mathbb{R}^k$ be an upper triangular matrix and let z be its approximate maximum right singular vector. Let

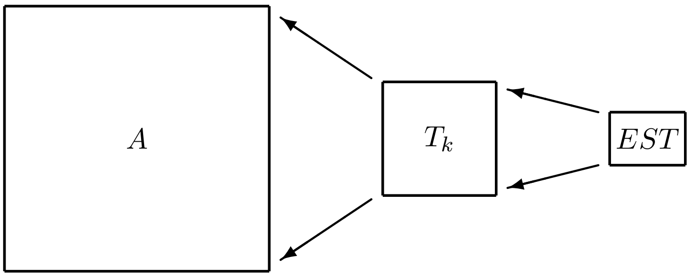
$$\hat{R} = \begin{bmatrix} R & v \\ & \mu \end{bmatrix}, \qquad v \in \mathbb{R}^k, \quad \mu \in \mathbb{R}.$$

We consider the new approximate maximum right singular vector in the form $\hat{z} = \left[\begin{array}{cc} sz & c \end{array} \right]^T$, where $s^2 + c^2 = 1$ are chosen such that the norm of the vector

$$\hat{R}\hat{z} = \left[\begin{array}{c} sRz + cv \\ c\mu \end{array} \right]$$

is maximum. The numbers s, c, and the corresponding maximum norm of $\hat{R}\hat{z}$ can easily be determined from the eigenvalue problem for a 2×2 symmetric matrix having the entries $\|Rz\|^2$, $v^T Rz$, and $v^T v + \mu^2$.

Specialization to bidiagonal matrices and their inverses. In general, the previous technique requires to store the matrix R, the vector z, and to perform $\mathcal{O}(k^2)$ operations per update. In this presentation, we apply the previous technique to the upper triangular matrices R_k and R_k^{-1}, which are available in CG. Thanks to their special structure (R_k is bidiagonal, R_k^{-1} is semiseparable) we will show that the incremental estimates of $\|R_k\|$ and $\|R_k^{-1}\|$ can be computed in $\mathcal{O}(1)$ operations, without storing the corresponding matrices and the approximate maximum right singular vectors. In other words, in each iteration of CG we just need to update a few scalars to compute incremental estimates of $\|R_k\|$ and $\|R_k^{-1}\|$. Note that the incremental estimates provide a lower bound on the largest Ritz value and an upper bound on the smallest Ritz value.



The estimators have been suggested to be computationally cheap and efficient, but, on the other hand, one cannot expect a very high relative accuracy of the estimates. Numerical experiments predict that the incremental estimates agree with the approximated Ritz values to 2 or 3 valid digits. If one needs to improve the accuracy, one can store R_k and compute, from time to time, a better approximation of the minimum and the maximum right singular vector.

Acknowledgments. This work was supported by the GACR grant P201/13-06684 S.

References

[1] I.S. Duff, C. Vömel: *Incremental norm estimation for dense and sparse matrices*, BIT, 42 (2002), pp. 300–322.

[2] M.R. Hestenes, E. Stiefel: *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), 1953, pp. 409–436.

[3] G. Meurant: *The Lanczos and conjugate gradient algorithms*, vol. 19 of Software, Environments, and Tools, SIAM, Philadelphia, PA, 2006.

[4] P. Tichý: *On error estimation in the conjugate gradient method: Normwise backward error*, in Proceedings of Algoritmy 2016, 2016, pp. 323–332.

Parallelization of the PragTic software for an automated fatigue damage calculation

J. Tomčala, M. Pecha

IT4Innovations National Supercomputing Centre, VŠB - Technical university of Ostrava

1 Introduction

Repeated service loading of machine parts leads to the reduction of their load-carrying capacity and often to the fatigue failure. Because the experimental verification of their long-term functionality is very expensive and time consuming, the engineers are interested in virtual modelling and simulating fatigue failure. The quality of these simulations is limited by:

- reliability of phenomenological models - used prediction criteria, cyclic plasticity model
- involved numerical methods, algorithms, and their implementations
- computational resources being at disposal for the simulations.

Fatigue analysis requires very accurate results of the stress-strain calculation at critical locations of the finite element (FE) model. After results are obtained, the fatigue analysis takes place, being realized usually in another software. One of available tools for computational fatigue analysis is PragTic.

PragTic software is developed at CTU in Prague. It serves as a fatigue analysis tool using the computation results of the FE-solution. It is provided as a freeware on PragTic1. Many new fatigue criteria and approaches have been implemented into this code during last fifteen years. PragTic also has a direct connection to the large material and experimental databases FatLim (Fatigue Limits) and FinLiv (Finite Lives). The major focus of PragTic is multiaxial fatigue analysis. It is very important in cases where more load channels act on the component simultaneously. The analysis is therefore usually realized by evaluating stress or strain components on various planes in the analyzed point. The damage parameter built from these components is either maximized over all possible planes or integrated over them. Obtaining the single equivalent stress or strain by some criterion gets complicated, because many candidate planes have to be evaluated. In addition to it, the computation time is increased even more due the need to correctly evaluate shear stress or strain components in cases of non-proportional loading, when the shear stress or strain vector rotates. The current usual solution is to compute the position and radius of the minimum circle circumscribed to the shear stress vector tip trajectory in each load cycle.

The major feature of the PragTic software is the multiaxial fatigue analysis. This analysis type is used if multiple load channels simultaneously act on a mechanical component. In these cases, the analysis process is usually proceeded by the stress-strain component evaluation on various planes, which go through the analyzed point. Then, the damage parameter is set up from these components and further processed as follows: it is maximized or integrated over all evaluated planes [1].

The PragTic software predicts mechanical fatigue failure by generating large-scale computational simulations. This approach demands extensive computational resources and large amount of the time. By the parallelization of this software at the various levels, we can save a lot of the run time. It allows to find much more detailed solution. In this paper we present the parallelization at the node level.

2 Former version and its inefficiency

Former version of the PragTic software is accessing disc very often through the simulations. Therefore, it wastes huge amount of the run time by writing, reading, writing, reading, . . . etc. of partial results to/from disc. The PragTic uses files on disc as the temporary buffers, which cause significant slowdown of the program. These temporary buffer files are erased at the end of the run.

Development of the PragTic fatigue solver was started by Jan Papuga in 2000, and various institutions got involved throughout the past years - above all the Czech Technical University, the Evektor company, or VŠB-Technical University of Ostrava. Since the last two decades, a whole range of new criteria and approaches on fatigue analysis have been implemented. Nowadays, it is distributed as a freeware application, and it can be downloaded from the PragTic webpages [3].

3 Main idea

The PragTic sequentially generates simulations node by node. Main idea of our first parallelization was to assign the particular subset of nodes to every parallel process, then compute partial solutions and finally merge the results into the one result file. If we use for example 192 processor cores, then the theoretical maximum speed up would be 192. Of course, there is always the run time overhead cost, so the real speed up is lower. It is caused by distributing the particular subsets of nodes at the beginning and then merging the results after the partial computations.

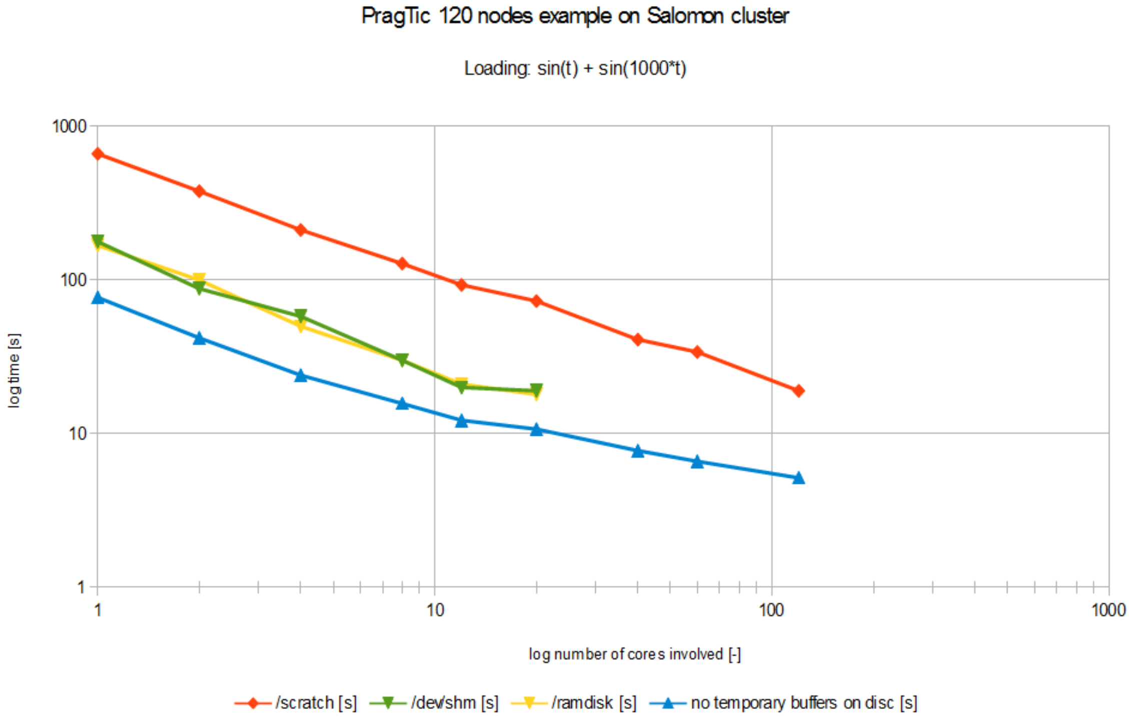
Further significant speed up can be reached by not saving the temporary buffers to the disc, but keeping them just in the memory. It is a nowadays approach compared to the 20 years old one, when computers had significantly less operating memory than today.

4 Implementation and results

First step was to compile the PragTic on Salomon cluster. Then we have parallelized the PragTic using MPI and got the first results. And finally we optimized temporary buffers operations, which was the hardest part, because this optimization required changes at circa 500 lines of the PragTic's source code.

We have chosen to test the small benchmark of 120 nodes on the Salomon cluster. The loading was sin(t)+ sin(1000*t) where t means time.

The result run times are shown in the Figure 1. The red color represents run time of the version saving temporary buffers to the real disc (former version of the PragTic), the green color represents run time of the version saving temporary buffers to the shared memory (/dev/shm) and the yellow color represents run time of the version saving temporary buffers to the ramdisk (/ramdisk). As the compute nodes do not see each other's shared memory and ramdisk, their



Processor cores	*1*	*2*	*4*	*8*	*12*	*20*	*40*	*60*	*120*
/scratch [s]	663	379	212	128	93	73	41	34	19
/dev/shm [s]	178	88	58	30	20	19			
/ramdisk [s]	168	100	50	30	21	18			
no temporary buffers on	77	42	24	15,7	12,2	10,7	7,75	6,6	5,15

Figure 1: Result run times for 120 nodes simple example

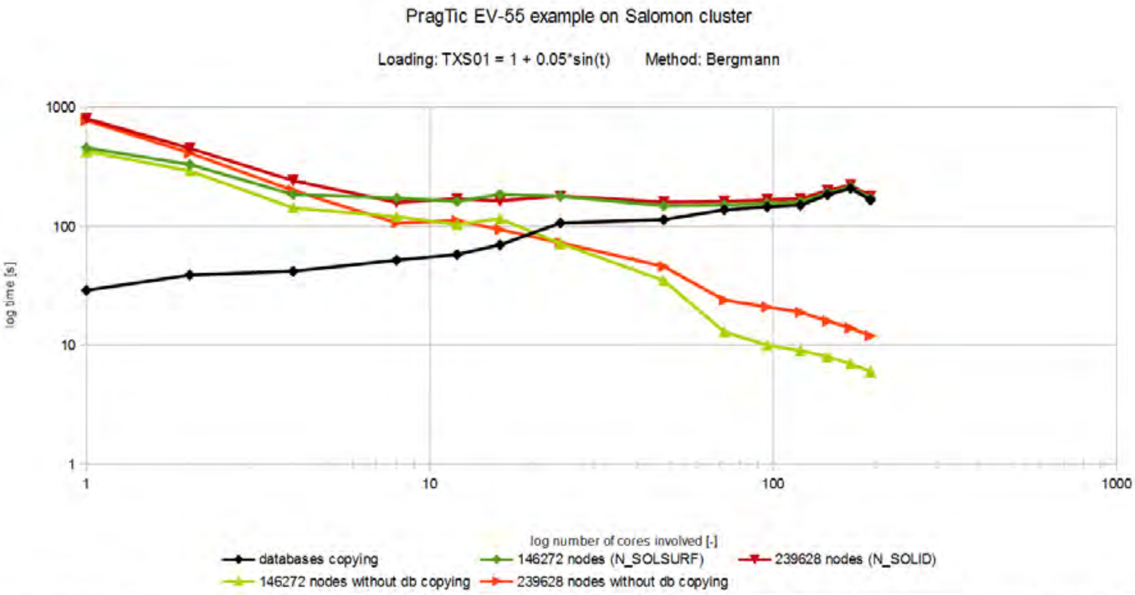
run time graph line ends at 20 processor cores. The Salomon's compute nodes contain just 24 processor cores [2]. The blue color represents run times of the optimized version with no temporary buffers on disc (they are just in memory).

Another example is EV-55 airplane [7]. Its result run times are shown in the Figure 2. Black color represents database copying time, which appeared as the main issue in parallelization of this example. This figure also contains graph lines with lighter colors, they represents run times without delay caused by database copying.

5 Conclusion

As it can be seen from the results, by the parallelization and optimization of the PragTic we reached speed up 128 on 120 processor cores in first, simple 120 nodes example and speed up 64 on 192 processor cores in 239628 nodes EV-55 airplane example. We also can see, that using bigger databases causes significant delay and that without this delay our version scales well.

It is obvious, that much higher speed up could be reached on bigger problem with more processor cores. We also expect much higher speed up if we will be successful with the plane and method parallelization.



Processor cores	*1*	*2*	*4*	*8*	*12*	*16*	*24*	*48*	*72*	*96*	*120*	*144*	*168*	*192*
databases copying	29	39	42	52	58	70	107	114	138	146	151	184	208	167
146272 nodes (N_SOLSURF)	457	331	185	173	162	186	179	149	151	156	160	192	215	173
239628 nodes (N_SOLID)	806	453	242	159	170	164	180	160	162	167	170	200	222	179
146272 nodes without db copying	428	292	143	121	104	116	72	35	13	10	9	8	7	6
239628 nodes without db copying	777	414	200	107	112	94	73	46	24	21	19	16	14	12

Figure 2: Result run times for EV-55 airplane example

Acknowledgement: This work has been supported by the grant of the Czech Science Foundation (GACR) project no. 15-18274S.

References

[1] J. Papuga, R. Halama, M. Fusek, J. Rojíček, F. Fojtík, D. Horák, M. Pecha, J. Tomčala, M. Čermák, V. Hapla, R. Sojka, J. Kružík: *Efficient Lifetime Estimation Techniques for General Multiaxial Loading.* Proceedings ICNAAM 2016 conference, Rhodos, 19.-26.7.2016.

[2] IT4Innovations: *National Supercomputing Center, VSB-Technical University of Ostrava, Salomon Cluster Documentation – Hardware Overview.* https://docs.it4i.cz/salomon-cluster-documentation/hardware-overview, 2016.

[3] *PragTic Project.* http://www.pragtic.com.

[4] J. Papuga, M. Lutovinov: *Help for FinLiv.VBA Excel Database.* [FAD/12/001, ver. B]. FME CTU in Prague and Evektor, spol. s r.o., Prague 2014.

[5] R. Halama, M. Fusek, M. Šofer, M. Poruba, Z. Matušek, P. Fajkoš: *Ratcheting Behavior of Class C Wheel Steel and Its Prediction by Modified AbdelKarim-Ohno Model.* In Proceedings of the 10th International Conference on Contact Mechanics CM2015, Colorado Springs, Colorado, USA, August 30 – September 3, 2015.

[6] *PERMON webpages.* http://permon.it4i.cz/.

[7] *Evektor EV-55 airplane webpages.* http://www.evektor.cz/cz/ev-55-outback.

3D simulation of a wheel tracker test of asphalt concrete described by the Burgers model

K. Tůma

Mathematical Institute, Faculty of Mathematics and Physics, Charles University in Prague

1 Introduction

Asphalt concrete is important for its usage in the construction of the roads, highways or runways. When the cars are running over the surface of the road, the material is being repeatedly compressed. Therefore, it is important to study how the response of the material depends on the applied load and its speed.

This contribution reflects joint research with V. Průša, J. Málek and J.M. Krishnan who studied the response of the asphalt concrete in a wheel tracker test. The experiment was performed for asphalt concrete confined with 200 kPa and containing 2% of air voids. In this abstract we present a numerical simulation of this experiment described by th Burgers model.

2 Description of the experiment

In the experiment, that is being simulated, the sample of the shape of a brick with the dimensions 30 cm × 13.8 cm × 5 cm is subject to the applied stress that starts at $t = 0$ at the top of the brick according to Figure 1 and it moves to the right and then back (this is one cycle) with a constant velocity. The dimensions of the contact area are equal to 2.5 cm × 6 cm and 1 000 cycles are performed. The experiment is performed with two different applied stresses 540 kPa and 800 kPa moving with two different speeds 1 km/h and 10 km/h. The experimental setup is depicted in Figure 1, the dashed line shows the trajectory of the applied stress.

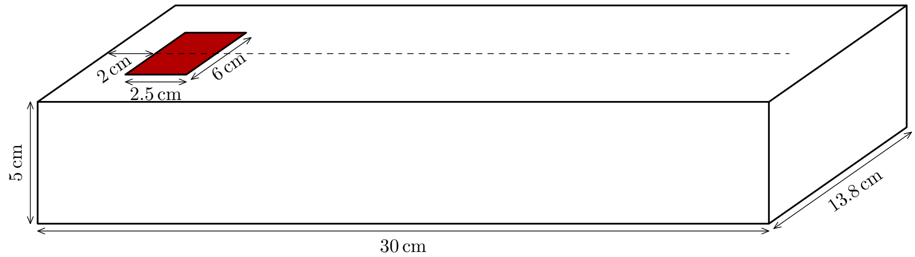


Figure 1: Schematic description of the experimental setup.

3 Mathematical model

Asphalt concrete is simulated by the viscoelastic fluid-like Burgers model that is capable of capturing two different relaxation mechanisms that appear in asphalt concrete. It is assumed that the density ρ of the material is constant, then the balance of mass and balance of linear and angular momentum are in the form

$$\operatorname{div}\mathbf{v} = 0, \tag{1}$$

$$\rho\left(\frac{\partial\mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\mathbf{v}\right) = \operatorname{div}\mathbf{T}, \quad \mathbf{T} = \mathbf{T}^{\mathrm{T}}, \tag{2}$$

where $\mathbf{v}$ is the fluid velocity. Next, $\mathbf{T}$ is the symmetric Cauchy stress tensor in the form

$$\mathbf{T} = -p\mathbf{I} + \mu_{\mathrm{s}}\left(\nabla\mathbf{v} + (\nabla\mathbf{v})^{\mathrm{T}}\right) + G_1(\mathbf{B}_1 - \mathbf{I}) + G_2(\mathbf{B}_2 - \mathbf{I}), \tag{3}$$

where μ_{s} is the solvent viscosity, G_1 and G_2 are the elastic moduli and $\mathbf{B}_1, \mathbf{B}_2$ are the additional stress tensors. They satisfy the set of evolution differential equations (fully coupled through the velocity field $\mathbf{v}$)

$$\overset{\triangledown}{\mathbf{B}}_1 + \frac{1}{\tau_1}(\mathbf{B}_1 - \mathbf{I}) = 0, \tag{4}$$

$$\overset{\triangledown}{\mathbf{B}}_2 + \frac{1}{\tau_2}(\mathbf{B}_2 - \mathbf{I}) = 0, \tag{5}$$

where $\overset{\triangledown}{\mathbf{B}} = \dfrac{\partial\mathbf{B}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{B} - (\nabla\mathbf{v})\mathbf{B} - \mathbf{B}(\nabla\mathbf{v})^{\mathrm{T}}$ is the objective upper convected Oldroyd derivative and τ_1, τ_2 are two relaxation times describing two different relaxation mechanisms of the material.

It is worth mentioning that the Burgers model can be derived using the framework of thermomechanics of continuum based on two notions that assure that the second law of thermodynamics is automatically satisfied. The first notion is the principle of the maximum rate of entropy production and the other one is the natural configuration that splits the total deformation into the dissipative part and the purely elastic part that corresponds to that of the compressible neo-Hookean solid. For more details see [1].

We obtain the material parameters $\mu_{\mathrm{s}}, G_1, G_2, \tau_1, \tau_2$ by comparing the predictions of the model to the simple compression experiment. The fitting procedure is based on the minimization of the difference between the measured experimental data and the numerical simulation, for more details see [2, 3]. Fitted material parameters $\mu_{\mathrm{s}} = 2.81\,\mathrm{MPa\ s}$, $G_1 = 52.4\,\mathrm{MPa}$, $G_2 = 9.81\,\mathrm{MPa}$, $\tau_1 = 101.3\,\mathrm{s}$, $\tau_2 = 109.3\,\mathrm{s}$ are then used in the simulation of the wheel tracker test.

4 Numerical implementation

In the present experiment the top boundary of the asphalt concrete brick was deforming. In order to simulate it, the arbitrary Lagrangian-Eulerian (ALE) method is employed. By using a new unknown — arbitrary deformation $\hat{\mathbf{u}}$ — the standard weak formulation in deforming Eulerian domain Ω_x is transformed to a fixed ALE domain Ω_χ. It is assumed that all points on the boundaries are material points, i.e. the time derivative of $\hat{\mathbf{u}}$ is equal to the fluid velocity $\mathbf{v}$. For more details on the transformation to the ALE domain see [4, 5], where this method is applied in two spacial dimensions.

In three dimensional space the weak formulation in $\Omega_\chi \in \mathbb{R}^3$ is in the form

$$\hat{\mathbf{F}} = \mathbf{I} + \nabla_\chi\hat{\mathbf{u}}, \quad \hat{J} = \det\hat{\mathbf{F}}, \quad \int_{\Omega_\chi} \hat{J}\,\mathrm{tr}\left((\nabla_\chi\mathbf{v})\hat{\mathbf{F}}^{-1}\right) q\,\mathrm{d}\chi = 0,$$

$$\int_{\Omega_\chi} \hat{J}\rho\left[\frac{\partial\mathbf{v}}{\partial t} + (\nabla_\chi\mathbf{v})\left(\hat{\mathbf{F}}^{-1}\left(\mathbf{v} - \frac{\partial\hat{\mathbf{u}}}{\partial t}\right)\right)\right]\cdot\mathbf{q}\,\mathrm{d}\chi + \int_{\Omega_\chi} \hat{J}\hat{\mathbf{T}}\hat{\mathbf{F}}^{-\mathrm{T}}\cdot\nabla\mathbf{q}\,\mathrm{d}\chi = \int_{\partial\Omega_\chi} \hat{\mathbf{t}}_n\cdot\mathbf{q}\,\mathrm{d}S_\chi,$$

$$\hat{\mathbf{T}} = -p\mathbf{I} + \mu_{\mathrm{s}}\left((\nabla_\chi\mathbf{v})\hat{\mathbf{F}}^{-1} + \hat{\mathbf{F}}^{-\mathrm{T}}(\nabla_\chi\mathbf{v})^{\mathrm{T}}\right) + G_1(\mathbf{B}_1 - \mathbf{I}) + G_2(\mathbf{B}_2 - \mathbf{I}),$$

$$\int_{\Omega_\chi} \hat{J} \left[\frac{\partial \mathbf{B}_i}{\partial t} + (\nabla_\chi \mathbf{B}_i)\left(\hat{\mathbf{F}}^{-1}\left(\mathbf{v} - \frac{\partial \hat{\mathbf{u}}}{\partial t}\right)\right) - (\nabla_\chi \mathbf{v})\hat{\mathbf{F}}^{-1}\mathbf{B}_i - \mathbf{B}_i\hat{\mathbf{F}}^{-\mathrm{T}}(\nabla_\chi \mathbf{v})^{\mathrm{T}} + \frac{1}{\tau_i}(\mathbf{B}_i - \mathbf{I})\right] \cdot \mathbf{Q}_i \,\mathrm{d}\chi = 0, \;\; i = 1, 2,$$

$$\int_{\Omega_\chi} \nabla_\chi \hat{\mathbf{u}} \cdot \nabla \mathbf{r} \,\mathrm{d}\chi = 0,$$

which holds for all admissible test functions $q, \mathbf{q}, \mathbf{Q}_1, \mathbf{Q}_2$ and $\mathbf{r}$. The vector $\hat{\mathbf{t}}_n$ is used for prescribing the time-dependent Neumann boundary condition, i.e. for prescribing the moving compression on the top side of the domain. All other sides of the domain are fixed, i.e. described by zero Dirichlet boundary conditions for $\mathbf{v}$ and $\hat{\mathbf{u}}$.

The numerical implementation is based on this weak formulation and is has been performed using *AceGen/AceFEM* system [6, 7]. *AceGen* is a code generation system and *AceFEM* is a finite element environment that uses the generated code. The system provides the automatic differentiation to compute the exact tangent matrix from the residuum which implies the efficient and robust implementation of the Newton solver.

Due to the symmetry of the problem with respect to the dashed line in Figure 1 only one half is computed. The symmetric part of Ω_χ is discretized by regular hexahedra, pressure p, parts of the Cauchy stress tensor $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are approximated by piecewise discontinuous linear $\mathrm{P1}^{\mathrm{disc}}$ elements, the velocity $\mathbf{v}$ is approximated by piecewise triquadratic H2 elements, and in order to decrease the size of the problem the arbitrary deformation $\hat{\mathbf{u}}$ is approximated by piecewise trilinear H1 elements. The time derivatives are approximated by backward Euler method, nonlinearities are solved with the Newton method and the consequent set of linear equations are solved with the iterative CGS solver with a LU decomposition used as a constant preconditioner (MKL Pardiso). The linear iterations are stopped when the relative residuum reaches 10^{-4} and the stopping condition for the Newton iterations is 10^{-9}.

5 Results

The results are computed on the mesh containing 1 680 hexahedra and described by 88 052 degrees of freedom. The problem is calculated parallely with 24 threads on the system with two Intel Xeon E5-2620 v2, the typical time of the assembly of the residuum and the tangent matrix is 0.85 s, LU decomposition that is needed only once takes 5.8 s and because the solution between two Newton iterations is not changing a lot, usually it is enough to perform only two CGS iterations which take 0.13 s. One compression cycle is approximated by 200 time steps, hence all together 200 000 time steps has to be performed to compute the whole simulation.

Figure 2 shows the snapshot in the 1000^{th} cycle at the time when the 800 kPa compression is at the top in the middle going to the left with the lower speed 1 km/h. The pressure is localized mainly under the compression area. The cumulated deformation of the domain is very small and thus the dependence of the deformation on the cycle number Figure 3a) and also coordinate x Figure 3b) are plotted. The graph in 3b) shows that the upper side is mostly depreciated at $x \doteq 16\,\mathrm{cm}$ which is on the right from the middle and which shows the inertia of the material. The graph in 3a) shows that the deformation of the material is bigger when it is pressed with higher stress or when it moves with a slower speed.

In the next step of our research the simulation will be compared to the experimental results.

Acknowledgement: This research is supported by the ERC-CZ project LL1202.

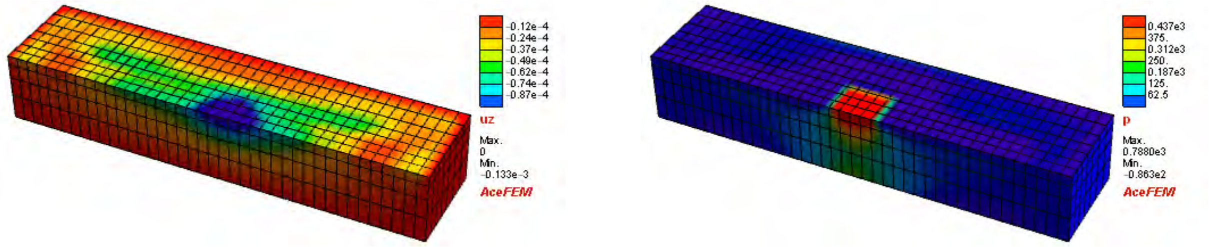
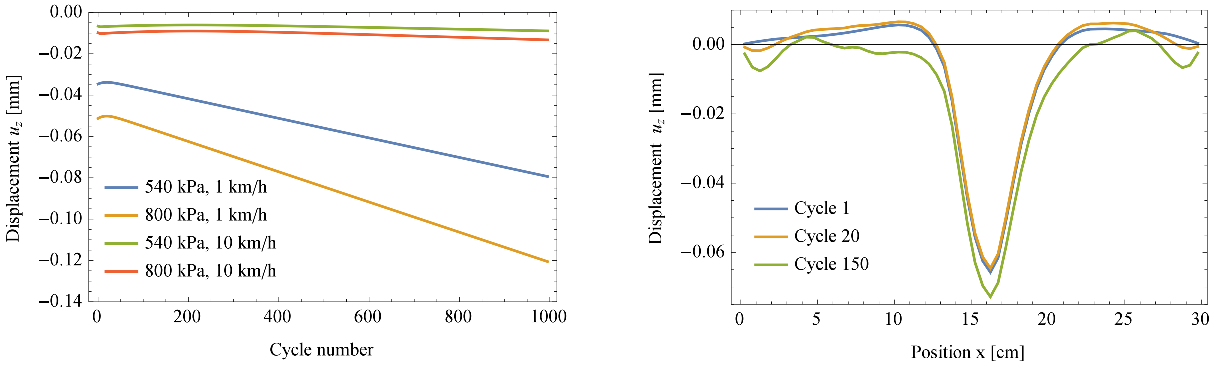


Figure 2: A snapshot of the asphalt concrete pressed with 800 kPa and the speed 1 km/h in cycle 1000 going to the left, displacement u_z [m] (left) and pressure p [kPa] (right).



a) b)

Figure 3: Graph of the dependence of the displacement u_z on the number of the cycle measured at the top in the middle for different speeds and applied pressures (left), and the dependence of the displacement u_z on the position x obtained for the applied pressure 800 kPa and the speed 1 km/h (right).

References

[1] J. Málek, K.R. Rajagopal, K. Tůma: *On a variant of the Maxwell and Oldroyd-B models within the context of a thermodynamic basis.* Int J Nonlinear Mech 76, 2015, pp. 42–47.

[2] J. Hron, J. Kratochvíl, J. Málek, K.R. Rajagopal, K. Tůma: *A thermodynamically compatible rate type fluid to describe the response of asphalt.* Math Comput Simulat 82(10), 2012, pp. 1853–1873.

[3] J. Málek, K.R. Rajagopal, K. Tůma: *A thermodynamically compatible model for describing the response of asphalt binders.* Int J Pavement Eng 16(4), 2015, pp. 297–314.

[4] J. Hron, K.R. Rajagopal, K. Tůma: *Flow of a Burgers fluid due to time varying loads on deforming boundaries.* J Nonnewton Fluid Mech 210, 2014, pp. 66–77.

[5] J. Málek, K.R. Rajagopal, K. Tůma: *A thermodynamically compatible model for describing asphalt binders: solutions of problem.* Int J Pavement Eng 17(6), 2016, pp. 550–564.

[6] J. Korelc: *Multi-language and multi-environment generation of nonlinear finite element codes.* Eng Comput 18, 2002, pp. 312–327.

[7] J. Korelc: *Automation of primal and sensitivity analysis of transient coupled problems.* Comput Mech 44, 2009, pp. 631–649.

Lower bounds on eigenvalues of linear elliptic operators

T. Vejchodský

Institute of Mathematics of the CAS, Prague

1 Introduction

Eigenvalue problems for linear second-order partial differential elliptic operators are often solved by the conforming finite element method. This approach is a special case of the standard Galerkin method and, thus, it yields approximate eigenvalues that are guaranteed to be greater or equal to the exact eigenvalues. The natural question how to compute guaranteed lower bounds on eigenvalues is much more difficult to answer. The problem of lower bounds has been studied for decades and approached by many authors. Lower bounds of Weinstein [5] and Kato [1] belong among the oldest.

In this contribution, we recall these two classical lower bounds and discuss their properties. A straightforward application of these two bounds in the context of the finite element method can be problematic, because the differential operator has to be applied in the strong (point-wise) form. Therefore, we generalized Weinstein's and Kato's bounds to the weak setting [4], which can be easily implemented within the standard conforming finite element method.

2 Weinstein's bound

Let us consider a Hilbert space V, its dense subset $D(A)$ and a linear symmetric operator $A : D(A) \to V$. Eigenfunctions $u_i \in D(A) \setminus \{0\}$ and the corresponding eigenvalues $\lambda_i \in \mathbb{R}$ of A satisfy

$$Au_i = \lambda_i u_i.$$

We assume that eigenvalues form a countable sequence $0 < \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \cdots$ and that the corresponding eigenfunctions $\{u_i\}$ form an orthonormal basis in V.

Theorem 1 (Weinstein 1934). *Let $u_* \in D(A) \setminus \{0\}$ and $\lambda_* \in \mathbb{R}$ be arbitrary. Let $\delta = \|Au_* - \lambda_* u_*\|/\|u_*\|$. Then there exists λ_i such that $\lambda_* - \delta \leq \lambda_i \leq \lambda_* + \delta$.*

Proof. Since eigenfunctions $\{u_i\}$ form an orthonormal basis, we use Parseval's identity and the symmetry of A to derive the estimate

$$\|Au_* - \lambda_* u_*\|^2 = \sum_{j=1}^{\infty} \langle Au_* - \lambda_* u_*, u_j \rangle^2 = \sum_{j=1}^{\infty} |\lambda_j - \lambda_*|^2 \langle u_*, u_j \rangle^2 \geq \min_j |\lambda_j - \lambda_*|^2 \|u_*\|^2.$$

Thus, there exists λ_i such that

$$|\lambda_i - \lambda_*| = \min_j |\lambda_j - \lambda_*| \leq \frac{\|Au_* - \lambda_* u_*\|}{\|u_*\|} = \delta.$$

□

3 Kato's bound

Weinstein's lower bound is simple and elegant, but it does not provide the index information. Having the approximation λ_*, the corresponding λ_i is the exact eigenvalue closest to λ_*, but we do not know which one it is, i.e. we do not know the particular value for the index i. Moreover, the accuracy of Weinstein's bound is suboptimal, because it is linear in δ. This is solved by Kato's bound, which is quadratic in δ.

Theorem 2 (Kato 1949). *Let $u_* \in D(A) \setminus \{0\}$ be arbitrary and let $\lambda_* = \langle Au_*, u_* \rangle / \langle u_*, u_* \rangle$. Let $\delta = \|Au_* - \lambda_* u_*\| / \|u_*\|$ and $\mu, \nu \in \mathbb{R}$ satisfy*

$$\lambda_{i-1} \le \mu < \lambda_* < \nu \le \lambda_{i+1} \quad \textit{for some } i.$$

Then $\lambda_* - \dfrac{\delta^2}{\nu - \lambda_*} \le \lambda_i \le \lambda_* + \dfrac{\delta^2}{\lambda_* - \mu}$.

Proof. For simplicity, we prove the lower bound only. Notice that $(\lambda_j - \lambda_i)(\lambda_j - \nu) \ge 0$ for all $j = 1, 2, \dots$. Thus,

$$\begin{aligned} 0 \le \sum_{j=1}^{\infty} (\lambda_j - \lambda_i)(\lambda_j - \nu) \langle u_*, u_j \rangle^2 &= \sum_{j=1}^{\infty} (\lambda_j^2 - (\lambda_i + \nu)\lambda_j + \lambda_i \nu) \langle u_*, u_j \rangle^2 \\ &= \|Au_*\|^2 - (\lambda_i + \nu) \langle Au_*, u_* \rangle + \lambda_i \nu \|u_*\|^2 = \left(\delta^2 + \lambda_*^2 - (\lambda_i + \nu)\lambda_* + \lambda_i \nu\right) \|u_*\|^2, \end{aligned}$$

where we use the fact that $\|Au_*\|^2 = (\delta^2 + \lambda_*^2)\|u_*\|^2$. Consequently, we derived the inequality $0 \le \delta^2 + \lambda_*^2 - (\lambda_i + \nu)\lambda_* + \lambda_i \nu$ and the claimed statement follows by expressing λ_i. □

The order of accuracy of Kato's lower bound is optimal and it solves – in a sense – the index problem as well. However, this is a consequence of the strong assumption of having a lower bound ν on the exact eigenvalue λ_{i+1}. This assumption cannot be verified unless we have an additional knowledge about the spectrum. One possibility how to verify this assumption is the homotopy method proposed in [2].

4 Weak setting

The weak formulation of an elliptic eigenvalue problem is based on two continuous bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ defined on a Hilbert space V and reads: find $u_i \in V \setminus \{0\}$ and $\lambda_i \in \mathbb{R}$ such that

$$a(u_i, v) = \lambda_i b(u_i, v) \quad \forall v \in V.$$

For example, in the case of the Laplace eigenvalue problem, we have $V = H_0^1(\Omega)$, $a(u,v) = (\nabla u, \nabla v)$ and $b(u,v) = (u,v)$, where $\Omega \subset \mathbb{R}^d$ is a domain and $(\cdot,\cdot)$ stands for the $L^2(\Omega)$ scalar product.

In general, we assume the bilinear form $a(\cdot,\cdot)$ to be symmetric and V-elliptic and the bilinear form $b(\cdot,\cdot)$ to be symmetric and nonnegative. We use the notation $\|v\|_a^2 = a(v,v)$ and $|v|_b^2 = b(v,v)$ for the induced norm and seminorm, respectively. If the seminorm $\|\cdot\|_b$ is compact with respect to $\|\cdot\|_a$ then we can use the spectral theory of compact operators to show that there is a countable sequence $0 < \lambda_1 \le \lambda_2 \le \lambda_3 \le \cdots$ of eigenvalues, the corresponding eigenfunctions can be normalized such that $b(u_i, u_j) = \delta_{ij}$, and Parseval's identity $\|v\|_b^2 = \sum_{i=1}^{\infty} |b(v, u_i)|^2$ holds true for all $v \in V$.

Within this setting Weinstein's bound can be generalized as follows.

Theorem 3. *Let $u_* \in V \setminus \{0\}$ and $\lambda_* \in \mathbb{R}$ be arbitrary and let $w \in V$ be the unique solution of the problem*

$$a(w,v) = a(u_*,v) - \lambda_* b(u_*,v) \quad \forall v \in V.$$

If

$$\sqrt{\lambda_{i-1}\lambda_i} \le \lambda_* \le \sqrt{\lambda_i \lambda_{i+1}} \tag{1}$$

and if there exists $\eta > 0$ such that $\|w\|_a \le \eta$ then

$$\ell_i \le \lambda_i, \quad \text{where} \quad \ell_i = \frac{1}{4\|u_*\|_b^2}\left(-\eta + \sqrt{\eta^2 + 4\lambda_*\|u_*\|_b^2}\right)^2.$$

Function w is a representative of the residual and its energy norm $\|w\|_a$ cannot be obtain exactly, in general. Therefore, we assume the existence of the computable bound η. The quantity η can be computed by the complementarity approach (or two-energy principle) using techniques of a posteriori error estimates, in particular, a suitable flux reconstruction. This theorem is a generalization of the result [3] and more details including the proof can be found in [4].

In a similar way, we can generalize Kato's bound.

Theorem 4. *Let $u_* \in V \setminus \{0\}$ be arbitrary and let $\lambda_* = \|u_*\|_a^2/\|u_*\|_b^2$. Let there be $\nu \in \mathbb{R}$ such that*

$$\lambda_{i-1} < \lambda_* < \nu \le \lambda_{i+1} \quad \text{for a fixed index } i. \tag{2}$$

Let $w \in V$ be the same as in Theorem 3 and let there is η such that $\|w\|_a \le \eta$. Then

$$L_i \le \lambda_i, \quad \text{where} \quad L_i = \lambda_* \left(1 + \frac{\nu}{\lambda_*(\nu - \lambda_*)}\frac{\eta^2}{\|u_*\|_b^2}\right)^{-1}.$$

5 Numerical example

For illustration, we consider Laplace eigenvalue problem in the dumbbell shaped domain $\Omega = (0,\pi)^2 \cup [\pi, 5\pi/4] \times (3\pi/8, 5\pi/8) \cup (5\pi/4, 9\pi/4) \times (0,\pi)$ with zero Dirichlet boundary conditions. Upper bounds $\lambda_* = \Lambda_{h,i}$ on eigenvalues are computed by conforming piecewise linear finite elements based on adaptive triangular meshes. Lower bounds ℓ_i, L_i with $\nu = \ell_{i+1}$, and L_i' with $\nu = L_{i+1}$ are computed as described above. Figure 1 presents convergence curves of eigenvalue enclosures $\Lambda_{h,i} - \ell_i$ (squares), $\Lambda_{h,i} - L_i$ (circles), and $\Lambda_{h,i} - L_i'$ for $i = 1$ and 2. The curve for L_i is missing in the left panel, because eigenvalues λ_1 and λ_2 form a tight cluster close to 2 and λ_3 is close to 5. Consequently, ℓ_2 is below Λ_1 even on the finest mesh, the middle inequality in (2) is not satisfied, and L_1 cannot be defined. Fortunately, L_2 with $\nu = \ell_3$ works well and L_1' with $\nu = L_2$ can be successfully computed.

6 Conclusions

Lower bound ℓ_i converges with a suboptimal rate, because it is based on the subotimal Weinstein's estimate. Lower bound L_i converges with the optimal rate, however, its accuracy depends heavily on the size of the spectral gap $\lambda_{i+1} - \lambda_i$ and on the choice of ν sufficiently close to λ_{i+1}. As we observed in the numerical example, if the spectral gap is small then the accuracy of the bound L_i is relatively low, unless the mesh is sufficiently fine and the small spectral gap well resolved. On the other hand, if the spectral gap is relatively large then the bound L_i produces very accurate

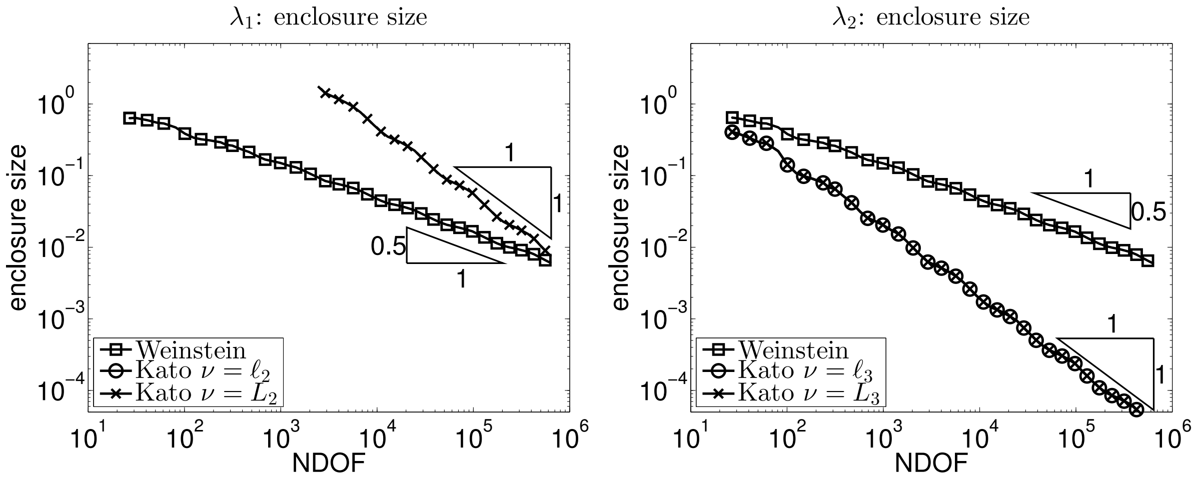


Figure 1: Squares correspond to ℓ_i, circles to L_i with $\nu = \ell_{i+1}$, and crosses to L_i with $\nu = L_{i+1}$.

results. Since both lower bounds ℓ_i and L_i are compute by simple formulas based on the same quantities, we recommend to compute both of them and use the larger one as the final lower bound.

Lower bound ℓ_i is guaranteed to be below λ_i if the relative closeness assumption (1) is satisfied. A similar assumption for Kato's bound is (2). These assumption are difficult to verify, but our numerical experiments indicate that even if they are not satisfied the bounds ℓ_i and L_i are very often below the exact eigenvalue λ_i. A natural approach is to combine the bounds ℓ_i and L_i and use $\nu = \ell_{i+1}$ to compute the bound L_i.

The disadvantage of the bound L_i is its dependence on the size of the spectral gap $\lambda_{i+1} - \lambda_i$. This bound even fails in the case of multiple eigenvalues. However, a version of Kato's bound that improves this disadvantage and that is suitable for multiple eigenvalues is presented already in [1] and it is generalized to the weak setting in [4]. In the future research we plan to prove the convergence of the adaptive algorithm mentioned in Section 5.

Acknowledgement: This work has been supported by Neuron Fund for Support of Science, project no. 24/2016.

References

[1] T. Kato: *On the upper and lower bounds of eigenvalues.* J. Phys. Soc. Japan 4 (1949), pp. 334–339.

[2] M. Plum: *Eigenvalue inclusions for second-order ordinary differential operators by a numerical homotopy method.* Z. Angew. Math. Phys. 41 (1990), pp. 205–226.

[3] I. Šebestová, T. Vejchodský: *Two-sided bounds for eigenvalues of differential operators with applications to Friedrichs, Poincaré, trace, and similar constants.* SIAM J. Numer. Anal. 52 (2014), pp. 308–329.

[4] T. Vejchodský, I. Šebestová: *New guaranteed lower bounds on eigenvalues by conforming finite elements.* In preparation.

[5] D.H. Weinstein: *Modified Ritz method.* Proc. Natl. Acad. Sci. USA 20 (1934), pp. 529–532.

Winter school lectures

R. Blaheta:
Overlapping domain decomposition preconditioners for elliptic and parabolic problems in primal and mixed form

T. Brzobohatý, Z. Dostál, D. Horák et al.:
FETI methods for large problems – algorithms, scalability, and software implementation

M. Hladík:
Introduction to interval computation and numerical verification

Parallel linear algebra

M. Faverge:
Dense linear algebra - From BLAS to Chameleon
Sparse linear algebra
PASTIX: A sparse direct solver based on super-nodal method

G. Marait:
MaPHyS: a Massively Parallel Hybrid Solver

Z. Strakoš:
Krylov subspace methods from the historical, analytic, application, and high performance computing perspective

Overlapping domain decomposition preconditioners for elliptic and parabolic problems in primal and mixed form

R. Blaheta

Institute of Geonics of the CAS, Ostrava

1 Introduction

In this lecture, we concern the numerical solution of PDE problems and describe overlapping domain decomposition, which provides a tool for the construction of parallelizable Schwarz type iterative solvers and preconditioners. The idea of using overlapping domain decomposition goes back to Schwarz alternating method from 1870, see [1]. The analysis of this alternating iterative method was evolved by great mathematicians, see e.g. S.L. Sobolev (1936), R. Courant and D. Hilbert (1937), S.G. Michlin (1951), M. Práger (1958), I. Babuška (1958), F.E. Browder (1958). The use of overlapping domain decomposition for parallel computations started in the late eighties in the work of M. Dryja and O. Widlund [4], P.L. Lions [5, 6], S. Nepomnyaschikh [7] and others and continue up to the present days. The origin of the alternating Schwarz method is nicely described in [8].

The classical results about the Schwarz type methods are described in several domain decomposition textbooks, see e.g. Chan and Mathew [9], Smith, Bjørstadt and Gropp [10], Toselli and Widlund [11], Mathew [12], Brenner and Scott [13].

The Schwarz method is also for a long time investigated and utilized at the Institute of Geonics, see e.g. [14, 15, 16, 17, 18]. The author would like to thank all co-workers contributed to this work. The implementation for massively parallel and robust computations is nowadays in progress starting collaboration with the PERMON team from IT4Innovations Centre at TU Ostrava.

In this lecture, we shall repeat the classical results which provide necessary framework. Then we touch new topics concerning robustness of additive Schwarz preconditioners as well as applications to mixed FEM and multiphysics problems.

2 Solved problems

In this paper, we are interested in numerical solution of boundary and initial-boundary value problems for elliptic and parabolic PDEs of the form

$$-\nabla \cdot \alpha \nabla p = f \ \text{ in } \Omega \tag{1}$$

and

$$c_0 \frac{\partial p}{\partial t} - \nabla \cdot \alpha \nabla p = f \ \text{ in } \Omega, \tag{2}$$

respectively. Above p is the unknown function and α and c_0 are coefficients which can vary in the domain Ω.

We assume that $\Omega \subset R^d$, $d = 1, 2, 3$ and that these problems are discretized by linear P_1 finite elements from the space

$$V_h = \left\{ v_h \in H^1(\Omega) : \ v_h|_E \in P_1 \ \forall E \in \mathcal{T}_h \right\},$$

where $\mathcal{T}_h$ is a division of Ω into simplexes. The discretization in time can be done by backward Euler or higher order Radau time integration methods. The FEM for elliptic problems and FEM with backward Euler with timestep τ for parabolic problems lead to linear systems with SPD matrices A and $M + \tau A$, respectively.

We shall also consider mixed formulation of the above PDE problems arising from introduction of a dual variable v,

$$\begin{array}{lll} \alpha^{-1}v & +\nabla p & = 0 \\ \nabla \cdot v & & = f \ \text{ in } \Omega \end{array}$$

and

$$\begin{array}{lll} \alpha^{-1}v & +\nabla p & = 0 \\ \nabla \cdot v & -c_0 \frac{\partial p}{\partial t} & = -f \ \text{ in } \Omega \end{array}$$

The discretization in space will use the same division $\mathcal{T}_h$ of Ω, piecewise constant pressure $p = p_h \in L_2(\Omega)$ and lowest order Raviart-Thomas elements for velocity $v = v_h \in H(div, \Omega)$. The discretization in time provides differential-algebraic equations (DAE) which can be again discretized by backward Euler or higher order Radau time integration methods. Discretization provide saddle point systems with the matrices of the form

$$\begin{bmatrix} M_v & B^T \\ B & 0 \end{bmatrix} \text{ or } \begin{bmatrix} M_v & B^T \\ B & -M_p \end{bmatrix},$$

respectively.

3 Overlapping domain decomposition

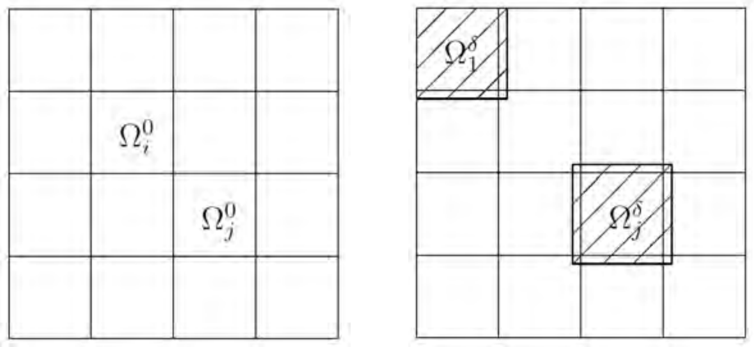


Figure 1: Overlapping domain decomposition.

Let us consider a two step construction of overlapping domain decomposition of Ω, see Fig. 1:

- first, decomposition into nonoverlapping subdomains Ω_i^0, $i = 1, \ldots, m$: $\bar{\Omega} = \bigcup \bar{\Omega}_i^0$, $\Omega_i^0 \cap \Omega_j^0 = \emptyset$ for $i \neq j$
- second, decomposition into overlapping subdomains Ω_i^δ, $i = 1, \ldots, m$, where $\delta > 0$, $\Omega_i^\delta = \{x \in \Omega,\ \mathrm{dist}(x, \Omega_i^0) \leq \delta\} \supset \Omega_i^0$

In the case of finite element division $\mathcal{T}_h$, we assume that all subdomains are aligned with the finite element division

$$\bar{\Omega}_j^0 = \bigcup \{E \subset \mathcal{T}_h : \ E \subset \bar{\Omega}_j^0 \neq \emptyset\}, \ \bar{\Omega}_j^\delta = \bigcup \left\{E \subset \mathcal{T}_h : \ E \subset \bar{\Omega}_j^\delta \neq \emptyset\right\}$$

Note that aligned extension is created by adding layers of elements,

$$\bar{\Omega}_j^{fe(1)} = \bigcup \left\{ E \subset \mathcal{T}_h : \; E \cap \bar{\Omega}_j^0 \neq \emptyset \right\}, \text{ etc.}$$

Beside the parameters m - number of subdomains and $\delta > 0$ - size of the overlap, there is another important parameter of the decomposition:

$$m_0 = \max_{i=1,\dots,m} m_{0i}, \;\; m_{0i} = card\left\{ j : \; \Omega_i^\delta \cap \Omega_j^\delta \neq \emptyset \right\}$$

Theorem 1. *Let us consider an overlapping domain decomposition $\{\Omega_j^\delta\}_{j=1}^m$. Then there is a partition of unity θ_j, $j = 1, \dots, m$ such that*

$$supp\, \theta_j \subset \overline{\Omega}_j^\delta \tag{3}$$

$$0 \leq \theta_j(x) \leq 1 \tag{4}$$

$$1 = \sum \theta_j(x) \quad \forall x \in \overline{\Omega} \tag{5}$$

Moreover, all θ_j are continuous in $\overline{\Omega}$, $\nabla\theta_j$ exists a.e. in Ω and there is a constant C independent on δ and $diam(\Omega_j)$ such that

$$\|\nabla\theta_j\|_\infty \leq C/\delta \quad \forall j = 1, \cdots, m. \tag{6}$$

Proof. The proof can be found in [11], see also [20, 10]. It is based on the construction

$$\theta_i(x) = d_i(x) \Big/ \sum_{k=1}^m d_k(x),$$

where

$$d_i(x) = \begin{cases} \text{dist}(x, \partial\Omega_i^\delta \setminus \partial\Omega) & \text{for } x \in \Omega_i^\delta, \\ 0 & \text{otherwise.} \end{cases}$$

□

4 Function spaces and bilinear forms

We consider two types of bilinear forms. First,

$$a(u,v) = \int_\Omega \alpha\nabla u \cdot \nabla v + \beta\, uv, \;\; \alpha = \alpha(x) \geq \alpha_0 > 0, \;\; \beta = \beta(x) \geq \beta_0 \geq 0. \tag{7}$$

This form is defined, bilinear and symmetric on a subspace V of $H^1(\Omega)$. If $\beta_0 = 0$, then we consider such subspace V that the Friedrichs or Poincare inequalities guarantee the positive definiteness of the bilinear form a on V. For simplicity we restrict to the case with Friedrichs inequality, i.e. we assume that there is a constant c_F independent on $v \in V$,

$$\|v\|_{2,0} \leq c_F \, |v|_{2,1} \;\; \forall v \in V. \tag{8}$$

If $\beta_0 > 0$ then the bilinear form can be considered on $V = H^1(\Omega)$ where it is positive definite and represents a weighted H^1norm.

Second considered bilinear form is the following

$$a(u,v) = \int_{\Omega} \kappa u \cdot v + \lambda \operatorname{div}(u) \operatorname{div}(v) \tag{9}$$

$\kappa \geq \kappa_0 > 0,\ \lambda \geq \lambda_0 > 0$. Then a is an SPD bilinear form which defines a weighted $H(div)$ inner product in the space $V = H(\mathrm{div}, \Omega)$ of vector functions $u :\ \Omega \to R^d$, $d = 2, 3$.

For the numerical realization, we use FE subspaces $V_h \subset V$. For example, we assume decomposition of Ω into triangles/tetrahedra and consider spaces

$$V_h = \{v \in C(\overline{\Omega}),\ v|_E \in P_1 \quad \forall E \in \mathcal{T}_h\} \subset V \subset H^1(\Omega), \tag{10}$$

$$V_h = \{v \in C(\,\overline{\Omega}\,)^d,\ v|_E \in RT_1 \quad \forall E \in \mathcal{T}_h\} \subset H(\mathrm{div}, \Omega), \tag{11}$$

where P_1 is the set of polynomials of order les or equal of one, RT_1 is the set of vector functions

$$v \in RT_1 \Leftrightarrow v(x) = cx + \xi, \quad x,\, \xi \in R^d,\, c \in R^1.$$

Using the bilinear form a, the space V and a continuous linear functional $b \in V'$, we can consider the variational problem,

$$\text{find}\, u \in V :\ a(u,v) = b(v) \quad \forall v \in V \tag{12}$$

Proposition 1. *Let V be a Hilbert space with the norm $\|\cdot\|_1$, a be bounded and positive definite on V , i.e. there are two positive constants γ_0, γ_1 such that*

$$|a(u,v)| \leq \gamma_1 \|u\|_V \|v\|_V \quad \forall u, v \in V \tag{13}$$

$$\gamma_0 \|u\|_V^2 \leq a(u,u) \quad \forall u \in V \tag{14}$$

Then the Lax-Milgram theorem guarantees existence and uniqueness of the solution of (12).

The problem (12) can be rewritten into the operator form

$$\mathcal{A}u = b,\ \mathcal{A} :\ V \to V',\ u \in V,\ b \in V', \tag{15}$$

where

$$\langle \mathcal{A}u,\, v \rangle = a(u,\, v)\ \forall u,\, v \in V, \tag{16}$$

$\langle \cdot, \cdot \rangle$ is the duality pairing.

The same construction is possible for the finite dimensional case $V = V_h$. Further, a basis $\{\phi_i\}_1^n$ in V_h define an isomorphism $u \equiv \boldsymbol{u}$ between V_h and R^n, and (15) become equivalent to the linear algebraic system

$$A\boldsymbol{u} = \boldsymbol{b}, \quad \boldsymbol{u}, \boldsymbol{b} \in R^n, \tag{17}$$

$$\langle A\boldsymbol{u},\, \boldsymbol{v} \rangle_n = a(u,v)\ \forall \boldsymbol{u}, \boldsymbol{v} \in R^n,\ \ \boldsymbol{u} \equiv u,\ \boldsymbol{v} \equiv v,\ \ \langle \boldsymbol{b},\, \boldsymbol{v} \rangle_n = b(v)\ \forall \boldsymbol{v} \in R^n,\ \ \boldsymbol{v} \equiv v,$$

where $\langle \cdot, \cdot \rangle_n$ is the Euclidean inner product in R^n.

5 Space decomposition

Let us consider the space V of functions in Ω defined in Section 4 and domain decomposition into Ω_i^δ, $i = 1, \dots, m$ defined in Section 3. Then a stable decomposition means that

$$V = V_1 + \dots + V_m, \;\; V_i = \{v \in V : \;v = 0 \text{ in } \Omega \setminus \Omega_i^\delta\}$$

The existence of stable decomposition is crucial for the infinite dimensional case, see e.g. [5]. Note that due to the nonempty overlap, the decomposition of V is not a direct sum. The inclusion $V_i \subset V$ defines a natural operator $S_i : V_i \to V$.

The bilinear form a and the operator $\mathcal{A}$ can be restricted to V_i,

$$a_i(u_i, v_i) = a(S_i u_i, S_i v_i) \;\; \forall u_i, v_i \in V_i,$$

$$\langle \mathcal{A}_i u_i, v_i \rangle = a_i(u_i, v_i) = a(S_i u_i, S_i v_i) = \langle \mathcal{A} S_i u_i, S_i v_i \rangle = \langle R_i \mathcal{A} S_i u_i, v_i \rangle\,,$$

thus $\mathcal{A}_i = R_i \mathcal{A} S_i = R_i \mathcal{A} R_i^\star$, where R_i (a restriction) is the adjoint operator to S_i, $R_i : V \to V_i$.

In the case of aligned FE discretization and domain decomposition and FE spaces with nodal degrees of freedom, the decomposition

$$V = V_1 + \dots + V_m, \;\; i.e.\; V_h = V_{h_1} + \dots + V_{h_m}$$

trivially exists. If $V_h = \text{span}\,\{\phi_i,\; i \in N_h\}$, where $\{\phi_i,\; i \in N_h\}$ is a nodal FE basis, $N_h = \{1 \dots, n\}$, it holds that $V_{h_i} = \text{span}\,\{\phi_i,\;\; i \in N_i\}$, $N_i \subset N_h$, $\text{card}(N_i) = n_i$. Then the isomorphism $u \equiv \boldsymbol{u}$ between V_h and $R^n \equiv \boldsymbol{V}$ can be completed by isomorphisms $u_i \equiv \boldsymbol{u}_i$ between V_{h_i} and $R^{n_i} \equiv \boldsymbol{V}_i$. The inclusion $V_{h_i} \subset V_h$ and these isomorphisms define the prolongation operators $\boldsymbol{S}_i : R^{n_i} \to R^n$ and the restriction operators $\boldsymbol{R}_i : R^n \to R^{n_i}$.

Note that $\boldsymbol{R}_i^T = \boldsymbol{S}_i$,

$$(\boldsymbol{S}_i)_{jk} = \left\{ \begin{array}{ll} 1 & if\ j \in N_i\ and\ k\ is\ the\ order\,(index)\ of\ j\ in\ N_i \\ 0 & \qquad\qquad\qquad otherwise \end{array} \right. ,$$

The decomposition now has the form

$$\boldsymbol{V} = \sum_i \boldsymbol{R}_i^T \boldsymbol{V}_i$$

and $\boldsymbol{A}_i = \boldsymbol{R}_i \boldsymbol{A} \boldsymbol{S}_i = \boldsymbol{R}_i \boldsymbol{A} \boldsymbol{R}_i^T$.

6 Schwarz-type methods and preconditioners

Let us consider the problem

$$\text{find}\, u \in V : \;\; a(u, v) = b(v) \quad \forall v \in V \tag{18}$$

and note that if $\tilde{u} \in V$ is an approximation of u, which is the solution of (18), then a correction from V_k can be computed as

$$w_k \in V_k, \;\; a(w_k, v) = b(v) - a(\tilde{u}, v) \quad \forall v \in V_k.$$

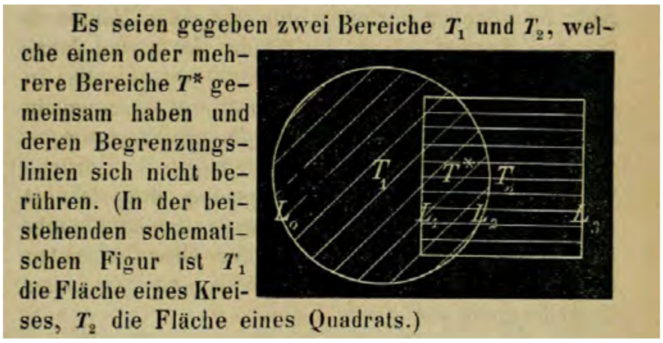
The corrected approximation to u has the form $\tilde{u}_C = \tilde{u} + w_k$. Note that w_k is the a-orthogonal projection of the error $u - \tilde{u}$ to V_k because

$$a(u - \tilde{u} - w_k,\, v) = 0 \quad \forall v \in V_k.$$

Schwarz algorithm, which uses the correction from all subspaces V_k, can be defined as follows

```
Let $u^0$ be given
for $i = 0, 1, 2, \cdots$ until convergence
    $w = 0$
    for $k = 1, \cdots, m$
        compute $w^k$
        $a(w^k, v) = b(v) - a(u^i + \sigma w, v) \quad \forall v \in V_k$
        $w = w + w^k$
    end
    $u^{i+1} = u^i + \omega w$
end
```

The multiplicative algorithm uses the choice $\sigma = 1$, $\omega = 1$ and it is convergent under described setting. The additive algorithm uses $\sigma = 0$ and for convergence we need suitable damping by $0 < \omega < 1$. Note that the multiplicative algorithm was suggested by H.A. Schwarz in 1870 for proving existence of the solution on a composite domain, see Fig. 2.



Es seien gegeben zwei Bereiche T_1 und T_2, welche einen oder mehrere Bereiche T^* gemeinsam haben und deren Begrenzungslinien sich nicht berühren. (In der beistehenden schematischen Figur ist T_1 die Fläche eines Kreises, T_2 die Fläche eines Quadrats.)

T_1
T^*
T_2
L_0
L_1
L_2
L_3

Figure 2: A picture from the original paper by H.A. Schwarz published in 1870.

The Schwarz algorithm can be rewriten into the operator or matrix form

Operator form

```
Let $u^0$ be given
for $i = 0, 1, 2, \cdots$ until convergence
    $w = 0$
    for $k = 1, \cdots, m$
        compute $w^k = \mathcal{A}_k^{-1}\left(b - \mathcal{A}(u^i + \sigma w)\right)$
        $w = w + w^k$
    end
    $u^{i+1} = u^i + \omega w$
end
```

Matrix form

```
Let $\boldsymbol{u}^0$ be given
for $i = 0, 1, 2, \cdots$ until convergence
    $\boldsymbol{w} = 0$
    for $k = 1, \cdots, m$
        compute $\boldsymbol{w}^k = \boldsymbol{A}_k^{-1}\left(\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{u}^i + \sigma \boldsymbol{w})\right)$
        $\boldsymbol{w} = \boldsymbol{w} + \boldsymbol{w}^k$
    end
    $\boldsymbol{u}^{i+1} = \boldsymbol{u}^i + \omega \boldsymbol{w}$
end
```

Application of one iteration of the Schwarz method starting from zero initial gues provides the Schwarz preconditioner. We shall be especially interested in the additive Schwarz preconditioner. In the operator form it provides $\mathcal{B} : V \to V'$ where

$$\mathcal{B}^{-1} = \sum R_k^\star \mathcal{A}_k^{-1} R_k. \tag{19}$$

In the matrix form

$$\boldsymbol{B}^{-1} = \sum \boldsymbol{R}_k^T \boldsymbol{A}_k^{-1} \boldsymbol{R}_k \tag{20}$$

Note that the local inverses represent solving local systems. It is possible to generalize the Schwarz method in this respect that local systems are solved only inaccuratelly.

7 Tools for analysis of the additive Schwarz preconditioner

Let us consider the preconditioner (19) providing the preconditioned operator

$$\mathcal{B}^{-1}\mathcal{A} = \sum_k R_k^\star \mathcal{A}_k^{-1} R_k \mathcal{A} = \sum_k P_k,$$

where P_k are a-orthogonal projections. To get spectral information about $\mathcal{B}^{-1}\mathcal{A}$, which is symmetric in a-inner product, we shall investigate the form $a(\mathcal{B}^{-1}\mathcal{A}v, v)$. A trivial upper bound for is as follows

$$a(\mathcal{B}^{-1}\mathcal{A}v,\, v) \leq \left\| \sum_k P_k v \right\|_a \|v\|_a \leq m\|v\|_a^2,$$

where m is the number of subdomains. A sharper estimate, not depending on m, is provided by the following theorem.

Theorem 2. *Let*

$$a(v_i,\, v_j) \leq \varepsilon_{ij} \sqrt{a(v_i,\, v_i)} \sqrt{a(v_j,\, v_j)}$$

for all $v_i \in V_i$, $v_j \in V_j$, $0 \leq \varepsilon_{ij} \leq 1$. For $\mathcal{E} = (\varepsilon_{ij})$, let $\rho(\mathcal{E})$ be the spectral radius of $\mathcal{E}$. Then for all $v \in V$,

$$a(\mathcal{B}^{-1}\mathcal{A}v,\, v) = a\left(\sum_1^m P_k v,\, v\right) \leq \rho(\mathcal{E})\, a(v,v).$$

Proof. It holds

$$\begin{aligned}
a\left(\sum_1^m P_i v,\, v\right) &\leq a\left(\sum_i P_i v,\, \sum_j P_j v\right)^{1/2} a(v,v)^{1/2} \leq \left[\sum_{ij} a(P_i v,\, P_j v)\right]^{1/2} a(v,v)^{1/2} \leq \\
&\leq \left[\sum_{ij} \varepsilon_{ij} \|P_i v\|_a \|P_j v\|_a\right]^{1/2} a(v,v)^{1/2} \leq \left[\varrho(\mathcal{E}) \sum_i \|P_i v\|_a^2\right]^{1/2} a(v,v)^{1/2} \leq \\
&\leq \rho(\mathcal{E})^{1/2} a\left(\sum P_i v,\, v\right)^{1/2} a(v,v)^{1/2}
\end{aligned}$$

□

Note that

$$\rho(\mathcal{E}) \leq \|\mathcal{E}\|_\infty = \max_i \sum \varepsilon_{ij} \leq m_0$$

where m_0 is the maximal number of overlapping subdomains (colouring). Thus

$$\lambda_{\max}(\mathcal{B}^{-1}\mathcal{A}) \leq m_0 = K_1. \tag{21}$$

Theorem 3. *(Lions 1988, Nepomnyaschikh 1986) Let K_0 be a positive constant such that*

$$\forall v \in V \ \exists v_k \in V_k : \ v = v_1 + \cdots + v_k \quad \sum a(v_k, v_k) \leq K_0 \, a(v, v).$$

Then

$$a(v, v) \leq K_0 \, a(\mathcal{B}^{-1}\mathcal{A}v, v) \quad \forall v \in V$$

and consequently

$$\lambda_{\min}(\mathcal{B}^{-1}\mathcal{A}) \geq 1/K_0. \tag{22}$$

Proof. It holds

$$\begin{aligned} a(v, v) &= a\left(v, \sum_k v_k\right) = \sum_k a(v, P_k v_k) = \sum_k a(P_k v, v_k) \leq \\ &\leq \left\{\sum_k a(P_k v, P_k v)\right\}^{1/2} \left\{\sum_k a(v_k, v_k)\right\}^{1/2} \\ &= \left\{\sum_k a(P_k v, v)\right\}^{1/2} k_0^{1/2} a(v, v)^{1/2} \end{aligned}$$

$$a(v, v) \leq k_0 \, a(\mathcal{B}^{-1}\mathcal{A}v, v)$$

□

As a counterpart to this theorem, we also have

Theorem 4. *(Bjørstadt, Mandel 1991) Let us assume that there is a constant K_1 such that*

$$\forall v \in V \ \forall v_k \in V_k : \ v = v_1 + \cdots + v_m \quad a(v, v) \leq K_1 \sum a(v_k, v_k).$$

Then

$$a(\mathcal{B}^{-1}\mathcal{A}v, v) \leq K_1 \, a(v, v) \ \ \forall v \in V$$

and

$$\lambda_{\max}(\mathcal{B}^{-1}\mathcal{A}) \leq K_1.$$

Note that $K_1 \leq \varrho(\mathcal{E})$ for $\mathcal{E}$ introduced in Theorem (2).

For the algebraic case, the preconditioned system has the form

$$\boldsymbol{B}^{-1}\boldsymbol{A} = \sum \boldsymbol{R}_k^T \boldsymbol{A}_k^{-1} \boldsymbol{R}_k \boldsymbol{A} = \sum \boldsymbol{P}_k,$$

where $\boldsymbol{P}_k$ are now $\boldsymbol{A}$-orthogonal projections. All the above theorems are applicable to this case, getting a bit modified form. As an example, we reformulate Theorem (3).

Theorem 5. *Let K_0 be a positive constants, such that*

$$\forall \boldsymbol{v} \in \boldsymbol{V}, \exists \boldsymbol{v}_k \in \boldsymbol{V}_k, \quad \boldsymbol{v} = \sum_{k=k_0}^{m} \boldsymbol{v}_k : \sum_k \| \boldsymbol{R}_k \boldsymbol{v}_k \|_{\boldsymbol{A}}^2 \leq K_0 \| \boldsymbol{v} \|_{\boldsymbol{A}}^2, \tag{23}$$

Then

$$\lambda_{min}\left(\boldsymbol{B}^{-1}\boldsymbol{A}\right) \geq 1/K_0, \ \lambda_{max}\left(\boldsymbol{B}^{-1}\boldsymbol{A}\right) \leq K_1 = m_0, \ cond\left(\boldsymbol{B}^{-1}\boldsymbol{A}\right) \leq K_0 K_1.$$

8 Applications

The developed theory can be applied to analysis of particular cases. The condition number estimate is not favourable for elliptic problems with bilinear form of the type (7) with $\beta = 0$. In this case,

$$\operatorname{cond}\left(\boldsymbol{B}^{-1}\boldsymbol{A}\right) \leq 2\left(1 + \frac{1}{\delta^2}\frac{\alpha_{\max}}{\alpha_{\min}}c_{F,\Omega}\right),$$

which is not favourable for two reasons. First, the estimate involve contrast in the coefficient α over whole domain. Note that this can be localized into subdomains if the Friedrichs inequality holds on the subdomains with uniformly bounded constant c_F. Second, the term δ^{-2} naturaly increase if we increase number of subdomains.

This unfavourable dependence on δ^{-2} is usually compensated by introducing auxiliary global coarse space into the space decomposition. Such space can be defined e.g. by a coarser finite element grid [4], by agregations [22, 14] or by smoothed aggregations [20].

The situation is more favourable for the bilinear form of the type (7) with $\beta = \beta(x) \geq \beta_0 > 0$. In this case

$$\operatorname{cond}\left(\boldsymbol{B}^{-1}\boldsymbol{A}\right) \leq 2\left(1 + \frac{1}{\delta^2}\max\frac{\alpha}{\beta}\right),$$

where $\max\frac{\alpha}{\beta}$ can effectivelly compensate the term δ^{-2} if $\alpha \ll \beta$, e.g. for parabolic problems with small time step. See [15, 12].

The presence of L_2 part in the $H(div)$ norm has a similar effect. This enables to construct efficient preconditioners for mixed FEM systems and apply one-level Schwarz preconditioning to elliptic problems discretized by mixed FEM. See [16, 17]

9 Schwarz preconditioner for poroelasticity

The Biot model of poroelasticity arises from interconnecting elasticity and time dependent Darcy flow in deformable porous media. The fluid pressure contributes to elastic stress, deformation of porous space serves as a source or sink in fluid conservation. Frequently the elasticity is discretized in space by standard Lagrangian (Courant) finite elements and Darcy flow is discretized by more accurate and conservative mixed elements (Raviart - Thomas). The combination with time discretization by implicit Euler method then provides a time stepping scheme, when the system with the matrix $\mathcal{A}_E$ or $\overline{\mathcal{A}}_E$

$$\mathcal{A}_E = \begin{bmatrix} A & 0 & B_u^T \\ 0 & M_v & B^T \\ \frac{1}{\tau}B_u & B & -\frac{1}{\tau}C \end{bmatrix}, \quad \overline{\mathcal{A}}_E = \begin{bmatrix} 1 & & \\ & \tau & \\ & & \tau \end{bmatrix} \mathcal{A}_E = \begin{bmatrix} A & 0 & B_u^T \\ 0 & \tau M_v & \tau B^T \\ B_u & \tau B & -C \end{bmatrix},$$

is solved in each time step. The blocks are finite element matrices, A corresponds to elasticity discretized by Lagrangian elements, M_v is a weighted mass matrix corresponding to velocities discretized by Raviart-Thomas mixed finite elements, C corresponds to a weighted mass matrix for piecewise constant finite elements, B_u and B are constraint matrices from coupling of elasticity and flow equations. For the iterative solution, it is favourable to scale the matrix $\mathcal{A}_E$ to get system with symmetric matrix $\overline{\mathcal{A}}_E$. We can use GMRES or after symmetrization the MINRES iterative methods with the following matrix $\mathcal{P}$ as an efficient positive definite preconditioner to $\overline{\mathcal{A}}_E$,

$$\mathcal{P} = \begin{bmatrix} A & 0 & 0 \\ 0 & M_a & 0 \\ 0 & 0 & C \end{bmatrix}, \quad M_a = \tau M + \tau^2 B^T C^{-1} B.$$

More details can be founfd e.g. in [19, 18].

For implementation of the preconditioner $\mathcal{P}$, we have to (approximately) solve the systems with A and the augmented matrix M_a. The solution of the elasticity block system can be done by two-level additive Schwarz method. The solution of the system with matrix M_a has been described and analysed in [19]. The analyse shows that the use of one-level method is enough in this case for several reasons. First is damping of the differential part of M_a by the time step factor. This is similar to application of the one-level Schwarz method to the systems appearing in the implicit Euler solution of parabolic equations, which was mentioned earlier. Second reason is that the matrix M_v is weighted by inverse of permeabilities, which make it dominating over the differential part of M_a .

Acknowledgement: The work was done within the projects LD15105 “Ultrascale computing in geo-sciences” and LQ1602 “IT4Innovations excellence in science” supported by the Ministry of Education, Youth and Sports of the Czech Republic.

References

[1] H.A. Schwarz: *Über einen Grenzübergang durch alternierendes Verfahren.* Vierteljahrsschrift der Naturforschenden Gesellschaft in Züurich, 15, Mai 1870, pp. 272–286.

[2] S.L. Sobolev: *L'Algorithme de Schwarz dans la Thórie de l'Elasticité.* Doklady Akademii Nauk SSSR, IV((XIII) 6), 1936, pp. 243–246.

[3] S.G. Mikhlin: *On the Schwarz algorithm*, Doklady Akademii Nauk SSSR, n. Ser., (in Russian), 77, 1951, pp. 569–571.

[4] M. Dryja, O.B. Widlund: *An additive variant of the Schwarz alternating method for the case of many subregions.* Technical Report 339, also Ultracomputer Note 131, Department of Computer Science, Courant Institute, 1987.

[5] P.L. Lions: *On the Schwarz alternating method.* I: First international symposium on domain dcecomposition methods for partial differential equations. SIAM, 1988.

[6] P.L. Lions: *On the Schwarz alternating method.* II: Second international symposium on domain dcecomposition methods for partial differential equations. SIAM, 1989.

[7] S. Nepomnyaschikh: *Domain decomposition and Schwarz methods in a subspace for the approximate solution of elliptic boundary value problems.* PhD thesis. Computing Center, USSR Academy of Sciences. Novosibirsk, 1986.

[8] M.J. Gander, G. Wanner: *The Origins of the Alternating Schwarz Method.* Lecture Notes in Computational Science and Engineering 98, 2014, pp. 487–496.

[9] T.F. Chan, T.P. Mathew: *Domain decomposition algorithms.* Acta Numerica. Cambridge University Press, 1994, pp. 61–143.

[10] B. Smith, P.E. Bjørstad, W.D. Gropp: *Domain decomposition: Parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press, 1996.

[11] A. Toselli, O.B. Widlund: *Domain decomposition methods: Algorithms and theory.* Springer, 2004.

[12] T.P.A. Mathew: *Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations.* Springer-Verlag, Berlin, Heidelberg, 2008.

[13] S.C. Brenner, L.R. Scott: *Mathematical theory of finite element methods.* Springer-Verlag, third edition, 2008.

[14] R. Blaheta: *Space decomposition preconditioners and parallel solvers.* In: M. Feistauer et al.: Numerical Mathematics and Advanced Applications, Springer, Berlin, 2004, pp. 20–38.

[15] R. Blaheta, R. Kohut, M. Neytcheva, J. Starý: *Schwarz methods for discrete elliptic and parabolic problems with an application to nuclear waste repository modelling.* Mathematics and Computers in Simulation 76 (2007), pp. 18–27.

[16] O. Axelsson, R. Blaheta: *Preconditioning of matrices partitioned in 2 x 2 block form: Eigenvalue estimates and Schwarz DD for mixed FEM.* Numerical Linear Algebra with Applications 17(2010), pp. 787–810.

[17] O. Axelsson, R. Blaheta, P. Byczanski, J. Karátson, B. Ahmad: *Preconditioners for regularized saddle point problems with an application for heterogeneous Darcy flow problems.* Journal of Computational and Applied Mathematics 280 (2015), pp. 141–157.

[18] R. Blaheta, J. Starý, O. Jakl: *Parallel Schwarz domain decomposition solvers with applications in elasticity and poroelasticity.* ICNAAM 2016, to appear in AIP Proceedings.

[19] O. Axelsson, R. Blaheta, P. Byczanski: *Stable discretization of poroelasticity problems and efficient preconditioners for arising saddle point type matrices.* Comput. Visual. Sci. 15, 2012, pp. 191–207. DOI 10.1007/s00791-013-0209-0

[20] M. Brezina, P. Vanek: *A black box iterative solver based on a two level Schwarz method.* Computing 63, No. 3, 1999, pp. 233–263.

[21] P. Bjørstad, J. Mandel: *On the spectra of sums of orthogonal projections with applications to parallel computing.* BIT. 31 (1991), pp. 76–88.

[22] E.W. Jenkins, C.T. Kelley, C.T. Miller, C.E. Kees: *An aggregation based domain decomposition preconditioner for groundwater flow.* SIAM J. Sci. Comput. 23(2001), pp. 430–441.

FETI methods for large problems – algorithms, scalability, and software implementation

T. Brzobohatý, Z. Dostál, D. Horák, V. Hapla,
T. Kozubek, J. Kružík, A. Markopoulos, L. Říha

IT4Innovations, VŠB - Technical university of Ostrava
Department of applied mathematics, VŠB - Technical university of Ostrava

1 Introduction

The purpose of this lecture is to give an overview of the basic FETI (Finite Element Tearing and Interconnecting) methods for the solution of extremely large (currently some hundreds of billions) systems of linear equations arising from the discretization of the boundary value problems for elliptic partial differential equations. In the first part of the lecture, we shall review the basic algorithms and scalability results. We shall also briefly mention some interesting features of these methods arising in the solution of more complex engineering problems including the problems with plasticity, contact problems of elasticity with or without friction [7], shape optimization including contact shape optimization, problems with varying material properties etc.

The FETI domain decomposition methods can effectively exploit the parallel facilities provided by modern supercomputers. However, this task is far from trivial and straightforward. The FETI methods appeared in the early 90s, when the parallel computers were not assumed to have some tens or even hundreds of thousands of cores, and an immediate goal was to use them for the solution of the problems discretized by a few millions of the degrees of freedom. Thus it is not surprising that we face new problems. For example, the cost of the assembling of the projector to the "natural coarse grid", which is nearly negligible for smaller problems, starts to essentially affect the cost of the solution when the dimension of the dual problem reaches some tens of millions. New challenges are posed also by the emerging exascale technologies, the effective exploitation of which has to take into account a hierarchical organization of memory, the varying cost of operations depending on the position of arguments in memory, and the increasing role of communication costs. Last but not least, it is important to exploit an up-to-date software, either open-source or commercial, as the effective implementation of some standard steps, such as the application of direct solvers, is highly nontrivial and affects the overall performance of algorithms.

In the second part of the lecture we present some hints concerning the parallel implementation of FETI-type algorithms for the solution of very large problems, including the implementation of the action of a generalized inverse K^+ of the stiffness matrix K and the action of the projector to the "natural coarse grid" P. We briefly discuss the possibility to overcome the bottleneck by introducing the third level grid by a variant of HTFETI (Hybrid TFETI). The third level is introduced by the decomposition of TFETI subdomains into smaller subdomains that are partly glued in corners or by averages at the primal level as proposed by Klawonn and Rheinbach. The presentation will use two packages developed at IT4Innovations, National Supercomputing Center Ostrava, namely PERMON based on PETSc, and ESPRESO based on Intel MKL and Cilk.

2 FETI methods

The basic FETI (also FETI-1) method was was proposed by Farhat and Roux [1] in 1990. The FETI-1 method is based on the decomposition of the spatial domain into non-overlapping subdomains that are "glued" by Lagrange multipliers. After eliminating the primal variables, the original problem is reduced to a small, better conditioned system in Lagrange multipliers that is solved iteratively. The original FETI-1 method became numerically scalable after introducing the projectors to the natural coarse space (kernels of local stiffness matrices) by Farhat, Mandel, and Roux [2]. The latter authors proved the bounds on the spectrum in terms of the ratio of the decomposition and discretization parameters.

By projecting the Lagrange multipliers in each iteration onto an auxiliary space to enforce continuity of the primal solutions at the crosspoints, Farhat, Mandel and Tezaur obtained a faster converging FETI method for plate and shell problems - FETI-2.

Similar effect was achieved by a variant called the Dual-Primal FETI method FETI-DP, introduced by Farhat et al. The continuity of the primal solution at crosspoints is implemented directly into the formulation of the primal problem so that one degree of freedom is considered at each crosspoint shared by two and more adjacent subdomains. The continuity of the primal variables across the rest of the subdomain interfaces is once again enforced by the Lagrange multipliers. After eliminating the primal variables, the problem reduces to a small, relatively well conditioned strictly convex quadratic programming problem that is again solved iteratively.

Implementation of the FETI-1 and FETI-2 method into general purpose packages requires an effective method for automatic identification of the kernels of the stiffness matrices of the subdomains as these kernels are used both in elimination of the primal variables and in definition of the natural coarse grid projectors. This problem motivated the development of FETI-DP (dual-primal). FETI-DP manipulates with the subdomains joined in the some nodes called corners, so that the stiffness matrices of the subdomains are invertible. However, even though FETI-DP may be efficiently preconditioned so that it scales better than the original FETI for plates and shells, the coarse grid defined by the corners without additional preconditioning is less efficient than that defined by the rigid body motions, which is important for some applications, and the FETI-DP method is more difficult to implement as it requires special treatment of the corners which are not local variables associated with the subdomains.

An alternative solution was proposed in [3]. It is easier to implement and it preserves efficiency of the coarse grid of the classical FETI-1. The basic idea is to use the Lagrange multipliers not only for gluing of the subdomains along the auxiliary interfaces, but also for implementation of the Dirichlet boundary conditions. The resulting TFETI method thus works with a priori known kernels of the local stiffness matrices. Heuristic arguments and the results of numerical experiments indicate that the new method is not only much easier to implement, but also more efficient than the original FETI-1.

The parallel scalability of TFETI deteriorates with the increasing number of subdomains. The reason is the increasing cost of the implementation of projectors to the coarse grid. It seems that the most powerful tool for the solution of very large problems is a combination of TFETI and FETI-DP that is called HTFETI (hybrid). A few subdomains are joined by nodes or averages into so called clusters which have in HTFETI the same role as subdomains in FETI. The stiffness matrix of each cluster shares the dimension with any of its subdomains, i.e., six in 3D elasticity. Thus the dimension of the coarse space is reduced by tens, opening the way for effective exploitation of tens or hundreds of thousands of cores. HTFETI is thus a powerful tool for the exploitation of the hierarchical structure of modern supercomputers.

3 PERMON

PERMON (Parallel, Efficient, Robust, Modular, Object-oriented, Numerical) [5] is a software package which aims at the massively parallel solution of problems of constrained quadratic programming (QP). PERMON is based on PETSc and combines aforementioned TFETI method and QP algorithms. The core solver layer consists of the PermonQP package for QP and its PermonFLLOP extension for FETI. PermonQP supports the separation of QP problems, their transformations, and solvers. It contains all QP solvers described in [7]. More can be found on the PERMON website: permon.it4i.cz.

An example of numerical and weak parallel scalability of TFETI on model 3D linear elastic cube up to 701 millions of unknowns and 10,648 subdomains with one subdomain per one computational core on Archer is demonstrated in the graphs in Fig. 1. The contact problem was solved using SMALBE and MPRGP with our new adaptive expansion steplength which significantly improved this scalability and reduced not only the number of expansion steps but also the number of CG steps.

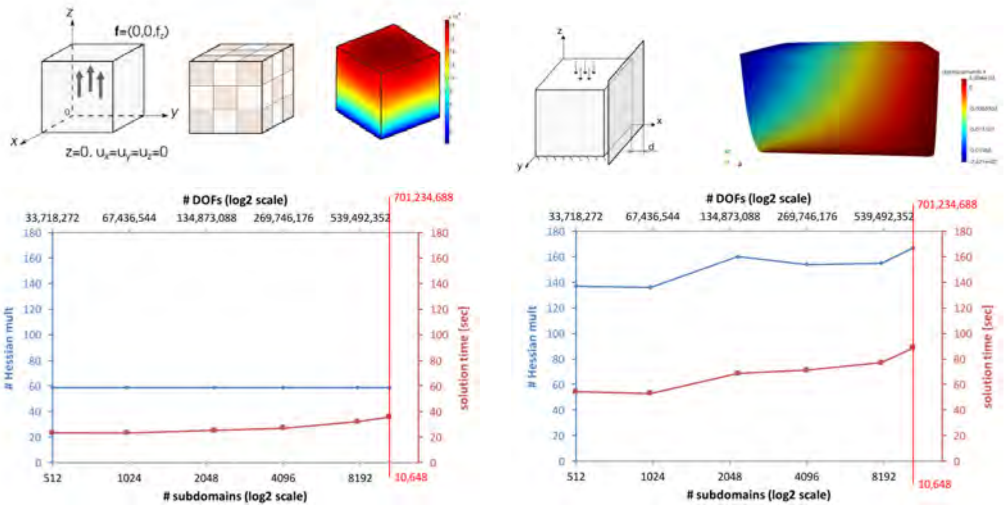


Figure 1: Scalability highlights - linear and contact 3D elastic cube problems

4 ESPRESO

ESPRESO [6] is an ExaScale PaRallel FETI SOlver developed at IT4Innovations. The main focus is to create a highly efficient parallel solver. Apart from the algorithms used by MatSol and PERMON, it also enhances the HFETI method, which is designed to run on massively parallel machines with thousands of compute nodes and hundreds of thousands of CPU cores. The algorithms can be seen as a multilevel FETI method designed to overcome the main bottleneck of standard FETI methods, a large coarse problem, which arises when solving large problems decomposed into the large number of subdomains. ESPRESO can exploit modern many-core accelerators.

There are three major versions of the solver. ESPRESO CPU is a CPU version that uses the sparse representation of system matrices. It contains an efficient communication layer on the top of MPI 3.0 combined with the shared memory parallelization inside nodes. The communication layer was developed specifically for FETI solvers and uses several state-of-the-art communication hiding and avoiding techniques to achieve better scalability.

The ESPRESO solver can take advantage of many-core accelerators to speedup the solver runtime. To achieve this, it uses a dense representation of sparse system matrices in the form of Schur complements. The main advantage of using this approach in FETI solvers is the reduction of the iteration time. Instead of calling a solve routine of the sparse direct solver in every iteration, which by its nature is a sequential operation, the solver can use the dense matrix-vector multiplication (GEMV) routine. The GEMV offers the parallelism required by many-core accelerators and delivers up to 4× speedup depending on the hardware configuration. There are two versions: ESPRESO MIC for Intel Xeon Phi and ESPRESO GPU for graphic accelerators. More information can be found at the ESPRESO website: espreso.it4i.cz

Acknowledgement: This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070); Project of major infrastructures for research, development and innovation of Ministry of Education, Youth and Sports with reg. num. LM2011033; by the internal student grant competition project SP2017/169 "PERMON toolbox development III" and by the Czech Science Foundation (GACR) project no. 15-18274S.

References

[1] C. Farhat, F.-X. Roux: *An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems*, SIAM J. Sci. Statist. Comput. 13, 1992, pp. 379–396.

[2] C. Farhat, J. Mandel, F.-X. Roux: *Optimal convergence properties of the FETI domain decomposition method*, Comput. Methods Appl. Mech. Engrg. 115, 1994, pp. 365–385.

[3] Z. Dostál, D. Horák, R. Kucera: *Total FETI - an easier implementable variant of the FETI method for numerical solution of elliptic PDE*, Communications in Numerical Methods in Engineering 22 (2006), 12, pp. 1155–1162.

[4] Z. Dostál: *Optimal quadratic programming algorithms: with applications to variational inequalities*, SOIA 23, Springer US, New York, 2009.

[5] PERMON web page, URL: `http://permon.it4i.cz`

[6] ESPRESSO web page, URL: `http://espreso.it4i.cz`

[7] Z. Dostál, T. Kozubek, M. Sadowská, V. Vondrák: *Scalable Algorithms for Contact Problems*, 1st edition, Springer US, New York, 2017, AMM 36.

Introduction to Interval Computation and Numerical Verification

part I.

Milan Hladík

Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

Seminář numerické analýzy a zimní škola – SNA'17
Ostrava, 30. ledna – 3. února 2017

Outline

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Interval Computation

What is interval computation

Solving problems with interval data
(or using interval techniques for non-interval problems)

What is **not** interval computation

- stochastic computation
- fuzzy computation

Interval paradigm

Take into account all possible realizations rigorously.

Where interval data do appear

1. numerical analysis (handling rounding errors)
2. computer-assisted proofs
3. global optimization
4. modelling uncertainty

Numerical Analysis

Example (Rump, 1988)

Consider the expression

$$f = 333.75b^6 + a^2(11a^2b^2 - b^6 - 121b^4 - 2) + 5.5b^8 + \frac{a}{2b},$$

with

$$a = 77617, \quad b = 33096.$$

Calculations from 80s gave

single precision	$f \approx 1.172603\ldots$
double precision	$f \approx 1.1726039400531\ldots$
extended precision	$f \approx 1.172603940053178\ldots$
the true value	$f = -0.827386\ldots$

Computer-Assisted Proofs

Kepler conjecture

What is the densest packing of balls? (Kepler, 1611)

That one how the oranges are stacked in a shop.

The conjecture was proved by T.C. Hales (2005).

Double bubble problem

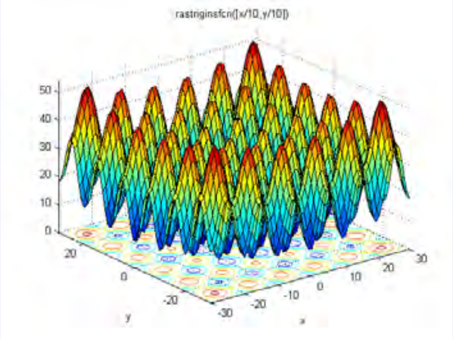
What is the minimal surface of two given volumes?

Two pieces of spheres meeting at an angle of 120°.

Hass and Schlafly (2000) proved the equally sized case.
Hutchings et al. (2002) proved the general case.

Global Optimization

Rastrigin's function $f(x) = 20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2))$



Further Sources of Intervals

- Mass number of chemical elements (sue to several stable isotopes)
  - $[12.0096, 12.0116]$ for the carbon
- physical constants
  - $[9.78,\ 9.82]\ ms^{-2}$ for the gravitational acceleration
- mathematical constants
  - $\pi \in [3.1415926535897932384,\ 3.1415926535897932385]$.
- measurement errors
  - temperature measured $23°\mathrm{C} \pm 1°\mathrm{C}$
- discretization
  - time is split in days
  - temperature during the day in $[-8, 3]°\mathrm{C}$ for Ostrava in January
- missing data
  - What was the temperature in Ostrava on January 31, 1999?
  - Very probably in $[-25, 15]°\mathrm{C}$.
- processing a state space
  - find robot singularities, where it may breakdown
  - check joint angles $[0, 180]°$.

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Interval Computations

Notation

An interval matrix

$$\boldsymbol{A} := [\underline{A}, \overline{A}] = \{A \in \mathbb{R}^{m\times n} \mid \underline{A} \le A \le \overline{A}\}.$$

The center and radius matrices

$$A^c := \frac{1}{2}(\overline{A} + \underline{A}), \quad A^\Delta := \frac{1}{2}(\overline{A} - \underline{A}).$$

The set of all $m \times n$ interval matrices: $\mathbb{IR}^{m\times n}$.

Main problem

Let $f\colon \mathbb{R}^n \mapsto \mathbb{R}^m$ and $\boldsymbol{x} \in \mathbb{IR}^n$. Determine the image

$$f(\boldsymbol{x}) = \{f(x) : x \in \boldsymbol{x}\}.$$

Interval Arithmetic

Interval arithmetic (incl. rounding, IEEE standard)

$$\begin{aligned}
\boldsymbol{a} + \boldsymbol{b} &= [\underline{a} + \underline{b}, \overline{a} + \overline{b}],\\
\boldsymbol{a} - \boldsymbol{b} &= [\underline{a} - \overline{b}, \overline{a} - \underline{b}],\\
\boldsymbol{a} \cdot \boldsymbol{b} &= [\min(\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}), \max(\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b})],\\
\boldsymbol{a} / \boldsymbol{b} &= [\min(\underline{a}/\underline{b}, \underline{a}/\overline{b}, \overline{a}/\underline{b}, \overline{a}/\overline{b}), \max(\underline{a}/\underline{b}, \underline{a}/\overline{b}, \overline{a}/\underline{b}, \overline{a}/\overline{b})], \quad 0 \notin \boldsymbol{b}.
\end{aligned}$$

Theorem (Basic properties of interval arithmetic)

- *Interval addition and multiplication is commutative and associative.*
- *It is not distributive in general, but sub-distributive instead,*

$$\forall \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \in \mathbb{IR} : \boldsymbol{a}(\boldsymbol{b} + \boldsymbol{c}) \subseteq \boldsymbol{ab} + \boldsymbol{ac}.$$

Example ($\boldsymbol{a} = [1, 2]$, $\boldsymbol{b} = 1$, $\boldsymbol{c} = -1$)

$$\begin{aligned}
\boldsymbol{a}(\boldsymbol{b} + \boldsymbol{c}) &= [1, 2] \cdot (1 - 1) = [1, 2] \cdot 0 = 0,\\
\boldsymbol{ab} + \boldsymbol{ac} &= [1, 2] \cdot 1 + [1, 2] \cdot (-1) = [1, 2] - [1, 2] = [-1, 1].
\end{aligned}$$

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Images of Functions

Monotone functions

If $f : \boldsymbol{x} \to \mathbb{R}$ is non-decreasing, then $f(\boldsymbol{x}) = [f(\underline{x}), f(\overline{x})]$.

Example

$\exp(\boldsymbol{x}) = [\exp(\underline{x}), \exp(\overline{x})]$, $\log(\boldsymbol{x}) = [\log(\underline{x}), \log(\overline{x})]$, ...

Some basic functions

Images $\boldsymbol{x}^2$, $\sin(\boldsymbol{x})$, ..., are easily calculated, too.

$$\boldsymbol{x}^2 = \begin{cases} [\min(\underline{x}^2, \overline{x}^2), \max(\underline{x}^2, \overline{x}^2)] & \text{if } 0 \notin \boldsymbol{x}, \\ \boldsymbol{x}^2 = [0, \max(\underline{x}^2, \overline{x}^2)] & \text{otherwise} \end{cases}$$

But...

...what to do for more complex functions?

Images of Functions

Notice

$f(\boldsymbol{x})$ need not be an interval (neither closed nor connected).

Interval hull $\square f(\boldsymbol{x})$

Compute the interval hull instead

$$\square f(\boldsymbol{x}) = \bigcap_{\boldsymbol{v} \in \mathbb{IR}^n \,:\, f(\boldsymbol{x}) \subseteq \boldsymbol{v}} \boldsymbol{v}.$$

Bad news

Computing $\square f(\boldsymbol{x})$ is still very difficult (NP-hard, undecidable).

Interval enclosure

Compute as tight as possible $\boldsymbol{v} \in \mathbb{IR}^n : f(\boldsymbol{x}) \subseteq \boldsymbol{v}$.

Interval Functions

Definition (Inclusion isotonicity)

$\boldsymbol{f} : \mathbb{IR}^n \mapsto \mathbb{IR}$ is *inclusion isotonic* if for every $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{IR}^n$:

$$\boldsymbol{x} \subseteq \boldsymbol{y} \Rightarrow \boldsymbol{f}(\boldsymbol{x}) \subseteq \boldsymbol{f}(\boldsymbol{y}).$$

Definition (Interval extension)

$\boldsymbol{f} : \mathbb{IR}^n \mapsto \mathbb{IR}$ is *an interval extension* of $f : \mathbb{R}^n \mapsto \mathbb{R}$ if for every $x \in \mathbb{R}^n$:

$$f(x) = \boldsymbol{f}(x).$$

Theorem (Fundamental theorem of interval analysis)

If $\boldsymbol{f} : \mathbb{IR}^n \mapsto \mathbb{IR}$ satisfies both properties, then

$$f(\boldsymbol{x}) \subseteq \boldsymbol{f}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathbb{IR}^n.$$

Proof.

For every $x \in \boldsymbol{x}$, one has by interval extension and inclusion isotonicity that $f(x) = \boldsymbol{f}(x) \subseteq \boldsymbol{f}(\boldsymbol{x})$, whence $f(\boldsymbol{x}) \subseteq \boldsymbol{f}(\boldsymbol{x})$. □

Natural Interval Extension

Definition (Natural interval extension)

Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ be a function given by an arithmetic expression. The corresponding *natural interval extension* $\boldsymbol{f}$ of f is defined by that expression when replacing real arithmetic by the interval one.

Theorem

Natural interval extension of an arithmetic expression is both an interval extension and inclusion isotonic.

Proof.

It is easy to see that interval arithmetic is both an interval extension and inclusion isotonic. Next, proceed by mathematical induction. □

Natural Interval Extension

Example

$$f(x) = x^2 - x, \quad x \in \boldsymbol{x} = [-1, 2].$$

Then

$$\begin{aligned} \boldsymbol{x}^2 - \boldsymbol{x} &= [-1, 2]^2 - [-1, 2] = [-2, 5], \\ \boldsymbol{x}(\boldsymbol{x} - 1) &= [-1, 2]([-1, 2] - 1) = [-4, 2], \\ \text{Best one?}(\boldsymbol{x} - \tfrac{1}{2})^2 - \tfrac{1}{4} &= ([-1, 2] - \tfrac{1}{2})^2 - \tfrac{1}{4} = [-\tfrac{1}{4}, 2]. \end{aligned}$$

Theorem

Suppose that in an expression of $f : \mathbb{R}^n \mapsto \mathbb{R}$ each variable $x_1, \dots, x_n$ appears at most once. The corresponding natural interval extension $\boldsymbol{f}(\boldsymbol{x})$ satisfies for every $\boldsymbol{x} \in \mathbb{IR}^n$: $f(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x})$.

Proof.

Inclusion “$\subseteq$” by the previous theorems.
Inclusion “$\supseteq$” by induction and exactness of interval arithmetic. □

Software

Matlab/Octave libraries

- *Intlab* (by S.M. Rump),
  interval arithmetic and elementary functions
  `http://www.ti3.tu-harburg.de/~rump/intlab/`
- *Versoft* (by J. Rohn),
  verification software written in Intlab
  `http://uivtx.cs.cas.cz/~rohn/matlab/`
- *Lime* (by M. Hladík, J. Horáček et al.),
  interval methods written in Intlab, under development
  `http://kam.mff.cuni.cz/~horacek/projekty/lime/`

Other languages libraries

- *Int4Sci Toolbox* (by Coprin team, INRIA),
  A Scilab Interface for Interval Analysis
  `http://www-sop.inria.fr/coprin/logiciels/Int4Sci/`
- *C++ libraries*: C-XSC, PROFIL/BIAS, BOOST interval, FILIB++,...
- *many others*: for Fortran, Pascal, Lisp, Maple, Mathematica,...

References – books

- G. Alefeld and J. Herzberger.
  Introduction to Interval Computations.
  Academic Press, New York, 1983.
- L. Jaulin, M. Kieffer, O. Didrit, and É. Walter.
  Applied Interval Analysis.
  Springer, London, 2001.
- R. E. Moore.
  Interval Analysis.
  Prentice-Hall, Englewood Cliffs, NJ, 1966.
- R. E. Moore, R. B. Kearfott, and M. J. Cloud.
  Introduction to Interval Analysis.
  SIAM, Philadelphia, PA, 2009.
- A. Neumaier.
  Interval Methods for Systems of Equations.
  Cambridge University Press, Cambridge, 1990.
- J. Rohn.
  A handbook of results on interval linear problems.
  Tech. Rep. 1163, Acad. of Sci. of the Czech Republic, Prague, 2012.
  `http://uivtx.cs.cas.cz/~rohn/publist/!aahandbook.pdf`

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Solution Set

Interval linear equations

Let $\boldsymbol{A} \in \mathbb{IR}^{m\times n}$ and $\boldsymbol{b} \in \mathbb{IR}^{m}$. The family of systems

$$Ax = b, \quad A \in \boldsymbol{A}, \ b \in \boldsymbol{b}.$$

is called interval linear equations and abbreviated as $\boldsymbol{A}x = \boldsymbol{b}$.

Solution set

The solution set is defined

$$\Sigma := \{x \in \mathbb{R}^n : \exists A \in \boldsymbol{A} \exists b \in \boldsymbol{b} : Ax = b\}.$$

Important notice

We do not want to compute $\boldsymbol{x} \in \mathbb{IR}^n$ such that $\boldsymbol{Ax} = \boldsymbol{b}$.

Theorem (Oettli–Prager, 1964)

The solution set Σ is a non-convex polyhedral set described by

$$|A^c x - b^c| \le A^\Delta |x| + b^\Delta.$$

Proof of Oettli–Prager Theorem ($|A^c x - b^c| \le A^\Delta |x| + b^\Delta$)

Let $x \in \Sigma$, that is, $Ax = b$ for some $A \in \boldsymbol{A}$ and $b \in \boldsymbol{b}$. Now,

$$\begin{aligned}|A^c x - b^c| &= |(A^c - A)x + (Ax - b) + (b - b^c)| = |(A^c - A)x + (b - b^c)|\\ &\le |A^c - A||x| + |b - b^c| \le A^\Delta |x| + b^\Delta.\end{aligned}$$

Conversely, let $x \in \mathbb{R}^n$ satisfy the inequalities. Define $y \in [-1,1]^m$ as

$$y_i = \begin{cases} \frac{(A^c x - b^c)_i}{(A^\Delta |x| + b^\Delta)_i} & \text{if } (A^\Delta |x| + b^\Delta)_i > 0,\\ 1 & \text{otherwise.}\end{cases}$$

Now, we have $(A^c x - b^c)_i = y_i (A^\Delta |x| + b^\Delta)_i$, or,

$$A^c x - b^c = \operatorname{diag}(y)(A^\Delta |x| + b^\Delta).$$

Define $z := \operatorname{sgn}(x)$, then $|x| = \operatorname{diag}(z)x$ and we can write

$$A^c x - b^c = \operatorname{diag}(y) A^\Delta \operatorname{diag}(z) x + \operatorname{diag}(y) b^\Delta,$$

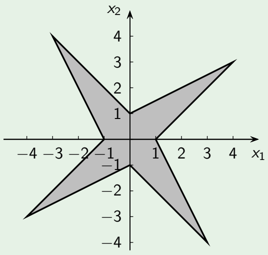
or

$$(A^c - \operatorname{diag}(y) A^\Delta \operatorname{diag}(z))x = b^c + \operatorname{diag}(y) b^\Delta. \qquad \square$$

Example of the Solution Set

Example (Barth & Nuding, 1974))

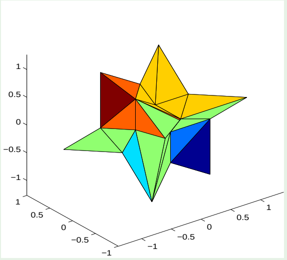
$$\begin{pmatrix} [2,4] & [-2,1]\\ [-1,2] & [2,4]\end{pmatrix} \begin{pmatrix} x_1\\ x_2\end{pmatrix} = \begin{pmatrix} [-2,2]\\ [-2,2]\end{pmatrix}$$



Example of the Solution Set

Example

$$\begin{pmatrix} [3,5] & [1,3] & -[0,2]\\ -[0,2] & [3,5] & [0,2]\\ [0,2] & -[0,2] & [3,5]\end{pmatrix} \begin{pmatrix} x_1\\ x_2\\ x_3\end{pmatrix} = \begin{pmatrix} [-1,1]\\ [-1,1]\\ [-1,1]\end{pmatrix}.$$



Topology of the Solution Set

Proposition

In each orthant, Σ is either empty or a convex polyhedral set.

Proof.

Restriction to the orthant given by $s \in \{\pm 1\}^n$:

$$|A^c x - b^c| \le A^\Delta |x| + b^\Delta, \ \operatorname{diag}(s)x \ge 0.$$

Since $|x| = \operatorname{diag}(s)x$, we have

$$|A^c x - b^c| \le A^\Delta \operatorname{diag}(s)x + b^\Delta, \ \operatorname{diag}(s)x \ge 0.$$

Using $|a| \le b \ \Leftrightarrow a \le b, \ -a \le b$, we get

$$(A^c - A^\Delta \operatorname{diag}(s))x \le \overline{b}, \ (-A^c - A^\Delta \operatorname{diag}(s))x \le -\underline{b}, \ \operatorname{diag}(s)x \ge 0. \quad \square$$

Corollary

The solutions of $\boldsymbol{A}x = \boldsymbol{b}$, $x \ge 0$ *is described by* $\underline{A}x \le \overline{b}, \ \overline{A}x \ge \underline{b}, \ x \ge 0$.

Remark

Checking $\Sigma \ne \emptyset$ and boundedness are NP-hard.

Interval Hull $\square\Sigma$

Goal

Seeing that Σ is complicated, compute $\square\Sigma$ instead.

First idea

Go through all 2^n orthants of $\mathbb{R}^n$, determine interval hull of restricted sets (by solving $2n$ linear programs), and then put together.

Theorem

If $\boldsymbol{A}$ *is regular (each* $A \in \boldsymbol{A}$ *is nonsingular),* Σ *is bounded and connected.*

Theorem (Jansson, 1997)

When $\Sigma \ne \emptyset$*, then exactly one of the following alternatives holds true:*

1. Σ *is bounded and connected.*
2. *Each topologically connected component of* Σ *is unbounded.*

Second idea – Jansson's algorithm

Check the orthant with $(A^c)^{-1}b^c$ and then all the topologically connected.

Polynomial Cases

Two basic polynomial cases

1. $A^c = I_n$,
2. $\boldsymbol{A}$ is inverse nonnegative, i.e., $A^{-1} \ge 0 \ \forall A \in \boldsymbol{A}$.

Theorem (Kuttler, 1971)

$\boldsymbol{A} \in \mathbb{IR}^{n \times n}$ *is inverse nonnegative if and only if* $\underline{A}^{-1} \ge 0$ *and* $\overline{A}^{-1} \ge 0$.

Theorem

Let $\boldsymbol{A} \in \mathbb{IR}^{n \times n}$ *be inverse nonnegative. Then*

1. $\square\Sigma = [\overline{A}^{-1}\underline{b}, \underline{A}^{-1}\overline{b}]$ *when* $\underline{b} \ge 0$,
2. $\square\Sigma = [\underline{A}^{-1}\underline{b}, \overline{A}^{-1}\overline{b}]$ *when* $\underline{b} \le 0$,
3. $\square\Sigma = [\underline{A}^{-1}\underline{b}, \underline{A}^{-1}\overline{b}]$ *when* $0 \in \boldsymbol{b}$.

Proof.

1. Let $A \in \boldsymbol{A}$ and $b \in \boldsymbol{b}$. Since $\overline{b} \ge b \ge \underline{b} \ge 0$ and $\underline{A}^{-1} \ge A^{-1} \ge \overline{A}^{-1} \ge 0$, we get $\overline{A}^{-1}\underline{b} \le A^{-1}b \le \underline{A}^{-1}\overline{b}$. $\square$

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Preconditioning

Enclosure

Since Σ is hard to determine and deal with, we seek for enclosures

$$\boldsymbol{x} \in \mathbb{IR}^n \text{ such that } \Sigma \subseteq \boldsymbol{x}.$$

Many methods for enclosures exists, usually employ preconditioning.

Preconditioning (Hansen, 1965)

Let $C \in \mathbb{R}^{n \times n}$. The preconditioned system of equations:

$$(C\boldsymbol{A})x = C\boldsymbol{b}.$$

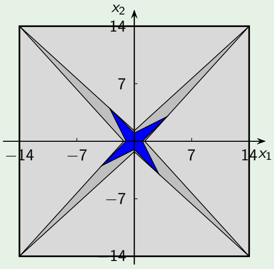
Remark

- the solution set of the preconditioned systems contains Σ
- usually, we use $C \approx (A^c)^{-1}$
- then we can compute the best enclosure (Hansen, 1992, Bliek, 1992, Rohn, 1993)

Preconditioning

Example (Barth & Nuding, 1974))

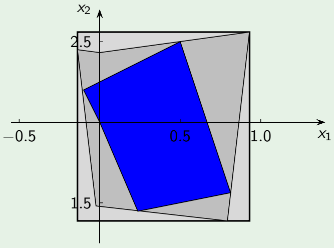
$$\begin{pmatrix} [2,4] & [-2,1] \\ [-1,2] & [2,4] \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} [-2,2] \\ [-2,2] \end{pmatrix}$$



Preconditioning

Example (typical case)

$$\begin{pmatrix} [6,7] & [2,3] \\ [1,2] & -[4,5] \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} [6,8] \\ -[7,9] \end{pmatrix}$$



Interval Gaussian Elimination

Interval Gaussian elimination = Gaussian elimination + interval arithmetic.

Example (Barth & Nuding, 1974))

$$\begin{pmatrix} [2,4] & [-2,1] \\ [-1,2] & [2,4] \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} [-2,2] \\ [-2,2] \end{pmatrix}$$

Then we proceed as follows

$$\begin{pmatrix} [2,4] & [-2,1] & [-2,2] \\ [-1,2] & [2,4] & [-2,2] \end{pmatrix} \sim \begin{pmatrix} [2,4] & [-2,1] & [-2,2] \\ 0 & [1,6] & [-4,4] \end{pmatrix}.$$

By back substitution, we compute

$$\boldsymbol{x}_2 = [-4,4],$$
$$\boldsymbol{x}_1 = ([-2,2] - [-2,1] \cdot [-4,4]) \,/\, [2,4] = [-5,5].$$

Interval Jacobi and Gauss-Seidel Iterations

Idea

From the ith equation of $Ax = b$ we get

$$x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j - \sum_{j=i+1}^{n} a_{ij} x_j \right).$$

If $\boldsymbol{x}^0 \supseteq \Sigma$ is an initial enclosure, then

$$x_i \in \frac{1}{\boldsymbol{a}_{ii}} \left(\boldsymbol{b}_i - \sum_{j \neq i} \boldsymbol{a}_{ij} \boldsymbol{x}_j^0 \right), \quad \forall x \in \Sigma.$$

Thus, we can tighten the enclosure by iterations

Interval Jacobi / Gauss–Seidel iterations ($k = 1, 2, \dots$)

```
for $i = 1, \dots, n$ do
    $\boldsymbol{x}_i^k := \frac{1}{\boldsymbol{a}_{ii}} \left( \boldsymbol{b}_i - \sum_{j \neq i} \boldsymbol{a}_{ij} \boldsymbol{x}_j^{k-1} \right) \cap \boldsymbol{x}_i^{k-1}$;
end for
```

Krawczyk Iterations

Krawczyk operator

Krawczyk operator $K \colon \mathbb{IR}^n \to \mathbb{IR}^n$ reads

$$K(\boldsymbol{x}) := C\boldsymbol{b} + (I_n - C\boldsymbol{A})\boldsymbol{x}$$

Proposition

If $x \in \boldsymbol{x} \cap \Sigma$, then $x \in K(\boldsymbol{x})$.

Proof.

Let $x \in \boldsymbol{x} \cap \Sigma$, so $Ax = b$ for some $A \in \boldsymbol{A}$ and $b \in \boldsymbol{b}$. Thus $CAx = Cb$, whence $x = Cb + (I_n - CA)x \in C\boldsymbol{b} + (I_n - C\boldsymbol{A})\boldsymbol{x} = K(\boldsymbol{x})$. □

Krawczyk iterations

Let $\boldsymbol{x}^0 \supseteq \Sigma$ is an initial enclosure, and iterate ($k = 1, 2, \dots$):

```
$\boldsymbol{x}^k := K(\boldsymbol{x}^{k-1}) \cap \boldsymbol{x}^{k-1}$;
```

ε-inflation

Theorem

Let $\boldsymbol{x} \in \mathbb{IR}^n$ and $C \in \mathbb{R}^{n \times n}$. If

$$K(\boldsymbol{x}) = C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{x} \subseteq \mathrm{int}\, \boldsymbol{x},$$

then C is nonsingular, $\boldsymbol{A}$ is regular, and $\Sigma \subseteq \boldsymbol{x}$.

Proof.

Existence of a solution based on Brouwer's fixed-point theorem.
Nonsingularity and uniqueness based on the Perron–Frobenius theory. □

Remark

- A reverse iteration method to the Krawczyk method.
- It starts with a small box around $(A^c)^{-1} b^c$, and then iteratively inflates the box.
- Implemented in Intlab v. 6.

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Regularity

Definition (Regularity)

$\boldsymbol{A} \in \mathbb{IR}^{n\times n}$ is regular if each $A \in \boldsymbol{A}$ is nonsingular.

Theorem

Checking regularity of an interval matrix is co-NP-hard.

Forty necessary and sufficient conditions for regularity of $\boldsymbol{A}$ by Rohn (2010):

1. The system $|A^c x| \le A^\Delta |x|$ has the only solution $x = 0$.
2. $\det(A^c - \operatorname{diag}(y) A^\Delta \operatorname{diag}(z))$ is constantly either positive or negative for each $y, z \in \{\pm 1\}^n$.
3. For each $y \in \{\pm 1\}^n$, the system $A^c x - \operatorname{diag}(y) A^\Delta |x| = y$ has a solution.
4. ...

Regularity – Sufficient / Necessary Conditions

Theorem (Beeck, 1975)

If $\rho(|(A^c)^{-1}|A^\Delta) < 1$, then $\boldsymbol{A}$ is regular.

Proof.

Precondition $\boldsymbol{A}$ by the midpoint inverse: $\boldsymbol{M} := (A^c)^{-1}\boldsymbol{A}$. Now,

$$M^c = I_n, \quad M^\Delta = |(A^c)^{-1}|A^\Delta,$$

and for each $M \in \boldsymbol{M}$ we have

$$|M - M^c| = |M - I_n| \le M^\Delta.$$

From the theory of eigenvalues of nonnegative matrices it follows

$$\rho(M - I_n) \le \rho(M^\Delta) < 1,$$

so M has no zero eigenvalue and is nonsingular. □

Necessary condition

If $0 \in \boldsymbol{A}x$ for some $0 \ne x \in \mathbb{R}^n$, then $\boldsymbol{A}$ is not regular. (Try $x := (A^c)^{-1}_{*i}$)

Next Section

1. Motivation
2. Interval Computations
3. Interval Functions
4. Interval Linear Equations – Solution Set
5. Interval Linear Equations – Enclosure Methods
6. Regularity of Interval Matrices
7. Parametric Interval Systems

Parametric Interval Systems

Parametric interval systems

$$A(p)x = b(p),$$

where the entries of $A(p)$ and $b(p)$ depend on parameters $p_1 \in \boldsymbol{p}_1, \dots, p_K \in \boldsymbol{p}_K$.

Definition (Solution set)

$$\Sigma_p = \{x \in \mathbb{R}^n : A(p)x = b(p) \text{ for some } p \in \boldsymbol{p}\}.$$

Relaxation

Compute (enclosures of) the ranges $\boldsymbol{A} := A(\boldsymbol{p})$ and $\boldsymbol{b} := b(\boldsymbol{p})$ and solve

$$\boldsymbol{A}x = \boldsymbol{b}.$$

May overestimate a lot!

Special Case: Parametric Linear Interval Systems

Parametric linear interval systems

$$A(p)x = b(p),$$

where

$$A(p) = \sum_{k=1}^{K} A_k p_k, \quad b(p) = \sum_{k=1}^{K} b_k p_k$$

and $p \in \boldsymbol{p}$ for some given interval vector $\boldsymbol{p} \in \mathbb{IR}^K$, matrices $A_1, \dots, A_K \in \mathbb{R}^{n\times n}$ and vectors $b_1, \dots, b_n \in \mathbb{R}^n$.

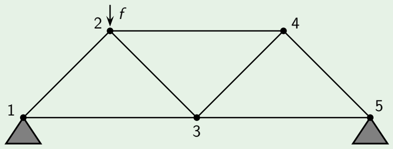
Remark

It covers many structured matrices: symmetric, skew-symmetric, Toeplitz or Hankel.

Parametric Linear Interval Systems – Example

Example (Displacements of a truss structure (Skalna, 2006))

The 7-bar truss structure subject to downward force.
The stiffnesses s_{ij} of bars are uncertain.
The displacements d of the nodes, are solutions of the system $Kd = f$, where f is the vector of forces.



Parametric Linear Interval Systems – Example

Example (Displacements of a truss structure (Skalna, 2006))

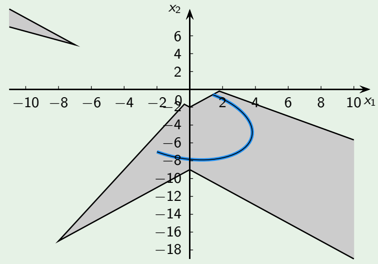
The 7-bar truss structure subject to downward force.
The stiffnesses s_{ij} of bars are uncertain.
The displacements d of the nodes, are solutions of the system $Kd = f$, where f is the vector of forces.

$$K = \begin{pmatrix} \frac{s_{12}}{2} + s_{13} & -\frac{s_{12}}{2} & -\frac{s_{12}}{2} & -s_{13} & 0 & 0 & 0 \\ -\frac{s_{21}}{2} & \frac{s_{21}+s_{23}}{2} + s_{24} & \frac{s_{21}-s_{23}}{2} & -\frac{s_{23}}{2} & \frac{s_{23}}{2} & -s_{24} & 0 \\ -\frac{s_{21}}{2} & \frac{s_{21}-s_{23}}{2} & \frac{s_{21}+s_{23}}{2} & \frac{s_{23}}{2} & -\frac{s_{23}}{2} & 0 & 0 \\ -s_{31} & -\frac{s_{32}}{2} & \frac{s_{32}}{2} & s_{31} + \frac{s_{32}+s_{34}}{2} + s_{35} & \frac{s_{34}-s_{32}}{2} & -\frac{s_{34}}{2} & -\frac{s_{34}}{2} \\ 0 & \frac{s_{32}}{2} & -\frac{s_{32}}{2} & \frac{s_{34}-s_{32}}{2} & \frac{s_{34}+s_{32}}{2} & -\frac{s_{34}}{2} & -\frac{s_{34}}{2} \\ 0 & -s_{42} & 0 & -\frac{s_{43}}{2} & -\frac{s_{43}}{2} & s_{42} + \frac{s_{43}+s_{45}}{2} & 0 \\ 0 & 0 & 0 & -\frac{s_{43}}{2} & -\frac{s_{43}}{2} & 0 & \frac{s_{43}+s_{45}}{2} \end{pmatrix}$$

Parametric Linear Interval Systems – Example

Example

$$\begin{pmatrix} 1-2p & 1 \\ 2 & 4p-1 \end{pmatrix} x = \begin{pmatrix} 7p-9 \\ 3-2p \end{pmatrix}, \quad p \in \boldsymbol{p} = [0,1].$$



Parametric Linear Interval Systems – Solution Set

Theorem

If $x \in \Sigma_p$, then it solves

$$|A(p^c)x - b(p^c)| \le \sum_{k=1}^{K} p_k^{\Delta} |A^k x - b^k|.$$

Proof.

$$|A(p^c)x - b(p^c)| = \left|\sum_{k=1}^{K} p_k^c (A^k x - b^k)\right| = \left|\sum_{k=1}^{K} p_k^c (A^k x - b^k) - \sum_{k=1}^{K} p_k (A^k x - b^k)\right|$$

$$= \left|\sum_{k=1}^{K} (p_k^c - p_k)(A^k x - b^k)\right| \le \sum_{k=1}^{K} |p_k^c - p_k| |A^k x - b^k| \le \sum_{k=1}^{K} p_k^{\Delta} |A^k x - b^k|. \quad \square$$

- Popova (2009) showed that it is the complete characterization of Σ_p as long as no interval parameter appears in more than one equation.
- Checking $x \in \Sigma_p$ for a given $x \in \mathbb{R}^n$ is a polynomial problem via linear programming.

Parametric Linear Interval Systems – Enclosures

Relaxation and preconditioning – First idea

Evaluate $\boldsymbol{A} := A(\boldsymbol{p})$, $\boldsymbol{b} := b(\boldsymbol{p})$, choose $C \in \mathbb{R}^{n \times n}$ and solve

$$(C\boldsymbol{A})x = C\boldsymbol{b}.$$

Relaxation and preconditioning – Second idea

Solve $\boldsymbol{A}'x = \boldsymbol{b}'$, where

$$\boldsymbol{A}' := \sum_{k=1}^{K} (CA^k)\boldsymbol{p}_k, \quad \boldsymbol{b}' := \sum_{k=1}^{K} (Cb^k)\boldsymbol{p}_k.$$

Second idea is provably better

Due to sub-distributivity law,

$$\boldsymbol{A}' := \sum_{k=1}^{K} (CA^k)\boldsymbol{p}_k \subseteq C\left(\sum_{k=1}^{K} A^k \boldsymbol{p}_k\right) = (C\boldsymbol{A}).$$

Special Case: Symmetric Systems

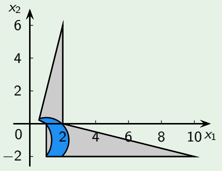
The symmetric solution set of $\boldsymbol{A}x = \boldsymbol{b}$

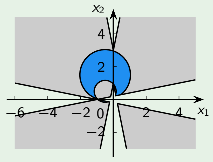
$$\{x \in \mathbb{R}^n : Ax = b \text{ for some symmetric } A \in \boldsymbol{A} \text{ and } b \in \boldsymbol{b}\}.$$

Described by $\frac{1}{2}(4^n - 3^n - 2 \cdot 2^n + 3) + n$ nonlinear inequalities (H., 2008).

Example

$$\boldsymbol{A} = \begin{pmatrix} [1,2] & [0,a] \\ [0,a] & -1 \end{pmatrix}, \ \boldsymbol{b} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}. \qquad \boldsymbol{A} = \begin{pmatrix} -1 & [-5,5] \\ [-5,5] & 1 \end{pmatrix}, \ \boldsymbol{b} = \begin{pmatrix} 1 \\ [1,3] \end{pmatrix}.$$





Application: Least Square Solutions

Least square solution

Let $A \in \mathbb{IR}^{m \times n}$, $b \in \mathbb{IR}^m$ and $m > n$. The least square solution of

$$Ax = b,$$

is defined as the optimal solution of

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2,$$

or, alternatively as the solution to

$$A^T Ax = A^T b.$$

Interval least square solution set

Let $\boldsymbol{A} \in \mathbb{IR}^{m \times n}$ and $\boldsymbol{b} \in \mathbb{IR}^m$ and $m > n$. The LSQ solution set is defined

$$\Sigma_{LSQ} := \{x \in \mathbb{R}^n : \exists A \in \boldsymbol{A} \exists b \in \boldsymbol{b} : A^T Ax = A^T b\}.$$

Proposition

Σ_{LSQ} is contained in the solution set to $\boldsymbol{A}^T\boldsymbol{A}x = \boldsymbol{A}^T\boldsymbol{b}$.

Application: Least Square Solutions

Proposition

Σ_{LSQ} is contained in the solution set to

$$\begin{pmatrix} 0 & \boldsymbol{A}^T \\ \boldsymbol{A} & I_m \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{b} \end{pmatrix}. \tag{1}$$

Proof.

Let $A \in \boldsymbol{A}$, $b \in \boldsymbol{b}$. If x, y solve

$$A^T y = 0, \;\; Ax + y = b,$$

then

$$0 = A^T(b - Ax) = A^T b - A^T Ax,$$

and vice versa. □

Proposition

Relaxing the dependencies, the solution set to $\boldsymbol{A}^T \boldsymbol{A} x = \boldsymbol{A}^T \boldsymbol{b}$ is contained in the solution set to (1).

Introduction to Interval Computation and Numerical Verification
part II.

Milan Hladík

Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
`http://kam.mff.cuni.cz/~hladik/`

Seminář numerické analýzy a zimní škola – SNA'17
Ostrava, 30. ledna – 3. února 2017

Outline

1. More on Interval Functions
2. Application: Solving Nonlinear Equations
3. Application: Verification
4. AE Solution Set
5. Eigenvalues of Symmetric Interval Matrices
6. Conclusion

Next Section

1. More on Interval Functions
2. Application: Solving Nonlinear Equations
3. Application: Verification
4. AE Solution Set
5. Eigenvalues of Symmetric Interval Matrices
6. Conclusion

Mean value form

Theorem

Let $f:\mathbb{R}^n \mapsto \mathbb{R}$, $\boldsymbol{x} \in \mathbb{IR}^n$ and $a \in \boldsymbol{x}$. Then

$$f(\boldsymbol{x}) \subseteq f(a) + \nabla f(\boldsymbol{x})^T(\boldsymbol{x} - a),$$

Proof.

By the mean value theorem, for any $x \in \boldsymbol{x}$ there is $c \in \boldsymbol{x}$ such that

$$f(x) = f(a) + \nabla f(c)^T(x-a) \in f(a) + \nabla f(\boldsymbol{x})^T(\boldsymbol{x} - a). \qquad \square$$

Improvements

- successive mean value form

$$\begin{aligned} f(\boldsymbol{x}) \subseteq f(a) &+ f'_{x_1}(\boldsymbol{x}_1, a_2, \dots, a_n)(\boldsymbol{x}_1 - a_1) \\ &+ f'_{x_2}(\boldsymbol{x}_1, \boldsymbol{x}_2, a_3 \dots, a_n)(\boldsymbol{x}_2 - a_2) + \dots \\ &+ f'_{x_n}(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n-1}, \boldsymbol{x}_n)(\boldsymbol{x}_n - a_n). \end{aligned}$$

- replace derivatives by slopes

Slopes

Slope form enclosure

$$f(\boldsymbol{x}) \subseteq f(a) + S(\boldsymbol{x}, a)(\boldsymbol{x} - a),$$

where $a \in \boldsymbol{x}$ and

$$S(x,a) := \begin{cases} \frac{f(x)-f(a)}{x-a} & \text{if } x \neq a, \\ f'(x) & \text{otherwise.} \end{cases}$$

Remarks

- Slopes can be replaced by derivatives, but slopes are tighter.
- Slopes can be computed in a similar way as derivatives.

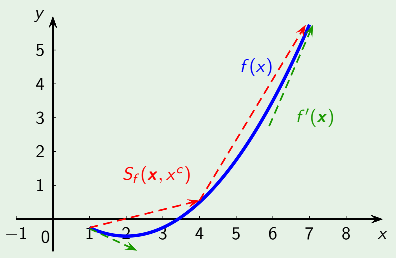
function	its slope $S(x,a)$
x	1
$f(x) \pm g(x)$	$S_f(x,a) \pm S_g(x,a)$
$f(x) \cdot g(x)$	$S_f(x,a)g(a) + f(x)S_g(x,a)$
$e^{f(x)}$	$e^{f(x)} S_f(x,a)$

Slopes

Example

$$f(x) = \tfrac{1}{4}x^2 - x + \tfrac{1}{2}, \qquad \boldsymbol{x} = [1,7].$$

$$f'(\boldsymbol{x}) = [-\tfrac{1}{2}, \tfrac{5}{2}], \qquad S_f(\boldsymbol{x}, x^c) = [\tfrac{1}{4}, \tfrac{7}{4}].$$



Notice: Slopes cannot be used for monotonicity checking.

Next Section

1. More on Interval Functions
2. Application: Solving Nonlinear Equations
3. Application: Verification
4. AE Solution Set
5. Eigenvalues of Symmetric Interval Matrices
6. Conclusion

Nonlinear Equations

Problem statement

Find all solutions to

$$f_j(x_1,\dots,x_n)=0,\quad j=1,\dots,j^*$$

inside the box $\boldsymbol{x}^0\in\mathbb{IR}^n$.

Theorem (Zhu, 2005)

For a polynomial $p(x_1,\dots,x_n)$, there is no algorithm solving

$$p(x_1,\dots,x_n)^2+\sum_{i=1}^{n}\sin^2(\pi x_i)=0.$$

Proof.

From Matiyasevich's theorem solving the 10th Hilbert problem. □

Remark

Using the arithmetical operations only, the problem is decidable by Tarski's theorem (1951).

Interval Newton method

Classical Newton method

…is an iterative method

$$x^{k+1}:=x^k-\nabla f(x^k)^{-1}f(x^k),\quad k=0,\dots$$

Cons

- Can miss some solutions
- Not verified (Are we really close to the true solution?)

Interval Newton method – Stupid intervalization

$$\boldsymbol{x}^{k+1}:=\boldsymbol{x}^k-\nabla f(\boldsymbol{x}^k)^{-1}f(\boldsymbol{x}^k),\quad k=0,\dots$$

Interval Newton method – Good intervalization

$$N(x^k,\boldsymbol{x}^k):=x^k-\nabla f(\boldsymbol{x}^k)^{-1}f(x^k),$$
$$\boldsymbol{x}^{k+1}:=\boldsymbol{x}^k\cap N(\boldsymbol{x}^k),\qquad k=0,\dots$$

Interval Newton method

Theorem (Moore, 1966)

If $x,x^0\in\boldsymbol{x}$ and $f(x)=0$, then $x\in N(x^0,\boldsymbol{x})$.

Proof.

By the Mean value theorem,

$$f_i(x)-f_i(x^0)=\nabla f_i(c_i)^T(x-x^0),\quad \forall i=1,\dots,n.$$

If x is a root, we have

$$-f_i(x^0)=\nabla f_i(c_i)^T(x-x^0).$$

Define $A\in\mathbb{R}^{n\times n}$ such that its ith row is equal to $\nabla f_i(c_i)^T$. Hence

$$-f(x^0)=A(x-x^0),$$

from which

$$x=x^0-A^{-1}f(x^0)\in x^0-\nabla f(\boldsymbol{x})^{-1}f(x^0).$$

Notice, that this does not mean that there is $c\in\boldsymbol{x}$ such that

$$-f(x^0)=\nabla f(c)(x-x^0).$$ □

Interval Newton method

Theorem (Nickel, 1971)

If $\emptyset\neq N(x^0,\boldsymbol{x})\subseteq\boldsymbol{x}$, then there is a unique root in $\boldsymbol{x}$ and $\nabla f(\boldsymbol{x})$ is regular.

Proof.

"Regularity." Easy.

"Existence." By Brouwer's fixed-point theorem.

[Any continuous mapping of a compact convex set into itself has a fixed point.]

"Uniqueness." If there are two roots $y_1\neq y_2$ in $\boldsymbol{x}$, then by the Mean value theorem,

$$f(y_1)-f(y_2)=A(y_1-y_2)$$

for some $A\in\nabla f(\boldsymbol{x})$;. Since $f(y_1)=f(y_2)=0$, we get

$$A(y_1-y_2)=0$$

and by the nonsingularity of A, the roots are identical. □

Interval Newton method

Practical implementation

Instead of

$$N(x^k,\boldsymbol{x}^k):=x^k-\nabla f(\boldsymbol{x}^k)^{-1}f(x^k)$$

let $N(x^k,\boldsymbol{x}^k)$ be an enclosure of the solution set (with respect to x) of

$$\nabla f(\boldsymbol{x})(x-x^0)=-f(x^0).$$

Extended interval arithmetic

So far

$$\frac{[12,15]}{[-2,3]}=(-\infty,\infty).$$

Now,

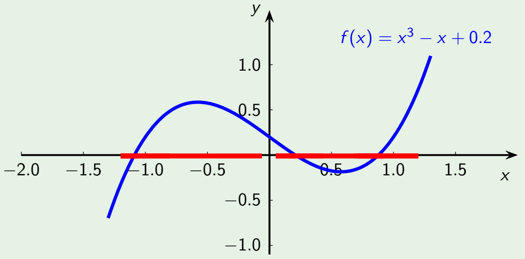
$$\boldsymbol{a}/\boldsymbol{b}:=\{a/b\colon a\in\boldsymbol{a},0\neq b\in\boldsymbol{b}\}.$$

So,

$$\frac{[12,15]}{[-2,3]}=(-\infty,-6]\cup[4,\infty).$$

Interval Newton method

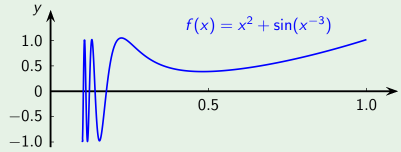
Example



In six iterations precision 10^{-11} (quadratic convergence).

Interval Newton method

Example (Moore, 1993)



All 318 roots of in the interval $[0.1, 1]$ found with accuracy 10^{-10}.
The left most root is contained in $[0.10003280626, 0.10003280628]$.

Summary

- $N(x^0, \boldsymbol{x})$ contains all solutions in $\boldsymbol{x}$
- If $\boldsymbol{x} \cap N(x^0, \boldsymbol{x}) = \emptyset$, then there is no root in $\boldsymbol{x}$
- If $\emptyset \neq N(x^0, \boldsymbol{x}) \subseteq \boldsymbol{x}$, then there is a unique root in $\boldsymbol{x}$

Krawczyk method

Krawczyk operator

Let $x^0 \in \boldsymbol{x}$ and $C \in \mathbb{R}^{n\times n}$, usually $C \approx \nabla f(x^0)^{-1}$. Then

$$K(\boldsymbol{x}) := x^0 - Cf(x^0) + (I_n - C\nabla f(\boldsymbol{x}))(\boldsymbol{x} - x^0).$$

Theorem

Any root of $f(x)$ in $\boldsymbol{x}$ is included in $K(\boldsymbol{x})$.

Proof.

If x^1 is a root of $f(x)$, then it is a fixed point of

$$g(x) := x - Cf(x).$$

By the mean value theorem,

$$g(x^1) \in g(x^0) + \nabla g(\boldsymbol{x})(x^1 - x^0),$$

whence

$$\begin{aligned} x^1 \in g(\boldsymbol{x}) &\subseteq g(x^0) + \nabla g(\boldsymbol{x})(\boldsymbol{x} - x^0) \\ &= x^0 - Cf(x^0) + (I_n - C\nabla f(\boldsymbol{x}))(\boldsymbol{x} - x^0). \end{aligned}$$

□

Krawczyk method

Theorem

If $K(\boldsymbol{x}) \subseteq \boldsymbol{x}$, then there is a root in $\boldsymbol{x}$.

Proof.

Recall

$$g(x) := x - Cf(x).$$

By the proof of the previous Theorem, $K(\boldsymbol{x}) \subseteq \boldsymbol{x}$ implies

$$g(\boldsymbol{x}) \subseteq \boldsymbol{x}.$$

Thus, there is a fixed point $x^0 \in \boldsymbol{x}$ of $g(x)$,

$$g(x^0) = x^0 - Cf(x^0) = x^0,$$

so x^0 is a root of $f(x)$. □

Krawczyk method

Theorem (Kahan, 1968)

If $K(\boldsymbol{x}) \subseteq int\, \boldsymbol{x}$, then there is a unique root in $\boldsymbol{x}$ and $\nabla f(\boldsymbol{x})$ is regular.

Recall Theorem from "ε-inflation" (for solving $\boldsymbol{A}x = \boldsymbol{b}$)

Let $\boldsymbol{x} \in \mathbb{IR}^n$ and $C \in \mathbb{R}^{n\times n}$. If

$$K(\boldsymbol{x}) = C\boldsymbol{b} + (I_n - C\boldsymbol{A})\boldsymbol{x} \subseteq int\, \boldsymbol{x},$$

then C is nonsingular, $\boldsymbol{A}$ is regular, and $\Sigma \subseteq \boldsymbol{x}$.

Proof.

The inclusion $K(\boldsymbol{x}) \subseteq int\, \boldsymbol{x}$ reads

$$-Cf(x^0) + (I_n - C\nabla f(\boldsymbol{x}))(\boldsymbol{x} - x^0) \subseteq int\,(\boldsymbol{x} - x^0)$$

Apply the above Theorem for

$$\boldsymbol{b} := -f(x^0), \quad \boldsymbol{A} := \nabla f(\boldsymbol{x}), \quad \boldsymbol{x} := \boldsymbol{x} - x^0$$

We have that $\nabla f(\boldsymbol{x})$ is regular, which implies uniqueness. □

More general constraints

Constraints

- equations $h_i(x) = 0$, $i = 1, \dots, I$
- inequalities $g_j(x) \leq 0$, $j = 1, \dots, J$
- may be others, but not considered here
  ($\neq$, quantifications, logical operators, lexicographic orderings, ...)

Problem

Denote by Σ the set of solutions in an initial box $\boldsymbol{x}^0 \in \mathbb{IR}^n$?
Problem: How to describe Σ?

Subpavings

Split $\boldsymbol{x}$ into a union of three sets of boxes such that

- the first set has boxes provably containing no solution
- the second set has boxes that provably consist of only solutions
- the third set has boxes which may or may not contain a solution

Subpaving Example

Example

$$x^2 + y^2 \leq 16,$$
$$x^2 + y^2 \geq 9$$

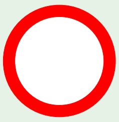


Figure: Exact solution set

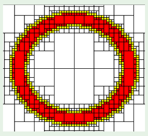


Figure: Subpaving approximation

Subpaving Example

Example

$$(x-1)^2 + (y-2)^2 \leq \tfrac{1}{7},$$
$$(x^2 + y^2 - 9)(\tfrac{1}{3}x - y^2) \geq \tfrac{1}{2}$$

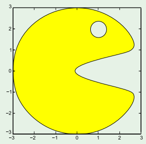


Figure: Exact solution set

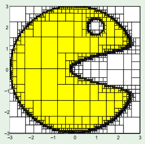


Figure: Subpaving approximation

Subpaving Algorithm

Branch & Bound approach

- divide $\boldsymbol{x}^0$ recursively into sub-boxes,
- remove sub-boxes with provably no solutions
- contract sub-boxes

Some simple tests

- Test for $\boldsymbol{x} \subseteq \Sigma$:
  - no equations and $\overline{g}_j(\boldsymbol{x}) \leq 0\ \forall j$
- Test for $\boldsymbol{x} \cap \Sigma = \emptyset$:
  - $0 \notin h_i(\boldsymbol{x})$ for some i
  - $\underline{g}_j(\boldsymbol{x}) > 0$ for some j

Also very important

- Which box to choose (data structure fo $\mathcal{L}$)?
- How to divide the box? (which coordinate, which place, how many sub-boxex)

A Simple Contractor – Constraint Propagation

Example

Consider the constraint

$$x + yz = 7, \quad x \in [0,3],\ y \in [3,5],\ z \in [2,4].$$

- Express x

$$x = 7 - yz \in 7 - [3,5][2,4] = [-13,1].$$

Thus, the domain for x is $[0,3] \cap [-13,1] = [0,1]$.

- Express y

$$y = (7-x)/z \in (7 - [0,1])/[2,4] = [1.5,\ 3.5].$$

Thus, the domain for y is $[3,5] \cap [1.5,\ 3.5] = [3,\ 3.5]$.

- Express z

$$z = (7-x)/y \in (7 - [0,1])/[3,\ 3.5] = [\tfrac{12}{7}, \tfrac{7}{3}].$$

Thus, the domain for z is $[2,4] \cap [\tfrac{12}{7}, \tfrac{7}{3}] = [2, \tfrac{7}{3}]$.

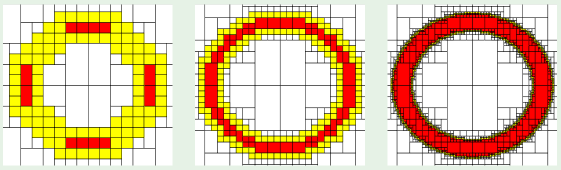
No further propagation needed as each variable appears just once.

Other Techniques

Other techniques

- Various kinds of consistencies (2B, 3B,...), shaving,...

Example (thanks to Elif Garajová)



$\varepsilon = 1.0$
time: 0.952 s

$\varepsilon = 0.5$
time: 2.224 s

$\varepsilon = 0.125$
time: 9.966 s

Software

Free constraint solving software

- *Alias* (by Jean-Pierre Merlet, COPRIN team),
  A C++ library for system solving, with Maple interface,
  `http://www-sop.inria.fr/coprin/logiciels/ALIAS/ALIAS-C++/ALIAS-C++.html`
- *Quimper* (by Gill Chabert and Luc Jaulin),
  written in an interval C++ library IBEX,
  a language for interval modelling and handling constraints,
  `http://www.emn.fr/z-info/ibex`
- *RealPaver* (by L. Granvilliers and F. Benhamou),
  a C++ package for modeling and solving nonlinear and nonconvex constraint satisfaction problems,
  `http://pagesperso.lina.univ-nantes.fr/info/perso/permanents/granvil/realpaver`
- *RSolver* (by Stefan Ratschan),
  solver for quantified constraints over the real numbers,
  implemented in the programming language OCaml,
  `http://rsolver.sourceforge.net/`

References

G. Alefeld and J. Herzberger.
Introduction to Interval Computations.
Academic Press, New York, 1983.

F. Benhamou and L. Granvilliers.
Continuous and interval constraints.
In *Handbook of Constraint Programming*, chap. 16. Elsevier, 2006.

G. Chabert and L. Jaulin.
Contractor programming.
Artif. Intell., 173(11):1079-1100, 2009.

F. Goualard and C. Jermann.
A reinforcement learning approach to interval constraint propagation.
Constraints, 13(1):206–226, 2008.

L. Jaulin, M. Kieffer, O. Didrit, and É. Walter.
Applied Interval Analysis.
Springer, London, 2001.

Next Section

1 More on Interval Functions

2 Application: Solving Nonlinear Equations

3 Application: Verification

4 AE Solution Set

5 Eigenvalues of Symmetric Interval Matrices

6 Conclusion

Introduction

Rigorous computation

What and why?

Can we obtain rigorous numerical results by using floating-point arithmetic?

Yes, by extending to interval arithmetic. Direct usage is however not effective!

Example (Amplification factor for the interval Gaussian elimination)

$n = 20$	$n = 50$	$n = 100$	$n = 170$
10^2	10^5	10^{10}	10^{16}

Advise

Postpone interval computation to the very end.

Verification

Verification

Compute a solution by floating-point arithmetic, and then to verify that the result is correct or determine rigorous distance to a true solution.

Typically, we can prove uniqueness (=the problem is well posed). Therefore, verifying singularity of a matrix cannot be performed!

What we will do

As an example, we show a verification method for the problem of finding a root of a function $f : \mathbb{R}^n \to \mathbb{R}^n$.

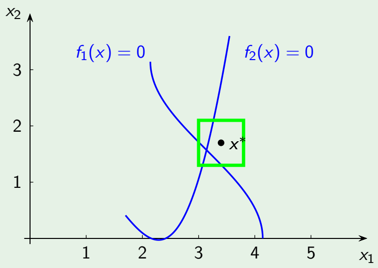
Problem statement

Given $x^* \in \mathbb{R}^n$ a numerically computed (=approximate) solution of $f(x) = 0$, find a small interval $0 \in \boldsymbol{y} \in \mathbb{IR}^n$ such that the true solution lies in $x^* + \boldsymbol{y}$.

Illustration of Verification

Example

Illustration of the verification of x^* to be a solution of $f(x) = 0$.



Ingredients

Brouwer fixed-point theorem

Let U be a convex compact set in $\mathbb{R}^n$ and $g\colon U \to U$ a continuous function. Then there is a fixed point, i.e., $\exists x \in U : g(x) = x$.

Observation

Finding a root of $f(x)$ is equivalent to finding a fixed-point of the function $g(y) \equiv y - C \cdot f(x^* + y)$, where C is any nonsingular matrix of order n.

Perron theory of nonnegative matrices

- If $|A| \le B$, then $\rho(A) \le \rho(B)$.
  ($\le$ is meant entrywise and $\rho(\cdot)$ is the spectral radius)
- If $A \ge 0$, $x > 0$ and $Ax < \alpha x$, then $\rho(A) < \alpha$.

Lemma

If $\boldsymbol{z} + \boldsymbol{R}\boldsymbol{y} \subseteq \operatorname{int} \boldsymbol{y}$, then $\rho(R) < 1$ for every $R \in \boldsymbol{R}$.

Proof. $|R|y^\Delta < y^\Delta$, whence by Perron theory $\rho(R) < 1$. □

Cooking

Theorem

Suppose $0 \in \mathbf{y}$. *Now if*

$$-C \cdot f(x^*) + (I - C \cdot \nabla f(x^* + \mathbf{y})) \cdot \mathbf{y} \subseteq \mathit{int}\, \mathbf{y},$$

then:

- *C and every matrix in $\nabla f(x^* + \mathbf{y})$ are nonsingular, and*
- *there is a unique root of $f(x)$ in $x^* + \mathbf{y}$.*

Proof.

By the mean value theorem,

$$f(x^* + y) \in f(x^*) + \nabla f(x^* + \mathbf{y})y.$$

By the assumptions, the function

$$g(y) = y - C \cdot f(x^* + y) \in -C \cdot f(x^*) + (I - C \cdot \nabla f(x^* + \mathbf{y}))\mathbf{y} \subseteq \mathit{int}\, \mathbf{y}$$

has a fixed point, which shows "existence".

By Lemma, C and $\nabla f(x^* + \mathbf{y})$ are nonsingular, implying "uniqueness". □

Cooking

Implementation

- take $C \approx \nabla f(x^*)^{-1}$ (numerically computed inverse),
- take $\mathbf{y} := C \cdot f(x^*)$ and repeat inflation

$$\mathbf{y} := \Big(-C \cdot f(x^*) + (I - C \cdot \nabla f(x^* + \mathbf{y})) \cdot \mathbf{y} \Big) \cdot [0.9, 1.1] + 10^{-20}[-1, 1]$$

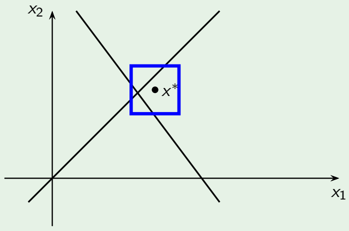
until the assumption of Theorem are satisfied.

Verification of a Linear System of Equations

Problem formulation

Given a real system $Ax = b$ and x^* approximate solution, find $\mathbf{y} \in \mathbb{IR}^n$ such that $A^{-1}b \in x^* + \mathbf{y}$.

Example



Verification of a Linear System of Equations

Given the system $Ax = b$ and an approximate solution x^*.

Theorem

Suppose $0 \in \mathbf{y}$. *Now if*

$$C(b - Ax^*) + (I - CA)\mathbf{y} \subseteq \mathit{int}\, \mathbf{y},$$

then:

- *C and A are nonsingular,*
- *there is a unique solution of $Ax = b$ in $x^* + \mathbf{y}$.*

Proof.

Use the previous result with $f(x) = Ax - b$. □

Implementation

- take $C \approx A^{-1}$ (numerically computed inverse),

Verification of a Linear System of Equations

ε-inflation method (Caprani and Madsen, 1978, Rump, 1980)

Repeat inflating $\mathbf{y} := [0.9, 1.1]\mathbf{x} + 10^{-20}[-1, 1]$ and updating

$$\mathbf{x} := C(b - Ax^*) + (I - CA)\mathbf{y}$$

until $\mathbf{x} \subseteq \mathit{int}\, \mathbf{y}$.

Then, $\Sigma \subseteq x^* + \mathbf{x}$.

Results

- Verification is about 7 times slower than solving the original problem (for random instances of dimension 100 to 2000).

Verification of a Linear System of Equations

Example

Let A be the Hilbert matrix of size 10 (i.e., $a_{ij} = \frac{1}{i+j-1}$), and $b := Ae$.

Then $Ax = b$ has the solution $x = e = (1, \dots, 1)^T$.

Approximate solution by Matlab:	Enclosing interval by ε-inflation method (2 iterations):
0.999999999235452	[0.99999973843401, 1.00000026238575]
1.000000065575364	[0.99999843048508, 1.00000149895660]
0.999998607887449	[0.99997745481481, 1.00002404324710]
1.000012638750021	[0.99978166603900, 1.00020478046370]
0.999939734980300	[0.99902374408278, 1.00104070076742]
1.000165704992114	[0.99714060702796, 1.00268292103727]
0.999727989024899	[0.99559932282378, 1.00468935360003]
1.000263042205847	[0.99546972629357, 1.00425202249136]
0.999861803020249	[0.99776781605377, 1.00237789028988]
1.000030414871015	[0.99947719419921, 1.00049082925529]

Verification of a Linear System of Equations

Challenge

- verification for large systems
  (one cannot use preconditioning by the inverse matrix)

References

S.M. Rump.
Verification methods: Rigorous results using floating-point arithmetic.
Acta Numerica, 19:187–449, 2010.

Next Section

1 More on Interval Functions

2 Application: Solving Nonlinear Equations

3 Application: Verification

4 AE Solution Set

5 Eigenvalues of Symmetric Interval Matrices

6 Conclusion

Tolerable Solutions

Motivation

So far, existentially quantified interval systems

$$\Sigma := \{x \in \mathbb{R}^n : \exists A \in \boldsymbol{A} \, \exists b \in \boldsymbol{b} : Ax = b\}.$$

Now, incorporate universal quantification as well!

Definition (Tolerable solutions)

A vector $x \in \mathbb{R}^n$ is a tolerable solution to $\boldsymbol{A}x = \boldsymbol{b}$ if for each $A \in \boldsymbol{A}$ there is $b \in \boldsymbol{b}$ such that $Ax = b$.

In other words,

$$\forall A \in \boldsymbol{A} \, \exists b \in \boldsymbol{b} : Ax = b.$$

Equivalent characterizations

- $\boldsymbol{A}x \subseteq \boldsymbol{b}$,
- $|A^c x - b^c| \le -A^\Delta |x| + b^\Delta$.

Tolerable Solutions

Theorem (Rohn, 1986)

A vector $x \in \mathbb{R}^n$ is a tolerable solution if and only if $x = x_1 - x_2$, where

$$\overline{A}x_1 - \underline{A}x_2 \le \overline{b}, \ \underline{A}x_1 - \overline{A}x_2 \ge \underline{b}, \ x_1, x_2 \ge 0.$$

Proof.

"$\Leftarrow$" Let $A \in \boldsymbol{A}$. Then

$$Ax = Ax_1 - Ax_2 \le \overline{A}x_1 - \underline{A}x_2 \le \overline{b},$$
$$Ax = Ax_1 - Ax_2 \ge \underline{A}x_1 - \overline{A}x_2 \ge \underline{b}$$

Thus, $Ax \in \boldsymbol{b}$ and $Ax = b$ for some $b \in \boldsymbol{b}$.

"$\Rightarrow$" Let $x \in \mathbb{R}^n$ be a tolerable solution. Define $x_1 := \max\{x, 0\}$ and $x_2 := \max\{-x, 0\}$ the positive and negative part of x, respectively. Then $x = x_1 - x_2$, $|x| = x_1 + x_2$, and $|A^c x - b^c| \le -A^\Delta |x| + b^\Delta$ draws

$$A^c(x_1 - x_2) - b^c \le -A^\Delta(x_1 + x_2) + b^\Delta,$$
$$-A^c(x_1 - x_2) + b^c \le -A^\Delta(x_1 + x_2) + b^\Delta.$$

□

Tolerable Solutions – Application

Example (Leontief's Input–Output Model of Economics)

- economy with n sectors (e.g., agriculture, industry, transportation, etc.),
- sector i produces a single commodity of amount x_i,
- production of each unit of the jth commodity will require a_{ij} (amount) of the ith commodity
- d_i the final demand in sector i.

Now the model draws

$$x_i = a_{i1}x_1 + \cdots + a_{in}x_n + d_i.$$

or, in a matrix form

$$x = Ax + d.$$

The solution $x = (I_n - A)^{-1}d = \sum_{k=0}^{\infty} A^k d$ is nonnegative if $\rho(A) < 1$.

Question: Exists x such that for any $A \in \boldsymbol{A}$ there is $d \in \boldsymbol{d}$: $(I_n - A)x = d$?

AE Solutions

Quantified system $\boldsymbol{A}x = \boldsymbol{b}$

- each interval parameter $\boldsymbol{a}_{ij}$ and $\boldsymbol{b}_i$ is quantified by $\forall$ or $\exists$
- the universally quantified parameters are denoted by $\boldsymbol{A}^\forall$, $\boldsymbol{b}^\forall$,
- the existentially quantified parameters are denoted by $\boldsymbol{A}^\exists$, $\boldsymbol{b}^\exists$
- the system reads $(\boldsymbol{A}^\forall + \boldsymbol{A}^\exists)x = \boldsymbol{b}^\forall + \boldsymbol{b}^\exists$

Definition (AE solution set)

$$\Sigma_{AE} := \{x \in \mathbb{R}^n : \\ \forall A^\forall \in \boldsymbol{A}^\forall \, \forall b^\forall \in \boldsymbol{b}^\forall \, \exists A^\exists \in \boldsymbol{A}^\exists \, \exists b^\exists \in \boldsymbol{b}^\exists : (A^\forall + A^\exists)x = b^\forall + b^\exists\}.$$

AE Solutions

Theorem (Shary, 1995)

$$\Sigma_{AE} = \{x \in \mathbb{R}^n : \boldsymbol{A}^\forall x - \boldsymbol{b}^\forall \subseteq \boldsymbol{b}^\exists - \boldsymbol{A}^\exists x\}. \qquad (1)$$

Proof.

$$\begin{aligned}\Sigma_{AE} &= \{x \in \mathbb{R}^n : \forall A^\forall \in \boldsymbol{A}^\forall\, \forall b^\forall \in \boldsymbol{b}^\forall\, \exists A^\exists \in \boldsymbol{A}^\exists\, \exists b^\exists \in \boldsymbol{b}^\exists : A^\forall x - b^\forall = b^\exists - A^\exists x\}\\ &= \{x \in \mathbb{R}^n : \forall A^\forall \in \boldsymbol{A}^\forall\, \forall b^\forall \in \boldsymbol{b}^\forall : A^\forall x - b^\forall \in \boldsymbol{b}^\exists - \boldsymbol{A}^\exists x\}\\ &= \{x \in \mathbb{R}^n : \boldsymbol{A}^\forall x - \boldsymbol{b}^\forall \subseteq \boldsymbol{b}^\exists - \boldsymbol{A}^\exists x\}.\end{aligned}$$

□

Theorem (Rohn, 1996)

$$\Sigma_{AE} = \left\{x \in \mathbb{R}^n : |A^c x - b^c| \le \big((\boldsymbol{A}^\exists)^\Delta - (\boldsymbol{A}^\forall)^\Delta\big)|x| + (\boldsymbol{b}^\exists)^\Delta - (\boldsymbol{b}^\forall)^\Delta\right\}.$$

Proof.

Using (1) and the fact $\boldsymbol{p} \subseteq \boldsymbol{q} \;\Leftrightarrow\; |p^c - q^c| \le q^\Delta - p^\Delta$, we get

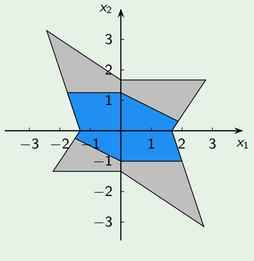
$$\begin{aligned}|(\boldsymbol{A}^\forall x - \boldsymbol{b}^\forall)^c - (\boldsymbol{b}^\exists - \boldsymbol{A}^\exists x)^c| &\le \left(\boldsymbol{A}^\exists x - \boldsymbol{b}^\exists\right)^\Delta - \left(\boldsymbol{b}^\forall - \boldsymbol{A}^\forall x\right)^\Delta\\ &= (\boldsymbol{A}^\exists)^\Delta|x| + \boldsymbol{b}^{\exists\Delta} - (\boldsymbol{A}^\forall)^\Delta x| - \boldsymbol{b}^{\forall\Delta}.\end{aligned}$$

□

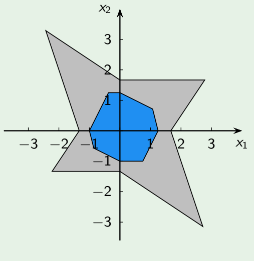
AE Solutions

Example

$$\begin{pmatrix}[3,4]^\exists & [-2,1]^\exists\\ [0,2]^\forall & [3,4]^\forall\end{pmatrix} x = \begin{pmatrix}[-4,5]^\exists\\ [-4,5]^\exists\end{pmatrix}. \qquad \begin{pmatrix}[3,4]^\forall & [-2,1]^\forall\\ [0,2]^\forall & [3,4]^\forall\end{pmatrix} x = \begin{pmatrix}[-4,5]^\exists\\ [-4,5]^\exists\end{pmatrix}.$$



AE solution set.



Tolerable solution set.

Next Section

1. More on Interval Functions
2. Application: Solving Nonlinear Equations
3. Application: Verification
4. AE Solution Set
5. Eigenvalues of Symmetric Interval Matrices
6. Conclusion

Eigenvalues of Symmetric Interval Matrices

A symmetric interval matrix

$$\boldsymbol{A}^S := \{A \in \boldsymbol{A} : A = A^T\}.$$

Without loss of generality assume that $\underline{A} = \underline{A}^T$, $\overline{A} = \overline{A}^T$, and $\boldsymbol{A}^S \neq \emptyset$.

Eigenvalues of a symmetric interval matrix

Eigenvalues of a symmetric $A \in \mathbb{R}^{n\times n}$: $\lambda_1(A) \ge \dots \ge \lambda_n(A)$.
Eigenvalue sets of $\boldsymbol{A}^S$ are compact intervals

$$\boldsymbol{\lambda}_i(\boldsymbol{A}^S) := \left\{\lambda_i(A) : A \in \boldsymbol{A}^S\right\}, \quad i = 1, \dots, n.$$

Theorem

Checking whether $0 \in \boldsymbol{\lambda}_i(\boldsymbol{A}^S)$ for some $i = 1, \dots, n$ is NP-hard.

Proof.

$\boldsymbol{A}$ is singular iff $\boldsymbol{M}^S := \begin{pmatrix}0 & \boldsymbol{A}\\ \boldsymbol{A}^T & 0\end{pmatrix}^S$ is singular (has a zero eigenvalue). □

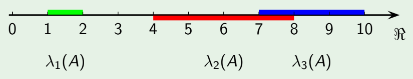
Eigenvalues – An Example

Example

Let

$$A \in \boldsymbol{A} = \begin{pmatrix}[1,2] & 0 & 0\\ 0 & [7,8] & 0\\ 0 & 0 & [4,10]\end{pmatrix}$$

What are the eigenvalue sets?
We have $\boldsymbol{\lambda}_1(\boldsymbol{A}^S) = [7,10]$, $\boldsymbol{\lambda}_2(\boldsymbol{A}^S) = [4,8]$ and $\boldsymbol{\lambda}_3(\boldsymbol{A}^S) = [1,2]$.



Eigenvalue sets are compact intervals. They may intersect or equal.

Eigenvalues – Some Exact Bounds

Theorem (Hertz, 1992)

We have

$$\begin{aligned}\overline{\lambda}_1(\boldsymbol{A}^S) &= \max_{z\in\{\pm1\}^n} \lambda_1(A^c + \operatorname{diag}(z)A^\Delta \operatorname{diag}(z)),\\ \underline{\lambda}_n(\boldsymbol{A}^S) &= \min_{z\in\{\pm1\}^n} \lambda_n(A^c - \operatorname{diag}(z)A^\Delta \operatorname{diag}(z)).\end{aligned}$$

Proof.

"Upper bound." By contradiction suppose that there is $A \in \boldsymbol{A}^S$ such that

$$\lambda_1(A) > \max_{z\in\{\pm1\}^n} \lambda_1(A_z), \qquad \Big[\text{where } A_z \equiv A^c + \operatorname{diag}(z)A^\Delta \operatorname{diag}(z)\Big]$$

Thus $Ax = \lambda_1(A)x$ for some x with $\|x\|_2 = 1$.
Put $z^* := \operatorname{sgn}(x)$, and by the Rayleigh–Ritz Theorem we have

$$\begin{aligned}\lambda_1(A) &= x^T A x \le x^T A_{z^*} x\\ &\le \max_{y:\|y\|_2=1} y^T A_{z^*} y = \lambda_1(A_{z^*}).\end{aligned}$$

□

Eigenvalues – Some Other Exact Bounds

Theorem

$\underline{\lambda}_1(\boldsymbol{A}^S)$ and $\overline{\lambda}_n(\boldsymbol{A}^S)$ are polynomially computable by semidefinite programming with arbitrary precision.

Proof.

We have

$$\overline{\lambda}_n(\boldsymbol{A}^S) = \max \alpha \quad \text{subject to} \quad A - \alpha I_n \text{ is positive semidefinite}, \ A \in \boldsymbol{A}^S.$$

Consider a block diagonal matrix $M(A, \alpha)$ with blocks

$$A - \alpha I_n, \ a_{ij} - \underline{a}_{ij}, \ \overline{a}_{ij} - a_{ij}, \ i \le j.$$

Then the semidefinite programming problem reads

$$\overline{\lambda}_n(\boldsymbol{A}^S) = \max \alpha \quad \text{subject to} \quad M(A, \alpha) \text{ is positive semidefinite.}$$

□

Eigenvalues – Enclosures

Theorem

We have

$$\boldsymbol{\lambda}_i(\boldsymbol{A}^S) \subseteq [\lambda_i(A^c) - \rho(A^\Delta), \lambda_i(A^c) + \rho(A^\Delta)], \quad i = 1, \dots, n.$$

Proof.

Recall for any $A, B \in \mathbb{R}^{n \times n}$,

$$|A| \le B \quad \Rightarrow \quad \rho(A) \le \rho(|A|) \le \rho(B),$$

and for A, B symmetric (Weyl's Theorem)

$$\lambda_i(A) + \lambda_n(B) \le \lambda_i(A + B) \le \lambda_i(A) + \lambda_1(B), \quad i = 1, \dots, n.$$

Let $A \in \boldsymbol{A}^S$, so $|A - A^c| \le A^\Delta$. Then

$$\begin{aligned} \lambda_i(A) &= \lambda_i(A^c + (A - A^c)) \le \lambda_i(A^c) + \lambda_1(A - A^c) \\ &\le \lambda_i(A^c) + \rho(|A - A^c|) \le \lambda_i(A^c) + \rho(A^\Delta). \end{aligned}$$

Similarly for the lower bound. □

Eigenvalues – Easy Cases

Theorem

1. *If A^c is essentially non-negative, i.e., $A^c_{ij} \ge 0 \ \forall i \ne j$, then*
$$\overline{\lambda}_1(\boldsymbol{A}^S) = \lambda_1(\overline{A}).$$
2. *If A^Δ is diagonal, then*
$$\overline{\lambda}_1(\boldsymbol{A}^S) = \lambda_1(\overline{A}), \quad \underline{\lambda}_n(\boldsymbol{A}^S) = \lambda_n(\underline{A}).$$

Proof.

1. For the sake of simplicity suppose $A^c \ge 0$. Then $\forall A \in \boldsymbol{A}^S$ we have $|A| \le \overline{A}$, whence
$$\lambda_1(A) = \rho(A) \le \rho(\overline{A}) = \lambda_1(\overline{A}).$$
2. By Hertz's theorem,
$$\begin{aligned} \overline{\lambda}_1(\boldsymbol{A}^S) &= \max_{z \in \{\pm 1\}^n} \lambda_1(A^c + \operatorname{diag}(z) A^\Delta \operatorname{diag}(z)), \\ &= \lambda_1(A^c + A^\Delta) = \lambda_1(\overline{A}). \end{aligned}$$

□

Positive Semidefiniteness

$\boldsymbol{A}^S$ is *positive semidefinite* if every $A \in \boldsymbol{A}^S$ is positive semidefinite.

Theorem

The following are equivalent

1. *$\boldsymbol{A}^S$ is positive semidefinite,*
2. *$A_z \equiv A^c - \operatorname{diag}(z) A^\Delta \operatorname{diag}(z)$ is positive semidefinite $\forall z \in \{\pm 1\}^n$,*
3. *$x^T A^c x - |x|^T A^\Delta |x| \ge 0$ for each $x \in \mathbb{R}^n$.*

Proof.

"(1) ⇒ (2)" Obvious from $A_z \in \boldsymbol{A}^S$.
"(2) ⇒ (3)" Let $x \in \mathbb{R}^n$ and put $z := \operatorname{sgn}(x)$. Now,

$$x^T A^c x - |x|^T A^\Delta |x| = x^T A^c x - x^T \operatorname{diag}(z) A^\Delta \operatorname{diag}(z) x = x^T A_z x \ge 0.$$

"(3) ⇒ (1)" Let $A \in \boldsymbol{A}^S$ and $x \in \mathbb{R}^n$. Now,

$$\begin{aligned} x^T A x = x^T A^c x + x^T (A - A^c) x &\ge x^T A^c x - |x^T (A - A^c) x| \\ &\ge x^T A^c x - |x|^T A^\Delta |x| \ge 0. \end{aligned}$$

□

Positive Definiteness

$\boldsymbol{A}^S$ is *positive definite* if every $A \in \boldsymbol{A}^S$ is positive definite.

Theorem

The following are equivalent

1. *$\boldsymbol{A}^S$ is positive definite,*
2. *$A_z \equiv A^c - \operatorname{diag}(z) A^\Delta \operatorname{diag}(z)$ is positive definite for each $z \in \{\pm 1\}^n$,*
3. *$x^T A^c x - |x|^T A^\Delta |x| > 0$ for each $0 \ne x \in \mathbb{R}^n$,*
4. *A^c is positive definite and $\boldsymbol{A}$ is regular.*

Proof.

"(1) ⇔ (2) ⇔ (3)" analogously.
"(1) ⇒ (4)" If there are $A \in \boldsymbol{A}$ and $x \ne 0$ such that $Ax = 0$, then

$$0 = x^T A x = x^T \tfrac{1}{2}(A + A^T) x,$$

and so $\frac{1}{2}(A + A^T) \in \boldsymbol{A}^S$ is not positive definite.
"(4) ⇒ (1)" Positive definiteness of A^c implies $\lambda_i(A^c) > 0 \ \forall i$, and regularity of $\boldsymbol{A}$ implies $\boldsymbol{\lambda}_i(\boldsymbol{A}^S) > 0 \ \forall i$. □

Complexity

Theorem (Nemirovskii, 1993)

Checking positive semidefiniteness of $\boldsymbol{A}^S$ is co-NP-hard.

Theorem (Rohn, 1994)

Checking positive definiteness of $\boldsymbol{A}^S$ is co-NP-hard.

Theorem (Jaulin and Henrion, 2005)

Checking whether there is a positive definite matrix in $\boldsymbol{A}^S$ is a polynomial time problem.

Proof.

There is a positive semidefinite matrix in $\boldsymbol{A}^S$ iff $\overline{\lambda}_n(\boldsymbol{A}^S) \ge 0$.
So we can check it by semidefinite programming. □

Sufficient Conditions

Theorem

1. *$\boldsymbol{A}^S$ is positive semidefinite if $\lambda_n(A^c) \geq \rho(A^\Delta)$.*
2. *$\boldsymbol{A}^S$ is positive definite if $\lambda_n(A^c) > \rho(A^\Delta)$.*
3. *$\boldsymbol{A}^S$ is positive definite if A^c is positive definite and $\rho(|(A^c)^{-1}|A^\Delta) < 1$.*

Proof.

1. $\boldsymbol{A}^S$ is positive semidefinite iff $\underline{\lambda}_n(\boldsymbol{A}^S) \geq 0$.
   Now, employ the smallest eigenvalue set enclosure
   $$\boldsymbol{\lambda}_n(\boldsymbol{A}^S) \subseteq [\lambda_n(A^c) - \rho(A^\Delta), \lambda_n(A^c) + \rho(A^\Delta)].$$
2. Analogous.
3. Use Beeck's sufficient condition for regularity of $\boldsymbol{A}$. □

Application: Convexity Testing

Theorem

A function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is convex on $\boldsymbol{x} \in \mathbb{IR}^n$ iff its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall x \in int\, \boldsymbol{x}$.

Corollary

A function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is convex on $\boldsymbol{x} \in \mathbb{IR}^n$ if $\nabla^2 f(\boldsymbol{x})$ is positive semidefinite.

Application: Convexity Testing

Example

Let
$$f(x,y,z) = x^3 + 2x^2y - xyz + 3yz^2 + 8y^2,$$
where $x \in \boldsymbol{x} = [2,3]$, $y \in \boldsymbol{y} = [1,2]$ and $z \in \boldsymbol{z} = [0,1]$. The Hessian of f reads
$$\nabla^2 f(x,y,z) = \begin{pmatrix} 6x+4y & 4x-z & -y \\ 4x-z & 16 & -x+6z \\ -y & -x+6z & 6y \end{pmatrix}$$
Evaluation the Hessian matrix by interval arithmetic results in
$$\nabla^2 f(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \subseteq \begin{pmatrix} [16,26] & [7,12] & -[1,2] \\ [7,12] & 16 & [-3,4] \\ -[1,2] & [-3,4] & [6,12] \end{pmatrix}$$
Now, both sufficient conditions for positive definiteness succeed. Thus, we can conclude that f si convex on the interval domain.

References

- M. Hladík, D. Daney, and E. Tsigaridas.
  Bounds on real eigenvalues and singular values of interval matrices.
  SIAM J. Matrix Anal. Appl., 31(4):2116–2129, 2010.
- M. Hladík, D. Daney, and E. P. Tsigaridas.
  Characterizing and approximating eigenvalue sets of symmetric interval matrices.
  Comput. Math. Appl., 62(8):3152–3163, 2011.
- L. Jaulin and D. Henrion.
  Contracting optimally an interval matrix without loosing any positive semi-definite matrix is a tractable problem.
  Reliab. Comput., 11(1):1–17, 2005.
- J. Rohn.
  Positive definiteness and stability of interval matrices.
  SIAM J. Matrix Anal. Appl., 15(1):175–184, 1994.
- J. Rohn.
  A handbook of results on interval linear problems.
  Tech. Rep. 1163, Acad. of Sci. of the Czech Republic, Prague, 2012.
  `http://uivtx.cs.cas.cz/~rohn/publist/!aahandbook.pdf`

Next Section

1. More on Interval Functions
2. Application: Solving Nonlinear Equations
3. Application: Verification
4. AE Solution Set
5. Eigenvalues of Symmetric Interval Matrices
6. Conclusion

Conclusion

Interval computation offers:

- nice theory, methods and applications
- many open problems
- interdisciplinarity

Thanks

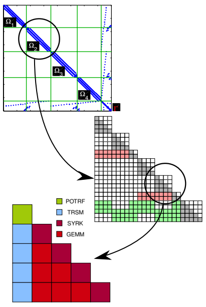
Any feedback is welcome!

Dense linear algebra - From BLAS to Chameleon

PATC Parallel Linear Algebra @ IT4Innovations
February 2nd, 2017

Mathieu Faverge

PRACE Advanced Training Course



MaPHyS (Tomorrow)
- Hybrid solver
- Domain decomposition

PaStiX (Today)
- Sparse direct solver
- Supernodal method

Chameleon (Today)
- Dense linear algebra
- Tile algorithms
- Heterogeneous distributed architectures

Parallel Linear Algebra

Thursday, February 2nd	
09:00 - 09:15	Introduction to PLA, *T. Kozubek*
09:00 - 10:30	Introduction to dense linear algebra - From BLAS to Chameleon, *M. Faverge*
10:30 - 11:00	Break
11:00 - 12:30	Sparse direct solvers - Analyze, *M. Faverge*
12:30 - 13:30	Lunch break
13:30 - 14:45	Sparse direct solvers - Supernodal Method, *M. Faverge*
14:45 - 15:00	Hands-on Spack - softwares installation, *F. Pruvost*
15:00 - 15:30	Break
15:30 - 17:00	Hands-on Chameleon library, *M. Faverge, G. Marait, F. Pruvost*

Parallel Linear Algebra

Thursday, February 2nd	
09:00 - 10:30	Krylov subspace methods, *Z. Strakos*
10:30 - 11:00	Break
11:00 - 12:30	Hybrid solvers, *G. Marait*
12:30 - 13:30	Lunch break
13:30 - 15:00	Hands-on PaStiX, *M. Faverge, G. Marait, F. Pruvost*
15:00 - 15:30	Break
15:30 - 17:00	Hands-on MaPHyS, *M. Faverge, G. Marait, F. Pruvost*

1

Introduction

What is the purpose of linear algebra libraries? (1/2)

- Many simulations codes solves a problem:
  - $Ax = b$
  - A is a matrix of size $M - by - N$
  - x and b, two vectors (or set of vectors) of sizes N and M respectively
- Goal: provide the users with libraries able to do this operation in the most efficient manner:
  - The fastest time to solution as possible
  - Numerical Accuracy
- Two major kinds of problems:
  - A is dense, all entries are considered non-zeroes
  - A is sparse, high percentage of zero entries

What is the purpose of linear algebra libraries? (2/2)

- Can we compute A^{-1}?

What is the purpose of linear algebra libraries? (2/2)

- Can we compute A^{-1}?
- Tow main classes of algorithms?

What is the purpose of linear algebra libraries? (2/2)

- Can we compute A^{-1}?
- Tow main classes of algorithms?
  - Direct or iterative methods

What is the purpose of linear algebra libraries? (2/2)

- Can we compute A^{-1}?
- Tow main classes of algorithms?
  - Direct or iterative methods
- When using direct methods, it also exists many factorization algorithms:
  - For general matrices: $A = LU$
  - For symmetric/hermitian definite positive matrices: $A = LL^t$, or LL^h (Cholesky)
  - For symmetric/hermitian non definite positive matrices: $A = LDL^t$, or LDL^h
  - But also: $A = QR$

A little bit of history

Computing in 1974

- High Performance Computers:
  - IBM 370/195, CDC 7600, Univac 1110, DEC PDP-10, Honeywell 6030
- Fortran 66
- Trying to achieve software portability
- Run efficiently
- BLAS (Level 1)
  - Vector operations
- Software released in 1979
  - About the time of the Cray 1

A little bit of history

- But the BLAS-1 weren't enough
  - Consider AXPY ($y = \alpha x + y$): $2n$ flops on $3n$ read/writes
  - Computational intensity $= (2n)/(3n) = 2/3$
  - Too low to run near peak speed (read/write dominates)

A little bit of history

- But the BLAS-1 weren't enough
  - Consider AXPY ($y = \alpha x + y$): $2n$ flops on $3n$ read/writes
  - Computational intensity = $(2n)/(3n) = 2/3$
  - Too low to run near peak speed (read/write dominates)
- So the BLAS-2 were developed (1984-1986)
  - Standard library of 25 operations (mostly) on matrix/vector pairs
    - GEMV: $y = \alpha Ax + \beta y$, GER: $A = A + \alpha x * y^t$,
    - Up to 4 versions of each (S/D/C/Z), 66 routines, 18K LOC
  - Why BLAS 2? They do $O(n^2)$ ops on $O(n^2)$ data *rightarrow* So computational intensity still just $(2n^2)/(n^2) = 2$

A little bit of history

- But the BLAS-1 weren't enough
  - Consider AXPY ($y = \alpha x + y$): $2n$ flops on $3n$ read/writes
  - Computational intensity = $(2n)/(3n) = 2/3$
  - Too low to run near peak speed (read/write dominates)
- So the BLAS-2 were developed (1984-1986)
  - Standard library of 25 operations (mostly) on matrix/vector pairs
    - GEMV: $y = \alpha Ax + \beta y$, GER: $A = A + \alpha x * y^t$,
    - Up to 4 versions of each (S/D/C/Z), 66 routines, 18K LOC
  - Why BLAS 2? They do $O(n^2)$ ops on $O(n^2)$ data *rightarrow* So computational intensity still just $(2n^2)/(n^2) = 2$
    - *OK for vector machines, but not for machines with caches*

Why higher level BLAS?

BLAS	Memory Refs	Flops	Ratio
Lvl 1 $y = y + \alpha x$	$3n$	$2n$	$2/3$
Lvl 2 $y = y + Ax$	n^2	$2n^2$	2
Lvl 3 $C = C + AB$	$4n^2$	$2n^3$	$n/2$

How do we measure the code performance/efficiency? What is a xflop/s?

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.
- kilo 10^3, mega 10^6, giga 10^9,

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.
- kilo 10^3, mega 10^6, giga 10^9, tera 10^{12},

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.
- kilo 10^3, mega 10^6, giga 10^9, tera 10^{12}, exa 10^{15},

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.
- kilo 10^3, mega 10^6, giga 10^9, tera 10^{12}, exa 10^{15}, zetta 10^{18},

How do we measure the code performance/efficiency? What is a xflop/s?

- xflop/s is a rate of execution, some number of floating point operations per second. Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.
- kilo 10^3, mega 10^6, giga 10^9, tera 10^{12}, exa 10^{15}, zetta 10^{18}, yotta 10^{21}

How do we measure the code performance/efficiency? What is the theoretical peak performance?

- The theoretical peak is based not on an actual performance from a benchmark run, but on a paper computation to determine the theoretical peak rate of execution of floating point operations for the machine.
- Flops $=$ cores $\times$ clock $\times$ $\frac{\text{FLOPs}}{\text{cycle}}$
- For example, an Intel Xeon 5570 quad core at 2.93 GHz can complete 4 floating point operations per cycle or a theoretical peak performance of 11.72 GFlop/s per core or 46.88 Gflop/s for the socket.
- A more recent example: an Intel Haswell architecture like the E5-2620v3 can complete up to 16 Flops per cycle (thanks to AVX2 and FMA3) at a frequency up to 3.2 GHz per core. So the theoretical peak is 51.2GFlop/s per core, and 201.6GFlop/s for the 6 cores (the frequency is limited to 2.1GHz when all cores are enabled with AVX2 and FMA3)

2

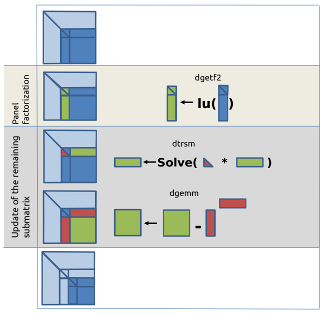
Software evolution

2.1

Software evolution

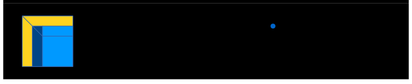
Single core architectures

Example of the LU factorization



Software evolution

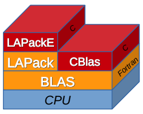
Software evolution



LAPACK

- http://www.netlib.org/lapack/
- LAPACK (Linear Algebra PACKage) provides routines for
  - solving systems of simultaneous linear equations,
  - least-squares solutions of linear systems of equations,
  - eigenvalue problems,
  - and singular value problems.
- it relies on BLAS
- it uses Fortran column major layout
- it is **sequential**
- it is a reference implementation
- It handle dense and banded matrices, but not general sparse matrices
- In all areas, similar functionality are provided for real and complex matrices, in both single and double precision.

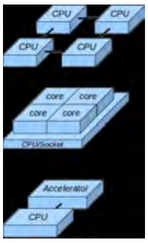
Summary for one single core



1. BLAS provides the basic linear algebra subroutines in Fortran
2. CBlas provides a C interface to BLAS
3. LAPACK provides a more advanced set of linear algebra routines on top of BLAS, and in Fortran
4. LAPACKE (since 2011) provides a C interface to LAPACK (*Do not use CLapack*)

Provided by Netlib, OpenBLAS, IBM ESSL, Intel MKL, AMD ACML, ...

What about more complex architectures?



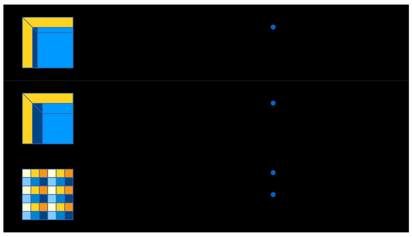
- Multiple CPUs in distributed memory
- Multi-core architectures
- Nodes enhanced with accelerators as GPUs and/or KNL

2.2

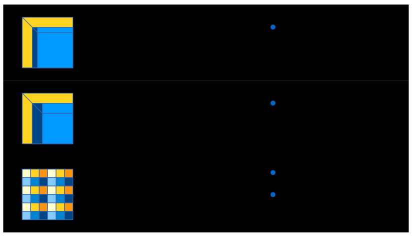
Software evolution

Distributed memory

Software evolution

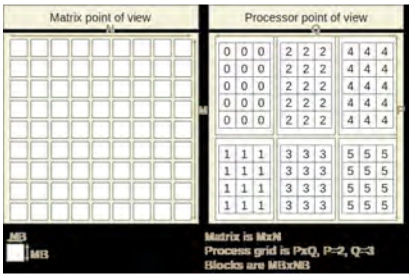


Software evolution



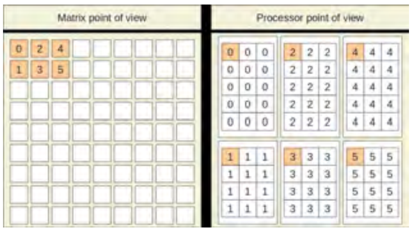
- The problem is: how to distribute the data?
  → 2D block cyclic layout

2D block-cyclic layout

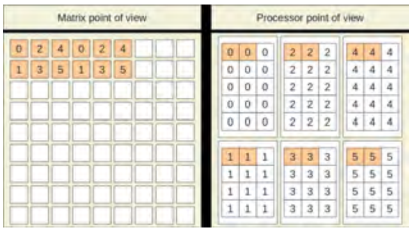


Matrix is MxN
Process grid is PxQ, P=2, Q=3
Blocks are MBxNB

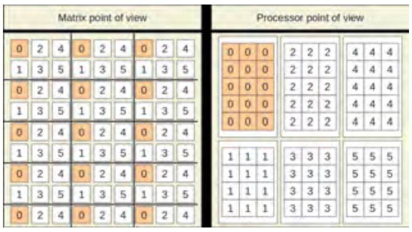
2D block-cyclic layout



2D block-cyclic layout

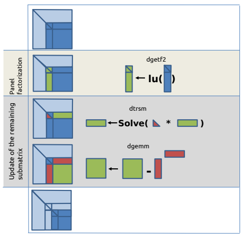


2D block-cyclic layout



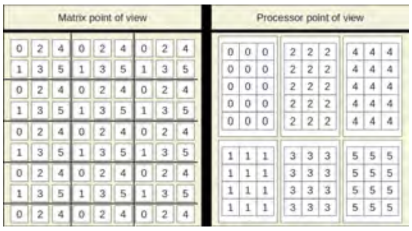
Example of LU factorization

How distributed memory implementation works?

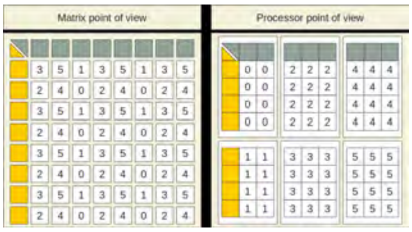


1. Panel: Communications between involved processors for each column
2. TRSM update:
   - Broadcast the triangle on the row
   - Local TRSM updates are made in parallel
3. GEMM update:
   - Broadcast the *U* part to the column
   - Local GEMM updates are made in parallel

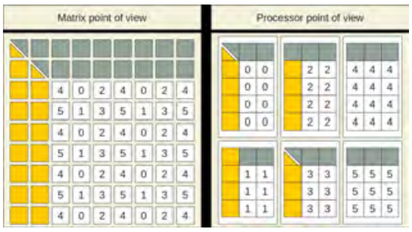
2D block-cyclic layout, Algorithm progression



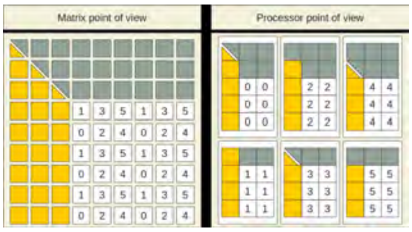
2D block-cyclic layout, Algorithm progression



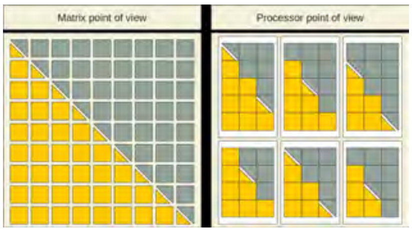
2D block-cyclic layout, Algorithm progression



2D block-cyclic layout, Algorithm progression



2D block-cyclic layout, Algorithm progression



Parallelism in ScaLAPACK

- Level 3 BLAS block operations
  - For all the reduction routines
- Pipelining/Look-ahead
  - QR Algorithm, Triangular Solvers, classic factorizations
- Redundant computations
  - Condition estimators
- Static work assignment
  - Bisection

- Task parallelism
  - Sign function eigenvalue computations
- Divide and Conquer
  - Tridiagonal and band solvers, symmetric eigenvalue problem and Sign function
- Cyclic reduction
  - Reduced system in the band solver
- Data parallelism
  - Sign function

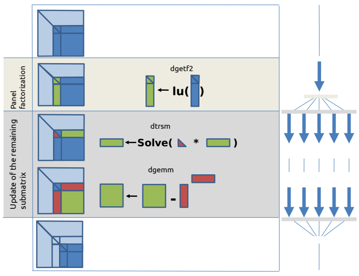
2.3

Software evolution

Multi-core architectures (shared memory)

Example of LU factorization

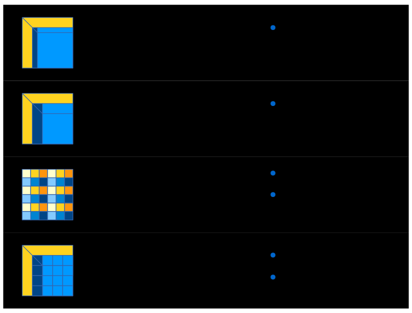
How to parallelize it in shared memory?



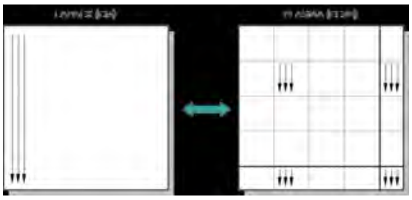
- Use fork-and-join parallelism (Bulk Sync Processing)
- Simple and easy to do in any reasonable software
- Parallelize the largest portion of the Flops
- Requires only to link to Multi-threaded BLAS library such as MKL

Software evolution

Plasma



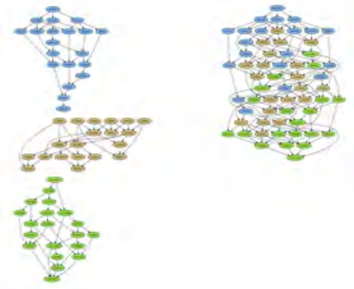
PLASMA: 1) Memory layout



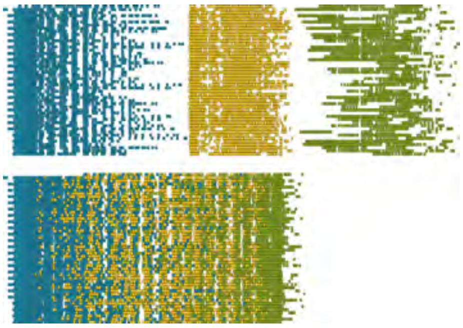
PLASMA: 2) Dataflow scheduling

- Rethink algorithms as dataflow algorithms
- Express the algorithm as a directed acyclic graph (DAG) where nodes are tasks, and edges data movements
- Rely on external runtime to:
  - discover data dependencies
  - schedule tasks in a coherent way
- Remove all possible synchronization points

Example of Cholesky Inversion



Example of Cholesky Inversion



2.4

Software evolution

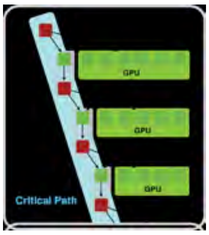
Heterogeneous architectures

Which library to use for accelerators?

- Nvidia CuBLAS on Nvidia GPUs
  - Cover the set of BLAS routines for Nvidia GPUs
  - A few extra routines from LAPACK
  - Set of Batched BLAS routines to apply many times the same operation of multiple data
- Intel MKL on Intel MIC architectures
  - Same coverage as classic MKL (with various efficiency)
  - Set of Batched BLAS routines to apply many times the same operation of multiple data
- MAGMA (ICL - UTK) for Nvidia GPUs and Intel MIC
  - Interface to BLAS routines + subset of internally implemented routines
  - Cover partially LAPACK routines
  - Set of Batched BLAS routines
- CULA, BLIS, clBLAS, . . .

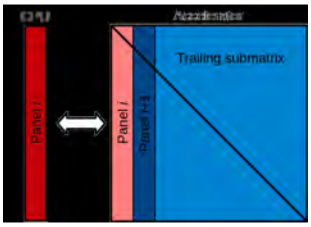
MAGMA: Methodology overview

- MAGMA uses hybridization methodology based on:
  - Representing linear algebra as collections of tasks and data dependencies among them
  - Properly scheduling tasks' execution over multicore and GPU hardware components
- Applied to fundamental linear algebra algorithms
  - One and two-sided factorizations and solver
  - Iterative solvers
  - Eigensolvers
- Productivity
  - High level
  - Leveraging prior developments
  - Exceeding in performance homogeneous solutions



MAGMA: Methodology overview

1. Perform panel computations (Level 2 BLAS) on CPUs using multi-threaded LAPACK
2. Perform trailing matrix updates (Level 3 BLAS) on the accelerator using look-ahead technique.



How to combine eveything?

1. Distributed memory system with 2D block-cyclic (or not)
2. New tile data layout for enhanced memory accesses
3. Tile algorithms to reduce synchronization points
4. Exploit both CPUs and accelerators

3 Chameleon

Matrices Over Runtime Systems @ Exascale

Linear algebra

$AX = B$

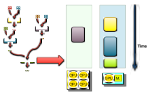
Sequential-Task-Flow

```
for (j = 0; j < N; j++)
  Task (A[j]);
```

Direct Acyclic Graph

→

Runtime systems



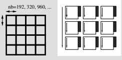
→

Heterogeneous platforms

The Chameleon Library

Sequential Task Flow (STF) design of *dense linear algebra* tiles algorithms (derived from PLASMA) on top of runtime systems

Tile matrix layout

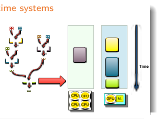


STF **PLASMA** algorithms

```
for (k = 0; k < N; k++){
    POTRF (A[k][k]);
    for (m = k+1; m < N; m++)
        TRSM (A[k][k], A[m][k]);
    for (n = k+1; n < N; n++) {
        SYRK (A[n][k], A[n][n]);
        for (m = n+1; m < N; m++)
            GEMM (A[m][k], A[n][k], A[m][n]);
    }
}
```

Runtime systems

- **QUARK**
- **StarPU**
- PaRSEC
- OmpSS



Optimized kernels

- BLAS, LAPACK
- cuBLAS, MAGMA

3.1

Chameleon

Programming model: Sequential Task Flow

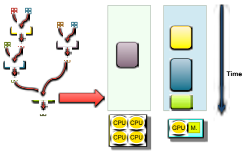
Task Scheduling

The runtime maps the graph of tasks (DAG) on the hardware

- Allocating computing resources
- Enforcing dependency constraints
- Handling data transfers

Adaptiveness

- A single DAG enables multiple scheduling strategies
- A single DAG can be mapped on multiple platforms



Sequential Task Flow / StarPU

- Express parallelism. . .
- . . . using the natural program flow

- **Submit** tasks in the sequential flow of the program. . .
- . . . then let the runtime:
  - infer the dependencies and,
  - schedule the tasks asynchronously

StarPU (http://starpu.gforge.inria.fr/)

- Storm Team – Inria Bordeaux - Sud-Ouest
- Computes cost models on the fly
- Kernels can be scheduled on either the CPU, and/or the accelerators
- Multiple scheduling strategies: Minimum Completion Time, Local Work Stealing, user defined...

Ex.: Sequential Cholesky Decomposition

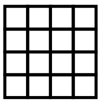
```
for (j = 0; j < N; j++) {
  POTRF (   A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (   A[i][j],   A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (   A[i][i],   A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (   A[i][k],
             A[i][j],   A[k][j]);
  }
}
```

Ex.: Task-Based Cholesky Decomposition

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

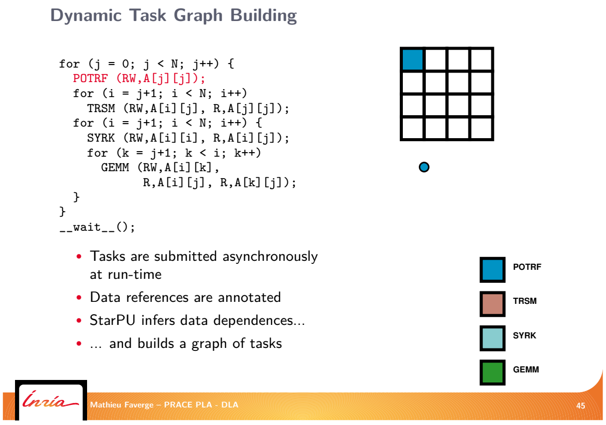
Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```



- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

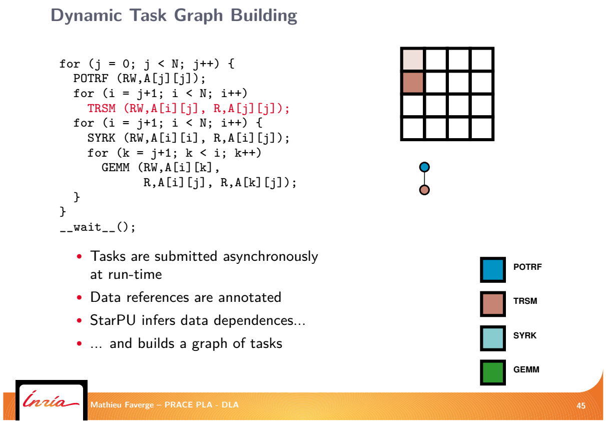
🟦 POTRF
🟫 TRSM
▨ SYRK
🟩 GEMM



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

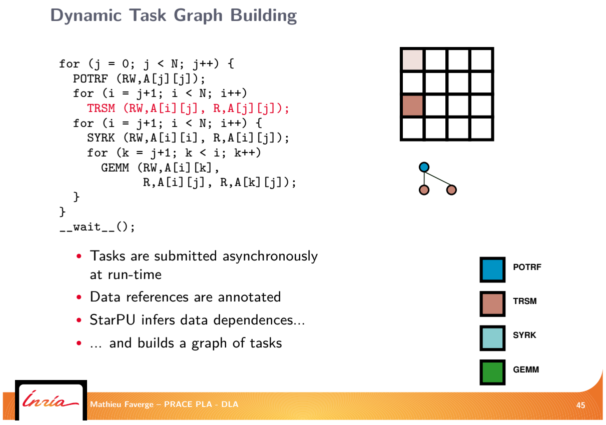
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

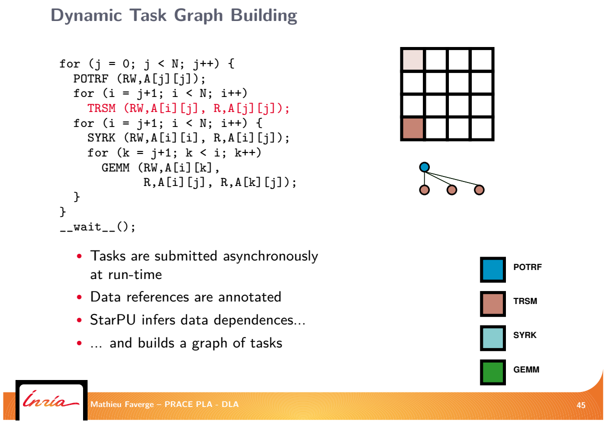
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

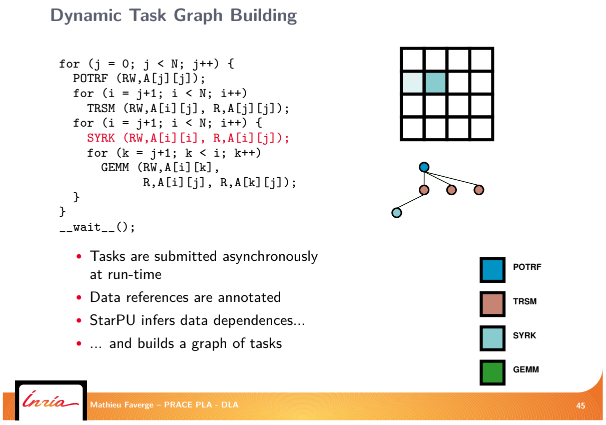
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

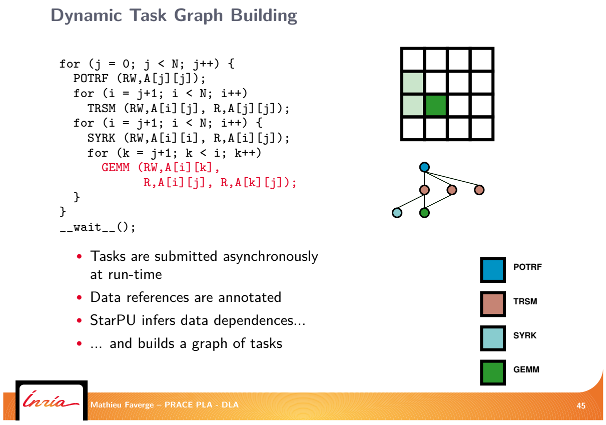
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

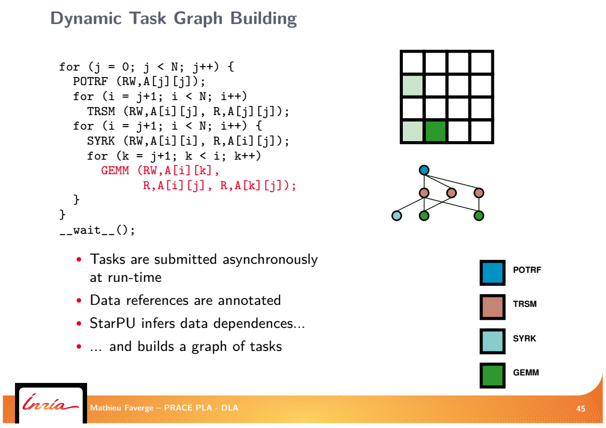
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks



Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

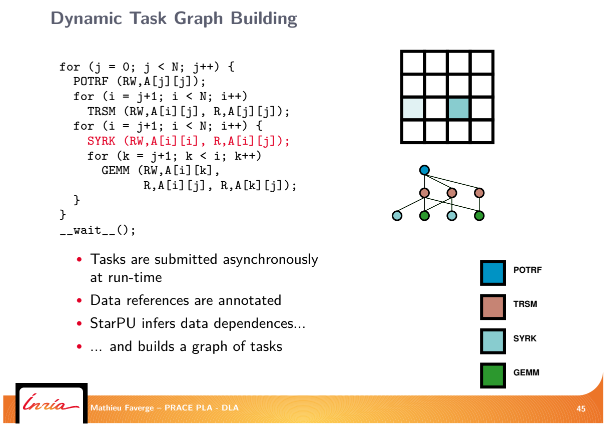


Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM

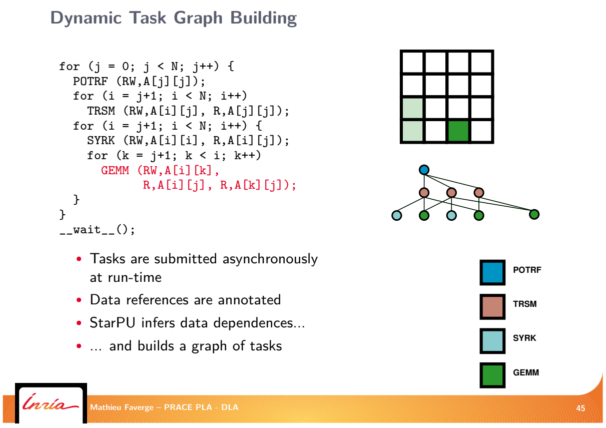


Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM

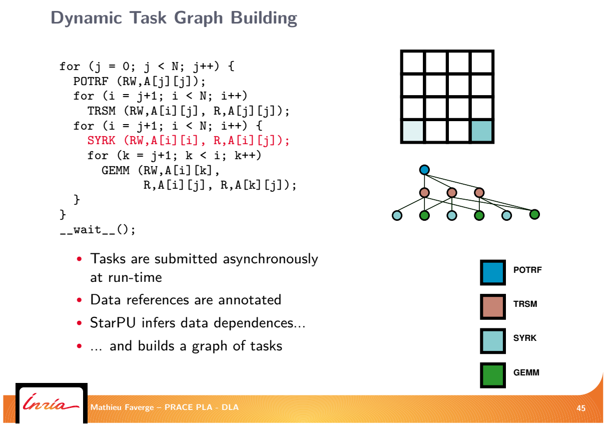


Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM

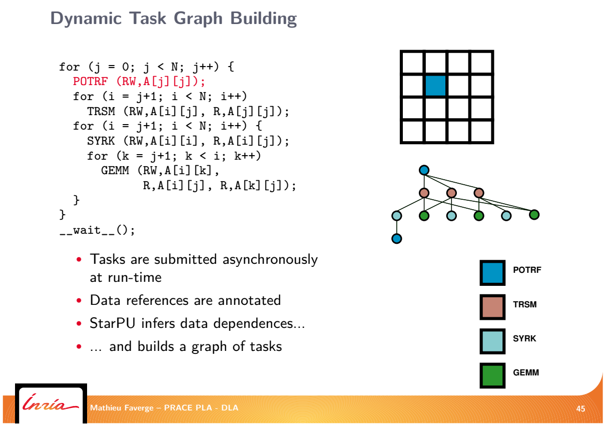


Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM

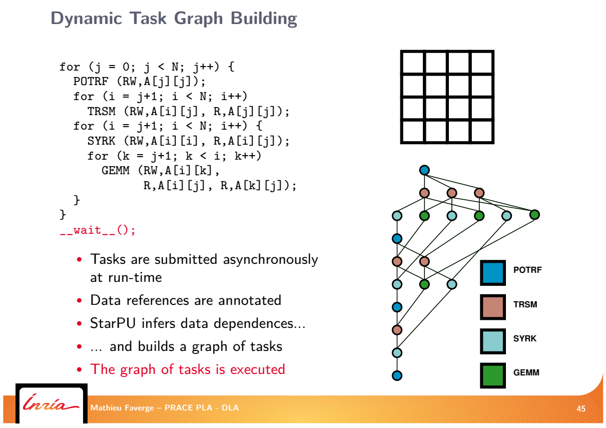


Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM



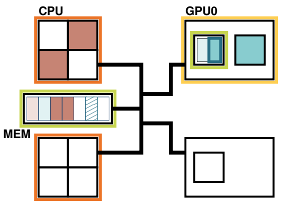
Dynamic Task Graph Building

```
for (j = 0; j < N; j++) {
  POTRF (RW,A[j][j]);
  for (i = j+1; i < N; i++)
    TRSM (RW,A[i][j], R,A[j][j]);
  for (i = j+1; i < N; i++) {
    SYRK (RW,A[i][i], R,A[i][j]);
    for (k = j+1; k < i; k++)
      GEMM (RW,A[i][k],
            R,A[i][j], R,A[k][j]);
  }
}
__wait__();
```

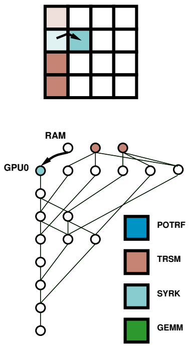
- Tasks are submitted asynchronously at run-time
- Data references are annotated
- StarPU infers data dependences...
- ... and builds a graph of tasks
- The graph of tasks is executed

🟦 POTRF
🟥 TRSM
🟦 SYRK
🟩 GEMM

Data Dependencies on Heterogeneous Nodes



- Handles dependencies
- Handles scheduling (policy)
- Handles data consistency (MSI protocol)



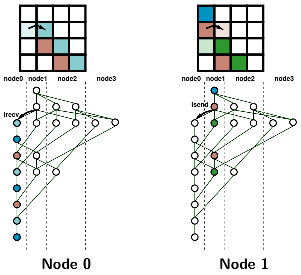
Data dependencies on distributed architectures

Task ↔ Node Mapping

- Provided by the application
- Can be altered dynamically

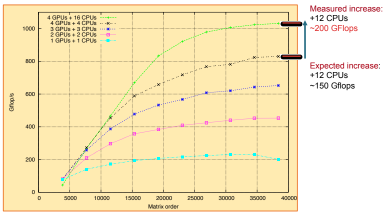
Communications

- Inferred from the task graph
  - Dependencies
- Automatic **Isend** and **Irecv** calls



Showcase with QR factorization on heterogeneous node

Shared memory with accelerators



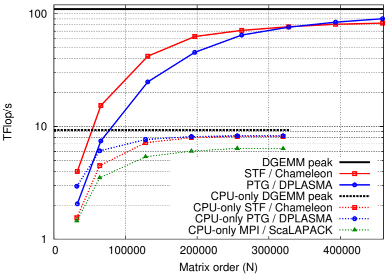
Measured increase:
+12 CPUs
~200 GFlops

Expected increase:
+12 CPUs
~150 Gflops

- QR decomposition on 16 CPUs (AMD) + 4 GPUs (C1060)

Showcase with Cholesky decomposition

Distributed memory with accelerators

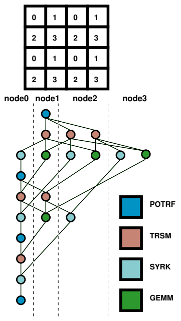


- 144 nodes of curie: 1152 cores + 288 Nvidia M2090

Goal: Separation of concerns

Main advantages:

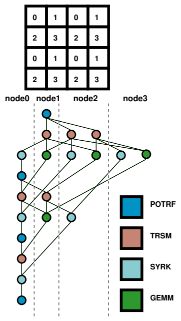
1. Data distribution



Goal: Separation of concerns

Main advantages:

1. Data distribution
2. Algorithm

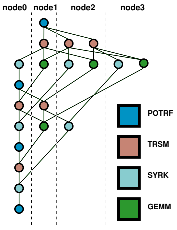


Goal: Separation of concerns

Main advantages:

1. Data distribution
2. Algorithm
3. Tasks distribution

0	1	0	1
2	3	2	3
0	1	0	1
2	3	2	3



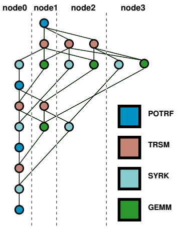
Goal: Separation of concerns

Main advantages:

1. Data distribution
2. Algorithm
3. Tasks distribution

- Allow for simpler composition of algorithms (FActorization and solve, Cholesky inversion, . . .)
- Allow to develop new algorithms without the burden of the communications

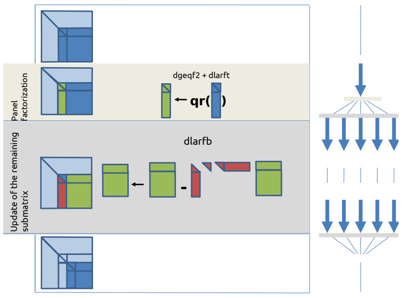
0	1	0	1
2	3	2	3
0	1	0	1
2	3	2	3



4

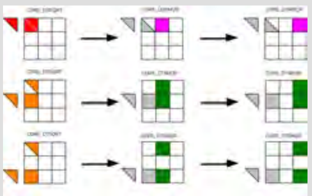
Advanced dense linear algebra algorithm

Example of QR factorization

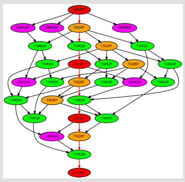


Tile QR Factorization

First panel factorization and corresponding updates



DAG for a 4×4 tiles matrix

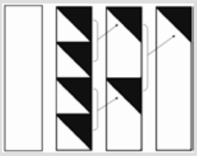


Tile QR Factorization

- Algorithm
  - the same R factor as LAPACK (absolute values)
  - different set of Householder reflectors
  - different Q matrix
  - different Q generation / application procedure
- Numerics
  - same as LAPACK
- Performance
  - comparable to vendor on few cores
  - much better than vendor on many cores

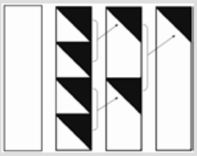
Communication Avoiding QR (CAQR) [Demmel et al.'08]

Tall and Skinny QR (TSQR)

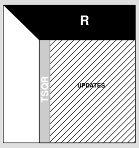


Communication Avoiding QR (CAQR) [Demmel et al.'08]

Tall and Skinny QR (TSQR)

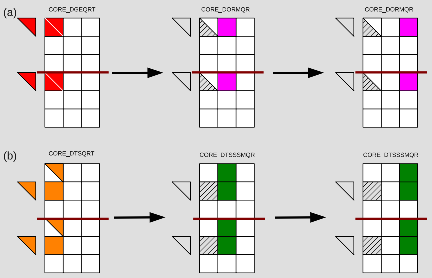


CAQR



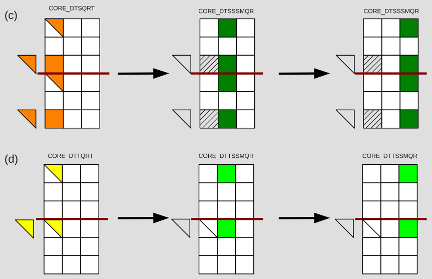
Tile CAQR factorization

First panel factorization and corresponding updates (a & b).



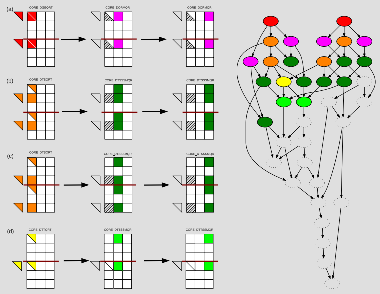
Tile CAQR factorization

First panel factorization and corresponding updates (c & d).



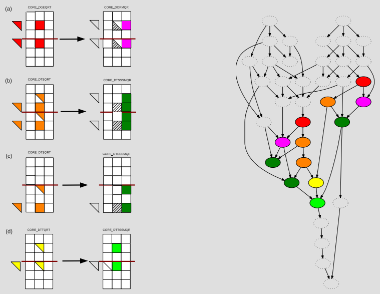
Tile CAQR factorization

First panel factorization and corresponding updates.



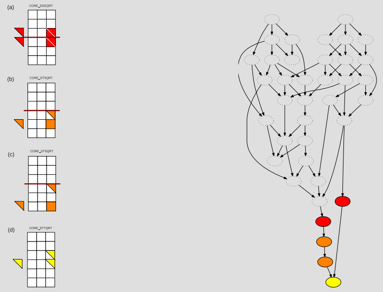
Tile CAQR factorization

Second panel factorization and corresponding updates.



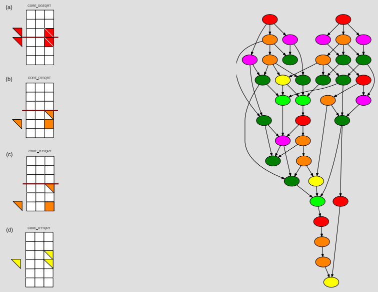
Tile CAQR factorization

Final panel factorization.

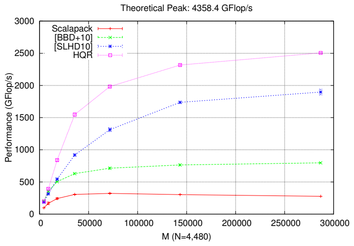


Tile CAQR factorization

Final panel factorization.

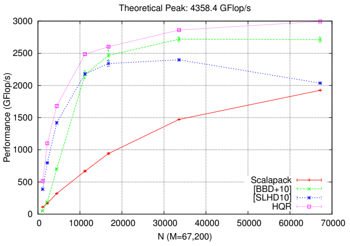


Tile QR factorization performance (N=4480, M varies)



- 2 Nehalem Xeon E5520 at 2.27GHz per node (8 cores)
- P=15, Q=4, MB=280
- Theoretical peak of 4.358 TFlop/s

Tile QR factorization performance (M=67200, N varies)



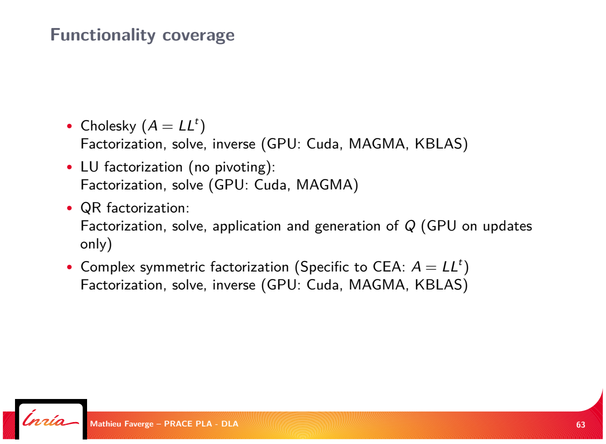
- 2 Nehalem Xeon E5520 at 2.27GHz per node (8 cores)
- P=15, Q=4, MB=280
- Theoretical peak of 4.358 TFlop/s

5

conclusion

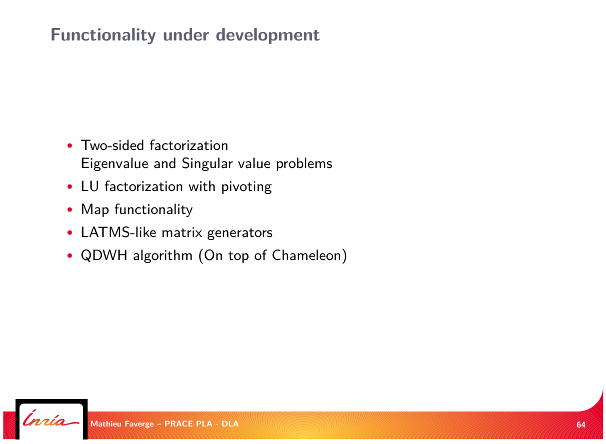
Functionality coverage

- Covers classic four precisions (zcds):
  double complex, single complex, double real, single real
- All BLAS 3 subroutines (CPU or GPU):
  GEMM, TRSM, TRMM, HEMM/SYMM, HERK/SYRK, HER2K/SYR2K
- Some auxiliary subroutines:
  - Matrix generation: random general (PLRNT), hermitian (PLGHE), symmetric (PLGSY)
  - Norms computation: Max, Infinite, One, Frobenius
  - A few extra functions: LASET, LACPY, GEADD, TRADD
- Data distribution is 2D Block Cyclic in tile layout



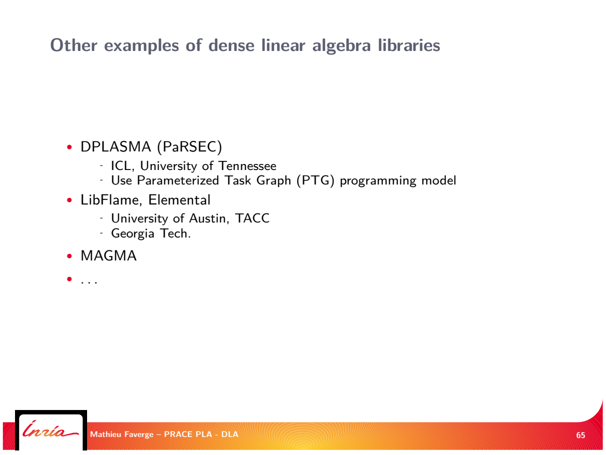
Functionality coverage

- Cholesky ($A = LL^t$)
  Factorization, solve, inverse (GPU: Cuda, MAGMA, KBLAS)
- LU factorization (no pivoting):
  Factorization, solve (GPU: Cuda, MAGMA)
- QR factorization:
  Factorization, solve, application and generation of Q (GPU on updates only)
- Complex symmetric factorization (Specific to CEA: $A = LL^t$)
  Factorization, solve, inverse (GPU: Cuda, MAGMA, KBLAS)



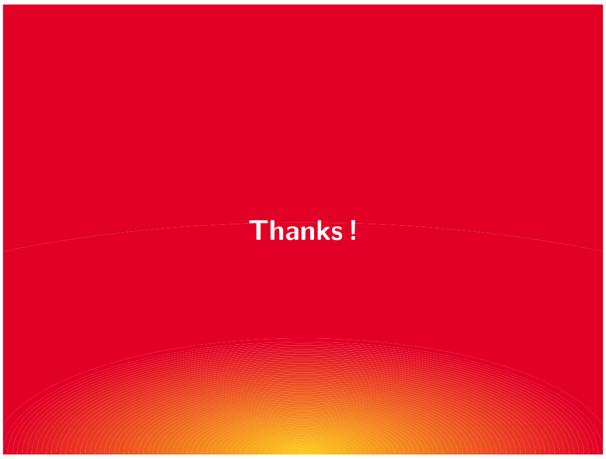
Functionality under development

- Two-sided factorization
  Eigenvalue and Singular value problems
- LU factorization with pivoting
- Map functionality
- LATMS-like matrix generators
- QDWH algorithm (On top of Chameleon)



Other examples of dense linear algebra libraries

- DPLASMA (PaRSEC)
  - ICL, University of Tennessee
  - Use Parameterized Task Graph (PTG) programming model
- LibFlame, Elemental
  - University of Austin, TACC
  - Georgia Tech.
- MAGMA
- . . .



Thanks !

Sparse Linear Algebra
PRACE PLA, Ostrava, Czech Republic

Mathieu Faverge

February 2nd, 2017

Contributions

Many thanks to Patrick Amestoy, Abdou Guermouche, Pascal Henon, and Jean-Yves l'Excellent for their large contribution to these slides.

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

A selection of references

- Books
  - Duff, Erisman and Reid, Direct methods for Sparse Matrices, Clarenton Press, Oxford 1986.
  - Dongarra, Duff, Sorensen and H. A. van der Vorst, Solving Linear Systems on Vector and Shared Memory Computers, SIAM, 1991.
  - Saad, Yousef, Iterative methods for sparse linear systems (2nd edition), SIAM press, 2003
- Articles
  - Gilbert and Liu, Elimination structures for unsymmetric sparse LU factors, SIMAX, 1993.
  - Liu, The role of elimination trees in sparse factorization, SIMAX, 1990.
  - Heath and E. Ng and B. W. Peyton, Parallel Algorithms for Sparse Linear Systems, SIAM review 1991.

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

Motivations

- solution of linear systems of equations → key algorithmic kernel

Continuous problem

↓

Discretization

↓

Solution of a linear system $Ax = b$

- Main parameters:
  - Numerical properties of the linear system (symmetry, pos. definite, conditioning, ...)
  - Size and structure:
    - Large ($> 1000000 \times 1000000$?), square/rectangular
    - Dense or sparse (structured / unstructured)
    - Target computer (sequential/parallel)

→ *Algorithmic choices are critical*

Motivations for designing efficient algorithms

- ⋆ Time-critical applications
- ⋆ Solve larger problems
- ⋆ Decrease elapsed time (parallelism ?)
- ⋆ Minimize cost of computations (time, memory)

Difficulties

- ⋆ Access to data :
  - Computer : complex memory hierarchy (registers, multilevel cache, main memory (shared or distributed), disk)
  - Sparse matrix : large irregular dynamic data structures.

  → *Exploit the locality of references to data on the computer (design algorithms providing such locality)*
- ⋆ Efficiency (time and memory)
  - Number of operations and memory depend very much on the algorithm used and on the numerical and structural properties of the problem.
  - The algorithm depends on the target computer (vector, scalar, shared, distributed, clusters of Symmetric Multi-Processors (SMP), GRID).

  → *Algorithmic choices are critical*

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

Sparse matrices

Example:

$$\begin{array}{ccccccc} 3\,x_1 & + & 2\,x_2 & & & = & 5 \\ & & 2\,x_2 & - & 5\,x_3 & = & 1 \\ 2\,x_1 & & & + & 3\,x_3 & = & 0 \end{array}$$

can be represented as

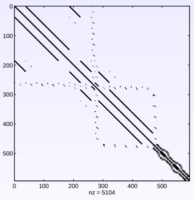
$$\mathbf{Ax} = \mathbf{b},$$

where $\mathbf{A} = \begin{pmatrix} 3 & 2 & 0 \\ 0 & 2 & -5 \\ 2 & 0 & 3 \end{pmatrix}$, $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$, and $\mathbf{b} = \begin{pmatrix} 5 \\ 1 \\ 0 \end{pmatrix}$

Sparse matrix: only nonzeros are stored.

Sparse matrix ?

Original matrix



Matrix dwt_592.rua (N=592, NZ=5104);
Structural analysis of a submarine

Factorization process (direct method)

Solution of $\mathbf{Ax} = \mathbf{b}$

- ⋆ $\mathbf{A}$ is unsymmetric :
  - $\mathbf{A}$ is factorized as: $\mathbf{A} = \mathbf{LU}$, where
    $\mathbf{L}$ is a lower triangular matrix, and
    $\mathbf{U}$ is an upper triangular matrix.
  - Forward-backward substitution: $\mathbf{Ly} = \mathbf{b}$ then $\mathbf{Ux} = \mathbf{y}$
- ⋆ $\mathbf{A}$ is symmetric:
  - $\mathbf{A} = \mathbf{LDL}^T$ or $\mathbf{LL}^T$
- ⋆ $\mathbf{A}$ is rectangular $m \times n$ with $m \geq n$ and $\min_x \|\mathbf{Ax} - \mathbf{b}\|_2$:
  - $\mathbf{A} = \mathbf{QR}$ where $\mathbf{Q}$ is orthogonal ($\mathbf{Q}^{-1} = \mathbf{Q}^T$ and $\mathbf{R}$ is triangular).
  - Solve: $\mathbf{y} = \mathbf{Q}^T\mathbf{b}$ then $\mathbf{Rx} = \mathbf{y}$

Factorization process (direct method)

Solution of $\mathbf{Ax} = \mathbf{b}$

- $\mathbf{A}$ is unsymmetric :
  - $\mathbf{A}$ is factorized as: $\mathbf{A} = \mathbf{LU}$, where
    $\mathbf{L}$ is a lower triangular matrix, and
    $\mathbf{U}$ is an upper triangular matrix.
  - Forward-backward substitution: $\mathbf{Ly} = \mathbf{b}$ then $\mathbf{Ux} = \mathbf{y}$
- $\mathbf{A}$ is symmetric:
  - $\mathbf{A} = \mathbf{LDL}^T$ or $\mathbf{LL}^T$
- $\mathbf{A}$ is rectangular $m \times n$ with $m \geq n$ and $\min_x \|\mathbf{Ax} - \mathbf{b}\|_2$:
  - $\mathbf{A} = \mathbf{QR}$ where $\mathbf{Q}$ is orthogonal ($\mathbf{Q}^{-1} = \mathbf{Q}^T$ and $\mathbf{R}$ is triangular).
  - Solve: $\mathbf{y} = \mathbf{Q}^T\mathbf{b}$ then $\mathbf{Rx} = \mathbf{y}$

Factorization process (direct method)

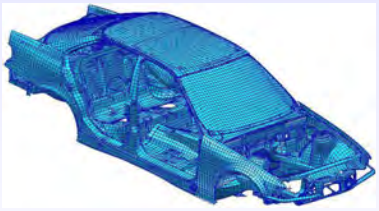
Solution of $\mathbf{Ax} = \mathbf{b}$

- $\mathbf{A}$ is unsymmetric :
  - $\mathbf{A}$ is factorized as: $\mathbf{A} = \mathbf{LU}$, where
    $\mathbf{L}$ is a lower triangular matrix, and
    $\mathbf{U}$ is an upper triangular matrix.
  - Forward-backward substitution: $\mathbf{Ly} = \mathbf{b}$ then $\mathbf{Ux} = \mathbf{y}$
- $\mathbf{A}$ is symmetric:
  - $\mathbf{A} = \mathbf{LDL}^T$ or $\mathbf{LL}^T$
- $\mathbf{A}$ is rectangular $m \times n$ with $m \geq n$ and $\min_x \|\mathbf{Ax} - \mathbf{b}\|_2$:
  - $\mathbf{A} = \mathbf{QR}$ where $\mathbf{Q}$ is orthogonal ($\mathbf{Q}^{-1} = \mathbf{Q}^T$ and $\mathbf{R}$ is triangular).
  - Solve: $\mathbf{y} = \mathbf{Q}^T\mathbf{b}$ then $\mathbf{Rx} = \mathbf{y}$

Difficulties

- Only non-zero values are stored
- Factors $\mathbf{L}$ and $\mathbf{U}$ have far more nonzeros than $\mathbf{A}$
- Data structures are complex
- Computations are only a small portion of the code (the rest is data manipulation)
- Memory size is a limiting factor
  → *out-of-core solvers*

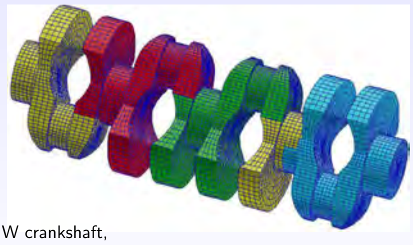
Typical test problems:



BMW car body,
227,362 unknowns,
5,757,996 nonzeros,
MSC.Software

Size of factors: 51.1 million entries
Number of operations: 44.9 $\times 10^9$

Typical test problems:



BMW crankshaft,
148,770 unknowns,
5,396,386 nonzeros,
MSC.Software

Size of factors: 97.2 million entries
Number of operations: 127.9 $\times 10^9$

Sources of parallelism

Several levels of parallelism can be exploited:

- At problem level: problem can de decomposed into sub-problems (e.g. domain decomposition)
- At matrix level arising from its sparse structure
- At submatrix level within dense linear algebra computations (parallel BLAS, . . .)

Data structure for sparse matrices

- Storage scheme depends on the pattern of the matrix and on the type of access required
  - band or variable-band matrices
  - “block bordered” or block tridiagonal matrices
  - general matrix
  - row, column or diagonal access

Data formats for a general sparse matrix $\mathbf{A}$

What needs to be represented

- Assembled matrices: MxN matrix $\mathbf{A}$ with NNZ nonzeros.
- Elemental matrices (unassembled): MxN matrix $\mathbf{A}$ with NELT elements.
- Arithmetic: Real (4 or 8 bytes) or complex (8 or 16 bytes)
- Symmetric (or Hermitian)
  → store only part of the data.
- Distributed format ?
- Duplicate entries and/or out-of-range values ?

Classical Data Formats for Assembled Matrices

- Example of a 3x3 matrix with NNZ=5 nonzeros

$$\begin{array}{c|c|c|c|}
 & 1 & 2 & 3 \\ \hline
1 & a11 & & \\ \hline
2 & & a22 & a23 \\ \hline
3 & a31 & & a33 \\ \hline
\end{array}$$

- Coordinate format

$$\begin{array}{llcccccc}
\text{IC} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{JC} & [1:NNZ] & = & 1 & 1 & 2 & 3 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33}
\end{array}$$

- Compressed Sparse Column (CSC) format

$$\begin{array}{llcccccc}
\text{IA} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33} \\
\text{JA} & [1:N+1] & = & 1 & 3 & 4 & 6 &
\end{array}$$

  column J is stored in IA and VAL at indices JA(J)...JA(J+1)-1
- Compressed Sparse Row (CSR) format:
  Similar to CSC, but row by row

Classical Data Formats for Assembled Matrices

- Example of a 3x3 matrix with NNZ=5 nonzeros

$$\begin{array}{c|c|c|c|}
 & 1 & 2 & 3 \\ \hline
1 & a11 & & \\ \hline
2 & & a22 & a23 \\ \hline
3 & a31 & & a33 \\ \hline
\end{array}$$

- Coordinate format

$$\begin{array}{llcccccc}
\text{IC} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{JC} & [1:NNZ] & = & 1 & 1 & 2 & 3 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33}
\end{array}$$

- Compressed Sparse Column (CSC) format

$$\begin{array}{llcccccc}
\text{IA} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33} \\
\text{JA} & [1:N+1] & = & 1 & 3 & 4 & 6 &
\end{array}$$

  column J is stored in IA and VAL at indices JA(J)...JA(J+1)-1
- Compressed Sparse Row (CSR) format:
  Similar to CSC, but row by row

Classical Data Formats for Assembled Matrices

- Example of a 3x3 matrix with NNZ=5 nonzeros

$$\begin{array}{c|c|c|c|}
 & 1 & 2 & 3 \\ \hline
1 & a11 & & \\ \hline
2 & & a22 & a23 \\ \hline
3 & a31 & & a33 \\ \hline
\end{array}$$

- Coordinate format

$$\begin{array}{llcccccc}
\text{IC} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{JC} & [1:NNZ] & = & 1 & 1 & 2 & 3 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33}
\end{array}$$

- Compressed Sparse Column (CSC) format

$$\begin{array}{llcccccc}
\text{IA} & [1:NNZ] & = & 1 & 3 & 2 & 2 & 3 \\
\text{VAL} & [1:NNZ] & = & a_{11} & a_{31} & a_{22} & a_{23} & a_{33} \\
\text{JA} & [1:N+1] & = & 1 & 3 & 4 & 6 &
\end{array}$$

  column J is stored in IA and VAL at indices JA(J)...JA(J+1)-1
- Compressed Sparse Row (CSR) format:
  Similar to CSC, but row by row

Sparse Matrix-vector products

Assume we want to comute $Y \leftarrow AX$.
Various algorithms for matrix-vector product depending on sparse matrix format:

- Coordinate format:

```
    Y(1:N) = 0
    DO i=1,NNZ
      Y(IC(i)) = Y(IC(i)) + VAL(i) * X(JC(i))
    ENDDO
```

- CSC format:

Sparse Matrix-vector products

Assume we want to comute $Y \leftarrow AX$.
Various algorithms for matrix-vector product depending on sparse matrix format:

- Coordinate format:

```
Y(1:N) = 0
DO i=1,NNZ
   Y(IC(i)) = Y(IC(i)) + VAL(i) * X(JC(i))
ENDDO
```

- CSC format:

```
Y(1:N) = 0
DO J=1,N
   DO I=JA(J),JA(J+1)-1
      Y(IA(I)) = Y(IA(I)) + VAL(I)*X(J)
   ENDDO
ENDDO
```

File storage: Rutherford-Boeing

- Standard ASCII format for files
- Header + Data (CSC format). key xyz:
  - x=[rcp] (real, complex, pattern)
  - y=[suhzr] (sym., uns., herm., skew sym., rectang.)
  - z=[ae] (assembled, elemental)
  - ex: M_T1.RSA, SHIP003.RSE
- Supplementary files: right-hand-sides, solution, permutations...
- Canonical format introduced to guarantee a unique representation (order of entries in each column, no duplicates).

Format description can be found at :
`http://math.nist.gov/MatrixMarket/formats.html`

File storage: Rutherford-Boeing

```
DNV-Ex 1 : Tubular joint-1999-01-17                                    M_T1
       1733710          9758        492558       1231394             0
rsa                    97578         97578       4925574             0
(10I8)          (10I8)          (3e26.16)
       1      49      96     142     187     231     274     346     417     487
     556     624     691     763     834     904     973    1041    1108    1180
    1251    1321    1390    1458    1525    1573    1620    1666    1711    1755
    1798    1870    1941    2011    2080    2148    2215    2287    2358    2428
    2497    2565    2632    2704    2775    2845    2914    2982    3049    3115
...
       1       2       3       4       5       6       7       8       9      10
      11      12      49      50      51      52      53      54      55      56
      57      58      59      60      67      68      69      70      71      72
     223     224     225     226     227     228     229     230     231     232
     233     234     433     434     435     436     437     438       2       3
       4       5       6       7       8       9      10      11      12      49
      50      51      52      53      54      55      56      57      58      59
...
  -0.2624989288237320E+10   0.6622960540857440E+09   0.2362753266740760E+11
   0.3372081648690030E+08  -0.4851430162799610E+08   0.1573652896140010E+08
   0.1704332388419270E+10  -0.7300763190874110E+09  -0.7113520995891850E+10
   0.1813048723097540E+08   0.2955124446119170E+07  -0.2606931100955540E+07
   0.1606040913919180E+07  -0.2377860366909130E+08  -0.1105180386670390E+09
   0.1610636280324100E+08   0.4230082475435230E+07  -0.1951280618776270E+07
   0.4498200951891750E+08   0.2066239484615530E+09   0.3792237438608430E+08
   0.9819999042370710E+08   0.3881169368090200E+08  -0.4624480572242580E+08
```

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

Gaussian elimination

$\mathbf{A} = \mathbf{A}^{(1)}$, $\mathbf{b} = \mathbf{b}^{(1)}$, $\mathbf{A}^{(1)}\mathbf{x} = \mathbf{b}^{(1)}$:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \qquad \begin{array}{l} 2 \leftarrow 2 - 1 \times a_{21}/a_{11} \\ 3 \leftarrow 3 - 1 \times a_{31}/a_{11} \end{array}$$

$\mathbf{A}^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(2)} \end{pmatrix} \qquad \begin{array}{l} b_2^{(2)} = b_2 - a_{21}b_1/a_{11} \dots \\ a_{32}^{(2)} = a_{32} - a_{31}a_{12}/a_{11} \dots \end{array}$$

Finally $\mathbf{A}^{(3)}\mathbf{x} = \mathbf{b}^{(3)}$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & 0 & a_{33}^{(3)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(3)} \end{pmatrix} \qquad a_{(33)}^{(3)} = a_{(33)}^{(2)} - a_{32}^{(2)}a_{23}^{(2)}/a_{22}^{(2)} \dots$$

Typical Gaussian elimination step k : $\boxed{a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)}a_{kj}^{(k)}}{a_{kk}^{(k)}}}$

Relation with $\mathbf{A} = \mathbf{LU}$ factorization

- One step of Gaussian elimination can be written:
  $\mathbf{A}^{(k+1)} = \mathbf{L}^{(k)}\mathbf{A}^{(k)}$, with
  $$\mathbf{L}^k = \begin{pmatrix} \mathbf{1} & & & & & \\ & \ddots & & & & \\ & & \mathbf{1} & & & \\ & & -\mathbf{l_{k+1,k}} & \ddots & & \\ & & \vdots & & \ddots & \\ & & -\mathbf{l_{n,k}} & & & \mathbf{1} \end{pmatrix} \text{ and } l_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}.$$
- Then, $\mathbf{A}^{(n)} = \mathbf{U} = \mathbf{L}^{(n-1)} \dots \mathbf{L}^{(1)}\mathbf{A}$, which gives $\boxed{\mathbf{A} = \mathbf{LU}}$,
  with $\mathbf{L} = [\mathbf{L}^{(1)}]^{-1} \dots [\mathbf{L}^{(n-1)}]^{-1} = \begin{pmatrix} \mathbf{1} & & \mathbf{0} & \\ & \ddots & & \\ & & \ddots & \\ & & \mathbf{l_{i,j}} & \ddots & \mathbf{1} \end{pmatrix}$,
- In dense codes, entries of $\mathbf{L}$ and $\mathbf{U}$ overwrite entries of $\mathbf{A}$.
- Furthermore, if $\mathbf{A}$ is symmetric, $\boxed{\mathbf{A} = \mathbf{LDL}^{\mathrm{T}}}$ with $d_{kk} = a_{kk}^{(k)}$:
  $A = LU = A^t = U^tL^t$ implies $(U)(L^t)^{-1} = L^{-1}U^t = D$ diagonal and $U = DL^t$, thus $A = L(DL^t) = LDL^t$

Gaussian elimination and sparsity

Step k of $\mathbf{LU}$ factorization (a_{kk} pivot):

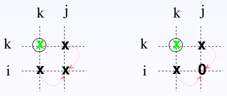
- ⋆ For $i > k$ compute $l_{ik} = a_{ik}/a_{kk}$ $(= a'_{ik})$,
- ⋆ For $i > k, j > k$

$$a'_{ij} = a_{ij} - \frac{a_{ik} \times a_{kj}}{a_{kk}}$$

or

$$a'_{ij} = a_{ij} - l_{ik} \times a_{kj}$$

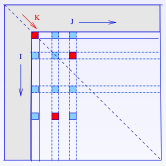
- ⋆ If $a_{ik} \neq 0$ et $a_{kj} \neq 0$ then $a'_{ij} \neq 0$
- ⋆ If a_{ij} was zero → its non-zero value must be stored



fill-in

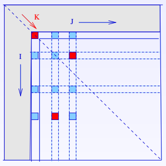
Factorisation LU (version KIJ)

```
for $k = 1$ to $n - 1$ do
    for $i = k + 1$ to $n$ do
        $a_{ik} := a_{ik}/a_{kk}$;
        for $j = k + 1$ to $n$ do
            $a_{ij} := a_{ij} - a_{ik} * a_{kj}$;
        end
    end
end
```



Factorisation LDLt (version KIJ)

```
for $k = 1$ to $n - 1$ do
    for $j = k + 1$ to $n$ do
        $t := a_{jk}/a_{kk}$;
        for $i = j$ to $n$ do
            $a_{ij} := a_{ij} - a_{ik} * t$;
        end
        $a_{jk} := t$;;
    end
end
```



Example

- ⋆ Original matrix

$$\begin{pmatrix} X & X & X & X & X \\ X & X & & & \\ X & & X & & \\ X & & & X & \\ X & & & & X \end{pmatrix}$$

- ⋆ Matrix is full after the first step of elimination
- ⋆ After reordering the matrix (1st row and column ↔ last row and column)

Example

$$\begin{pmatrix} X & & & & X \\ & X & & & X \\ & & X & & X \\ & & & X & X \\ X & X & X & X & X \end{pmatrix}$$

- ⋆ No fill-in
- ⋆ Ordering the variables has a strong impact on
  - ▸ the fill-in
  - ▸ the number of operations

Efficient implementation of sparse solvers

- ⋆ Indirect addressing is often used in sparse calculations: e.g. sparse SAXPY

```
do i = 1, m
   A( ind(i) ) = A( ind(i) ) + alpha * w( i )
enddo
```

- ⋆ Even if manufacturers provide hardware for improving indirect addressing
  - ▸ It penalizes the performance
- ⋆ Switching to dense calculations as soon as the matrix is not sparse enough

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

Symmetric matrices and graphs

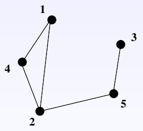
- ⋆ Assumptions: $\mathbf{A}$ symmetric and pivots are chosen on the diagonal
- ⋆ Structure of $\mathbf{A}$ symmetric represented by the graph $G = (V, E)$
  - Vertices are associated to columns: $V = \{1, ..., n\}$
  - Edges E are defined by: $(i, j) \in E \leftrightarrow a_{ij} \neq 0$
  - G undirected (symmetry of $\mathbf{A}$)

Symmetric matrices and graphs

- ⋆ Remarks:
  - Number of nonzeros in column $j = |Adj_G(j)|$
  - Symmetric permutation $\equiv$ renumbering the graph

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \times & \times & & \times & \\ 2 & \times & \times & & \times & \times \\ 3 & & & \times & & \times \\ 4 & \times & \times & & \times & \\ 5 & & \times & \times & & \times \end{array}$$

Symmetric matrix



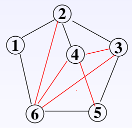
Corresponding graph

Outline

1. Introduction to Sparse Matrix Computations
   - Motivation and main issues
   - Sparse matrices
   - Gaussian elimination
   - Symmetric matrices and graphs
   - The elimination graph model

Introducing the filled graph $G^+(\mathbf{A})$

- ⋆ Let $\mathbf{F} = \mathbf{L} + \mathbf{L}^T$ be the filled matrix, and $G(\mathbf{F})$ the *filled graph* of $\mathbf{A}$ denoted by $G^+(\mathbf{A})$.
- ⋆ Lemma (Parter 1961) : $(v_i, v_j) \in G^+$ if and only if $(v_i, v_j) \in G$ or $\exists k < \min(i, j)$ such that $(v_i, v_k) \in G^+$ and $(v_k, v_j) \in G^+$.



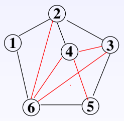
$\mathbf{G^{+}(A) = G(F)}$

$$\begin{bmatrix} \mathbf{1} & \times & & & & \times \\ \times & \mathbf{2} & \times & \times & & + \\ & \times & \mathbf{3} & + & \times & + \\ & \times & + & \mathbf{4} & + & + \\ & & \times & + & \mathbf{5} & \times \\ \times & + & + & + & \times & \mathbf{6} \end{bmatrix}$$

$\mathbf{F = L + L^T}$

Modeling elimination by reachable sets

- ⋆ The fill edge (v_4, v_6) is due to the path (v_4, v_2, v_6) in G_1. However (v_2, v_6) originates from the path (v_2, v_1, v_6) in G_0.
- ⋆ Thus the path (v_4, v_2, v_1, v_6) in the original graph is in fact responsible of the fill in edge (v_4, v_6).
- ⋆ Illustration :



$\mathbf{G^{+}(A) = G(F)}$

$$\begin{bmatrix} \mathbf{1} & \times & & & & \times \\ \times & \mathbf{2} & \times & \times & & + \\ & \times & \mathbf{3} & + & \times & + \\ & \times & + & \mathbf{4} & + & + \\ & & \times & + & \mathbf{5} & \times \\ \times & + & + & + & \times & \mathbf{6} \end{bmatrix}$$

$\mathbf{F = L + L^T}$

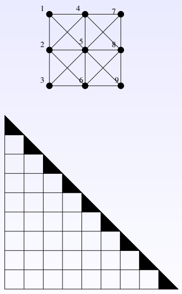
Fill-in theorem

Theorem

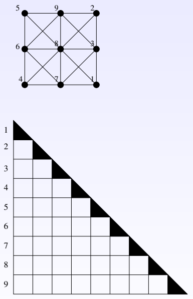
Any $A_{ij} = 0$ will become a non-null entry L_{ij} or $U_{ij} \neq 0$ in $A = L.U$ if and only if it exists a path in $G_A(V, E)$ from vertex i to vertex j that only goes through vertices with a lower number than i and j.

Exercise

Find the fill-in terms.



Matrice 3x3 : 1ere numerotation



Matrice 3x3 : 2eme numerotation

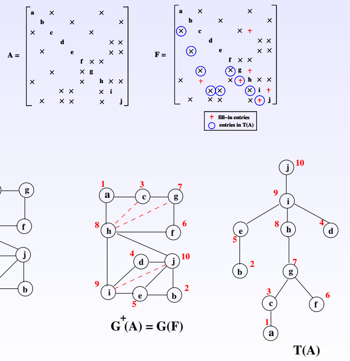
A first definition of the elimination tree

- A **spanning** tree of a connected graph G is a subgraph T of G such that if there is a path in G between i and j then there exists a path between i and j in T.
- Let $\mathbf{A}$ be a symmetric positive-definite matrix, $\mathbf{A} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$ its Cholesky factorization, and $G^+(\mathbf{A})$ its filled graph (graph of $\mathbf{F} = \mathbf{L} + \mathbf{L}^{\mathrm{T}}$).

Definition

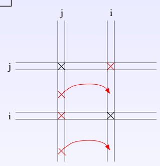
The **elimination tree** of $\mathbf{A}$ is a spanning tree of $G^+(\mathbf{A})$ satisfying the relation $PARENT[j] = min\{i > j | l_{ij} \neq 0\}$.

Graph structures



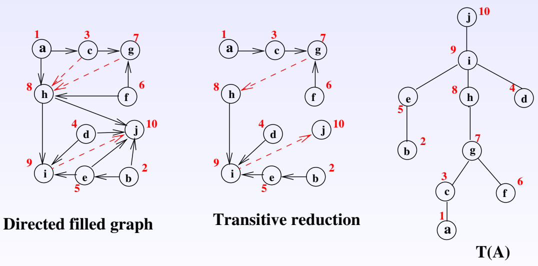
Properties of the elimination tree

- Another perspective also leads to the elimination tree



- Dependency between columns of $\mathbf{L}$:
  1. Column $i > j$ depends on column j iff $l_{ij} \neq 0$
  2. Use a directed graph to express this dependency
  3. Simplify redundant dependencies (*transitive reduction* in graph theory)
- *The transitive reduction of the directed filled graph gives the elimination tree structure*

Directed filled graph and its transitive reduction



Major steps for solving sparse linear systems

There are 4 steps :

1. Reordering :find a (symmetric) permutation P such that it minimizes fill-in in the factorization of $P.A.P^t$. Furthermore, in a parallel context, it should create as much as possible independent computation tasks.
2. Symbolic factorization : this step aims at computing the non-zeros structure of the factors before the actual numeric factorization. It avoids to manage a costly dynamic structure and allows to do some load-balancing.
3. Numerical factorization : compute the factorization by using the preallocated structure from the symbolic factorization.
4. Triangular solve : obtain the solution of $A.x = L.(U.x) = b$. Forward solve $L.y = b$ then a backward solve $U.x = y$. In some cases, it is required to use iterative refinements to increase the accuracy of the solution.

Outline

2. Ordering sparse matrices
   - Objectives/Outline
   - Impact of fill reduction algorithm on the shape of the tree

Outline

2. Ordering sparse matrices
   - Objectives/Outline
   - Impact of fill reduction algorithm on the shape of the tree

Fill-reducing orderings

Three main classes of methods for minimizing fill-in during factorization

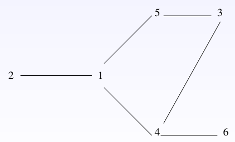
- ⋆ Selection of next best pivot (e. g. : minimum degree for symmetric matrices).
- ⋆ Cuthill-McKee (block tridiagonal matrix)
- ⋆ Nested dissections ("block bordered" matrix).

Cuthill-McKee and Reverse Cuthill-McKee

Consider the matrix:

$$\mathbf{A} = \begin{bmatrix} x & x & & x & x & \\ x & x & & & & \\ & & x & x & x & \\ x & & x & x & & x \\ x & & x & & x & \\ & & & x & & x \end{bmatrix}$$

The corresponding graph is



Cuthill-McKee algorithm

- ⋆ Goal: reduce the profile/bandwidth of the matrix
  (the fill is restricted to the band structure)
- ⋆ Level sets (such as Breadth First Search) are built from the vertex of minimum degree (priority to the vertex of smallest number)
  We get: $S_1 = \{2\}, S_2 = \{1\}, S_3 = \{4, 5\}, S_4 = \{3, 6\}$ and thus the ordering 2, 1, 4, 5, 3, 6.

The reordered matrix is:

$$A = \begin{bmatrix} x & x & & & & \\ x & x & x & x & & \\ & x & x & & x & x \\ & x & & x & x & \\ & & x & x & x & \\ & & x & & & x \end{bmatrix}$$

Reverse Cuthill-McKee

- The ordering is the reverse of that obtained using Cuthill-McKee i.e. on the example $\{6, 3, 5, 4, 1, 2\}$
- The reordered matrix is:

$$A = \begin{bmatrix} x & & & x & & \\ & x & x & x & & \\ & x & x & & x & \\ x & x & & x & x & \\ & & x & x & x & x \\ & & & & x & x \end{bmatrix}$$

- More efficient than Cuthill-McKee at reducing the envelop of the matrix.

Illustration: Reverse Cuthill-McKee on matrix dwt_592.rua

Harwell-Boeing matrix: dwt_592.rua, structural computing on a submarine. NZ(LU factors)=58202

Original matrix *Factorized matrix*

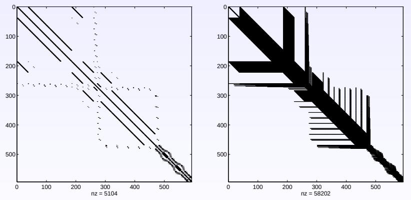
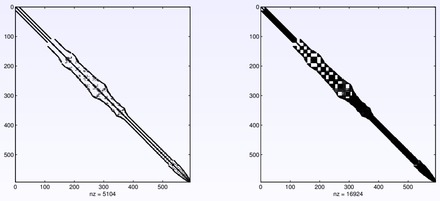


Illustration: Reverse Cuthill-McKee on matrix dwt_592.rua

NZ(LU factors)=16924

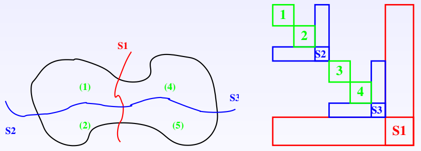
Permuted matrix (RCM) *Factorized permuted matrix*



Nested Dissection

Recursive approach based on graph partitioning.

Graph partitioning *Permuted matrix*



Nested Dissection : Algorithm

G(V,E) is the adjacency graph of A (V = vertices, E = edges).
In the recursive algorithm k is a global variable initialized to $n =$ card(G).
It represented the next number to be given.

```
NestedDissection(G) : ;
if G non dissecable then
    Number the vertices of V from $k$ to $k := k - |V| + 1$ ;
end
else
    Find a partition $V = A \bigcup B \bigcup S$ with S a separator of G;
    Number the vertices of S from $k$ to $k := k - |S| + 1$ ;
    NestedDissection(G(A)) : ;
    NestedDissection(G(B)) : ;
end
```

Ordering : efficient strategy

The modern software (e.g. METIS `http://glaros.dtc.umn.edu/gkhome/views/metis/` or SCOTCH `http://www.labri.fr/perso/pelegrin/scotch/`) are based on on the nested dissection algorithm but :

- they use hybrid ordering ND + local heuristics (e.g. Minimum degree) ;
- they use multilevel approaches : graph coarsening, reordering on the reduced graph, graph uncoarsening.

Outline

2. Ordering sparse matrices
   - Objectives/Outline
   - Impact of fill reduction algorithm on the shape of the tree

Impact of fill reduction on the shape of the tree (1/2)

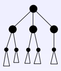
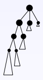
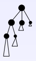
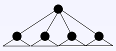
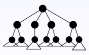
Reordering technique	Shape of the tree	observations
AMD		⋆ Deep well-balanced ⋆ Large frontal matrices on top
AMF		⋆ Very deep unbalanced ⋆ Small frontal matrices

Figure: Shape of the trees resulting from various reordering techniques.

Impact of fill reduction on the shape of the tree (2/2)

Reordering technique	Shape of the tree	observations
PORD		⋆ deep unbalanced ⋆ Small frontal matrices
SCOTCH		⋆ Very wide well-balanced ⋆ Large frontal matrices
METIS		⋆ Wide well-balanced ⋆ Smaller frontal matrices (than SCOTCH)

Importance of the shape of the tree

Suppose that each node in the tree corresponds to a task that:
- consumes temporary data from the children,
- produces temporary data, that is passed to the parent node.

- ⋆ Wide tree
  - Good parallelism
  - Many temporary blocks to store
  - Large memory usage
- ⋆ Deep tree
  - Less parallelism
  - Smaller memory usage

Outline

3. Symbolic factorization
   - Symbolic factorization : column algorithm
   - Symbolic factorization : column-block algorithm

Outline

3. Symbolic factorization
   - Symbolic factorization : column algorithm
   - Symbolic factorization : column-block algorithm

Symbolic factorization

The goal of this algorithm is to build the non-zero pattern of L (and U). We will consider the symmetric case (graph of $A + A^t$ if A has an unsymmetric NZ pattern). In this case the symbolic factorization is really cheaper than the factorization algorithm.

Fundamental property

The symbolic factorization relies on the elimination tree of A.

Symbolic factorization : Algorithm (1/3)

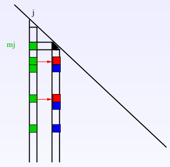
For a sparse matrix A we will denote by:

Definition

Row(A_{i*}) $= \{k < i / A_{ik} \neq 0\}$, for $i = 1..n$
Col(A_{*j}) $= \{k > j / A_{kj} \neq 0\}$, for $j = 1..n$
We will denote by SRow and SCol the sorted set.

Symbolic factorization : Algorithm (2/3)

```
for $j = 1$ to $n - 1$ do
    Build SCol($A_{*j}$)
end
;
for $j = 1$ to $n - 1$ do
    $m_j :=$ first elt of SCol($A_{*j}$) ;
    SCol($A_{*m_j}$) :=
      Merge(SCol($A_{*m_j}$), SCol($A_{*j}$) $- m_j$) ;
end
```



Symbolic factorization : Algorithm (3/3)

At the end of algorithm we have :
SCol(A_{*j}) for $j = 1..n$
The algorithm uses two loops :

Complexity

The complexity of the symbolic factorization is in $O(\|E^*\|)$ the number of edges in the elimination graph.

Outline

3. Symbolic factorization
   - Symbolic factorization : column algorithm
   - Symbolic factorization : column-block algorithm

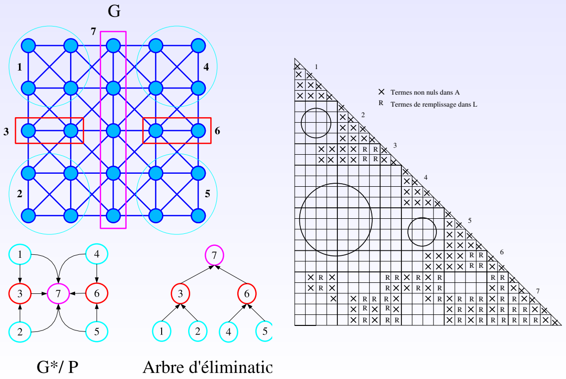
Block Symbolic factorization

The problem in the symbolic factorization is the memory needs.
It is of the same order than the factorization.
In fact, we can use the partition deduced from the ordering to compute a block structure of the matrix.

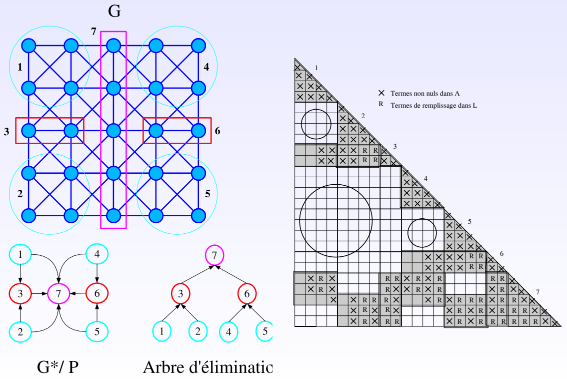
Definition

A *supernode* (or supervariable) is a set of contiguous columns in the factors $\mathbf{L}$ that share essentially the same sparsity structure.

Quotient graph and block elimination tree



Quotient graph and block elimination tree



Quotient graph and block elimination tree

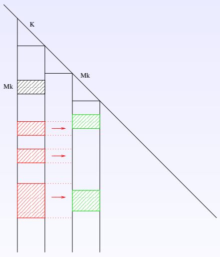
The block symbolic factorization relies on

Property of the elimination graph

$$Q(G,P)^* = Q(G^*,P)$$

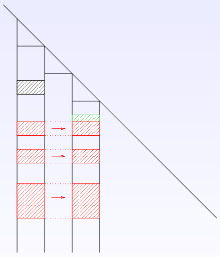
Block Symbolic factorization : Algo

```
for $k = 1$ to $N - 1$ do
    Build $I_k$ = the list of block intervals
end
;
for $k = 1$ to $N - 1$ do
    $m_k := n(k,1)$ (first extra-diagonal block in $k$) ;
    $I_{m_k} :=$ Merge($I_{m_k}, (I_k - [m_k])$) ;
end
```



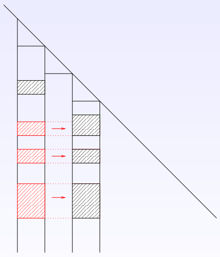
Block Symbolic factorization : Algo

```
for $k = 1$ to $N - 1$ do
    Build $I_k$ = the list of block intervals
end
;
for $k = 1$ to $N - 1$ do
    $m_k := n(k,1)$ (first extra-diagonal block in $k$) ;
    $I_{m_k} :=$ Merge($I_{m_k}, (I_k - [m_k])$) ;
end
```



Block Symbolic factorization : Algo

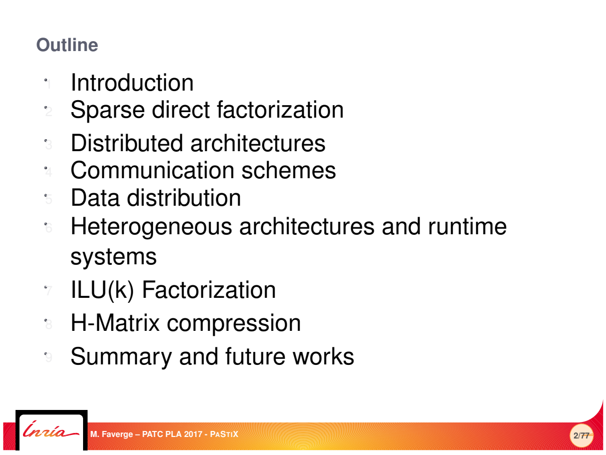
```
for $k = 1$ to $N - 1$ do
    Build $I_k$ = the list of block intervals
end
;
for $k = 1$ to $N - 1$ do
    $m_k := n(k,1)$ (first extra-diagonal block in $k$) ;
    $I_{m_k} :=$ Merge($I_{m_k}, (I_k - [m_k])$) ;
end
```



PaStiX: A sparse direct solver based on super-nodal method

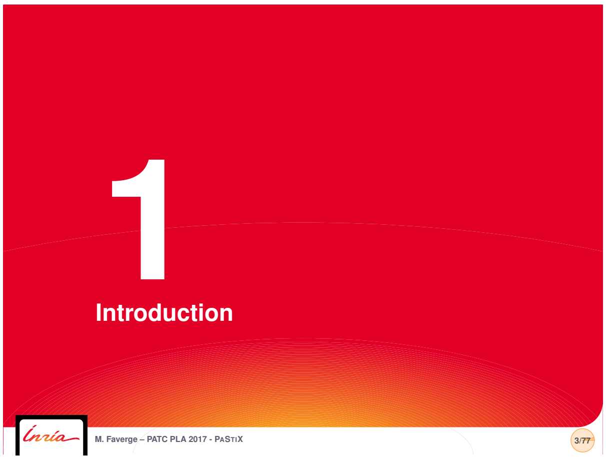
PATC Parallel Linear Algebra @ IT4Innovations, February 2nd, 2017

M. Faverge

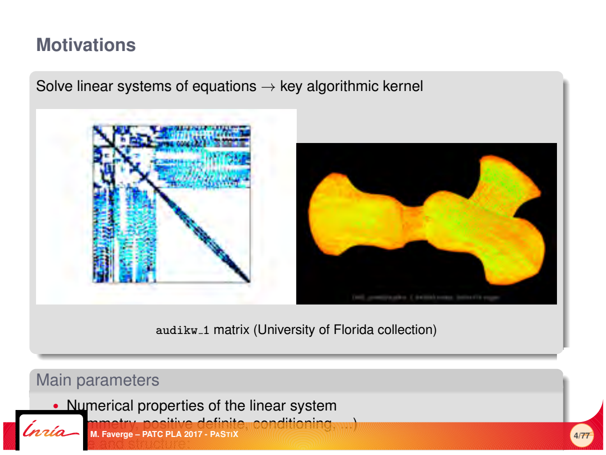


Outline

1. Introduction
2. Sparse direct factorization
3. Distributed architectures
4. Communication schemes
5. Data distribution
6. Heterogeneous architectures and runtime systems
7. ILU(k) Factorization
8. H-Matrix compression
9. Summary and future works



1 Introduction



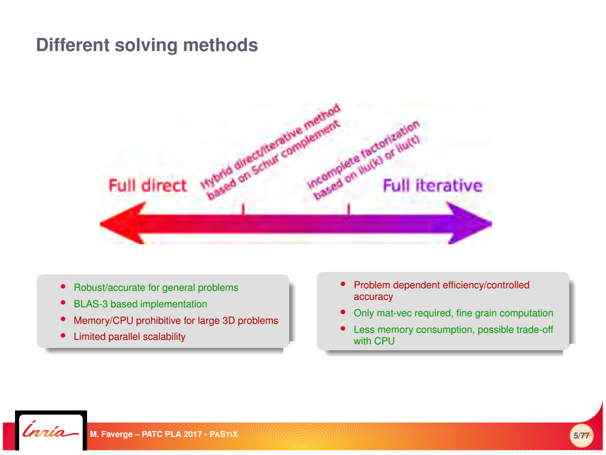
Motivations

Solve linear systems of equations → key algorithmic kernel

audikw_1 matrix (University of Florida collection)

Main parameters

- Numerical properties of the linear system
  mmetry, positive definite, conditioning, ...)
  and structure



Different solving methods

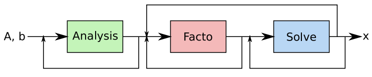
Full direct
Hybrid direct/iterative method based on Schur complement
incomplete factorization based on ilu(k) or ilu(t)
Full iterative

- Robust/accurate for general problems
- BLAS-3 based implementation
- Memory/CPU prohibitive for large 3D problems
- Limited parallel scalability

- Problem dependent efficiency/controlled accuracy
- Only mat-vec required, fine grain computation
- Less memory consumption, possible trade-off with CPU

2 Direct

Major steps for solving sparse linear systems



Direct methods steps

Steps of the factorization

1. Analysis (on the graph of the matrix):
   - **1.1** Ordering: minimize fill-in while maximizing parallelism
   - **1.2** Symbolic factorization: predict the structure of the factorized matrix
   - **1.3** Data distribution: optimize the data distribution to accelerate the factorization
2. Numerical factorization: decomposition of A into LU, LL^T, or LDL^T
3. Triangular systems solves: the solution x is computed by means of forward and backward substitutions

Ordering

Goals

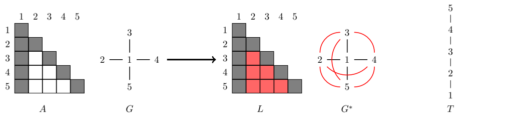
- reduce fill-in during factorization
- increase parallelism

Graph theory

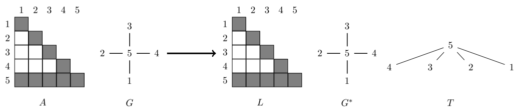
- Uses graph representation: $\exists (i,j) \in G \Leftrightarrow a_{ij} \neq 0$
- Characterization theorem:

$$l_{i,j} \neq 0 \Leftrightarrow \begin{cases} (i,j) \in G \\ or \\ \exists \text{ a path } (j, k_1, \dots, k_l, i) \text{ such that } \forall p \in [\![1,l]\!], k_p < min(i,j) \end{cases}$$

Ordering example



Without reordering



With reordering

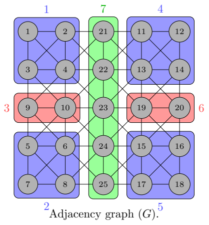
Nested dissection: a widely used algorithm in direct solvers

Algorithm

1. Find a list of vertices separating the graph into two pieces
2. Number this separator with higher indices
3. Iterate on sub-graphs

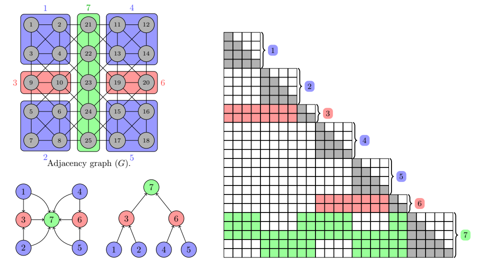
Advantages

- There is no fill-in between two separated graphs
- The two corresponding sub-parts of the matrix can be computed independently



Adjacency graph (G).

Symbolic factorization



Adjacency graph (G).

Quotient graph (G*/P).

Elimination tree (T).

Factorized matrix (L).

Columns are blocked into supernodes on which BLAS can be executed.

Symbolic factorization

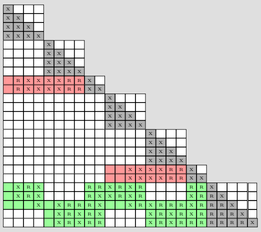
The goal of this algorithm is to build the non-zero pattern of L (and U). We will consider the symmetric case (graph of $A + A^t$ if A has an unsymmetric pattern). In this case the symbolic factorization is really cheaper than the factorization algorithm.

Fundamental property

The symbolic factorization relies on the elimination tree of A.

Quotient graph and block elimination tree

Property of the elimination graph : $Q(G,P)^* = Q(G^*,P)$



X: Non-zero terms of A, R: Fill-in terms of L.

Supernodal methods

Definition

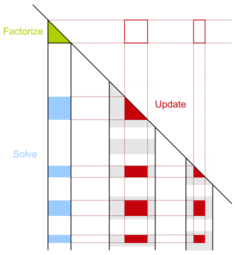
A supernode (or supervariable) is a set of contiguous columns in the factors L that share essentially the same sparsity structure.

- All algorithms (ordering, symbolic factor., factor., solve) generalized to block versions.
- Use of efficient matrix-matrix kernels (improve cache usage).
- Same concept as supervariables for elimination tree/minimum degree ordering.
- Supernodes and pivoting: pivoting inside a supernode does not increase fill-in.

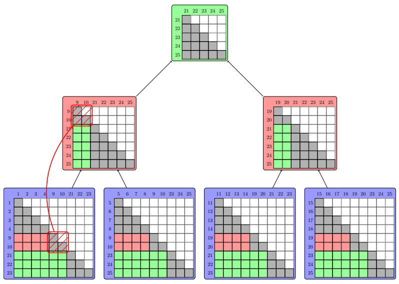
Numerical factorization: super-nodal method

Algorithm 1: Right looking blocked sequential factorization: $A = LL^T$.

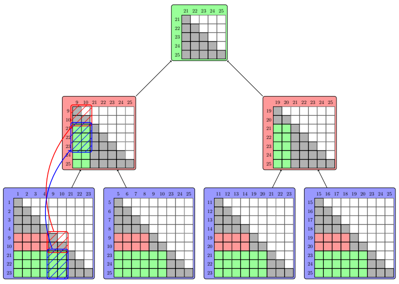
```
for $k = 1$ to $N$ do
    /* Factorize the column block  */
    Factorize $A_{k,k}$ in $L_{k,k}.L_{k,k}^T$;
    Solve $L_{(1-b_k),k}.L_{k,k}^T = A_{(1-b_k),k}$;
    /* Trailling supernodes updates
        */
    for $j = 1$ to $b_k$ do
        for $i = 1$ to $b_k$ do
            $A_{(i),(j)} = A_{(i),(j)} - L_{(i),k}.L_{k,(j)}^T$;
        end
    end
end
```



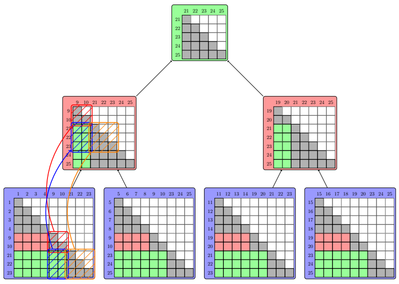
Numerical factorization: multi-frontal method



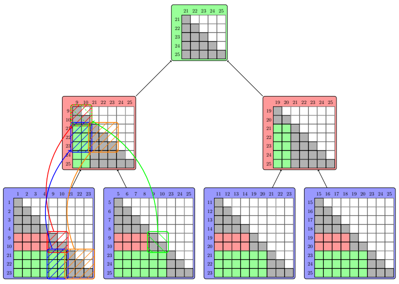
Numerical factorization: multi-frontal method



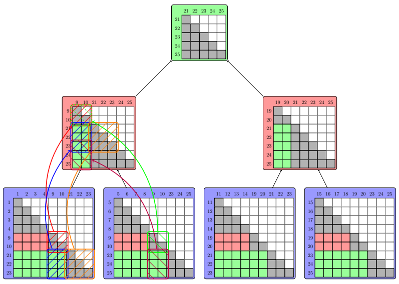
Numerical factorization: multi-frontal method



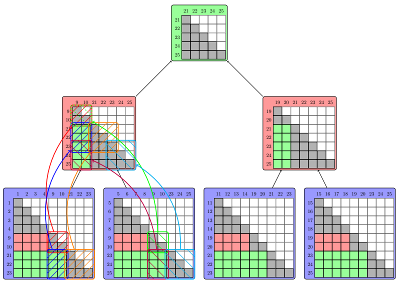
Numerical factorization: multi-frontal method



Numerical factorization: multi-frontal method



Numerical factorization: multi-frontal method



3

Distributed architectures

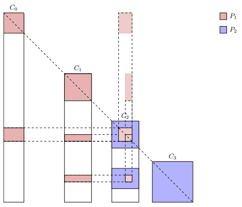
4

Communication schemes

Distributed matrix example (Fan-out)

Fan-out

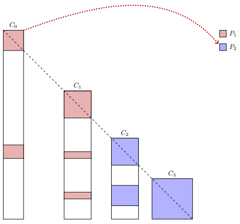
1. Column blocks are sent to target block's owner
2. Target block's owner performs the update



Distributed matrix example (Fan-out)

Fan-out

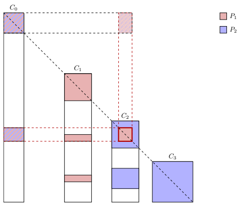
1. Column blocks are sent to target block's owner
2. Target block's owner performs the update



Distributed matrix example (Fan-out)

Fan-out

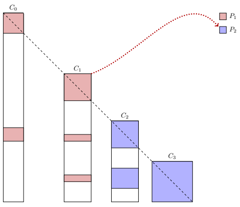
1. Column blocks are sent to target block's owner
2. Target block's owner performs the update



Distributed matrix example (Fan-out)

Fan-out

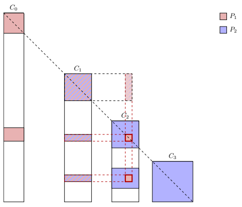
1. Column blocks are sent to target block's owner
2. Target block's owner performs the update



Distributed matrix example (Fan-out)

Fan-out

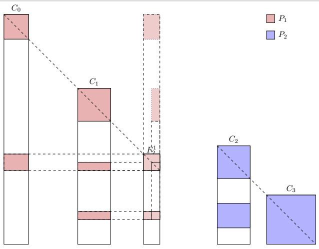
1. Column blocks are sent to target block's owner
2. Target block's owner performs the update



Distributed matrix example (Fan-in)

Fan-in

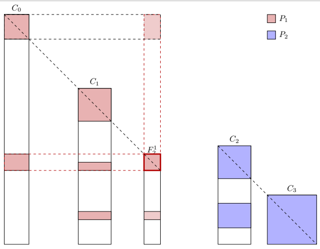
1. Contribution aggregated locally in a temporary buffer
2. Contribution sent to receiver when complete
3. Contribution added on receiver



Distributed matrix example (Fan-in)

Fan-in

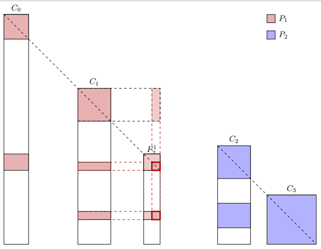
1. Contribution aggregated locally in a temporary buffer
2. Contribution sent to receiver when complete
3. Contribution added on receiver



Distributed matrix example (Fan-in)

Fan-in

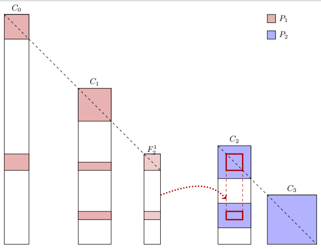
1. Contribution aggregated locally in a temporary buffer
2. Contribution sent to receiver when complete
3. Contribution added on receiver



Distributed matrix example (Fan-in)

Fan-in

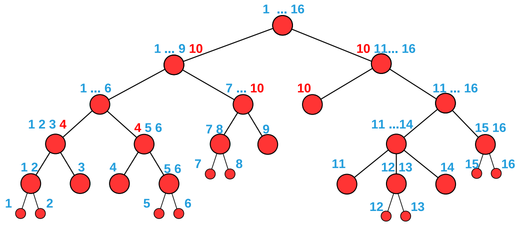
1. Contribution aggregated locally in a temporary buffer
2. Contribution sent to receiver when complete
3. Contribution added on receiver



5

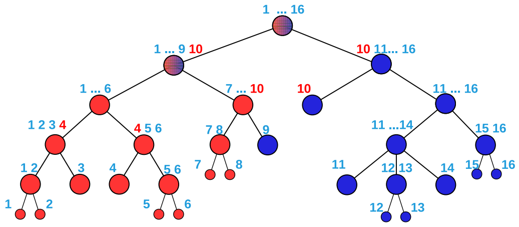
Data distribution

Proportional Mapping [A. Pothen, C. Sun 93]



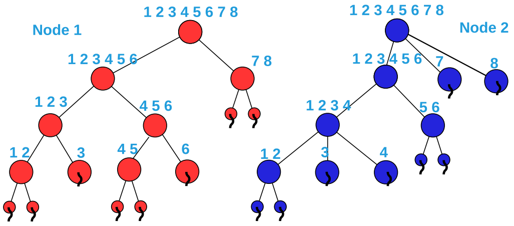
- Top-down strategy to build candidate processor groups for each block (ensure good locality of communications)
- Down-top mapping induced by a logical simulation of computations of the block solver (models for factorization time, communication and aggregation)

Dynamic Scheduling : New Mapping



- Need to map data on MPI process
- Two steps :
  - **A first proportional mapping step to map data**
  - A second step to build a file structure for the work stealing algorithm

Dynamic Scheduling : New Mapping

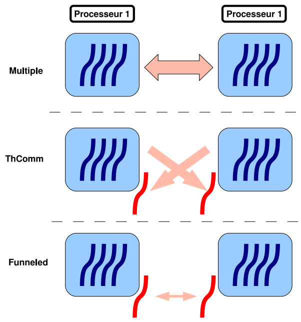


- Need to map data on MPI process
- Two steps :
  - A first proportional mapping step to map data
  - **A second step to build a file structure for the work stealing algorithm**

Thread support inside MPI libraries

- `MPI_THREAD_SINGLE`
  - Only one thread will execute.
- `MPI_THREAD_FUNNELED`
  - The process may be multi-threaded, but only the main thread will make MPI calls
    (all MPI calls are funneled to the main thread).
- `MPI_THREAD_SERIALIZED`
  - The process may be multi-threaded, and multiple threads may make MPI calls, but only one at a time: MPI calls are not made concurrently from two distinct threads
    (all MPI calls are serialized).
- `MPI_THREAD_MULTIPLE`
  - Multiple threads may call MPI, with no restrictions.

Communication schemes (Up to 10% efficiency)



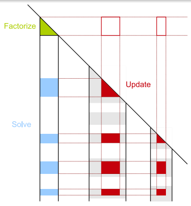
6

Heterogeneous architectures and runtime systems

Numerical Factorization

Algorithm to eliminate the block column k

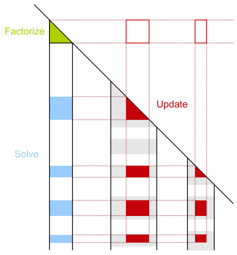
1. Factorize the diagonal block
2. Solve off-diagonal blocks in the current column (TRSM)
3. Update the underlying matrix with the column's contribution (GEMM)



Possible variants

- One single update ≈ multi-frontal
- 1D updates per block of columns
- 2D updates ≈ Dense factorization

Numerical Factorization on GPUs: PaStiX (historical) choices



- Supernodal method
- 1D updates
- Sequential BLAS
- Internal scheduler, or external runtime to support accelerators

Advantages of using a task-based runtime system

- Several computing kernels can be associated with the task (C, OpenCL, NVIDIA CUDA)
- Execute the task graph on the available resources
- Address the whole computing units and the whole potential parallelisms
- Insulate the algorithm from the architecture and data distribution
- Automatic handling of data transfers
- Finer parallelism handling

Runtime systems supported by PaStiX

StarPU

- Inria Storm Team
- **Dynamic Task Discovery**
- Computes cost models on the fly
- Multiple kernels on the accelerators
- Multiple scheduling strategies: Minimum Completion Time, Local Work Stealing, user defined...

PaRSEC

- ICL – University of Tennessee, Knoxville
- **Parameterized Task Graph**
- Only the most compute intensive kernel on accelerators
- Scheduling strategy based on static performance model
- GPU multi-stream enabled

Runtime systems supported by PaStiX

StarPU

- Inria Storm Team
- **Dynamic Task Discovery**
- Computes cost models on the fly
- Multiple kernels on the accelerators
- Multiple scheduling strategies: Minimum Completion Time, Local Work Stealing, user defined...

PaRSEC

- ICL – University of Tennessee, Knoxville
- **Parameterized Task Graph**
- Multiple kernels on the accelerators
- Scheduling strategy based on static performance model
- GPU multi-stream enabled

We consider only PaRSEC runtime in this talk

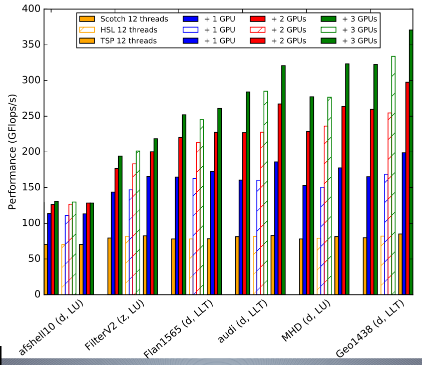
Test platform

Matrix	Size	nnz_A	nnz_L	TFlop
Afshell10	1.5e+6	27e+6	610e+6	0.12
FilterV2	0.6e+6	12e+6	536e+6	3.6
fault639	0.6e+6	29e+6	1234e+6	8.28
audi	0.9e+6	78e+6	1276e+6	5.12
boneS10	0.9e+6	55e+6	370e+6	0.29
inline	0.5e+6	37e+6	212e+6	0.14
ldoor	0.9e+6	47e+6	304e+6	1.08
nd24	0.07e+6	28e+6	335e+6	2.17

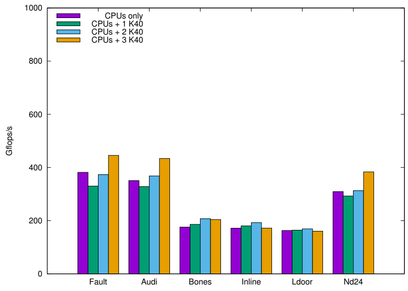
Matrices description from the Univ. of Florida collection

1. Fermi architecture
   - 2 hexacore Intel(R) Xeon(R) CPU X5650 @ 2.67
   - 3 Nvidia M2070
   - 32 GB of memory
2. Kepler architecture
   - 2 decacore Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30 - 3.00GHz
   - 3 Nvidia K40c (ECC=ON, no max boost)
   - 32 GB of memory

Performance on Fermi architecture

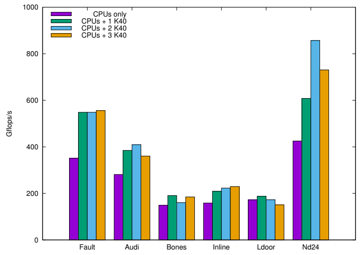


Performance on Kepler architecture



- Gives less improvement over the CPUs only version that on previous work with Fermi architecture

Performance of the hybrid 1D/2D DAG



- First GPU improves the CPUs performance
- Extra GPUs give less improvement (except for nd24k)
- Switch size is 240 (Might create some slow down on the CPUs-only version)

7

ILU(k) Factorization

Block ILU(k): supernode amalgamation algorithm

Derive a block incomplete LU factorization from the supernodal parallel direct solver

- Based on existing package PaStiX
- Level-3 BLAS incomplete factorization implementation
- Fill-in strategy based on level-fill among block structures identified thanks to the quotient graph
- **Amalgamation strategy to enlarge block size**

Highlights

- Handles efficiently high level-of-fill
- Solving time faster than with scalar ILU(k)
- Scalable parallel implementation

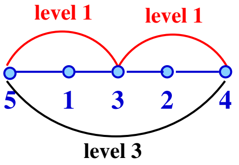
Fill-in theorem

Theorem

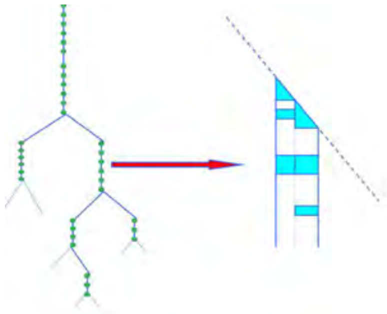
Any $A_{ij} = 0$ will become a non-null entry L_{ij} or $U_{ij} \neq 0$ in $A = iLU(k)$ if and only if it exists a shortest path of lengh $k + 1$ in $G_A(V, E)$ from vertex i to vertex j that only goes through vertices with a lower number than i and j.

Factorization ILU(k)

$$\begin{array}{c|ccccc|} & 1 & 2 & 3 & 4 & 5 \\ 1 & \times & & \times & & \times \\ 2 & & \times & \times & \times & \\ 3 & \times & \times & \times & \bullet & \bullet \\ 4 & & \times & \bullet & \times & \blacksquare \\ 5 & \times & & \bullet & \blacksquare & \times \end{array}$$

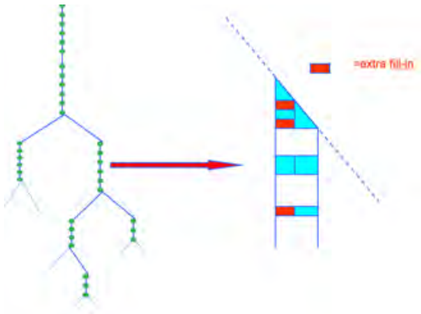


BILU(k): the amalgamation strategy

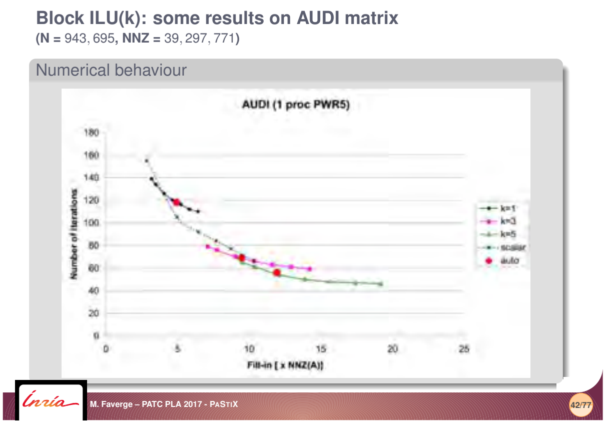


Find the exact supernode partition

BILU(k): the amalgamation strategy



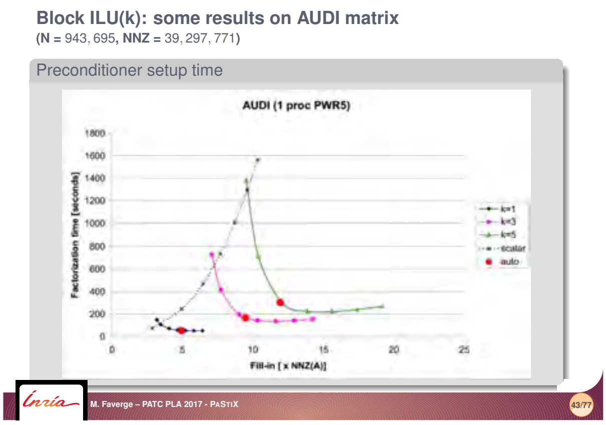
Merge the couple of supernodes that add the less extra fill-in



Block ILU(k): some results on AUDI matrix

(**N** = 943, 695, **NNZ** = 39, 297, 771)

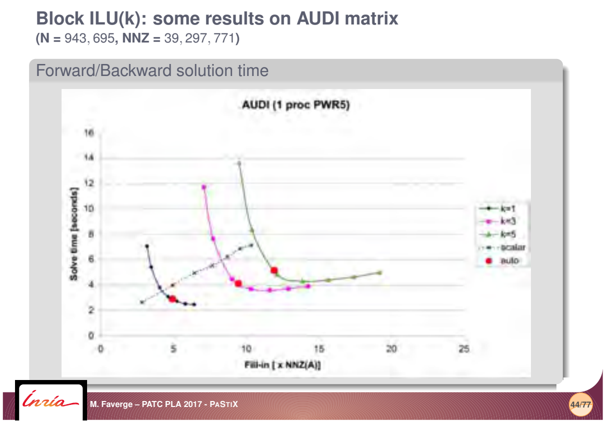
Numerical behaviour



Block ILU(k): some results on AUDI matrix

(**N** = 943, 695, **NNZ** = 39, 297, 771)

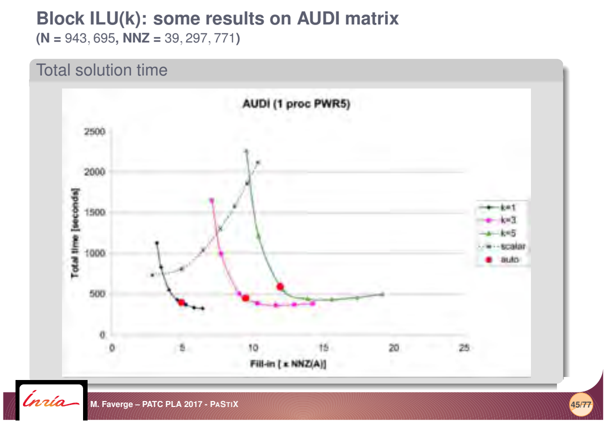
Preconditioner setup time



Block ILU(k): some results on AUDI matrix

(**N** = 943, 695, **NNZ** = 39, 297, 771)

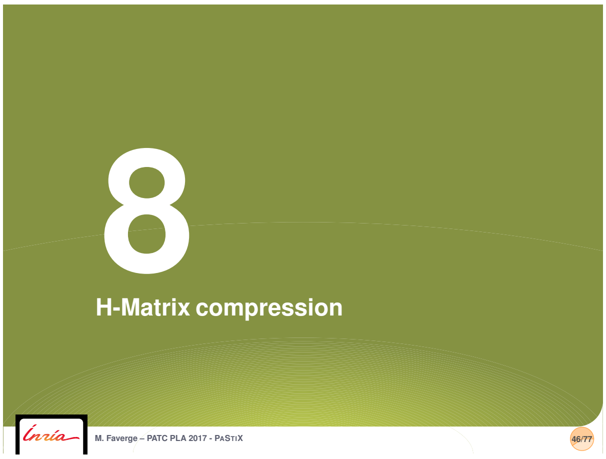
Forward/Backward solution time



Block ILU(k): some results on AUDI matrix

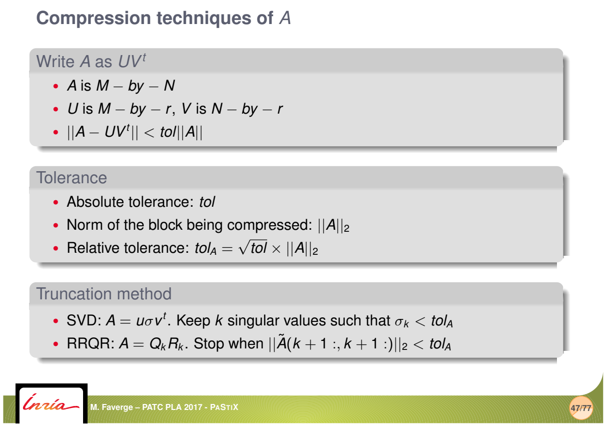
(**N** = 943, 695, **NNZ** = 39, 297, 771)

Total solution time



8

H-Matrix compression



Compression techniques of A

Write A as UV^t

- A is $M - by - N$
- U is $M - by - r$, V is $N - by - r$
- $||A - UV^t|| < tol||A||$

Tolerance

- Absolute tolerance: tol
- Norm of the block being compressed: $||A||_2$
- Relative tolerance: $tol_A = \sqrt{tol} \times ||A||_2$

Truncation method

- SVD: $A = u\sigma v^t$. Keep k singular values such that $\sigma_k < tol_A$
- RRQR: $A = Q_k R_k$. Stop when $||\tilde{A}(k+1:, k+1:)||_2 < tol_A$

Why using such techniques?

Dense matrix product updates

- $C+ = A * B$
- Time complexity: $\Theta(n^3)$, with N the block size
- Memory complexiity: $\Theta(n^2)$ memory complexity

Low rank matrix product updates

- $C+ = U_A * V_A^t * U_B * V_B^t$
- Time complexity: $\Theta(r^2 * n)$
- Memory complexity: $\Theta(r * n)$

Why using such techniques?

Current solver – for 3D problems

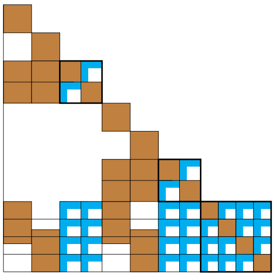
- $\Theta(n^2)$ time complexity
- $\Theta(n^{\frac{4}{3}})$ memory complexity
- BLAS 3 operations

Target solver – for 3D problems

- $\Theta(n^{\frac{4}{3}})$ time complexity
- $\Theta(n\ log(n))$ memory complexity
- BLAS 3 operations

Objective: build a black-box algebraic low-rank solver following the supernodal approach of PaStiX, with block-data structures.

Block-Low-Rank Compression – Symbolic Factorization



Large off-diagonal are low-rank, in the form uv^t.

Block-Low-Rank Algorithm

Approach

- Large supernodes are partitioned into a set of smaller supernodes
- Large off-diagonal blocks are represented as low-rank blocks

Operations

- Diagonal blocks are dense
- TRSM are performed on low-rank off-diagonal blocks
- GEMM are performed between low-rank off-diagonal blocks. It creates contributions to dense or low-rank blocks: this is the extend-add problem

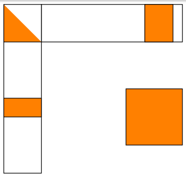
Compression techniques

- SVD, RRQR for now
- Easily extension to any algebraic method: ACA etc...
- Possible extension to randomized techniques etc...

Two different Scenarios: Scenario END

Scenario END: Compress L (similar to BLR-MUMPS: FCSU version)

1. Eliminate each column block
   - 1.1 Factorize the dense diagonal block
   - 1.2 Compress off-diagonal blocks belonging to the supernode
   - 1.3 Apply a TRSM on LR blocks (cheaper)
   - 1.4 LR update on dense matrices
2. Solve triangular systems with low-rank blocks



🟧Untouched

🟩READ

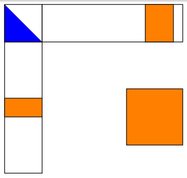
🟦READ/WRITE

🟥CREATED

Two different Scenarios: Scenario END

Scenario END: Compress L (similar to BLR-MUMPS: FCSU version)

1. Eliminate each column block
   - 1.1 Factorize the dense diagonal block
   - 1.2 Compress off-diagonal blocks belonging to the supernode
   - 1.3 Apply a TRSM on LR blocks (cheaper)
   - 1.4 LR update on dense matrices
2. Solve triangular systems with low-rank blocks



🟧Untouched

🟩READ

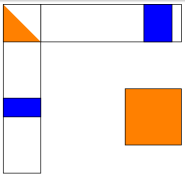
🟦READ/WRITE

🟥CREATED

Two different Scenarios: Scenario END

Scenario END: Compress *L* (similar to BLR-MUMPS: FCSU version)

1. Eliminate each column block
   - **1.1** Factorize the dense diagonal block
   - **1.2** Compress off-diagonal blocks belonging to the supernode
   - **1.3** Apply a TRSM on LR blocks (cheaper)
   - **1.4** LR update on dense matrices
2. Solve triangular systems with low-rank blocks



🟧 Untouched

🟩 READ

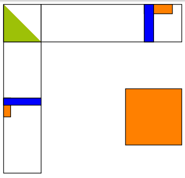
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario END

Scenario END: Compress *L* (similar to BLR-MUMPS: FCSU version)

1. Eliminate each column block
   - **1.1** Factorize the dense diagonal block
   - **1.2** Compress off-diagonal blocks belonging to the supernode
   - **1.3** Apply a TRSM on LR blocks (cheaper)
   - **1.4** LR update on dense matrices
2. Solve triangular systems with low-rank blocks



🟧 Untouched

🟩 READ

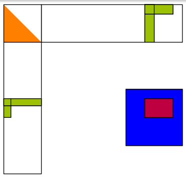
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario END

Scenario END: Compress *L* (similar to BLR-MUMPS: FCSU version)

1. Eliminate each column block
   - **1.1** Factorize the dense diagonal block
   - **1.2** Compress off-diagonal blocks belonging to the supernode
   - **1.3** Apply a TRSM on LR blocks (cheaper)
   - **1.4** LR update on dense matrices
2. Solve triangular systems with low-rank blocks



🟧 Untouched

🟩 READ

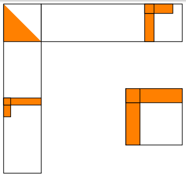
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario BEGIN

Scenario BEGIN: Compress *A*

1. Compress large off-diagonal blocks in *A* (exploiting sparsity)
2. Eliminate each column block
   - **2.1** Factorize the dense diagonal block
   - **2.2** Apply a TRSM on LR blocks (cheaper)
   - **2.3** LR update on LR matrices (extend-add)
3. Solve triangular systems with LR blocks



🟧 Untouched

🟩 READ

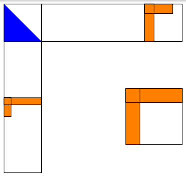
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario BEGIN

Scenario BEGIN: Compress *A*

1. Compress large off-diagonal blocks in *A* (exploiting sparsity)
2. Eliminate each column block
   - **2.1** Factorize the dense diagonal block
   - **2.2** Apply a TRSM on LR blocks (cheaper)
   - **2.3** LR update on LR matrices (extend-add)
3. Solve triangular systems with LR blocks



🟧 Untouched

🟩 READ

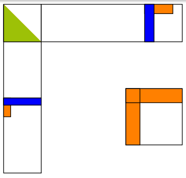
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario BEGIN

Scenario BEGIN: Compress *A*

1. Compress large off-diagonal blocks in *A* (exploiting sparsity)
2. Eliminate each column block
   - **2.1** Factorize the dense diagonal block
   - **2.2** Apply a TRSM on LR blocks (cheaper)
   - **2.3** LR update on LR matrices (extend-add)
3. Solve triangular systems with LR blocks



🟧 Untouched

🟩 READ

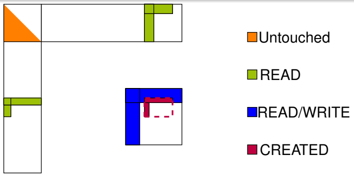
🟦 READ/WRITE

🟥 CREATED

Two different Scenarios: Scenario BEGIN

Scenario BEGIN: Compress A

1. Compress large off-diagonal blocks in A (exploiting sparsity)
2. Eliminate each column block
   - 2.1 Factorize the dense diagonal block
   - 2.2 Apply a TRSM on LR blocks (cheaper)
   - 2.3 LR update on LR matrices (extend-add)
3. Solve triangular systems with LR blocks



Extend-add: SVD Recompression

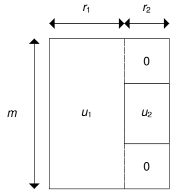
A low-rank structure $u_1 v_1^t$ receives a low-rank contribution $u_2 v_2^t$.

Algorithm

$$A = u_1 v_1^t + u_2 v_2^t = \left([u_1, u_2]\right) \times \left([v_1, v_2]\right)^t$$

- QR: $[u_1, u_2] = Q_1 R_1$ $\Theta(m(r_1 + r_2)^2)$
- QR: $[v_1, v_2] = Q_2 R_2$ $\Theta(n(r_1 + r_2)^2)$
- SVD: $R_1 R_2^t = u \sigma v^T$ $\Theta((r_1 + r_2)^3)$

$$A = \left(Q_1 u \sigma\right) \times \left(Q_2 v\right)^t$$



Extend-add: RRQR Recompression

A low-rank structure $u_1 v_1^t$ receives a low-rank contribution $u_2 v_2^t$.
u_1 and u_2 are orthogonal matrices

Algorithm

$$A = u_1 v_1^t + u_2 v_2^t = \left([u_1, u_2]\right) \times \left([v_1, v_2]\right)^t$$

Orthogonalize u_2 with respect to u_1 :

$$u_2^* = u_2 - u_1(u_1^t u_2) \quad \Theta(m r_1 r_2)$$

Form new orthogonal basis, and normalize each column :

$$[u_1, u_2] = [u_1, u_2^*] \times \begin{pmatrix} I & u_1^t u_2 \\ 0 & I \end{pmatrix}$$

Apply a RRQR on :

$$\begin{pmatrix} I & u_1^t u_2 \\ 0 & I \end{pmatrix} \times \left([v_1, v_2]\right)^t$$

RRQR with truncation in $\Theta(n(r_1 + r_2)r_1^*)$. Less stable?

Summary of both scenarios BEGIN and END

Memory consumption

- BEGIN scenario really saves memory
- END scenario reduces the size of L' factors, but supernodes are allocated dense at the beginning: no gain in pure *right-looking*

Update

- BEGIN scenario requires expensive extend-add algorithms to update (recompress) low-rank structures
- END scenario continues to apply dense update at a smaller cost

Potential optimizations

- BEGIN: Merge similar contributions together before applying a single recompression
- END: Use a *left-looking* algorithm to compress a block just before a supernode is eliminated. This approach may reduce the level of parallelism

Context

Machine: 2 INTEL Xeon E5-2680 v3 at 2.50 GHz

- 128 GB
- 24 threads

3D Matrices from The SuiteSparse Matrix Collection

- Atmosmodj: atmospheric model (1270432 dofs)
- Audi: structural problem (943695 dofs)
- Hook: model of a steel hook (1498023 dofs)
- Serena: gas reservoir simulation (1391349 dofs)
- Geo1438: geomechanical model of earth (1437960 dofs)
- + laplacian's generator: Poisson problem

Parallelism is obtained following PaStiX static scheduling strategy for multi-threaded architectures.

Parameters

Entry parameters

- Tolerance: absolute parameter (normalized for each block)
- Compression method: SVD or RRQR
- Compression scenario: BEGIN or END
- Blocking sizes: between 128 and 256 in following experiments

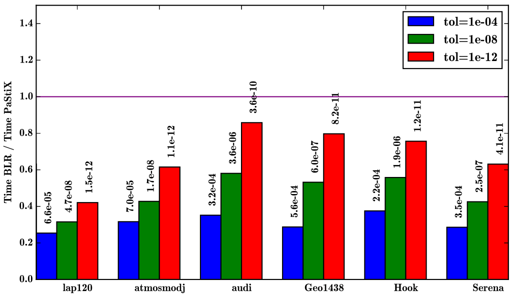
Scenario BEGIN

- Blocks are compressed at the beginning
- Each contribution implies a recompression

Scenario END

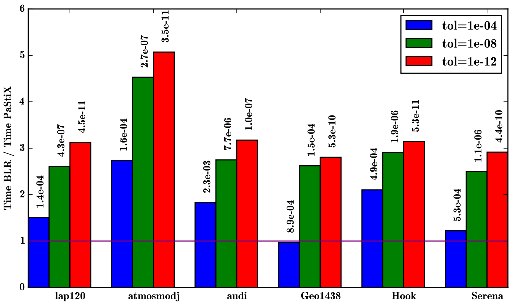
- Blocks are compressed just before a supernode is eliminated
- Those blocks are never uncompressed

Performance on general matrices: RRQR/END (24 CPUs)



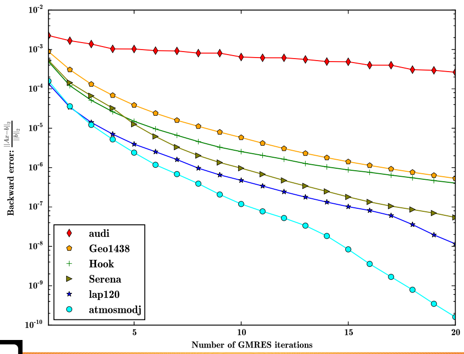
No iterative refinement.

Performance on general matrices: RRQR/BEGIN (24 CPUs)

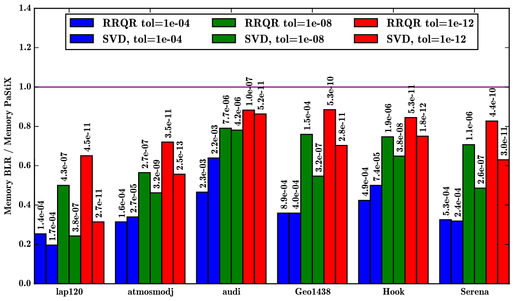


No iterative refinement.

Convergence of RRQR/BEGIN, tolerance=$1e-04$

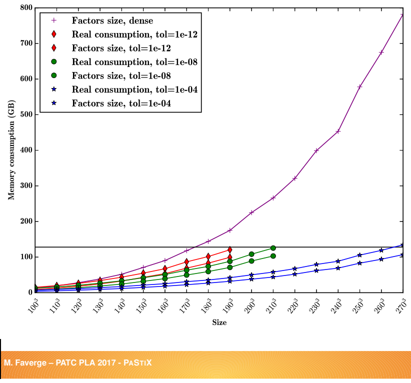


Memory on general matrices: BEGIN



No iterative refinement.

Scaling on laplacians: Memory Consumption RRQR/BEGIN

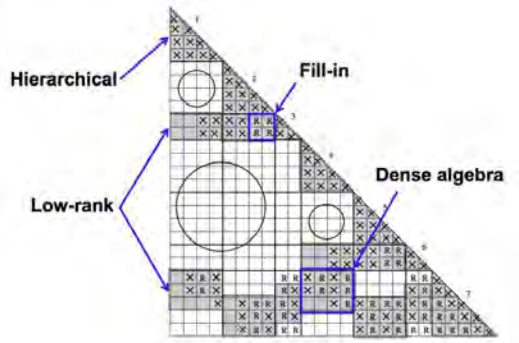


Toward low rank compressions in supernodal solver

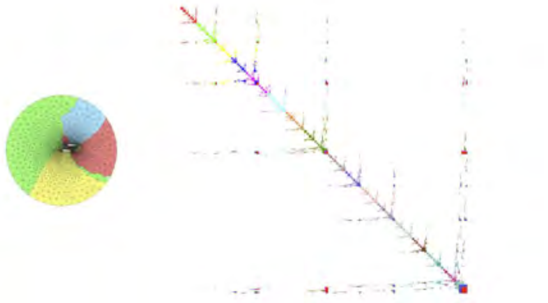
Many works on hierarchical matrices and direct solvers

- Eric Darve : Hierarchical matrices classifications (*Building O(N) Linear Solvers Using Nested Dissection*)
- Sherry Li : Multifrontal solver + HSS (*Towards an Optimal-Order Approximate Sparse Factorization Exploiting Data-Sparseness in Separators*)
- David Bindel : CHOLMOD + Low Rank (*An Efficient Solver for Sparse Linear Systems Based on Rank-Structured Cholesky Factorization*)
- Jean-Yves L'Excellent : MUMPS + Block Low Rank

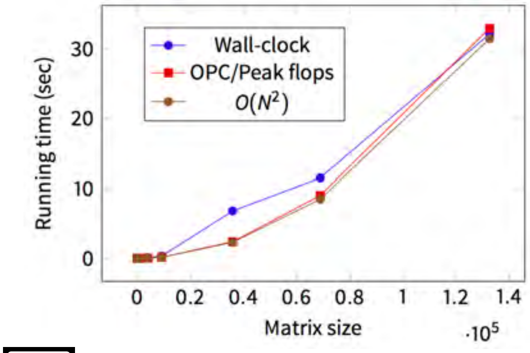
Symbolic factorization



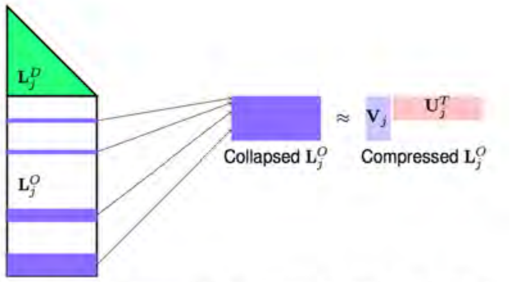
Nested dissection, 2D mesh/matrix



Computational cost ($O(N^2)$ for 3D PDE)



Low-Rank Compression of Supernodes



Collapsed off-diagonal block is a (nearly low-rank) dense matrix

FastLA associate team between INRIA/Berkeley/Stanford

Supernodal Solver - Hierarchical Matrices $O(N.log^a(N))$

1. Check the potential compression ratio on top level blocks
2. Develop a prototype with:
   - low-rank compression on the larger supernodes
   - compression tree built at each update
   - complexity analysis of the approach
3. Study coupling between nested dissection and compression tree ordering

Which algorithm to find low-rank approximation ?

SVD, RR-LU, RR-QR, ACA, CUR, Random ...

Which family of hierarchical matrix ?

$\mathcal{H}$, $\mathcal{H}^2$, HODLR ...

9

Summary and future works

Summary of the PaStiX 5.2.3 features

- LLt, LDLt, LU : supernodal implementation (BLAS3)
- Static pivoting + Refinement: CG/GMRES
- Simple/Double precision + Float/Complex operations
- Require MPI + Posix Thread (PETSc driver)

- MPI/Threads (Cluster/Multicore/SMP/NUMA)
- Centralized or Distributed interface
- **Dynamic scheduling NUMA (static mapping)**
- Support external ordering library (PT-Scotch/METIS)

- Multiple RHS (direct factorization)
- **Incomplete factorization with ILU(k) preconditionner**
- **Schur computation (hybrid method MaPHYS or HIPS)**
- Out-of Core implementation (shared memory only)

PaStiX 6.0 features

Factorization

	Real s/d	Complex c/z
POTRF	LL^t	LL^H
PXTRF	-	LL^t
HETRF	-	LDL^H
SYTRF	LDL^t	LDL^t
GETRF	LU	LU

Solve

- TRSM-like interface (LLT, LLN, LUN only for now)
- DIAG operation for LDL^t, LDL^H

Iterative refinement

- GMRES, BCG, CG

PaStiX 6.0

Which scheduler

	Seq.	P-Thread		Runtime	
		Static	Dyn	StarPU	PaRSEC
POTRF	SHM/LR	SHM/LR	-	-	SHM/LR (GPU)
PXTRF	Coming	Coming	-	-	Coming
HETRF	SHM	SHM	-	-	-
SYTRF	SHM	SHM	-	-	-
GETRF	SHM/LR	SHM/LR	-	-	SHM/LR (GPU)
TRSM	SHM/LR	SHM/LR	-	-	-
DIAG	SHM/LR	SHM/LR	-	-	-

Future development

- Integration of StarPU (SHM+GPU), similar coverage as PaRSEC
- Integration of MPI in all non runtime implementation

PaStiX 6.0

- Static pivoting + Refinement: CG/GMRES
- Multiple RHS (direct factorization)
- Schur computation (hybrid method MaPHYS or HIPS):
  - Improved parallelism
  - Solve operation on either interior and/or schur
- Support external ordering libraries:
  - Scotch, PT-Scotch, Metis, ParMetis (Under Dev.), Personal
- Open source git repository: https://gitlab.inria.fr/solverstack/pastix
- Open to external contributions

Softwares

Graph/Mesh partitioner and ordering :

`http://scotch.gforge.inria.fr`

Sparse linear system solvers :

`http://pastix.gforge.inria.fr`

`http://hips.gforge.inria.fr`
`https://wiki.bordeaux.inria.fr/maphys/doku.php`

Softwares

Fast Multipole Method :

`http://scalfmm-public.gforge.inria.fr/`

Matrices Over Runtime Systems (with University of Tenessee):

`http://icl.cs.utk.edu/projectsdev/morse`

Thanks !

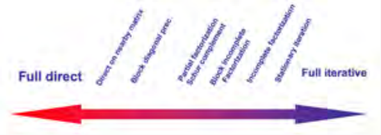
MaPHyS: a Massively Parallel Hybrid Solver

HiePACS team, Gilles Marait

PRACE 03/02/2017, Ostrava

Parallel sparse linear solver

Goal: solving $\mathcal{A}x = b$, where $\mathcal{A}$ is sparse, on distributed architectures



Usual trades off

Direct

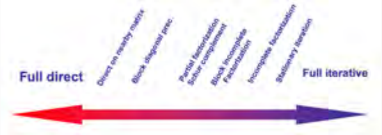
- Robust/accurate for general problems
- BLAS-3 based implementations
- Memory/CPU prohibitive for large $3D$ problems
- Limited weak scalability

Iterative

- Problem dependent efficiency / accuracy
- Sparse computational kernels
- Less memory requirements and possibly faster
- Possible high weak scalability

Parallel sparse linear solver

Goal: solving $\mathcal{A}x = b$, where $\mathcal{A}$ is sparse, on distributed architectures



Usual trades off

Direct

- Robust/accurate for general problems
- BLAS-3 based implementations
- Memory/CPU prohibitive for large $3D$ problems
- Limited weak scalability

Iterative

- Problem dependent efficiency / accuracy
- Sparse computational kernels
- Less memory requirements and possibly faster
- Possible high weak scalability

Outline

MaPHyS overview

Installation and current releases

Outline

MaPHyS overview

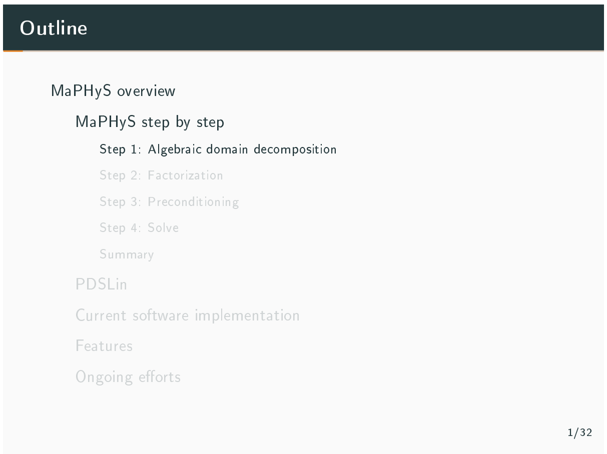
- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
- Ongoing efforts

Installation and current releases

Outline

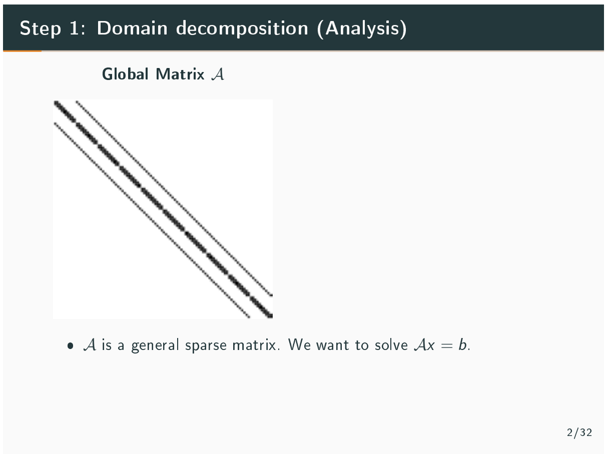
MaPHyS overview

- MaPHyS step by step
  - Step 1: Algebraic domain decomposition
  - Step 2: Factorization
  - Step 3: Preconditioning
  - Step 4: Solve
  - Summary
- PDSLin
- Current software implementation
- Features
- Ongoing efforts



Outline

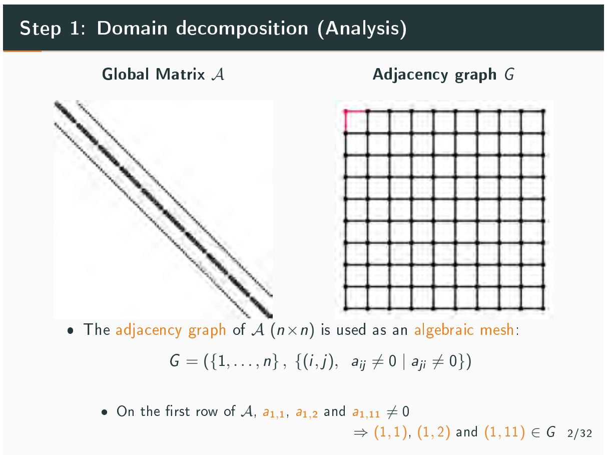
- MaPHyS overview
  - MaPHyS step by step
    - Step 1: Algebraic domain decomposition
    - Step 2: Factorization
    - Step 3: Preconditioning
    - Step 4: Solve
    - Summary
  - PDSLin
  - Current software implementation
  - Features
  - Ongoing efforts



Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$

- $\mathcal{A}$ is a general sparse matrix. We want to solve $\mathcal{A}x = b$.



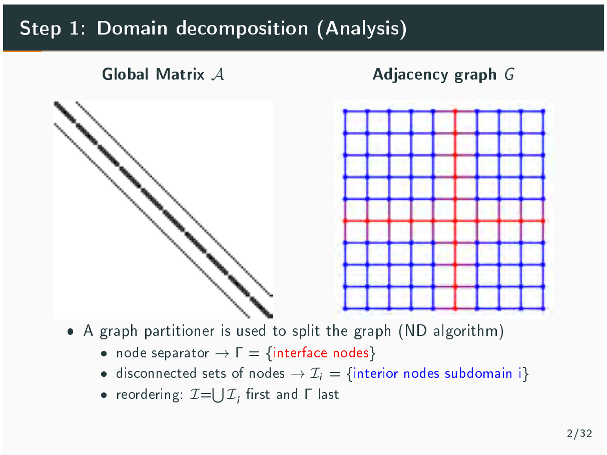
Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$ **Adjacency graph G**

- The adjacency graph of $\mathcal{A}$ $(n \times n)$ is used as an algebraic mesh:

$$G = (\{1,\ldots,n\}\,,\ \{(i,j),\quad a_{ij} \neq 0 \mid a_{ji} \neq 0\})$$

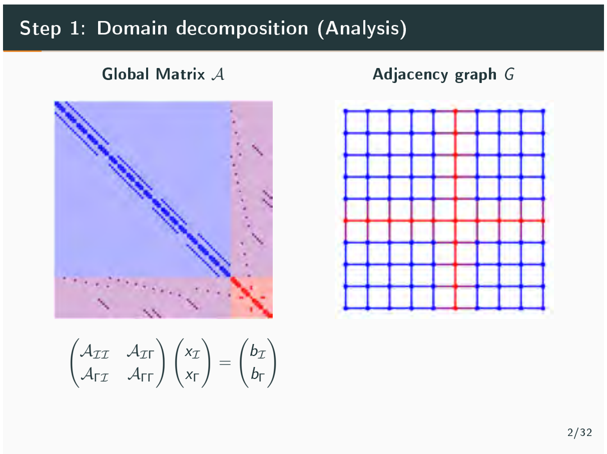
  - On the first row of $\mathcal{A}$, $a_{1,1}$, $a_{1,2}$ and $a_{1,11} \neq 0$ $\Rightarrow (1,1)$, $(1,2)$ and $(1,11) \in G$



Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$ **Adjacency graph G**

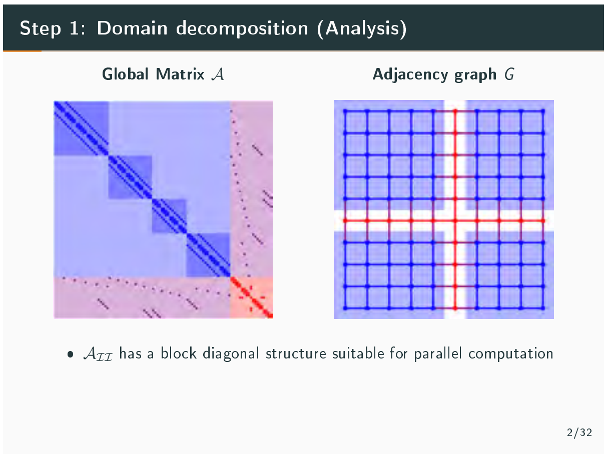
- A graph partitioner is used to split the graph (ND algorithm)
  - node separator → Γ = {interface nodes}
  - disconnected sets of nodes → $\mathcal{I}_i$ = {interior nodes subdomain i}
  - reordering: $\mathcal{I}{=}\bigcup \mathcal{I}_i$ first and Γ last



Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$ **Adjacency graph G**

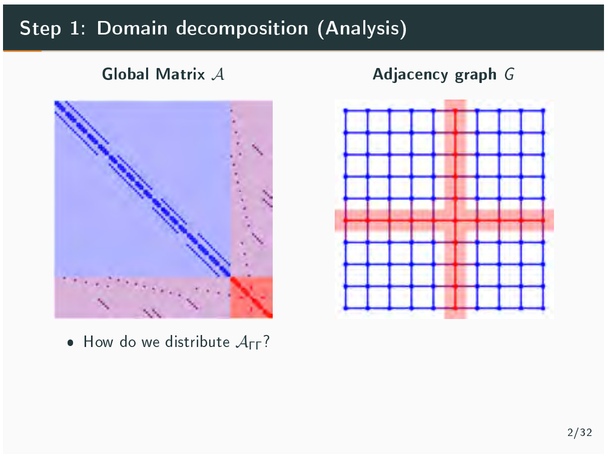
$$\begin{pmatrix} \mathcal{A}_{\mathcal{I}\mathcal{I}} & \mathcal{A}_{\mathcal{I}\Gamma} \\ \mathcal{A}_{\Gamma\mathcal{I}} & \mathcal{A}_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} x_{\mathcal{I}} \\ x_{\Gamma} \end{pmatrix} = \begin{pmatrix} b_{\mathcal{I}} \\ b_{\Gamma} \end{pmatrix}$$



Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$ **Adjacency graph G**

- $\mathcal{A}_{\mathcal{I}\mathcal{I}}$ has a block diagonal structure suitable for parallel computation

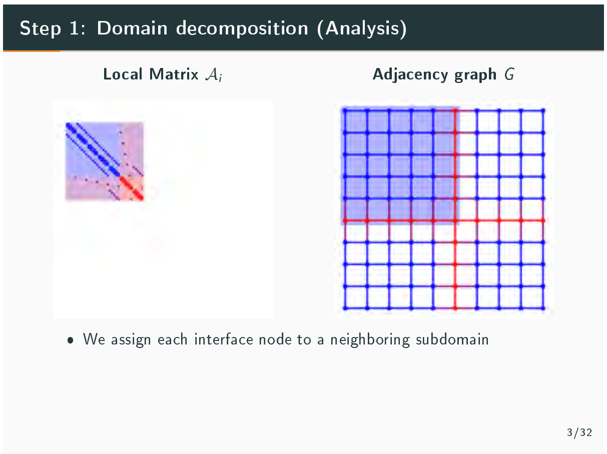


Step 1: Domain decomposition (Analysis)

Global Matrix $\mathcal{A}$

Adjacency graph G

- How do we distribute $\mathcal{A}_{\Gamma\Gamma}$?

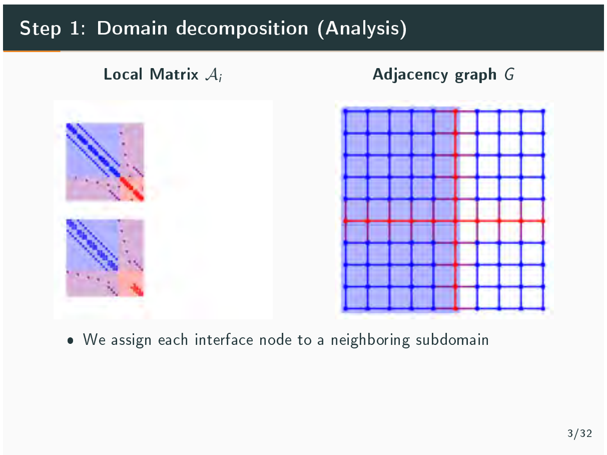


Step 1: Domain decomposition (Analysis)

Local Matrix $\mathcal{A}_i$

Adjacency graph G

- We assign each interface node to a neighboring subdomain

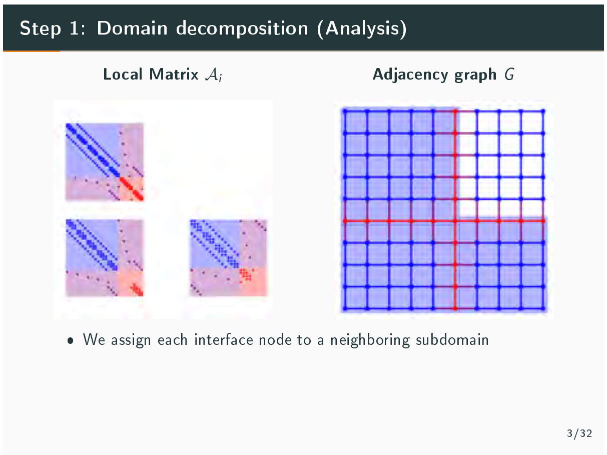


Step 1: Domain decomposition (Analysis)

Local Matrix $\mathcal{A}_i$

Adjacency graph G

- We assign each interface node to a neighboring subdomain

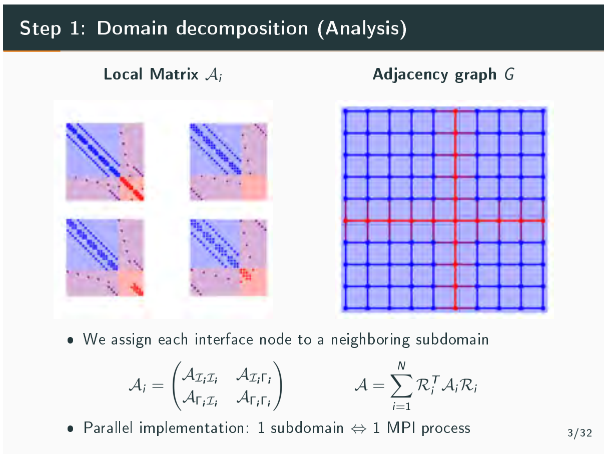


Step 1: Domain decomposition (Analysis)

Local Matrix $\mathcal{A}_i$

Adjacency graph G

- We assign each interface node to a neighboring subdomain



Step 1: Domain decomposition (Analysis)

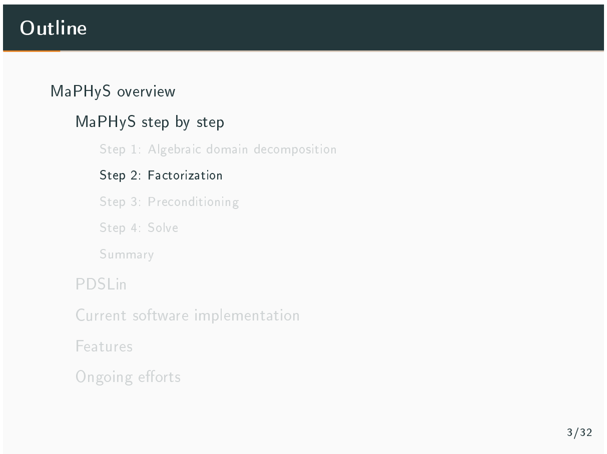
Local Matrix $\mathcal{A}_i$

Adjacency graph G

- We assign each interface node to a neighboring subdomain

$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix} \qquad \mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$$

- Parallel implementation: 1 subdomain ⇔ 1 MPI process



Outline

MaPHyS overview

MaPHyS step by step

Step 1: Algebraic domain decomposition

Step 2: Factorization

Step 3: Preconditioning

Step 4: Solve

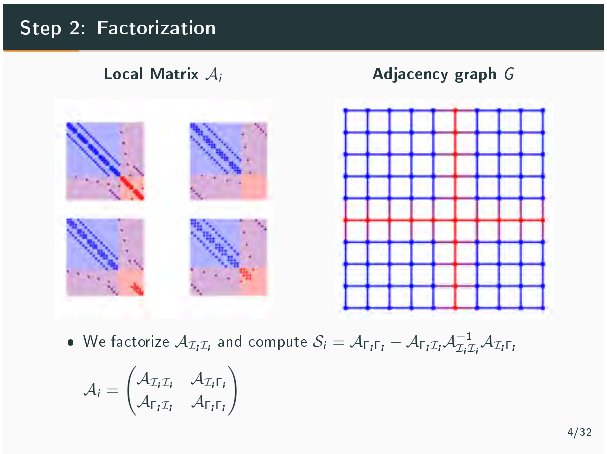
Summary

PDSLin

Current software implementation

Features

Ongoing efforts

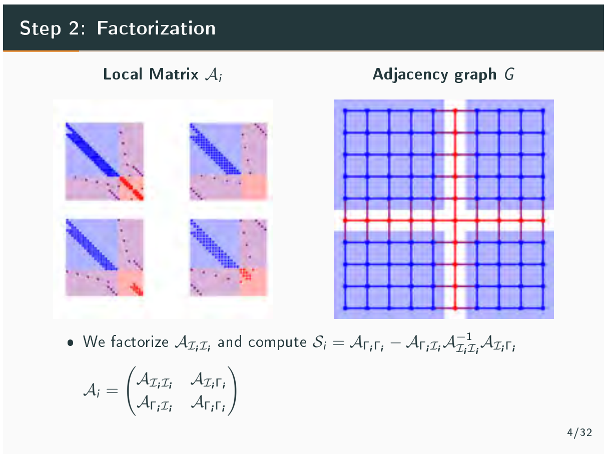


Step 2: Factorization

Local Matrix $\mathcal{A}_i$ **Adjacency graph G**

- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix}$$

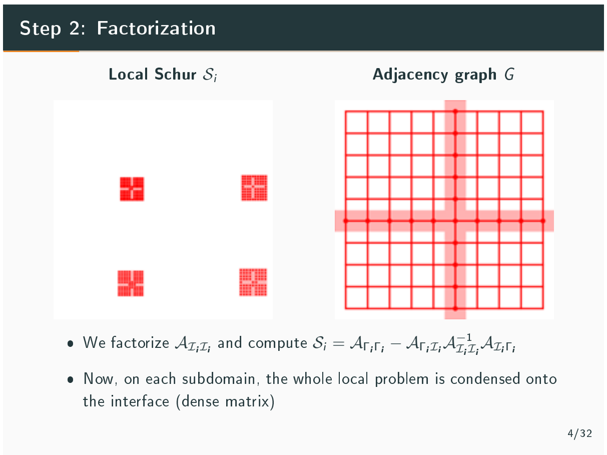


Step 2: Factorization

Local Matrix $\mathcal{A}_i$ **Adjacency graph G**

- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

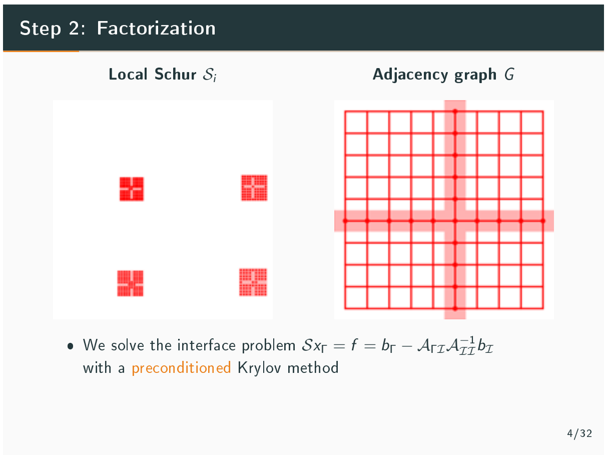
$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix}$$



Step 2: Factorization

Local Schur $\mathcal{S}_i$ **Adjacency graph G**

- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$
- Now, on each subdomain, the whole local problem is condensed onto the interface (dense matrix)



Step 2: Factorization

Local Schur $\mathcal{S}_i$ **Adjacency graph G**

- We solve the interface problem $\mathcal{S}x_\Gamma = f = b_\Gamma - \mathcal{A}_{\Gamma\mathcal{I}}\mathcal{A}_{\mathcal{I}\mathcal{I}}^{-1}b_\mathcal{I}$ with a preconditioned Krylov method

Outline

MaPHyS overview

MaPHyS step by step

Step 1: Algebraic domain decomposition

Step 2: Factorization

Step 3: Preconditioning

Step 4: Solve

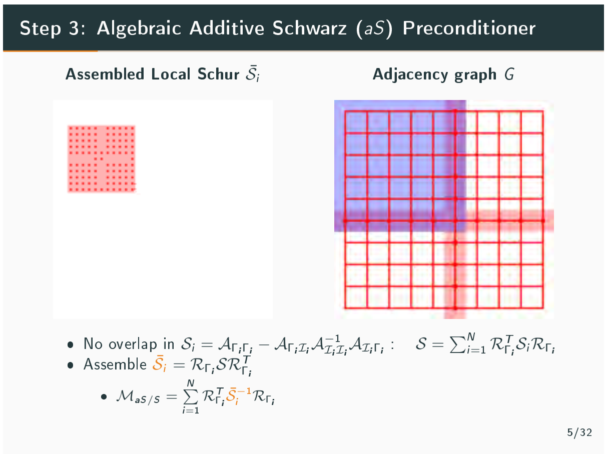
Summary

PDSLin

Current software implementation

Features

Ongoing efforts



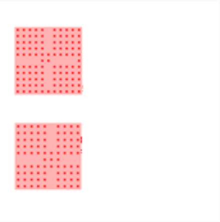
Step 3: Algebraic Additive Schwarz (aS) Preconditioner

Assembled Local Schur $\bar{\mathcal{S}}_i$ **Adjacency graph G**

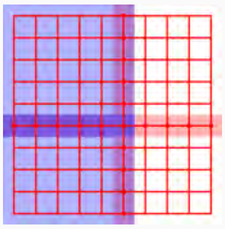
- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T\mathcal{S}_i\mathcal{R}_{\Gamma_i}$
- Assemble $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T$
  - $\mathcal{M}_{aS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T\bar{\mathcal{S}}_i^{-1}\mathcal{R}_{\Gamma_i}$

Step 3: Algebraic Additive Schwarz ($a\mathcal{S}$) Preconditioner

Assembled Local Schur $\bar{\mathcal{S}}_i$



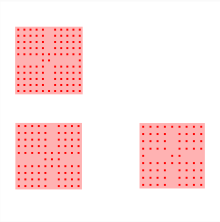
Adjacency graph G



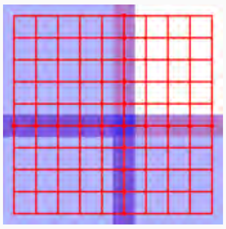
- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\quad \mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i \mathcal{R}_{\Gamma_i}$
- Assemble $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T$
  - $\mathcal{M}_{aS/S} = \sum\limits_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$

Step 3: Algebraic Additive Schwarz ($a\mathcal{S}$) Preconditioner

Assembled Local Schur $\bar{\mathcal{S}}_i$



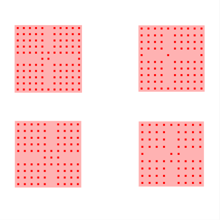
Adjacency graph G



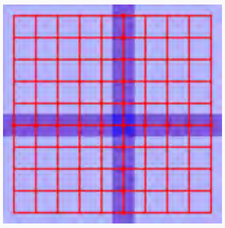
- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\quad \mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i \mathcal{R}_{\Gamma_i}$
- Assemble $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T$
  - $\mathcal{M}_{aS/S} = \sum\limits_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$

Step 3: Algebraic Additive Schwarz ($a\mathcal{S}$) Preconditioner

Assembled Local Schur $\bar{\mathcal{S}}_i$



Adjacency graph G



- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\quad \mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i \mathcal{R}_{\Gamma_i}$
- Assemble $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T$
  - $\mathcal{M}_{aS/S} = \sum\limits_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$

Assemble local schur computation

- Local Schur:

$$\mathcal{S}_i = \begin{pmatrix} \mathcal{S}_i^{kk} & \mathcal{S}_i^{kl} \\ \mathcal{S}_i^{lk} & \mathcal{S}_i^{ll} \end{pmatrix}$$

- Neighbor to neighbor communications:

$$\bar{\mathcal{S}}_i^{ll} = \sum_{k \in adj} \mathcal{S}_k^{ll}$$

- Assembled local Schur:

$$\bar{\mathcal{S}}_i = \begin{pmatrix} \bar{\mathcal{S}}_i^{kk} & \mathcal{S}_i^{kl} \\ \mathcal{S}_i^{lk} & \bar{\mathcal{S}}_i^{ll} \end{pmatrix}$$

- Algebraic Additive Schwarz Preconditionner:

$$\mathcal{M}_{aS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$$

Outline

MaPHyS overview

MaPHyS step by step

Step 1: Algebraic domain decomposition

Step 2: Factorization

Step 3: Preconditioning

Step 4: Solve

Summary

PDSLin

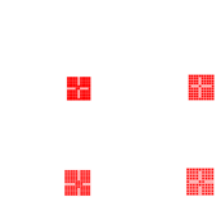
Current software implementation

Features

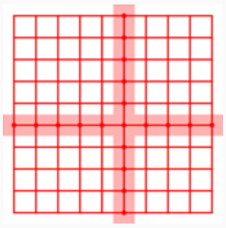
Ongoing efforts

Step 4: Solve

Local Schur $\mathcal{S}_i$



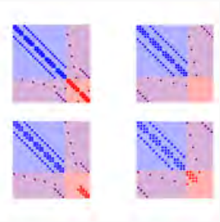
Adjacency graph G



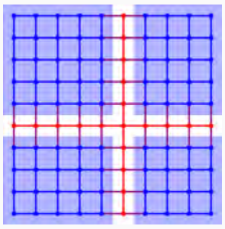
- on Γ: *Krylov method*
  - $\mathcal{S}\, x_\Gamma = f \quad$ preconditioned with $\mathcal{M}_{aS/S}$

Step 4: Solve

Local Matrix $\mathcal{A}_i$



Adjacency graph G



- on Γ: *Krylov method*
  - $\mathcal{S} x_\Gamma = f$ preconditioned with $\mathcal{M}_{as/s}$
- on $\mathcal{I}$: *Direct method*
  - $x_{\mathcal{I}_i} = \mathcal{A}^{-1}_{\mathcal{I}_i\mathcal{I}_i}(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i})$

Outline

MaPHyS overview

MaPHyS step by step

Step 1: Algebraic domain decomposition

Step 2: Factorization

Step 3: Preconditioning

Step 4: Solve

Summary

PDSLin

Current software implementation

Features

Ongoing efforts

Summary

Step 1: Algebraic domain decomposition

- $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$ with $\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix}$

Step 2: Factorization

- Computation of $\mathcal{A}^{-1}_{\mathcal{I}_i\mathcal{I}_i}$ and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}^{-1}_{\mathcal{I}_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioning

- Assembly and factorization of $\bar{\mathcal{S}}_i$

Step 4: Solve

- on Γ: *Krylov method*
  - $\mathcal{S} x_\Gamma = f$ preconditioned with $\mathcal{M}_{as/s} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$
- on $\mathcal{I}$: *Direct method*
  - $x_{\mathcal{I}_i} = \mathcal{A}^{-1}_{\mathcal{I}_i\mathcal{I}_i}(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i})$

Outline

MaPHyS overview

MaPHyS step by step

PDSLin

Presentation

Current software implementation

Features

Ongoing efforts

Outline

MaPHyS overview

MaPHyS step by step

PDSLin

Presentation

Current software implementation

Features

Ongoing efforts

Presentation

Parallel Domain decomposition Schur complement based Linear Solver

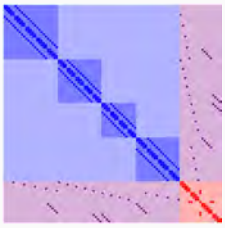
Hybrid solver developped in Berkley.

Developers:

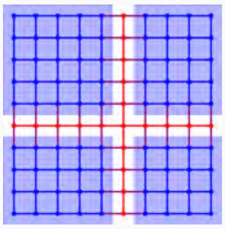
- Ichitaro Yamazaki
- X. Sherry Li

PSDLin domain decomposition

Local Matrix $\mathcal{A}_i$



Adjacency graph G



Analysis and domain decomposition like in MaPHyS:

$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix} \qquad \mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$$

Global algebraic view

- Global hybrid decomposition:

$$\mathcal{A} = \begin{pmatrix} \mathcal{A}_{\mathcal{I}\mathcal{I}} & \mathcal{A}_{\mathcal{I}\Gamma} \\ \mathcal{A}_{\Gamma\mathcal{I}} & \mathcal{A}_{\Gamma\Gamma} \end{pmatrix}$$

- Global Schur complement

$$\mathcal{S} = \mathcal{A}_{\Gamma\Gamma} - \mathcal{A}_{\Gamma\mathcal{I}}\mathcal{A}_{\mathcal{I}\mathcal{I}}^{-1}\mathcal{A}_{\mathcal{I}\Gamma}$$

Local algebraic view

- Local hybrid decomposition:

$$\mathcal{A}_i' = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & 0 \end{pmatrix}$$

- Local pre-Schur complement:

$$\mathcal{S}_i' = -\mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$$

- Global Schur complement:

$$\mathcal{S} = \mathcal{A}_{\Gamma\Gamma} - \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i' \mathcal{R}_{\Gamma_i}$$

- Global preconditioner: $M = S^{-1}$

Dropping

- Direct / direct method?

Dropping

- Direct / direct method?
- Three successive dropping techniques:
  - local dropping on $\mathcal{A}_{\mathcal{I}_i\Gamma_i}$
  - local dropping on $\mathcal{S}_i'$
  - global dropping on $\mathcal{S}$
  - $\mathcal{S} \rightarrow \tilde{\mathcal{S}}$

Dropping

- Direct / direct method?
- Three successive dropping techniques:
  - local dropping on $\mathcal{A}_{\mathcal{I}_i\Gamma_i}$
  - local dropping on $\mathcal{S}_i'$
  - global dropping on $\mathcal{S}$
  - $\mathcal{S} \rightarrow \tilde{\mathcal{S}}$
- $M = \tilde{\mathcal{S}}^{-1}$: direct/iterative method

Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
- Ongoing efforts

Current software implementation

Parallelism

- MPI: 1 subdomain ⇔ 1 MPI process

Partitioner

- Scotch [F. Pellegrini et al.]

Sparse direct solver

- MUMPS [P.R. Amestoy et al.]
- PaStiX [P. Ramet et al.]

Iterative Solvers

- CG/GMRES/FGMRES [V.Fraysse and L.Giraud]

Dense direct solver

- Your favorite BLAS and LAPACK implementations

Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
  - Multi-threading
  - Sparsification / approximation on the interface
  - MaPHyS interfaces
  - Coarse grid correction
- Ongoing efforts

Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
  - Multi-threading
  - Sparsification / approximation on the interface
  - MaPHyS interfaces
  - Coarse grid correction
- Ongoing efforts

Multi-threading: implementation

Parallelism

- MPI: 1 domain ⇔ 1 MPI process with multi-threading

Dense direct solver

- BLAS and LAPACK: multi-threaded MKL library

Sparse direct solver

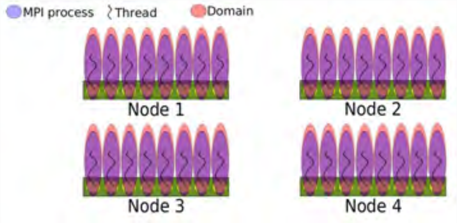
- MUMPS [P.R. Amestoy et al.] with multi-threaded MKL library
- PaStiX [P. Ramet et al.] with internal multi-threading

Iterative Solvers

- CG/GMRES/FGMRES [V.Fraysse and L.Giraud] with multi-threaded MKL library

Multi-threading: advantage

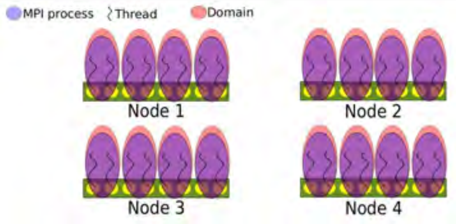
More flexibility to exploit multicore nodes



- 1 core per domain ⇒ 32 domains

Multi-threading: advantage

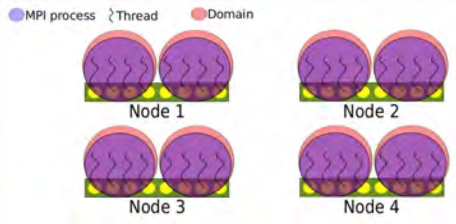
More flexibility to exploit multicore nodes



- 2 cores per domain ⇒ 16 domains

Multi-threading: advantage

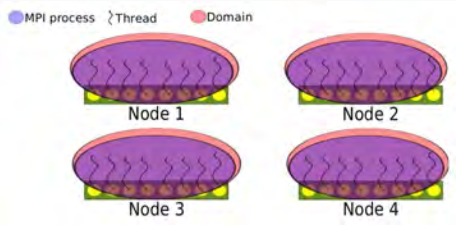
More flexibility to exploit multicore nodes



- 4 cores per domain ⇒ 8 domains

Multi-threading: advantage

More flexibility to exploit multicore nodes



- 8 cores per domain ⇒ 4 domains

Multi-threading: experiment

Hopper plateform

- Bi-socket nodes 12-core AMD MagnyCours @ 2.1GHz
- 32 GB DDR3 per node

Matrices

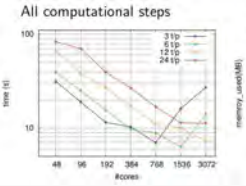
Matrix	Matrix211	Nachos4M
order	801K	4.147K
nnz	129.4M	256.4M

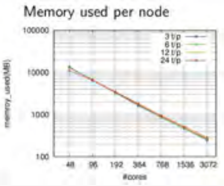
Multi-threading: experiment

Hopper plateform

- Bi-socket nodes 12-core AMD MagnyCours @ 2.1GHz
- 32 GB DDR3 per node

Matrix211



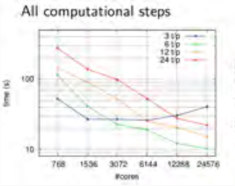


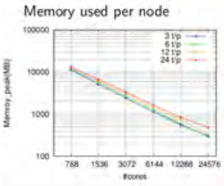
Multi-threading: experiment

Hopper plateform

- Bi-socket nodes 12-core AMD MagnyCours @ 2.1GHz
- 32 GB DDR3 per node

Nachos4M





Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
  - Multi-threading
  - Sparsification / approximation on the interface
  - MaPHyS interfaces
  - Coarse grid correction
- Ongoing efforts

Sparsification / approximation of the interface system

Local Schur complement $\mathcal{S}_i$ dense

1. preconditioner $\bar{\mathcal{S}}_i^{-1}$ construction expensive as a dense matrix
2. computation / memory requirements for $\mathcal{S}_i$ expensive

1. Sparsification of the local preconditioner

- dropping strategy to build $\tilde{\mathcal{S}}_i \approx \bar{\mathcal{S}}_i$ with $\tilde{\mathcal{S}}_i$ sparse

$$\tilde{s}_{lj} = \begin{cases} 0, & \text{if } |\bar{s}_{lj}| < \epsilon(|\bar{s}_{ll}| + |\bar{s}_{jj}|), \\ \bar{s}_{lj}, & \text{otherwise} \end{cases}$$

2. Approximation of the local Schur

- partial ILU(t,p) factorization of $\mathcal{A}_i$ to compute approximate Schur

Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
  - Multi-threading
  - Sparsification / approximation on the interface
  - MaPHyS interfaces
  - Coarse grid correction
- Ongoing efforts

Interfaces for MaPHyS

Application			↓		
Analysis			↓		
Factorization	↓	↓	↓	↓	↓
Preconditioner Setup	↓	↓	↓	↓	↓
Solve	↓	↓	↓	↓	↓

Centralized Matrix Input

- Application provides global matrix $\mathcal{A}$ on one MPI process
- MaPHyS performs algebraic domain decomposition and data distribution on MPI processes
- Convenient, but does not scale
  - problem size
  - number of processes

Interfaces for MaPHyS

Application	↓	↓	↓	↓	↓
Factorization	↓	↓	↓	↓	↓
Preconditioner Setup	↓	↓	↓	↓	↓
Solve	↓	↓	↓	↓	↓

Distributed subdomain interface

- Application performs domain decomposition and provides subdomain connectivity and local matrices $\mathcal{A}_i$ in a distributed way
- Algebraic domain decomposition is bypassed
- Naturally compliant with FEM, but also FV, HDG...
- Currently in use into TECSER ANR project, HPC4Energy project, ...

Outline

MaPHyS overview

- MaPHyS step by step
- PDSLin
- Current software implementation
- Features
  - Multi-threading
  - Sparsification / approximation on the interface
  - MaPHyS interfaces
  - Coarse grid correction
- Ongoing efforts

Motivation: Coarse Correction for MaPHyS

Need for Coarse Correction

- Good scalability of the direct part ☺
- The size and condition number of the iterative problem increases with the number of subdomains ☹

A proved robust coarse space for a larger class of methods

- Generalized Abstract Schwarz (GAS) methods
- Only works in the SPD case, with distributed input

Two implementations

- A python prototype, providing a framework for distributed GAS methods
- Integrated in MaPHyS 0.9.4

Example: 2D Test problem

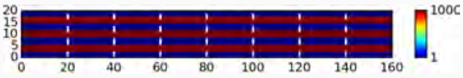
Heterogeneous diffusion

- $\nabla(K\nabla u) = q$
- 7 alternating conductivity layers
- Subdomain: 20×20 elements

Boundary conditions

- Dirichlet on the left
- Neumann elsewhere
- Source: $q = 1$

Conductivity K ($N = 8$ subdomains)



Example: 2D Test problem

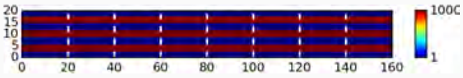
Heterogeneous diffusion

- $\nabla(K\nabla u) = q$
- 7 alternating conductivity layers
- Subdomain: 20×20 elements

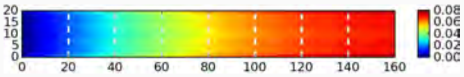
Boundary conditions

- Dirichlet on the left
- Neumann elsewhere
- Source: $q = 1$

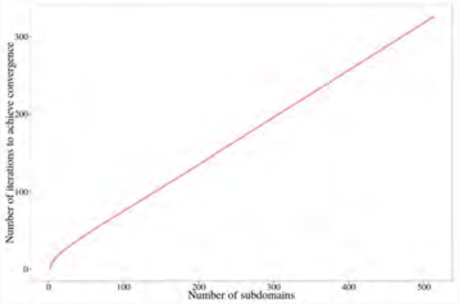
Conductivity K ($N = 8$ subdomains)



Solution x^* ($N = 8$ subdomains)



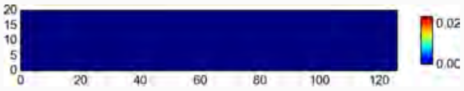
Weak Scalability



N	2	4	8	16	32	64	128	256	512
n_{iter}	1	7	13	21	33	54	92	169	325

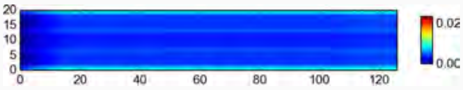
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 0$



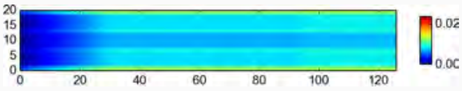
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 10$



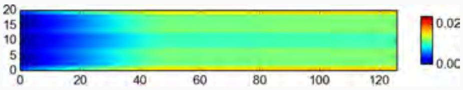
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 20$



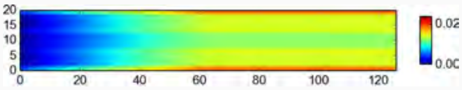
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 30$



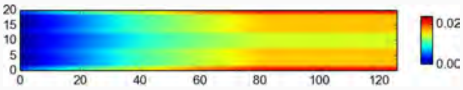
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 40$



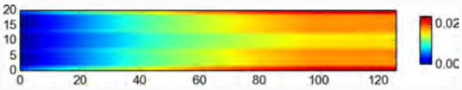
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 50$



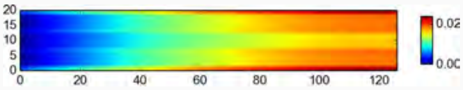
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 60$



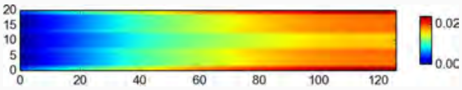
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$



Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$

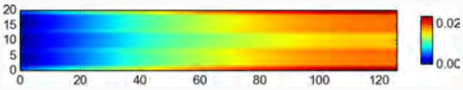


Problem

- No global exchange of information

Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$



Problem

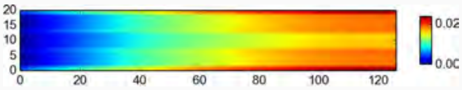
- No global exchange of information

Solution

- Use an exact direct solve on a coarse space V_0

Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$



Problem

- No global exchange of information

Solution

- Use an exact direct solve on a coarse space V_0

Contribution

- Coarse space for MaPHyS
  - but also for a wider class of methods
  - only in the SPD case

aS Step by step

Step 1: Domain Decomposition

- $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$

Step 2: Factorization

- Computation of $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}$ and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioner Setup

- $\mathcal{M}_{aS} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \left(\mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T\right)^{-1} \mathcal{R}_{\Gamma_i}$

Step 4: Solve

- on Γ: *Krylov method* $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{aS}$
- on $\mathcal{I}$: *Direct method* $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\left(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i}\right)$

aS, 2 Step by step

Step 1: Domain Decomposition (Application level)

- $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$

Step 2: Factorization

- Computation of $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}$ and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioner Setup

- $\mathcal{M}_{aS,2} = \mathcal{M}_0 + \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \left(\mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}^T\right)^{-1} \mathcal{R}_{\Gamma_i}$

Step 4: Solve

- on Γ: *Krylov method* $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{aS,2}$
- on $\mathcal{I}$: *Direct method* $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\left(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i}\right)$

3D Test problem

Heterogeneous diffusion

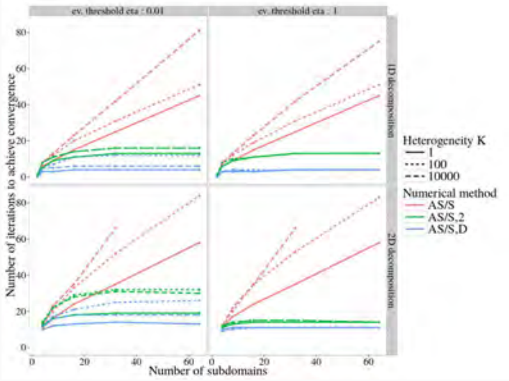
- $\nabla(K\nabla u) = 1$
- Alternating conductivity layers of 3 elements (ratio K between layers)
- Dirichlet on the left, Neumann elsewhere

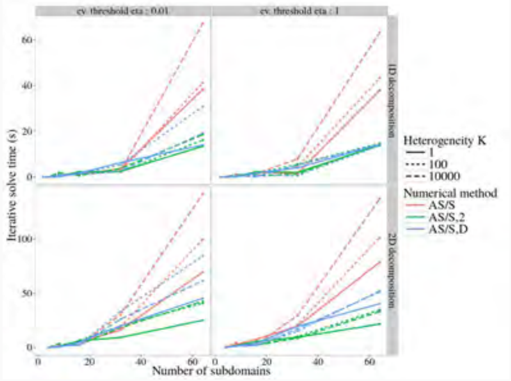
Domain decomposition

- $N \times 1 \times 1$ (1D decomposition)
- $N/2 \times 2 \times 1$ (2D decomposition)
- Constant subdomain size: $10 \times 10 \times 10$ elements

Implementation

- python/MPI





Outline

MaPHyS overview

MaPHyS step by step

PDSLin

Current software implementation

Features

Ongoing efforts

Ongoing efforts

1. Data-sparse preconditioner: consider H-matrix for local Schur complement representation (FASTLA with Stanford) - Funding Inria Région Aquitaine - Y. Harness
2. Multiple RHS: Inexact Breakdown Block GMRES with Deflated Restart standalone library IB-BGMRES-DR - Funding from DGA-RAPID/HI-BOX - C. Piacibello
3. Parallel algebraic domain decomposition for MaPHyS: consider input in any distributed format and perform parallel domain decomposition standalone library Paddle - Funding from ANR TECSER - M. Kuhn

Outline

MaPHyS overview

Installation and current releases

MaPHyS

Installing MaPHyS

- MaPHyS and its dependencies can be installed through `spack` in $\leq$ 15 minutes + coffee break

  `morse.gforge.inria.fr/spack/spack.html`

- From a laptop to an heterogeneous supercomputer

Bash

```
# Install and load spack
git clone https://github.com/fpruvost/spack.git
cd spack
. ./share/spack/setup-env.sh
# Install maphys
spack install maphys
```

MaPHyS versions

MaPHyS 0.9.3

- current version, multi-threading, distributed subdomain interface

MaPHyS 0.9.4

- integration of the Coarse Grid Correction
- already available through spack:

Bash

```
spack install maphys@0.9.4
```

MaPHyS 1.0 (work in progress)

- Redefinition of MaPHyS interface for current and future needs
- Integration of Paddle (Analysis step)
- Integration of IB-BGMRES-DR (multi-RHS iterative solver)

Krylov subspace methods from the historical, analytic, application, and high performance computing perspective

Zdeněk Strakoš
Charles University and Czech Academy of Sciences, Prague

PRACE course, Ostrava, February 2017

with thanks to, in particular

Erin Carson,
Tomáš Gergelits,
Jörg Liesen,
Josef Málek,
Jan Papež,
Miroslav Rozložník,
Petr Tichý,
Miroslav Tůma.

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods

  Matrices and operators in infinite dimensional Hilbert spaces.

  - Euler
  - Gauss
  - Jacobi
  - Chebyshev, Markov
  - Stieltjes
  - Hilbert, von Neumann
  - Krylov, Gantmakher
  - Lanczos, Hestenes, Stiefel

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods

Cornelius Lanczos, *Why Mathematics,* 1966

“ In a recent comment on mathematical preparation an educator wanted to characterize our backwardness by the following statement: ”Is it not astonishing that a person graduating in mathematics today knows hardly more than what Euler knew already at the end of the eighteenth century?”. On its face value this sounds a convincing argument. Yet it misses the point completely. Personally I would not hesitate not only to graduate with first class honors, but to give the Ph.D. (and with summa cum laude) without asking any further questions, to anybody who knew only one quarter of what Euler knew, provided that he knew it in the way in which Euler knew it. ”

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods
- *historical:* Without understanding the history we are confused in the presence and we will get lost in the future. This holds also for mathematics.

Cornelius Lanczos, *Linear Differential Operators,* 1961

“To get an explicit solution of a given boundary value problem is in this age of large electronic computers no longer a basic question. The problem can be coded for the machine and the numerical answer obtained. But of what value is the numerical answer if the scientist does not understand the peculiar analytical properties and idiosyncrasies of the given operator?”

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods
- *historical:* Without understanding the history we are confused in the presence and we will get lost in the future. This holds also for mathematics.
- *analytic:* The progress in computing technology and the need for solving practical problems forces us to think algorithmically and do things fast. Analytic view is by its nature slow and it does not keep up with the pace. But analytic view is absolutely crucial. It makes a little sense to progress fast in a wrong direction.

Cornelius Lanczos, *The Inspired Guess in the History of Physics,* 1964,

“Once the great mathematician Gauss was engaged in a particularly important investigation, but seemed to make little headway. His colleagues inquired when the publication was to appear. Gauss gave them an apparently paradoxical and yet perfectly correct answer: ‘I have all the results but I don’t know yet how I am going to get them’.”

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods
- *historical:* Without understanding the history we are confused in the presence and we will get lost in the future. This holds also for mathematics.
- *analytic:* The progress in computing technology and the need for solving practical problems forces us to think algorithmically and do things fast. Analytic view is by its nature slow and it does not keep up with the pace. But analytic view is absolutely crucial. It makes a little sense to progress fast in a wrong direction.
- *application:* Development and application of mathematics lives in an unbreakable symbiosis. I do not believe in “pure” against “applied” mathematics. This division is artificial, caused by proudness and ambitions. As a malign disease it leads mathematics to fragmentation and the fields of mathematics to dangerous isolation. Applications are like a fresh water. Any application must, however, honor the assumptions of the theory.

Meaning of the words - application

Henri Poincaré, 1909, graduate of the Polytechnique

“The scientist does not study nature because it is useful; he studies it because he delights in it, and he delights in it because it is beautiful. If nature were not beautiful, it would not be worth knowing, and if nature were not worth knowing, life would not be worth living. ...
I mean that deeper beauty coming from the harmonious order of the parts, and that a pure intelligence can grasp.

Science has had marvelous applications, but a science that would only have applications in mind would not be science anymore, it would be only cookery.”

Meaning of the words

- *Matrix iterative methods* ⟶ Krylov subspace methods
- *historical:* Without understanding the history we are confused in the presence and we will get lost in the future. This holds also for mathematics.
- *analytic:* The progress in computing technology and the need for solving practical problems forces us to think algorithmically and do things fast. Analytic view is by its nature slow and it does not keep up with the pace. But analytic view is absolutely crucial. It makes a little sense to progress fast in a wrong direction.
- *application:* Development and application of mathematics lives in an unbreakable symbiosis. I do not believe in “pure” against “applied” mathematics. This division is artificial, caused by proudness and ambitions. As a malign disease it leads mathematics to fragmentation and the fields of mathematics to dangerous isolation. Applications are like a fresh water. Any application must honor the assumptions of the theory.
- *computational:* Computing is a very involved process. Computers should serve in solving properly mathematically formulated problems. Mathematics must respect limitations of the computing technology.

Meaning of the words - computational

John von Neumann and Herman H. Goldstine, *Numerical ...* , 1947

“When a problem in pure or in applied mathematics is ‘solved’ by numerical computation, errors, that is, deviations of the numerical ‘solution’ obtained from the true, rigorous one, are unavoidable. Such a ‘solution’ is therefore meaningless, unless there is an estimate of the total error in the above sense.

Such estimates have to be obtained by a combination of several different methods, because the errors that are involved are aggregates of several different kinds of contributory, primary errors. These primary errors are so different from each other in their origin and character, that the methods by which they have to be estimated must differ widely from each other. A discussion of the subject may, therefore, advantageously begin with an analysis of the main kinds of primary errors, or rather of the sources from which they spring.

This analysis of the sources of errors should be objective and strict inasmuch as completeness is concerned, ...”

Cornelius Lanczos, March 9, 1947

On (what are now called) the Lanczos and CG methods:

“The reason why I am strongly drawn to such
approximation mathematics problems is ... the fact that
a very “economical” solution is possible only when it is very “adequate”.

To obtain a solution in very few steps
means nearly always that one has found a way
that does justice to the inner nature of the problem.”

Albert Einstein, March 18, 1947

“Your remark on the importance of
adapted approximation methods makes very
good sense to me, and I am convinced
that this is a fruitful mathematical aspect,
and not just a utilitarian one.”

Albert Einstein, March 18, 1947

"Your remark on the importance of *<u>adapted</u> approximation methods makes very good sense to me, and I am convinced that this is a fruitful mathematical aspect, and not just a utilitarian one."*

Main principle behind Krylov subspace methods:

Highly nonlinear adaptation of the iterations to the problem.

Conjugate Gradient (CG) method for $Ax = b$ with A HPD (1952)

$r_0 = b - Ax_0,\ p_0 = r_0\,.$ For $n = 1, \dots, n_{\max}$:

$$\begin{aligned} \alpha_{n-1} &= \frac{r_{n-1}^* r_{n-1}}{p_{n-1}^* A p_{n-1}} \\ x_n &= x_{n-1} + \alpha_{n-1} p_{n-1}\,, \quad \text{stop when the stopping criterion is satisfied} \\ r_n &= r_{n-1} - \alpha_{n-1} A p_{n-1} \\ \beta_n &= \frac{r_n^* r_n}{r_{n-1}^* r_{n-1}} \\ p_n &= r_n + \beta_n p_{n-1} \end{aligned}$$

Here α_{n-1} ensures the minimization of $\|x - x_n\|_A$ along the line

$$z(\alpha) = x_{n-1} + \alpha p_{n-1}\,.$$

Mathematical elegance of CG

- Provided that

$$p_i \perp_A p_j, \quad i \neq j,$$

the one-dimensional line minimizations at the individual steps 1 to n result in the n-dimensional minimization over the whole shifted Krylov subspace

$$x_0 + \mathcal{K}_n(A, r_0) \;=\; x_0 + \text{span}\{p_0, p_1, \dots, p_{n-1}\}\,.$$

Mathematical elegance of CG

- Provided that

$$p_i \perp_A p_j, \quad i \neq j,$$

the one-dimensional line minimizations at the individual steps 1 to n result in the n-dimensional minimization over the whole shifted Krylov subspace

$$x_0 + \mathcal{K}_n(A, r_0) \;=\; x_0 + \text{span}\{p_0, p_1, \dots, p_{n-1}\}\,.$$

- The orthogonality condition leads to short recurrences due to the relationship to the orthogonal polynomials that define the algebraic residuals and search vectors.

Mathematical elegance of CG

- Provided that

$$p_i \perp_A p_j, \quad i \neq j,$$

the one-dimensional line minimizations at the individual steps 1 to n result in the n-dimensional minimization over the whole shifted Krylov subspace

$$x_0 + \mathcal{K}_n(A, r_0) \;=\; x_0 + \text{span}\{p_0, p_1, \dots, p_{n-1}\}\,.$$

- The orthogonality condition leads to short recurrences due to the relationship to the orthogonal polynomials that define the algebraic residuals and search vectors.
- Inexact computation?

Antoine de Saint-Exupéry, *The Wisdom of the Sands*, 1944

"I would not bid you pore upon a heap of stones, and turn them over and over, in the vain hope of learning from them the secret of meditation. For on the level of the stones there is no question of meditation; for that, the temple must have come into being. But, once it is built, a new emotion sways my heart, and when I go away, I ponder on the relations between the stones. ...

I must begin by feeling love; and I must first observe a wholeness. After that I may proceed to study the components and their groupings. But I shall not trouble to investigate these raw materials unless they are dominated by something on which my heart is set. Thus I began by observing the triangle as a whole; then I sought to learn in it the functions of its component lines. ...

So, to begin with, I practise contemplation. After that, if I am able, I analyse and explain. ...

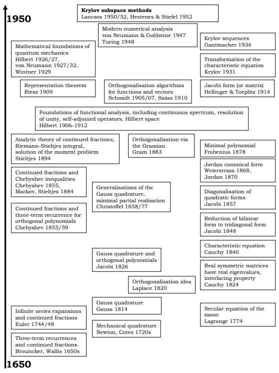
Little matter the actual things that are linked together; it is the links that I must begin by apprehending and interpreting."

Outline

1. What are the Krylov subspace methods and what kind of mathematics is involved?
2. Linear projections onto highly nonlinear Krylov subspaces
3. Model reduction and moment matching
4. Convergence and spectral information
5. Inexact computations and numerical stability
6. Functional analysis and infinite dimensional considerations
7. Operator preconditioning, discretization and algebraic computation
8. HPC computations with Krylov subspace methods?
9. Myths about Krylov subspace methods

1. What are the Krylov subspace methods and what kind of mathematics is involved?

1 Historical development and context



Krylov subspace methods
Lanczos 1950/52, Hestenes & Stiefel 1952

Modern numerical analysis
von Neumann & Goldstine 1947
Turing 1948

Krylov sequences
Gantmacher 1934

Mathematical foundations of quantum mechanics
Hilbert 1926/27,
von Neumann 1927/32,
Wintner 1929

Transformation of the characteristic equation
Krylov 1931

Representation theorem
Riesz 1909

Orthogonalisation algorithms for functions and vectors
Schmidt 1905/07, Szász 1910

Jacobi form (or matrix)
Hellinger & Toeplitz 1914

Foundations of functional analysis, including continuous spectrum, resolution of unity, self-adjoined operators, Hilbert space
Hilbert 1906-1912

Analytic theory of continued fractions, Riemann-Stieltjes integral, solution of the moment problem
Stieltjes 1894

Orthogonalisation via the Gramian
Gram 1883

Minimal polynomial
Frobenius 1878

Jordan canonical form
Weierstrass 1868,
Jordan 1870

Continued fractions and Chebyshev inequalities
Chebyshev 1855,
Markov, Stieltjes 1884

Generalisations of the Gauss quadrature, minimal partial realisation
Christoffel 1858/77

Diagonalisation of quadratic forms
Jacobi 1857

Continued fractions and three-term recurrence for orthogonal polynomials
Chebyshev 1855/59

Reduction of bilinear form to tridiagonal form
Jacobi 1848

Characteristic equation
Cauchy 1840

Gauss quadrature and orthogonal polynomials
Jacobi 1826

Real symmetric matrices have real eigenvalues, interlacing property
Cauchy 1824

Orthogonalisation idea
Laplace 1820

Gauss quadrature
Gauss 1814

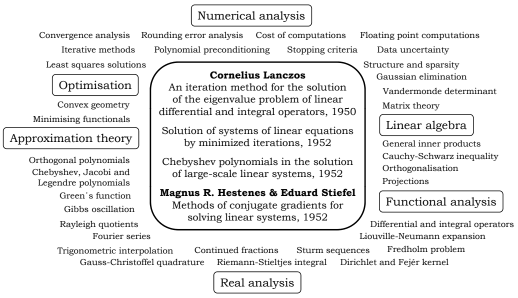
Secular equation of the moon
Lagrange 1774

Infinite series expansions and continued fractions
Euler 1744/48

Mechanical quadrature
Newton, Cotes 1720s

Three-term recurrences and continued fractions
Brouncker, Wallis 1650s

1 Lanczos, Hestenes and Stiefel



Numerical analysis

Convergence analysis · Rounding error analysis · Cost of computations · Floating point computations

Iterative methods · Polynomial preconditioning · Stopping criteria · Data uncertainty

Least squares solutions

Structure and sparsity

Optimisation

Convex geometry

Minimising functionals

Approximation theory

Orthogonal polynomials

Chebyshev, Jacobi and Legendre polynomials

Green's function

Gibbs oscillation

Rayleigh quotients

Fourier series

Cornelius Lanczos
An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, 1950

Solution of systems of linear equations by minimized iterations, 1952

Chebyshev polynomials in the solution of large-scale linear systems, 1952

Magnus R. Hestenes & Eduard Stiefel
Methods of conjugate gradients for solving linear systems, 1952

Gaussian elimination

Vandermonde determinant

Matrix theory

Linear algebra

General inner products

Cauchy-Schwarz inequality

Orthogonalisation

Projections

Functional analysis

Differential and integral operators

Liouville-Neumann expansion

Trigonometric interpolation · Continued fractions · Sturm sequences · Fredholm problem

Gauss-Christoffel quadrature · Riemann-Stieltjes integral · Dirichlet and Fejér kernel

Real analysis

1 Homework problem

Consider $2n$ real numbers $m_0, m_1, \ldots, m_{2n-1}$.
Solve the $2n$ equations

$$\sum_{j=1}^{n} \omega_j^{(n)} \, \{\theta_j^{(n)}\}^{\ell} \;=\; m_\ell \, , \qquad \ell = 0, 1, \ldots, 2n-1 \, ,$$

for the $2n$ real unknowns $\omega_j^{(n)} > 0, \; \theta_j^{(n)}$.

1 Homework problem

Consider $2n$ real numbers $m_0, m_1, \ldots, m_{2n-1}$.
Solve the $2n$ equations

$$\sum_{j=1}^{n} \omega_j^{(n)} \, \{\theta_j^{(n)}\}^{\ell} \;=\; m_\ell \, , \qquad \ell = 0, 1, \ldots, 2n-1 \, ,$$

for the $2n$ real unknowns $\omega_j^{(n)} > 0, \; \theta_j^{(n)}$.

Is this problem linear?
Does it look easy?
When does it have a solution?
How the solution can be determined?
How the solution can be computed?

1 Positive definite linear functionals on polynomials

Linear functional $\mathcal{L}(x)$ is positive definite on the space of polynomials $\mathcal{P}_n$ of degree at most n if its first $2n+1$ moments

$$\mathcal{L}(x^\ell) = m_\ell, \quad \ell = 0, 1, \ldots, 2n$$

are real and the Hankel matrix M_n of moments is positive definite, i.e., $\Delta_n > 0$, where

$$\Delta_n = |M_n| = \begin{vmatrix} m_0 & m_1 & \cdots & m_n \\ m_1 & m_2 & \cdots & m_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ m_n & m_{n+1} & \cdots & m_{2n} \end{vmatrix}.$$

1 Stieltjes moment problem (1894) of order n

With the positive definite $\mathcal{L}(x)$ we can restrict ourselves to real polynomials of a real variable and write, using a non-decreasing positive distribution function μ defined on the real axis having finite limits at $\pm\infty$,

$$\mathcal{L}(f) = \int f(x)\,\mathrm{d}\mu(x)\,,$$

with the inner product

$$(p,q) := \mathcal{L}(p(x)q(x)) = \int p(x)q(x)\,\mathrm{d}\mu(x)\,.$$

1 Stieltjes moment problem (1894) of order n

With the positive definite $\mathcal{L}(x)$ we can restrict ourselves to real polynomials of a real variable and write, using a non-decreasing positive distribution function μ defined on the real axis having finite limits at $\pm\infty$,

$$\mathcal{L}(f) = \int f(x)\,\mathrm{d}\mu(x)\,,$$

with the inner product

$$(p,q) := \mathcal{L}(p(x)q(x)) = \int p(x)q(x)\,\mathrm{d}\mu(x)\,.$$

Solution of the Stieltjes moment problem of order n exists and it is unique if and only if (with some $m_{2n} > 0$) we have $\Delta_n > 0$.

1 The unknown $\omega_j^{(n)}$, $\theta_j^{(n)}$?

- Cholesky decomposition of the matrix of moments $M_n = L_n L_n^T$
- The entries of the ℓth row of the the inverse L_n^{-1} give the coefficients of the ℓth orthonormal polynomial determined by the positive definite linear functional $\mathcal{L}(x)$ associated with the matrix of moments M_n .
- Roots of the ℓth orthogonal polynomial give the quadrature nodes $\theta_j^{(\ell)}$. The weights $\omega_j^{(\ell)}$ are given by the formula for the interpolatory quadrature.
- Computations are done differently

  (Gragg and Harrod, Gautschi, Laurie, ...)

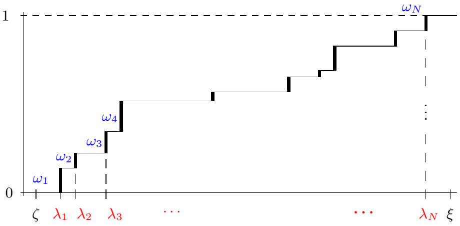
  O'Leary, S, Tichý, *On Sensitivity of Gauss-Christoffel quadrature,* Numerische Mathematik, 107, 2007, pp. 147 –174

  Pranic, Pozza, S, *Gauss quadrature for quasi-definite linear functionals,* IMA J. Numer. Anal, 2016 (to appear)

1 All we talk about is the conjugate gradient or the Lanczos method!

Distribution function $\omega(\lambda)$ associated with $Ax = b,\ r_0 = b - Ax_0,\ A$ SPD:

$$\lambda_i, s_i \text{ are the eigenpairs of } A\,,\ \omega_i = |(s_i, w_1)|^2$$



1 Spectral decomposition $A = \sum_{\ell=1}^N \lambda_\ell\, s_\ell s_\ell^*$

Symbolically

$$w_1^* A w_1 = w_1^* \left(\sum_{\ell=1}^N \lambda_\ell\, s_\ell s_\ell^* \right) w_1 \equiv w_1^* \left(\int_a^b \lambda\, dE(\lambda) \right) w_1$$

$$= \sum_{\ell=1}^N \lambda_\ell\, w_1^* s_\ell\, s_\ell^* w_1 = \sum_{\ell=1}^N \lambda_\ell\, \omega_\ell = \int_a^b \lambda\, d\omega(\lambda)\,,$$

where $dE(\lambda_\ell) \equiv s_\ell s_\ell^*$ and

$$I = \sum_{\ell=1}^N s_\ell s_\ell^* \equiv \int_a^b dE(\lambda)\,.$$

Hilbert (1906, 1912, 1928), Von Neumann (1927, 1932), Wintner (1929) .

2. Linear projections onto highly nonlinear Krylov subspaces

References:

J. Liesen. and Z.S., *Krylov Subspace Methods, Principles and Analysis.* Oxford University Press (2013), Chapter 2

2 Krylov sequences and (cyclic) Krylov subspaces

- The Krylov sequence generated by $A \in \mathbb{C}^{N\times N}$ and $v \in \mathbb{C}^N$

$$v, Av, A^2v, \ldots$$

- The nth Krylov subspace generated by $A \in \mathbb{C}^{N\times N}$ and $v \in \mathbb{C}^N$

$$\mathcal{K}_n(A,v) := \text{span}\{v, Av, \ldots, A^{n-1}v\}, \quad n = 1, 2, \ldots$$

- By construction,

$$\mathcal{K}_1(A,v) \subset \mathcal{K}_2(A,v) \subset \cdots \subset \mathcal{K}_d(A,v) = \mathcal{K}_{d+k}(A,v) \quad \text{for all } k \geq 1.$$

$$\begin{tikzcd}
v \arrow[r, bend left, "A"] & Av \arrow[r, bend left, "A"] & \cdots \arrow[r, bend left, "A"] & A^{d-2}v \arrow[r, bend left, "A"] & A^{d-1}v
\end{tikzcd}$$

$$\begin{tikzcd}
\mathcal{K}_1(A,v) \arrow[r, bend right, "A"'] & \mathcal{K}_2(A,v) \arrow[r, bend right, "A"'] & \cdots \arrow[r, bend right, "A"'] & \mathcal{K}_{d-1}(A,v) \arrow[r, bend right, "A"'] & \mathcal{K}_d(A,v) \arrow[loop below, "A"]
\end{tikzcd}$$

2 Krylov subspace methods

- Krylov subspace methods are based on a sequence of projections onto the nested Krylov subspaces that form the search spaces.
- Linear algebraic system $Ax = b$: x_0 (possibly zero), $r_0 = b - Ax_0$.

$$x_n \in x_0 + \mathcal{S}_n = x_0 + \mathcal{K}_n(A, r_0) \text{ such that } r_n = b - Ax_n \perp \mathcal{C}_n, \quad n = 1, 2, \ldots$$

- n-dimensional constraints space $\mathcal{C}_n$ determines the different methods.
- Eigenvalue problem $Ax = \lambda x$: v (nonzero), find (λ_n, x_n) such that

$$x_n \in \mathcal{K}_n(A,v) \quad \text{and} \quad r_n = Ax_n - \lambda_n x_n \perp \mathcal{C}_n.$$

- Examples: The Lanczos and Arnoldi methods, where $\mathcal{C}_n = \mathcal{K}_n(A,v)$.

2 Krylov subspace methods

- Krylov subspace methods are based on a sequence of projections onto the nested Krylov subspaces that form the search spaces.
- Linear algebraic system $Ax = b$: x_0 (possibly zero), $r_0 = b - Ax_0$.

$$x_n \in x_0 + \mathcal{S}_n = x_0 + \mathcal{K}_n(A, r_0) \text{ such that } r_n = b - Ax_n \perp \mathcal{C}_n, \quad n = 1, 2, \ldots$$

- n-dimensional constraints space $\mathcal{C}_n$ determines the different methods.
- Eigenvalue problem $Ax = \lambda x$: v (nonzero), find (λ_n, x_n) such that

$$x_n \in \mathcal{K}_n(A,v) \quad \text{and} \quad r_n = Ax_n - \lambda_n x_n \perp \mathcal{C}_n.$$

- Examples: The Lanczos and Arnoldi methods, where $\mathcal{C}_n = \mathcal{K}_n(A,v)$.

2 Examples of Krylov subspace methods for $Ax = b$

- Method is well defined when x_n is uniquely determined for $n = 1, 2, \ldots, d-1$, and $x_d = x$ (in exact arithmetic).
- Conjugate gradient (CG) method: $\mathcal{S}_n = \mathcal{C}_n = \mathcal{K}_n(A, r_0)$.
  - Well defined for HPD matrices A; short recurrences.
  - Orthogonality $r_n \perp \mathcal{K}_n(A,v)$ is equivalent to optimality:

$$\|x - x_n\|_A = \min_{z \in x_0 + \mathcal{K}_n(A,r_0)} \|x - z\|_A.$$

- GMRES method: $\mathcal{S}_n = \mathcal{K}_n(A, r_0)$, $\mathcal{C}_n = A\mathcal{K}_n(A, r_0)$.
  - Well defined for nonsingular matrices A; full recurrences.
  - Orthogonality $r_n \perp A\mathcal{K}_n(A,v)$ is equivalent to optimality:

$$\|b - Ax_n\|_2 = \min_{z \in x_0 + \mathcal{K}_n(A,r_0)} \|b - Az\|_2.$$

- Numerous other Krylov subspace methods. Some of them are not well defined in the above sense (e.g. BiCGStab or QMR).

2 Examples of Krylov subspace methods for $Ax = b$

- Method is well defined when x_n is uniquely determined for $n = 1, 2, \ldots, d-1$, and $x_d = x$ (in exact arithmetic).
- Conjugate gradient (CG) method: $\mathcal{S}_n = \mathcal{C}_n = \mathcal{K}_n(A, r_0)$.
  - Well defined for HPD matrices A; short recurrences.
  - Orthogonality $r_n \perp \mathcal{K}_n(A,v)$ is equivalent to optimality:

$$\|x - x_n\|_A = \min_{z \in x_0 + \mathcal{K}_n(A,r_0)} \|x - z\|_A.$$

- GMRES method: $\mathcal{S}_n = \mathcal{K}_n(A, r_0)$, $\mathcal{C}_n = A\mathcal{K}_n(A, r_0)$.
  - Well defined for nonsingular matrices A; full recurrences.
  - Orthogonality $r_n \perp A\mathcal{K}_n(A,v)$ is equivalent to optimality:

$$\|b - Ax_n\|_2 = \min_{z \in x_0 + \mathcal{K}_n(A,r_0)} \|b - Az\|_2.$$

- Numerous other Krylov subspace methods. Some of them are not well defined in the above sense (e.g. BiCGStab or QMR).

2 Examples of Krylov subspace methods for $Ax = b$

- Method is well defined when x_n is uniquely determined for $n = 1, 2, \dots, d-1$, and $x_d = x$ (in exact arithmetic).
- Conjugate gradient (CG) method: $\mathcal{S}_n = \mathcal{C}_n = \mathcal{K}_n(A, r_0)$.
  - Well defined for HPD matrices A; short recurrences.
  - Orthogonality $r_n \perp \mathcal{K}_n(A, v)$ is equivalent to optimality:

$$\|x - x_n\|_A = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|x - z\|_A.$$

- GMRES method: $\mathcal{S}_n = \mathcal{K}_n(A, r_0)$, $\mathcal{C}_n = A\mathcal{K}_n(A, r_0)$.
  - Well defined for nonsingular matrices A; full recurrences.
  - Orthogonality $r_n \perp A\mathcal{K}_n(A, v)$ is equivalent to optimality:

$$\|b - Ax_n\|_2 = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|b - Az\|_2.$$

- Numerous other Krylov subspace methods. Some of them are not well defined in the above sense (e.g. BiCGStab or QMR).

2 Conjugate gradients (CG), orthogonal projections and optimality

$$\|x - x_n\|_A \;=\; \min_{u \,\in\, x_0 + \mathcal{K}_n(A, r_0)} \|x - u\|_A$$

with the formulation via the Lanczos process, $w_1 = r_0/\|r_0\|$,

$$A\,W_n \;=\; W_n\,\mathbf{T_n} \;+\; \delta_{n+1} w_{n+1} \mathbf{e}_n^T\,, \quad \mathbf{T}_n \;=\; W_n^*(A, r_0)\,A\,W_n(A, r_0)\,,$$

and the CG approximation given by

$$\mathbf{T}_n\,\mathbf{y}_n \;=\; \|r_0\|\mathbf{e}_1\,, \quad x_n \;=\; x_0 \;+\; W_n\,\mathbf{y}_n\,.$$

$$A_n \;=\; Q_n\,A\,Q_n \;=\; W_n W_n^*\,A\,W_n W_n^* \;=\; W_n\,\mathbf{T}_n\,W_n^*\,,$$

Clearly, the projection process is very highly nonlinear in both A and r_0.

2 (Petrov-) Galerkin framework

Projection idea in Krylov subspace methods is analogous to the Galerkin framework in numerical solution of PDEs (here for convenience we take $\mathcal{C} = \mathcal{S}$).

Let $\mathcal{S}$ be an infinite dimensional Hilbert space, $a(\cdot, \cdot) : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ be a bounded and coercive bilinear form, $f : \mathcal{S} \to \mathbb{R}$ be a bounded linear functional.

- Weak formulation: Find $u \in \mathcal{S}$ with

$$a(u, v) = f(v) \quad \text{for all} \quad v \in \mathcal{S}.$$

- Discretization: Find $u_h \in \mathcal{S}_h \subset \mathcal{S}$ with

$$a(u_h, v_h) = f(v_h) \quad \text{for all} \quad v_h \in \mathcal{S}_h.$$

- Galerkin orthogonality:

$$a(u - u_h, v_h) = 0 \quad \text{for all} \quad v_h \in \mathcal{S}_h.$$

2 (Petrov-) Galerkin framework

Projection idea in Krylov subspace methods is analogous to the Galerkin framework in numerical solution of PDEs (here for convenience we take $\mathcal{C} = \mathcal{S}$).

Let $\mathcal{S}$ be an infinite dimensional Hilbert space, $a(\cdot, \cdot) : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ be a bounded and coercive bilinear form, $f : \mathcal{S} \to \mathbb{R}$ be a bounded linear functional.

- Weak formulation: Find $u \in \mathcal{S}$ with

$$a(u, v) = f(v) \quad \text{for all} \quad v \in \mathcal{S}.$$

- Discretization: Find $u_h \in \mathcal{S}_h \subset \mathcal{S}$ with

$$a(u_h, v_h) = f(v_h) \quad \text{for all} \quad v_h \in \mathcal{S}_h.$$

- Galerkin orthogonality:

$$a(u - u_h, v_h) = 0 \quad \text{for all} \quad v_h \in \mathcal{S}_h.$$

2 Operator problem formulation - motivation for part 7

- Equivalently, there exists a bounded and coercive operator $\mathcal{A} : \mathcal{S} \to \mathcal{S}^\#$, with the problem formulated as the following equation in the dual space:

$$\mathcal{A}u = f.$$

- Or, using the Riesz map $\tau : \mathcal{S}^\# \to \mathcal{S}$ defined by the inner product in $\mathcal{S}$, as the following operator preconditioned equation in the function space

$$\tau\mathcal{A}u = \tau f.$$

- Discretization then gives

$$\tau_h\mathcal{A}_h u_h - \tau_h f_h \;\perp\; \mathcal{S}_h.$$

Krylov subspace methods (here CG for $\mathcal{A}$ self-adjoint with respect to the duality pairing) can be formulated in infinite dimensional Hilbert spaces and extended to Banach spaces.

2 CG in infinite dimensional Hilbert spaces - motivation for part 7

$r_0 = f - \mathcal{A}u_0 \in \mathcal{S}^\#$, $p_0 = \tau r_0 \in \mathcal{S}$. For $n = 1, 2, \dots, n_{\max}$:

$$\begin{aligned} \alpha_{n-1} &= \frac{\langle r_{n-1}, \tau r_{n-1}\rangle}{\langle \mathcal{A}p_{n-1}, p_{n-1}\rangle} \\ u_n &= u_{n-1} + \alpha_{n-1}p_{n-1}\,, \quad \text{stop when the stopping criterion is satisfied} \\ r_n &= r_{n-1} - \alpha_{n-1}\mathcal{A}p_{n-1} \\ \beta_n &= \frac{\langle r_n, \tau r_n\rangle}{\langle r_{n-1}, \tau r_{n-1}\rangle} \\ p_n &= \tau r_n + \beta_n p_{n-1} \end{aligned}$$

Superlinear convergence for (identity + compact) operators.

Karush (1952), Hayes (1954), Vorobyev (1958)

Here the Riesz map τ indeed serves as a preconditioner.

2 Summary and motivation

- Krylov subspace methods for solving linear algebraic problems are based on linear projections onto nested subspaces.
- Krylov subspaces and therefore the resulting methods are highly nonlinear in the data defining the problem.
- The nonlinearity allows to adapt to the problem as the iteration proceeds. This is not apparent, e.g., from the derivation of CG based on the minimization of the quadratic functional, and this fact has affected negatively the presentation of Krylov subspace methods in textbooks.
- The adaptation can be better understood via the model reduction and moment matching properties of Krylov subspace methods.

3. Model reduction and moment matching

References:

J. Liesen. and Z.S., *Krylov Subspace Methods, Principles and Analysis.* Oxford University Press (2013), Chapter 3

3 Jacobi matrix and the conjugate gradient method

$$T_n = \begin{pmatrix} \gamma_1 & \delta_2 & & & \\ \delta_2 & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \delta_n \\ & & & \delta_n & \gamma_n \end{pmatrix}$$

is the Jacobi matrix of the orthogonalization coefficients and the CG method is formulated by

$$T_n t_n = \|r_0\|\, e_1 , \qquad x_n = x_0 + V_n t_n \,.$$

3 The projected system, A HPD, CG method

- Let the columns of $V_n = [v_1, \dots, v_n]$ form an orthonormal basis of $\mathcal{K}_n(A, r_0)$.
- Matrix formulation of $x_n \in x_0 + \mathcal{K}_n(A, r_0)$ and $r_n \perp \mathcal{K}_n(A, r_0)$:

$$x_n = x_0 + V_n t_n$$

and $\quad t_n \in \mathbb{C}^n \quad$ is found by solving

$$V_n^* A V_n \, t_n = \|r_0\| e_1 .$$

- This can be viewed as a model reduction from a (large) system of order N to a (small) system of order n.
- Intuition: Projected system should capture fast a sufficient part of information contained in the original data.
- Intuition: Powering the operator tends to transfer dominant information as quickly as possible into the projected system.

3 The projected system, A HPD, CG method

- Let the columns of $V_n = [v_1, \dots, v_n]$ form an orthonormal basis of $\mathcal{K}_n(A, r_0)$.
- Matrix formulation of $x_n \in x_0 + \mathcal{K}_n(A, r_0)$ and $r_n \perp \mathcal{K}_n(A, r_0)$:

$$x_n = x_0 + V_n t_n$$

and $\quad t_n \in \mathbb{C}^n \quad$ is found by solving

$$V_n^* A V_n \, t_n = \|r_0\| e_1 .$$

- This can be viewed as a model reduction from a (large) system of order N to a (small) system of order n.
- Intuition: Projected system should capture fast a sufficient part of information contained in the original data.
- Intuition: Powering the operator tends to transfer dominant information as quickly as possible into the projected system.

3 Distribution functions and moments

- Let A be HPD with spectral decomposition $A = Y \Lambda Y^*$, where $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_N$ (distinct eigenvalues for simplicity).
- Suppose $\omega_k = |(v_1, y_k)|^2 > 0$, $k = 1, \dots, N$, and define the distribution function

$$\omega(\lambda) = \begin{cases} 0, & \text{if } \lambda < \lambda_1, \\ \sum_{k=1}^{\ell} \omega_k, & \text{if } \lambda_\ell \le \lambda < \lambda_{\ell+1}, \text{ for } \ell = 1, \dots, N-1, \\ 1, & \text{if } \lambda_N \le \lambda. \end{cases}$$

- The moments of $\omega(\lambda)$ are given by

$$\int \lambda^k d\omega(\lambda) = \sum_{\ell=1}^{N} \omega_\ell \{\lambda_\ell\}^k = v_1^* A^k v_1 , \quad k = 0, 1, 2, \dots$$

- Analogous construction applied to $T_n = V_n^* A V_n$ yields a distribution function $\omega^{(n)}(\lambda)$ with moments given by

$$\int \lambda^k d\omega^{(n)}(\lambda) = \sum_{\ell=1}^{n} \omega_\ell^{(n)} \{\lambda_\ell^{(n)}\}^k = e_1^T T_n^k e_1 , \quad k = 0, 1, 2, \dots$$

3 Distribution functions and moments

- Let A be HPD with spectral decomposition $A = Y \Lambda Y^*$, where $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_N$ (distinct eigenvalues for simplicity).
- Suppose $\omega_k = |(v_1, y_k)|^2 > 0$, $k = 1, \dots, N$, and define the distribution function

$$\omega(\lambda) = \begin{cases} 0, & \text{if } \lambda < \lambda_1, \\ \sum_{k=1}^{\ell} \omega_k, & \text{if } \lambda_\ell \le \lambda < \lambda_{\ell+1}, \text{ for } \ell = 1, \dots, N-1, \\ 1, & \text{if } \lambda_N \le \lambda. \end{cases}$$

- The moments of $\omega(\lambda)$ are given by

$$\int \lambda^k d\omega(\lambda) = \sum_{\ell=1}^{N} \omega_\ell \{\lambda_\ell\}^k = v_1^* A^k v_1, \quad k = 0, 1, 2, \dots$$

- Analogous construction applied to $T_n = V_n^* A V_n$ yields a distribution function $\omega^{(n)}(\lambda)$ with moments given by

$$\int \lambda^k d\omega^{(n)}(\lambda) = \sum_{\ell=1}^{n} \omega_\ell^{(n)} \{\lambda_\ell^{(n)}\}^k = e_1^T T_n^k e_1, \quad k = 0, 1, 2, \dots$$

3 Stieltjes recurrence and Jacobi matrix

Let $\phi_0(\lambda) \equiv 1, \phi_1(\lambda), \dots, \phi_n(\lambda)$ be the first $n+1$ orthonormal polynomials corresponding to the distribution function $\omega(\lambda)$. Then, writing $\Phi_n(\lambda) = [\phi_0(\lambda), \dots, \phi_{n-1}(\lambda)]^*$,

$$\lambda \, \Phi_n(\lambda) \; = \; T_n \, \Phi_n(\lambda) \, + \, \delta_{n+1} \, \phi_n(\lambda) \, e_n$$

represents the Stieltjes recurrence (1893-4), see Chebyshev (1855), Brouncker (1655), Wallis (1656), Toeplitz and Hellinger (1914) with the Jacobi matrix

$$T_n \equiv \begin{pmatrix} \gamma_1 & \delta_2 & & \\ \delta_2 & \gamma_2 & \ddots & \\ & \ddots & \ddots & \delta_n \\ & & \delta_n & \gamma_n \end{pmatrix}, \quad \delta_l > 0 \, , \ell = 2, \dots, n \, .$$

3 Fundamental relationship with Gauss quadrature

$\omega^{(n)}(\lambda)$ is the distribution function determined by the n-node Gauss-Christoffel quadrature approximation of the Riemann-Stieltjes integral with $\omega(\lambda)$.

$$\begin{tikzcd}
A,\ v_1 = r_0/\|r_0\| \arrow[r, leftrightarrow] \arrow[d] & \omega(\lambda),\ \int f(\lambda)\, d\omega(\lambda) \arrow[d] \\
\text{Hermitian Lanczos/CG} \arrow[d] & \text{Gauss-Christoffel quadrature} \arrow[d] \\
T_n,\ e_1 \arrow[r, leftrightarrow] & \omega^{(n)}(\lambda),\ \sum_{j=1}^{n} \omega_j^{(n)} f(\theta_j^{(n)})
\end{tikzcd}$$

3 Continued fraction corresponding to $\omega(\lambda)$

$$\mathcal{F}_N(\lambda) \equiv \cfrac{1}{\lambda - \gamma_1 - \cfrac{\delta_2^2}{\lambda - \gamma_2 - \cfrac{\delta_3^2}{\lambda - \gamma_3 - \dots \cfrac{\ddots}{\lambda - \gamma_{N-1} - \cfrac{\delta_N^2}{\lambda - \gamma_N}}}}}$$

The entries $\gamma_1, \dots, \gamma_N$ and $\delta_2, \dots, \delta_N$ represent coefficients of the Stieltjes recurrence.

3 Partial fraction decomposition

$$b^*(\lambda I - A)^{-1} b \; = \; \int_L^U \frac{d\omega(\mu)}{\lambda - \mu} \; = \; \sum_{j=1}^{N} \frac{\omega_j}{\lambda - \lambda_j} = \frac{\mathcal{R}_N(\lambda)}{\mathcal{P}_N(\lambda)} \, ,$$

$$\frac{\mathcal{R}_N(\lambda)}{\mathcal{P}_N(\lambda)} \; \equiv \; \mathcal{F}_N(\lambda)$$

The denominator $\mathcal{P}_n(\lambda)$ corresponding to the nth convergent $\mathcal{F}_n(\lambda)$ of $\mathcal{F}_N(\lambda)$, $n = 1, 2, \dots$ is the nth orthogonal polynomial in the sequence determined by $\omega(\lambda)$; see Chebyshev (1855).

3 Fundamental relationship with Gauss quadrature

- The first $2n$ moments of the reduced model match those of the original model
- The n-node Gauss-Christoffel quadrature has algebraic degree $2n-1$, hence

$$v_1^* A^k v_1 = e_1^T T_n^k e_1 \quad \text{for} \quad k = 0, 1, \dots, 2n-1.$$

- Moment matching properties can also be derived for non-Hermitian matrices using the Vorobyev method of moments
- For the infinite dimensional Hilbert spaces and self-adjoint bounded operators it was described by Vorobyev (1958, 1965).

3 The problem of moments in Hilbert space

Let $z_0, z_1, \dots, z_n$ be $n+1$ linearly independent elements of Hilbert space V. Consider the subspace V_n generated by all possible linear combinations of $z_0, z_1, \dots, z_{n-1}$ and construct a linear operator $\mathcal{B}_n$ defined on V_n such that

$$\begin{aligned} z_1 &= \mathcal{B}_n z_0, \\ z_2 &= \mathcal{B}_n z_1, \\ &\vdots \\ z_{n-1} &= \mathcal{B}_n z_{n-2}, \\ E_n z_n &= \mathcal{B}_n z_{n-1}, \end{aligned}$$

where $E_n z_n$ is the (orthogonal or oblique) projection of z_n onto V_n.

3 Approximation of bounded linear operators

Let $\mathcal{B}$ be a bounded linear operator on Hilbert space V. Choosing an element z_0, we first form a sequence of elements $z_1, \dots, z_n, \dots$

$$z_0,\ z_1 = \mathcal{B}z_0,\ z_2 = \mathcal{B}z_1 = \mathcal{B}^2 z_0,\ \dots\ , z_n = \mathcal{B}z_{n-1} = \mathcal{B}^n z_{n-1},\ \dots$$

For the present $z_1, \dots, z_n$ are assumed to be linearly independent. Determine a sequence of operators $\mathcal{B}_n$ defined on the sequence of nested subspaces V_n such that

$$\begin{aligned} z_1 &= \mathcal{B}z_0 = \mathcal{B}_n z_0, \\ z_2 &= \mathcal{B}^2 z_0 = (\mathcal{B}_n)^2 z_0, \\ &\vdots \\ z_{n-1} &= \mathcal{B}^{n-1} z_0 = (\mathcal{B}_n)^{n-1} z_0, \\ E_n z_n &= E_n \mathcal{B}^n z_0 = (\mathcal{B}_n)^n z_0. \end{aligned}$$

3 Model reduction using Krylov subspaces

Using the projection E_n onto V_n we can write for the operators constructed above (here we need the linearity of $\mathcal{B}$)

$$\mathcal{B}_n \;=\; E_n \, \mathcal{B} \, E_n \,.$$

The finite dimensional operators $\mathcal{B}_n$ can be used to obtain approximate solutions to various linear problems. The choice of the elements $z_0, \dots, z_n, \dots$ as above gives Krylov subspaces that are determined by the operator and the initial element z_0 (e.g. by a partial differential equation, boundary conditions and outer forces).

Challenges:

- Convergence
- Krylov subspace methods in infinite dimensional Hilbert spaces?

4. Convergence and spectral information

References

- J. Liesen. and Z.S., *Krylov Subspace Methods, Principles and Analysis.* Oxford University Press (2013), Chapter 5, Sections 5.1 - 5.7
- T. Gergelits and Z.S., *Composite convergence bounds based on Chebyshev polynomials and finite precision conjugate gradient computations,* Numer. Alg. 65, 759-782 (2014)

4 Convergence bounds for the CG method

- The CG optimality property

$$\|x - x_n\|_A = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|x - z\|_A \;=\; \min_{p \in \mathcal{P}_n(0)} \|p(A)(x - x_0)\|_A$$

yields the convergence bounds

$$\frac{\|x - x_n\|_A}{\|x - x_0\|_A} \le \min_{p \in \mathcal{P}_n(0)} \max_{1 \le j \le N} |p(\lambda_j)| \le \min_{p \in \mathcal{P}_n(0)} \max_{\lambda \in [\lambda_1, \lambda_N]} |p(\lambda)|$$

$$\le 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^n, \quad \kappa = \frac{\lambda_N}{\lambda_1}.$$

- The worst-case behavior of the method is completely determined by the distribution of the eigenvalues of A.
- The widely known κ-bound is derived using Chebyshev polynomials on the interval $[\lambda_1, \lambda_N]$. It does not depend on any other properties of A, b, x_0.
- The κ-bound is linear and it can not capture the adaptation of the CG method to the problem!

4 Convergence bounds for the CG method

- The CG optimality property

$$\|x - x_n\|_A = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|x - z\|_A \;=\; \min_{p \in \mathcal{P}_n(0)} \|p(A)(x - x_0)\|_A$$

yields the convergence bounds

$$\frac{\|x - x_n\|_A}{\|x - x_0\|_A} \le \min_{p \in \mathcal{P}_n(0)} \max_{1 \le j \le N} |p(\lambda_j)| \le \min_{p \in \mathcal{P}_n(0)} \max_{\lambda \in [\lambda_1, \lambda_N]} |p(\lambda)|$$

$$\le 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^n, \quad \kappa = \frac{\lambda_N}{\lambda_1}.$$

- The worst-case behavior of the method is completely determined by the distribution of the eigenvalues of A.
- The widely known κ-bound is derived using Chebyshev polynomials on the interval $[\lambda_1, \lambda_N]$. It does not depend on any other properties of A, b, x_0.
- The κ-bound is linear and it can not capture the adaptation of the CG method to the problem!

4 CG, large outliers and condition numbers

Consider the desired accuracy ϵ, $\kappa_s(A) \equiv \lambda_{N-s}/\lambda_1$. Then

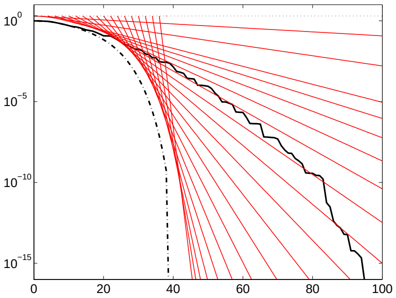
$$\mathbf{k} \;=\; \mathbf{s} + \left\lceil \frac{\ln(2/\epsilon)}{2} \sqrt{\kappa_s(A)} \right\rceil$$

CG steps will produce the approximate solution x_n satisfying

$$\|x - x_n\|_A \;\le\; \epsilon \, \|x - x_0\|_A \, .$$

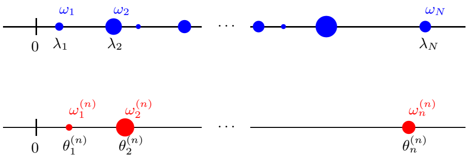
This statement qualitatively explains superlinear convergence of CG at the presence of large outliers in the spectrum, assuming exact arithmetic.

4 Adaptive Chebyshev bound?



4 Moment problem illustration

For a given n find a distribution function with n mass points in such a way that it in a best way captures the properties of the original distribution function



4 CG and Gauss quadrature errors

At any iteration step n, CG represents the matrix formulation of the n-point Gauss quadrature of the R-S integral determined by A and r_0 ,

$$\int f(\lambda)\, d\omega(\lambda) \;=\; \sum_{i=1}^{n} \omega_i^{(n)} f(\theta_i^{(n)}) \;+\; R_n(f)\, .$$

For $f(\lambda) \equiv \lambda^{-1}$ the formula takes the form

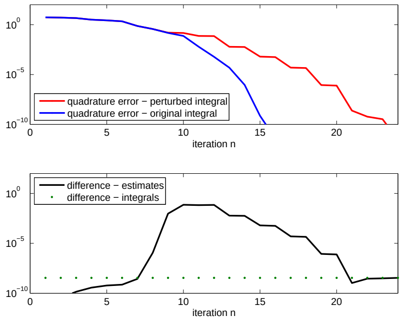
$$\frac{\|x - x_0\|_A^2}{\|r_0\|^2} \;=\; n\text{-th Gauss quadrature} \;+\; \frac{\|x - x_n\|_A^2}{\|r_0\|^2}\, .$$

This has became a base for the CG error estimation (see above); see the surveys in S and Tichý, 2002; Meurant and S, 2006; Liesen and S, 2013.

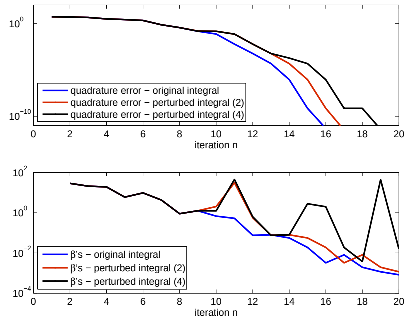
4 Gauss quadrature (CG) with tight clusters

- Replacing single eigenvalues by tight clusters can make a difference; see Greenbaum (1989); Greenbaum, S (1992); Golub, S (1994).
- The point is obvious. Orthogonal polynomials can be very sensitive to certain changes of the underlying distribution function.
- Otherwise CG behaves almost linearly and it can be described by contraction. In such case - is it worth using?

4 Sensitivity of the Gauss Quadrature



4 Simplified problem - only the largest eigenvalue is replaced



4 Theorem - O'Leary, S, Tichý (2007)

Consider distribution functions $\omega(x)$ and $\tilde{\omega}(x)$. Let

$$p_n(x) = (x - x_1) \dots (x - x_n) \quad \text{and} \quad \tilde{p}_n(x) = (x - \tilde{x}_1) \dots (x - \tilde{x}_n)$$

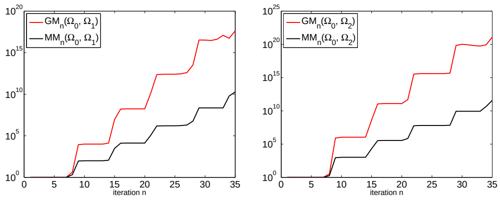
be the nth orthogonal polynomials corresponding to ω and $\tilde{\omega}$ respectively, with

$$\hat{p}_c(x) = (x - \xi_1) \dots (x - \xi_c)$$

their least common multiple. If f'' is continuous, then the difference $\Delta^n_{\omega,\tilde{\omega}} = |I^n_\omega - I^n_{\tilde{\omega}}|$ between the approximations I^n_ω to I_ω and $I^n_{\tilde{\omega}}$ to $I_{\tilde{\omega}}$, obtained from the n-point Gauss quadrature, is bounded as

$$\begin{aligned} |\Delta^n_{\omega,\tilde{\omega}}| \quad \leq \quad & \left| \int \hat{p}_c(x) f[\xi_1, \dots, \xi_c, x] \, d\omega(x) - \int \hat{p}_c(x) f[\xi_1, \dots, \xi_c, x] \, d\tilde{\omega}(x) \right| \\ + \quad & \left| \int f(x) \, d\omega(x) - \int f(x) \, d\tilde{\omega}(x) \right| . \end{aligned}$$

4 Modified moments do not tell the story



Condition numbers of the matrix of the modified moments (GM) and the matrix of the mixed moments (MM). Left - enlarged supports, right - shifted supports.

4 Summary of the CG/Lanczos part

1. Gauss-Christoffel quadrature for a small number of quadrature nodes can be highly sensitive to small changes in the distribution function enlarging its support.
2. In particular, the difference between the corresponding quadrature approximations (using the same number of quadrature nodes) can be many orders of magnitude larger than the difference between the integrals being approximated.
3. This sensitivity in Gauss-Christoffel quadrature can be observed for discontinuous, continuous, and even analytic distribution functions, and for analytic integrands uncorrelated with changes in the distribution functions and with no singularity close to the interval of integration.

4 Convergence results for the GMRES method

- For diagonalizable $A = Y \Lambda Y^{-1}$ the GMRES optimality property

$$\|r_n\|_2 = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|b - Az\|_2 = \min_{p \in \mathcal{P}_n(0)} \|p(A) r_0\|_2$$

yields the convergence bound

$$\frac{\|r_n\|_2}{\|r_0\|_2} \leq \kappa(Y) \min_{p \in \mathcal{P}_n(0)} \max_{1 \leq j \leq N} |p(\lambda_j)| .$$

- The eigenvalue distribution and the GMRES convergence are (closely) related only when $\kappa(Y)$ is small (A is close to normal).
- In general, the eigenvalues alone do not describe GMRES convergence:
- Any non-increasing convergence curve is attainable by GMRES for a matrix having any prescribed set of eigenvalues.

4 Convergence results for the GMRES method

- For diagonalizable $A = Y \Lambda Y^{-1}$ the GMRES optimality property

$$\|r_n\|_2 = \min_{z \in x_0 + \mathcal{K}_n(A, r_0)} \|b - Az\|_2 = \min_{p \in \mathcal{P}_n(0)} \|p(A) r_0\|_2$$

yields the convergence bound

$$\frac{\|r_n\|_2}{\|r_0\|_2} \leq \kappa(Y) \min_{p \in \mathcal{P}_n(0)} \max_{1 \leq j \leq N} |p(\lambda_j)| .$$

- The eigenvalue distribution and the GMRES convergence are (closely) related only when $\kappa(Y)$ is small (A is close to normal).
- In general, the eigenvalues alone do not describe GMRES convergence:
- Any non-increasing convergence curve is attainable by GMRES for a matrix having any prescribed set of eigenvalues.

4 Any GMRES convergence with any spectrum

Given any spectrum and any sequence of the nonincreasing residual norms, a complete parametrization is known of the set of all GMRES associated matrices and right hand sides.

The set of problems for which the distribution of eigenvalues alone does not correspond to convergence behavior is not of measure zero and it is not pathological.

- Widespread eigenvalues alone can not be identified with poor convergence.
- Clustered eigenvalues alone can not be identified with fast convergence.

Equivalent orthogonal matrices; pseudospectrum indication.

4 Any GMRES convergence with any spectrum

1° The spectrum of $\mathbf{A}$ is given by $\{\lambda_1, \dots, \lambda_N\}$ and $\mathrm{GMRES}(\mathbf{A}, \mathbf{b})$ yields residuals with the prescribed nonincreasing sequence $(\mathbf{x}_0 = 0)$

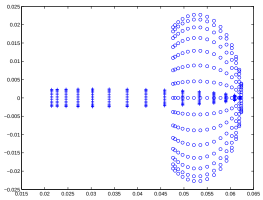
$$\|\mathbf{r}_0\| \geq \|\mathbf{r}_1\| \geq \cdots \geq \|\mathbf{r}_{N-1}\| > \|\mathbf{r}_N\| = 0 .$$

2° Let $\mathbf{C}$ be the spectral companion matrix, $h = (h_1, \dots, h_N)^T$, $h_i^2 = \|\mathbf{r}_{i-1}\|^2 - \|\mathbf{r}_i\|^2$, $i = 1, \dots, N$. Let $\mathbf{R}$ be a nonsingular upper triangular matrix such that $\mathbf{R}\mathbf{s} = \mathbf{h}$ with $\mathbf{s}$ being the first column of $\mathbf{C}^{-1}$, and let $\mathbf{W}$ be unitary matrix. Then

$$\mathbf{A} = \mathbf{W}\mathbf{R}\mathbf{C}\mathbf{R}^{-1}\mathbf{W}^* \quad \text{and} \quad \mathbf{b} = \mathbf{W}\mathbf{h} .$$

Greenbaum, Pták, Arioli and S (1994 - 98); Liesen (1999); Eiermann and Ernst (2001); Meurant (2012); Meurant and Tebbens (2012, 2014); .....

4 Convection-diffusion model problem



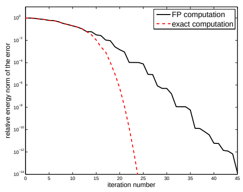
Quiz: In one case the convergence of GMRES is substantially faster than in the other; for the solution see Liesen, S (2005).

5. Inexact computation and numerical stability

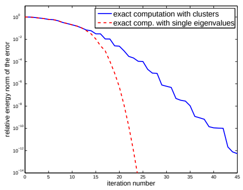
References

- J. Liesen. and Z.S., *Krylov Subspace Methods, Principles and Analysis.* Oxford University Press (2013), Chapter 5, Sections 5.8 - 5.11
- T. Gergelits and Z.S., *Composite convergence bounds based on Chebyshev polynomials and finite precision conjugate gradient computations,* Numer. Alg. 65, 759-782 (2014)

5 CG convergence behavior in finite precision arithmetic

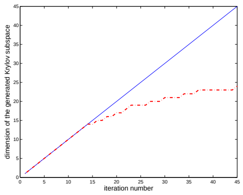


Rounding errors in finite precision CG computations cause a delay of convergence.

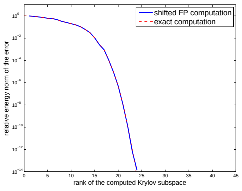


CG in finite precision corresponds to an exact CG computation for a matrix, where each eigenvalue is replaced by a tight cluster.

5 Delay of convergence and numerical rank of Krylov subspaces



The number of steps of the delay correspond to the rank-deficiency of the computed Krylov subspaces.



Shifting the finite precision curve by the number of delayed iteration steps yields the curve for the exact computation.

- The statements above can be proven by rigorous mathematical means!

5 Mathematical model of FP CG

CG in finite precision arithmetic can be seen as the exact arithmetic CG for the problem with the slightly modified distribution function with larger support, i.e., with single eigenvalues replaced by tight clusters.

Paige (1971-80), Greenbaum (1989),
Parlett (1990), S (1991), Greenbaum and S (1992), Notay (1993), ... , Druskin, Kniznermann, Zemke, Wülling, Meurant, ...
Recent reviews and updates in Meurant and S, Acta Numerica (2006); Meurant (2006); Liesen and S (2013).

One particular consequence is becoming very relevant: In FP computations, the composite convergence bounds eliminating large outlying eigenvalues at the cost of one iteration per eigenvalue (see Axelsson (1976), Jennings (1977)) are not valid.

5 Optimality in finite precision Lanczos (CG) computations?

- In exact arithmetic, local orthogonality properties of CG are equivalent to the global orthogonality properties and therefore also to the CG optimality recalled above.
- In finite precision arithmetic the local orthogonality properties are preserved proportionally to machine precision, but the global orthogonality and therefore the optimality wrt the underlying distribution function is lost.
- In finite precision arithmetic computations (or, more generally, in inexact Krylov subspace methods) the optimality property does not have any easily formulated meaning with respect to the subspaces generated by the computed residual (or direction) vectors.
- Using the results of Greenbaum from 1989, it does have, however, a well defined meaning with respect to the particular distribution functions defined by the original data and the rounding errors in the steps 1 through n.

5 Optimality in finite precision Lanczos (CG) computations?

Consider the following mathematically equivalent formulation of CG

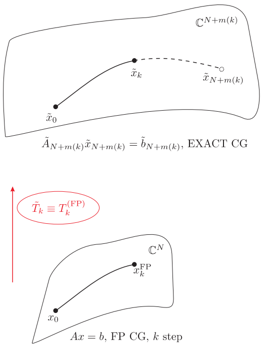
$$A\,W_n \;=\; W_n\,\mathbf{T_n} \;+\; \delta_{n+1} w_{n+1} \mathbf{e}_n^T\,, \qquad \mathbf{T}_n \;=\; W_n^*(A, r_0)\,A\,W_n(A, r_0)\,,$$

and the CG approximation given by

$$\mathbf{T}_n\,\mathbf{y}_n \;=\; \|r_0\|\mathbf{e}_1\,, \qquad x_n \;=\; x_0 \;+\; W_n\,\mathbf{y}_n\,.$$

- Greenbaum proved that the Jacobi matrix computed in finite precision arithmetic can be considered a left principal submatrix of a certain larger Jacobi matrix having all its eigenvalues close to the eigenvalues of the original matrix A.
- This is equivalent to saying that convergence behavior in the first n steps of the given finite precision Lanczos computation can equivalently be described as the result of the exact Gauss quadrature for certain distribution function that depends on n having tight clusters of points of increase around the original eigenvalues of A.

5 Analysis of the FP CG behaviour

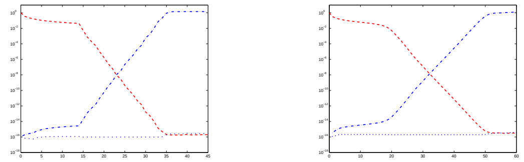

$\mathbb{C}^{N+m(k)}$
$\tilde{x}_k$
$\tilde{x}_{N+m(k)}$
$\tilde{x}_0$
$\tilde{A}_{N+m(k)}\tilde{x}_{N+m(k)} = \tilde{b}_{N+m(k)}$, EXACT CG
$\tilde{T}_k \equiv T_k^{(\mathrm{FP})}$
$\mathbb{C}^N$
x_k^{FP}
x_0
$Ax = b$, FP CG, k step

5 Numerical stability of GMRES

- In finite precision, the loss of orthogonality using the modified Gram-Schmidt GMRES is inversely proportional to the normwise relative backward error

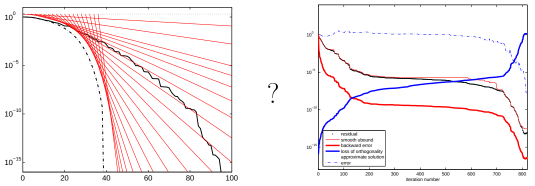
$$\frac{\|b - Ax_n\|_2}{\|b\|_2 + \|A\|_2\|x_n\|_2}.$$

Loss of orthogonality (blue) and normwise relative backward error (red) for a convection-diffusion model problem with two different “winds”:



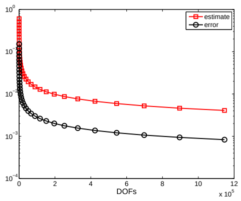
- It can be shown that the MGS-GMRES is normwise backward stable.

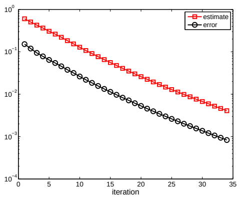
5 Delay of convergence due to inexactness



Here numerical inexactness due to roundoff. How much may we relax accuracy of the most costly operations without causing an unwanted delay and/or affecting the maximal attainable accuracy? That will be crucial in exascale computations.

5 Reaching an arbitrary accuracy in AFEM?





Inexactness and maximal attainable accuracy in matrix computations?

6. Functional analysis and infinite dimensional considerations

References

- J. Málek and Z.S., *Preconditioning and the Conjugate Gradient Method in the Context of Solving PDEs.* SIAM Spotlight Series, SIAM (2015), Chapter 9

6 Bounded invertible operators

Let V be an infinite dimensional Hilbert space, $\mathcal{B}$ a bounded linear operator on V that has a bounded inversion. Consider the problem

$$\mathcal{B}\,u \;=\; f\,, \quad f \in V\,.$$

- The identity operator on an infinite dimensional Hilbert space is not compact.
- Since $\mathcal{B}\mathcal{B}^{-1} = \mathcal{I}$, it follows that $\mathcal{B}$ can not be compact.
- Approximation of $\mathcal{B}$ by finite dimensional operators $\mathcal{B}_n : V \to V_n\,,$ V_n is finite dimensional?

6 Compact and finite dimensional operators

- A uniform (in norm) limit of finite dimensional operators $\mathcal{B}_n$ is a compact operator.
- Every compact operator on a Hilbert space is a uniform limit of a sequence of finite dimensional operators.
- A uniform limit of compact operators is a compact operator.

Bounded invertible operators in Hilbert (holds also for Banach) spaces can not be approximated in norm to an arbitrary accuracy by neither compact nor finite dimensional operators! Approximation can be considered only in the sense of strong convergence (pointwise limit); for the method of moments see Vorobyev (1958, 1965)

$$\|\mathcal{B}_n\, w - \mathcal{B}\, w\| \to 0 \quad \forall w \in V\,.$$

6 Invalid argument in convergence analysis

Let $\mathcal{Z}_h$ be a numerical approximation of the bounded operator $\mathcal{Z}$ such that, with an appropriate extension, $\|\mathcal{Z} - \mathcal{Z}_h\| \;=\; \mathcal{O}(h)\,.$

Then we have $[(\lambda - \mathcal{Z})^{-1} - (\lambda - \mathcal{Z}_h)^{-1}] \;=\; \mathcal{O}(h)$ uniformly for $\lambda \in \Gamma\,,$ where Γ surrounds the spectrum of $\mathcal{Z}$ with a distance of order $\mathcal{O}(h)$ or more. For any polynomial p

$$p(\mathcal{Z}) - p(\mathcal{Z}_h) \;=\; \frac{1}{2\pi i}\int_{\Gamma} p(\lambda)[(\lambda - \mathcal{Z})^{-1} - (\lambda - \mathcal{Z}_h)^{-1}]\, d\lambda\,,$$

and it seems that one can investigate $p(\mathcal{Z})$ instead of $p(\mathcal{Z}_h)\,.$

But the *assumption* $\|\mathcal{Z} - \mathcal{Z}_h\| \;=\; \mathcal{O}(h)\,,\ h \to 0$ does not hold for any bounded invertible infinite dimensional operator $\mathcal{Z}$.

6 Finite dimensional approximations of infinite dimensional operators

- If the infinite dimensional linear operator is bounded with the bounded inversion, then convergence of its finite dimensional approximations can be considered onlu in a pointwise sense.
- Spectral and norm equivalence of operators leads to bounds on the condition number of the discretized problems that are independent of the (Galerkin) discretization

V. Faber, T. Manteuffel and S. Parter, *On the Theory of Equivalent Operators and Application to the Numerical Solution of Uniformly Elliptic Partial Differential Equations.* Advances in Applied Math. 11, 109-163 (1990)

7. Operator preconditioning, discretization and algebraic computation

References

- J. Málek and Z.S., *Preconditioning and the Conjugate Gradient Method in the Context of Solving PDEs.* SIAM Spotlight Series, SIAM (2015)
- J. Papež, J.Liesen and Z.S., *Distribution of the discretization and algebraic error in numerical solution of partial differential equations,* Linear Alg. Appl. 449, 89-114 (2014)
- J. Papež, Z.S., and M. Vohralík, *Estimating and localizing the algebraic and total numerical errors using flux reconstructions,* (2016, submitted for publication)
- J. Papež and Z.S., *On a residual-based a posteriori error estimator for the total error,* (2016, submitted for publication, revised Dec. 2016)

7 Functional analysis and iterative methods

R. C. Kirby, SIREV (2010):

"We examine condition numbers, preconditioners and iterative methods for FEM discretization of coercive PDEs in the context of the solvability result, the Lax-Milgram lemma.

Moreover, useful insight is gained as to the relationship between Hilbert space and matrix condition numbers, and translating Hilbert space fixed point iterations into matrix computations provides new ways of motivating and explaining some classic iteration schemes. [...] This paper is [...] intending to bridge the functional analysis techniques common in finite elements and the linear algebra community."

7 Functional analysis and iterative methods

K. A. Mardal and R. Winther, NLAA (2011):

"The main focus will be on an abstract approach to the construction of preconditioners for symmetric linear systems in a Hilbert space setting [...] The discussion of preconditioned Krylov space methods for the continuous systems will be a starting point for a corresponding discrete theory.

By using this characterization it can be established that the conjugate gradient method converges [...] with a rate which can be bounded by the condition number [...] However, if the operator has a few eigenvalues far away from the rest of the spectrum, then the estimate is not sharp. In fact, a few 'bad eigenvalues' will have almost no effect on the asymptotic convergence of the method."

7 Functional analysis and iterative methods

O. Axelsson and J. Karátson, Numer. Alg. (2009):

"To preserve sparsity, the arising system is normally solved using an iterative solution method, commonly a preconditioned conjugate gradient method [...] the rate of convergence depends in general on a generalized condition number of the preconditioned operator [...]

- *if the two operators (original and preconditioner) are equivalent, then the corresponding PCG method provides mesh independent linear convergence [...]*
- *if the two operators (original and preconditioner) are compact-equivalent, then the corresponding PCG method provides mesh independent superlinear convergence."*

7 Mesh independent condition number

R. Hiptmair, CMA (2006):

"There is a continuous operator equation posed in infinite-dimensional spaces that underlines the linear system of equations [...] awareness of this connection is key to devising efficient solution strategies for the linear systems.

Operator preconditioning is a very general recipe [...]. It is simple to apply, but may not be particularly efficient, because in case of the [condition number] bound of Theorem 2.1 is too large, the operator preconditioning offers no hint how to improve the preconditioner. Hence, operator preconditioner may often achieve [...] the much-vaunted mesh independence of the preconditioner, but it may not perform satisfactorily on a given mesh."

7 Linear asymptotic behavior?

V. Faber, T. Manteuffel and S. V. Parter, Adv. in Appl. Math. (1990):

"For a fixed h, using a preconditioning strategy based on an equivalent operator may not be superior to classical methods [...] Equivalence alone is not sufficient for a good preconditioning strategy. One must also choose an equivalent operator for which the bound is small.

There is no flaw in the analysis, only a flaw in the conclusions drawn from the analysis [...] asymptotic estimates ignore the constant multiplier. Methods with similar asymptotic work estimates may behave quite differently in practice."

7 Notation

Let V be an infinite dimensional Hilbert space with the inner product

$$(\cdot,\cdot)_V : V \times V \to \mathbb{R}, \quad \text{the associated norm } \|\cdot\|_V,$$

$V^\#$ be the dual space of bounded (continuous) linear functionals on V with the duality pairing

$$\langle\cdot,\cdot\rangle : V^\# \times V \to \mathbb{R}.$$

For each $f \in V^\#$ there exists a unique $\tau f \in V$ such that

$$\langle f, v\rangle = (\tau f, v)_V \quad \text{for all } v \in V.$$

In this way the inner product $(\cdot,\cdot)_V$ determines the Riesz map

$$\tau : V^\# \to V.$$

7 Weak formulation of the BVP, assumptions

Let $a(\cdot,\cdot) = V \times V \to R$ be a bounded and coercive bilinear form. For $u \in V$ we can write the bounded linear functional $a(u,\cdot)$ on V as

$$\mathcal{A}u \equiv a(u,\cdot) \in V^\#, \quad \text{i.e.,}$$
$$\langle \mathcal{A}u, v\rangle = a(u,v) \quad \text{for all } v \in V.$$

This defines the bounded and coercive operator

$$\mathcal{A} : V \to V^\#, \quad \inf_{u\in V,\, \|u\|_V=1} \langle \mathcal{A}u, u\rangle = \alpha > 0, \ \|\mathcal{A}\| = C.$$

The Lax-Milgram theorem ensures that for any $b \in V^\#$ there exists a unique solution $x \in V$ of the problem

$$a(x,v) = \langle b, v\rangle \quad \text{for all } v \in V.$$

7 Operator problem formulation

Equivalently,

$$\langle \mathcal{A}x - b, v\rangle = 0 \quad \text{for all } v \in V,$$

which can be written as the equation in $V^\#$,

$$\mathcal{A}x = b, \qquad \mathcal{A} : V \to V^\#, \quad x \in V, \quad b \in V^\#.$$

We will consider $\mathcal{A}$ self-adjoint with respect to the duality pairing $\langle\cdot,\cdot\rangle$.

7 Discretization using $V_h \subset V$

Let $\Phi_h = (\phi_1^{(h)}, \dots, \phi_N^{(h)})$ be a basis of the subspace $V_h \subset V$,
let $\Phi_h^\# = (\phi_1^{(h)\#}, \dots, \phi_N^{(h)\#})$ be the canonical basis of its dual $V_h^\#$.

The Galerkin discretization then gives

$$\mathcal{A}_h x_h = b_h, \quad x_h \in V_h, \quad b_h \in V_h^\#, \quad \mathcal{A}_h : V_h \to V_h^\#.$$

Using the coordinates $x_h = \Phi_h \mathbf{x}$, $b_h = \Phi_h^\# \mathbf{b}$, the discretization results in the linear algebraic system

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

7 Computation

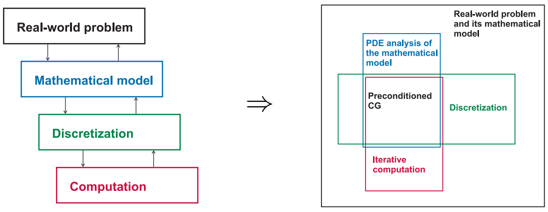
Preconditioning needed for accelerating the iterations is then often build up algebraically for the given matrix problem, giving (here illustrated as the left preconditioning)

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}.$$

Then the CG method is applied to the (symmetrized) preconditioned system, i.e., (PCG) (**M**-preconditioned CG) is applied to the unpreconditioned system. The schema of the solution process:

$$\mathcal{A}, \langle b,\cdot\rangle \to \mathbf{A}, \mathbf{b} \to \text{preconditioning} \to \text{PCG applied to } \mathbf{A}\mathbf{x} = \mathbf{b}.$$

7 A bit different view



Formulation of the model, discretization and algebraic computation, including the evaluation of the error, stopping criteria for the algebraic solver, adaptivity etc. are very closely related to each other.

7 Operator formulation of the problem

Recall that the inner product $(\cdot,\cdot)_V$ defines the Riesz map τ.
It can be used to transform the equation in $V^{\#}$

$$\mathcal{A}x = b\,, \qquad \mathcal{A}: V \to V^{\#}, \quad x \in V, \quad b \in V^{\#}\,.$$

into the equation in V

$$\tau\mathcal{A}x = \tau b, \qquad \tau\mathcal{A}: V \to V, \quad x \in V, \quad \tau b \in V\,,$$

This transformation is called operator preconditioning.

7 The mathematically best preconditioning?

With the choice of the inner product $(\cdot,\cdot)_V = a(\cdot,\cdot)$ we get

$$a(u,v) = \langle \mathcal{A}u, v\rangle = a(\tau\mathcal{A}\,u, v)$$

i.e.,

$$\tau = \mathcal{A}^{-1}\,, \quad \text{and the preconditioned system} \quad x = \mathcal{A}^{-1}b\,.$$

The inner product can be defined using an operator

$$\mathcal{B} \approx \mathcal{A}\,, \quad (\cdot,\cdot)_V \;=\; (\cdot,\cdot)_{\mathcal{B}} = \langle \mathcal{B}u, v\rangle\,.$$

Then

$$\tau = \mathcal{B}^{-1}\,, \quad \text{and the preconditioned system} \quad \mathcal{B}^{-1}\mathcal{A}x = \mathcal{B}^{-1}b\,.$$

What does it mean $\mathcal{B} \approx \mathcal{A}$?

Concept of norm equivalence and spectral equivalence of operators.

7 CG in infinite dimensional Hilbert spaces

$r_0 = b - \mathcal{A}x_0 \in V^{\#}$, $\quad p_0 = \tau r_0 \in V$. For $n = 1, 2, \ldots, n_{\max}$

$$\alpha_{n-1} = \frac{\langle r_{n-1}, \tau r_{n-1}\rangle}{\langle \mathcal{A}p_{n-1}, p_{n-1}\rangle}$$

$$x_n = x_{n-1} + \alpha_{n-1}p_{n-1}\,, \quad \text{stop when the stopping criterion is satisfied}$$

$$r_n = r_{n-1} - \alpha_{n-1}\mathcal{A}p_{n-1}$$

$$\beta_n = \frac{\langle r_n, \tau r_n\rangle}{\langle r_{n-1}, \tau r_{n-1}\rangle}$$

$$p_n = \tau r_n + \beta_n p_{n-1}$$

Hayes (1954); Vorobyev (1958, 1965); Karush (1952); Stesin (1954)
Superlinear convergence for (identity + compact) operators.
Here the Riesz map τ indeed serves as the preconditioner.

7 Discretization of the infinite dimensional CG

Using the coordinates in the bases Φ_h and $\Phi_h^{\#}$ of V_h and $V_h^{\#}$ respectively, $(\,V_h^{\#} = \mathcal{A}V_h\,)$,

$$\begin{aligned}
\langle f, v\rangle &\to \mathbf{v}^*\mathbf{f}\,, \\
(u,v)_V &\to \mathbf{v}^*\mathbf{M}\mathbf{u}, \qquad (\mathbf{M}_{ij}) = ((\phi_j,\phi_i)_V)_{i,j=1,\ldots,N}\,, \\
\mathcal{A}u &\to \mathbf{A}\mathbf{u}\,, \qquad \mathcal{A}u = \mathcal{A}\Phi_h\mathbf{u} = \Phi_h^{\#}\mathbf{A}\mathbf{u}\,; \quad (\mathbf{A}_{ij}) = (a(\phi_j,\phi_i))_{i,j=1,\ldots,N}\,, \\
\tau f &\to \mathbf{M}^{-1}\mathbf{f}\,, \qquad \tau f = \tau\Phi_h^{\#}\mathbf{f} = \Phi_h\mathbf{M}^{-1}\mathbf{f}\,;
\end{aligned}$$

we get with $b = \Phi_h^{\#}\mathbf{b}$, $x_n = \Phi_h\,\mathbf{x}_n$, $p_n = \Phi_h\,\mathbf{p}_n$, $r_n = \Phi_h^{\#}\,\mathbf{r}_n$
the algebraic CG formulation

7 Galerkin discretization gives matrix CG in V_h

$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\quad$ solve $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$, $\mathbf{p}_0 = \mathbf{z}_0$. For $n = 1, \ldots, n_{\max}$

$$\begin{aligned}
\alpha_{n-1} &= \frac{\mathbf{z}_{n-1}^*\mathbf{r}_{n-1}}{\mathbf{p}_{n-1}^*\mathbf{A}\mathbf{p}_{n-1}} \\
\mathbf{x}_n &= \mathbf{x}_{n-1} + \alpha_{n-1}\mathbf{p}_{n-1}\,, \quad \text{stop when the stopping criterion is satisfied} \\
\mathbf{r}_n &= \mathbf{r}_{n-1} - \alpha_{n-1}\mathbf{A}\mathbf{p}_{n-1} \\
\mathbf{M}\mathbf{z}_n &= \mathbf{r}_n\,, \qquad\qquad \text{solve for } \mathbf{z}_n \\
\beta_n &= \frac{\mathbf{z}_n^*\mathbf{r}_n}{\mathbf{z}_{n-1}^*\mathbf{r}_{n-1}} \\
\mathbf{p}_n &= \mathbf{z}_n + \beta_n\mathbf{p}_{n-1}
\end{aligned}$$

Günnel, Herzog, Sachs (2014); Málek, S (2015)

7 Philosophy of the a-priori robust bounds

The bound

$$\hat{\kappa}(\mathbf{M}^{-1}\mathbf{A}) = \frac{\lambda_{max}(\mathbf{M}^{-1}\mathbf{A})}{\lambda_{min}(\mathbf{M}^{-1}\mathbf{A})} \leq \frac{\sup_{u,v\in V,\,\|u\|_V=1,\|v\|_V=1} |\langle \mathcal{A}u, v\rangle|}{\inf_{u\in V,\,\|u\|_V=1}\langle \mathcal{A}u, u\rangle}$$

is valid independently of the discretization, see, e.g., Hiptmair (2006). If the bound is small enough, then the matter about the rate of convergence and its monitoring is resolved.

7 Observations

- Unpreconditioned CG, i.e. $\mathbf{M} = \mathbf{I}$, corresponds to the discretization basis Φ orthonormal wrt $(\cdot,\cdot)_V$.
- Orthogonalization of the discretization basis with respect to the given inner product in V will result in the unpreconditioned CG that is applied to the transformed (preconditioned) algebraic system. The resulting orthogonal discretization basis functions do not have local support and the transformed matrix is not sparse.
- Orthogonalization is not unique. For the same inner product we can get different bases and different discretized systems with exactly the same convergence behaviour.

7 Algebraic preconditioning?

Consider an algebraic preconditioning with the (SPD) preconditioner

$$\widehat{\mathbf{M}} = \widehat{\mathbf{L}}\widehat{\mathbf{L}}^* = \widehat{\mathbf{L}}\,(\mathbf{Q}\mathbf{Q}^*)\,\widehat{\mathbf{L}}^*$$

Where $\mathbf{Q}\mathbf{Q}^* = \mathbf{Q}^*\mathbf{Q} = \mathbf{I}$.

Question: Can any algebraic preconditioning be expressed in the operator preconditioning framework? How does it link with the discretization and the choice of the inner product in V ?

7 Change of the basis and of the inner product

Transform the discretization bases

$$\widehat{\Phi} = \Phi\,((\widehat{\mathbf{L}}\mathbf{Q})^*)^{-1}, \quad \widehat{\Phi}^\# = \Phi^\#\,\widehat{\mathbf{L}}\mathbf{Q}\,.$$

with the change of the inner product in V (recall $(u,v)_V = \mathbf{v}^*\mathbf{M}\mathbf{u}$)

$$(u,v)_{\mathrm{new},V} = (\widehat{\Phi}\widehat{\mathbf{u}}, \widehat{\Phi}\widehat{\mathbf{v}})_{\mathrm{new},V} := \widehat{\mathbf{v}}^*\widehat{\mathbf{u}} = \mathbf{v}^*\widehat{\mathbf{L}}\mathbf{Q}\mathbf{Q}^*\widehat{\mathbf{L}}^*\mathbf{u} = \mathbf{v}^*\widehat{\mathbf{L}}\widehat{\mathbf{L}}^*\mathbf{u} = \mathbf{v}^*\widehat{\mathbf{M}}\mathbf{u}\,.$$

The discretized Hilbert space formulation of CG gives the algebraically preconditioned matrix formulation of CG with the preconditioner $\widehat{\mathbf{M}}$

(more specifically, it gives the unpreconditioned CG applied to the algebraically preconditioned discretized system).

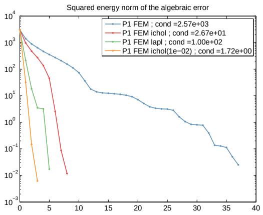
7 Sparsity, locality, global transfer of information

Sparsity of matrices of the algebraic systems is always presented as an advantage of the FEM discretizations.

Sparsity means locality of information in the individual matrix rows/columns. Getting a sufficiently accurate approximation to the solution may then require many matrix-vector multiplications (a large dimension of the Krylov space).

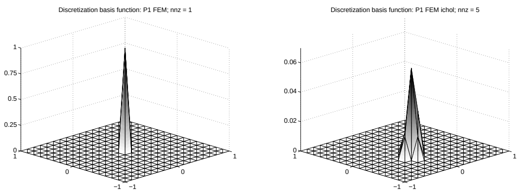
Preconditioning can be interpreted in part as addressing the unwanted consequence of sparsity (locality of the supports of the basis functions). Globally supported basis functions (hierarchical bases preconditioning, DD with coarse space components, multilevel methods, hierarchical grids etc.) can efficiently handle the transfer of global information.

7 Example - Nonhomogeneous diffusion tensor



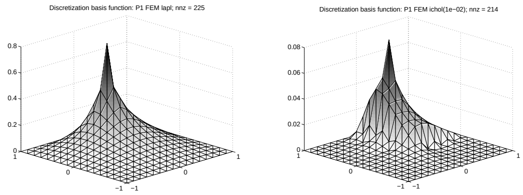
PCG convergence: unpreconditioned; ichol (no fill-in); Laplace operator preconditioning; ichol (drop-off tolerance 1e-02). Uniform mesh, condition numbers 2.5e03, 2.6e01, 1.0e02, 1.7e00.

7 Transformed basis elements



Original discretization basis element and its transformation corresponding to the ichol preconditioning.

7 Transformed basis elements



Transformed discretization basis elements corresponding to the lapl (left) and ichol(tol) preconditioning (right).

8. HPC computations with Krylov subspace methods?

References

- E. Carson, M. Rozložník, Z.S., P. Tichý, and M. Tůma, *On the numerical stability analysis of pipelined Krylov subspace methods,* (2016, submitted for publication).

8 Personal prehistory

Strakos, Z., *Efficiency and Optimizing of Algorithms and Programs on the Host Computer / Array Processor System*, Parallel Computing, 4, 1987, pp. 189-209.

- Host Computer (0.2 MFlops) / Array Processor (up to 10 MFlops).
- Large instruction overhead and slow data transfers.
- Pipelining, several arithmetic units.
- Possible overlap of data transfers and arithmetic.
- Slow scalar operations.

8 Challenges for efficient HPC computations

- Synchronized recursions.
- Matrix-vector multiplication and vector updates are linear and (possibly) fast. Preconditioning is expensive (substantial global communication).
- Scalar coefficients require inner products and synchronization points.
- Nonlinearity causes trouble. For the approximation power of the methods, nonlinearity is essential.
- Parallelization can lead to numerical instabilities.
- Algorithmic improvements are good, but a more general view is needed. Is sparsity of the system matrix always good? Is it time to reconsider the FEM civilization with its technology and axioms?

8 Parallel (communication sensitive) algorithms?

- Block recursion in order to increase arithmetic/communication ratio.
- Numerical stability is crucial.
- Stopping criteria can save the case. Size of the blocks?
- Preconditioning means an approximate solution of a part of the problem.

State-of-the-art in the algorithmic developments:

E. Carson, Communication-Avoiding Krylov Subspace Methods in Theory and Practice, PhD Thesis, UC at Berkeley, CA, 2015.

8 from HS CG towards pipelined CG

$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, solve $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$, $\mathbf{p}_0 = \mathbf{z}_0$. For $n = 1, \ldots, n_{\max}$

$$\begin{aligned}
\alpha_{n-1} &= \frac{\mathbf{z}_{n-1}^* \mathbf{r}_{n-1}}{\mathbf{p}_{n-1}^* \mathbf{A} \mathbf{p}_{n-1}} \\
\mathbf{x}_n &= \mathbf{x}_{n-1} + \alpha_{n-1} \mathbf{p}_{n-1} \, , \quad \text{stop when the stopping criterion is satisfied} \\
\mathbf{r}_n &= \mathbf{r}_{n-1} - \alpha_{n-1} \mathbf{A} \mathbf{p}_{n-1} \\
\mathbf{M}\mathbf{z}_n &= \mathbf{r}_n \, , \qquad \text{solve for } \mathbf{z}_n \\
\beta_n &= \frac{\mathbf{z}_n^* \mathbf{r}_n}{\mathbf{z}_{n-1}^* \mathbf{r}_{n-1}} \\
\mathbf{p}_n &= \mathbf{z}_n + \beta_n \mathbf{p}_{n-1}
\end{aligned}$$

In later CG variants we will not consider preconditioning
(for simplicity of presentation).

8 ST CG (see Rosser (1953)), using the notation in Carson et al. (2016)

Initialization: $r_0 = b - Ax_0$, $p_0 = r_0$, $x_{-1} = x_0$, $r_{-1} = r_0$, $e_{-1} = 0$

$$q_{i-1} = \frac{(r_{i-1}, Ar_{i-1})}{(r_{i-1}, r_{i-1})} - e_{i-2}$$

$$x_i = x_{i-1} + \tfrac{1}{q_{i-1}}[r_{i-1} + e_{i-2}(x_{i-1} - x_{i-2})]$$

$$r_i = r_{i-1} + \tfrac{1}{q_{i-1}}[-Ar_{i-1} + e_{i-2}(r_{i-1} - r_{i-2})]$$

evaluate the stopping criterion

$$e_{i-1} = q_{i-1}\frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

Three-term recurrence, a single synchronization point per iteration, used in Strakos (1987).

8 Pipelined CG (see Ghysels and Vanroose (2014))

Initialization: r_0, $p_0 = r_0$, $s_0 = Ap_0$, $w_0 = Ar_0$, $z_0 = Aw_0$, $\alpha_0 = \dfrac{(r_0, r_0)}{(p_0, s_0)}$

$$x_i = x_{i-1} + \alpha_{i-1}p_{i-1},$$
$$r_i = r_{i-1} - \alpha_{i-1}s_{i-1}$$
$$w_i = w_{i-1} - \alpha_{i-1}z_{i-1}$$

evaluate the stopping criterion

$$q_i = Aw_i$$
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$
$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$

$$p_i = r_i + \beta_i p_{i-1}$$
$$s_i = w_i + \beta_i s_{i-1}$$
$$z_i = q_i + \beta_i z_{i-1}$$

Auxiliary recursions for w_i, s_i, z_i in order to reduce the synchronization points and overlap the inner product computation with the matrix-vector multiplication.

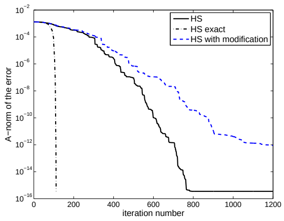
8 Individual added sources of instabilities

- Three term recurrences are less stable than coupled two term recurrences.
- Auxiliary recurrences do not recompute the recurrence coefficients. This harms the local orthogonality relations and it can possibly destabilize the whole computation.
- Modification of the computation of recurrence coefficients can have negative effect to the rate of convergence.
- Residual replacement strategy is not well understood and it needs further substantial analysis.

8 Numerical illustrations

- Matrix bcsstk03 (HB collection), $N = 112, x_0 = 0, \|b\| = 1.$
- The starting vector b has equal components in the individual invariant subspaces.
- We concentrate on the delay of convergence.
- Apart from some very particular cases, maximal attainable accuracy is not of practical importance.

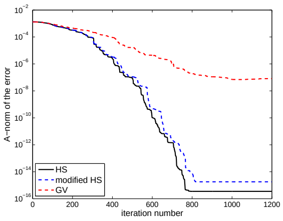
8 An innocent-looking (three-term) recurrence modification



Replacing $p_i = r_i + \beta_i p_{i-1}$ by $p_i = r_i + \dfrac{\beta_i}{\alpha_{i-1}}(x_i - x_{i-1})$

can significantly change behavior in finite precision arithmetic.

8 Adding auxiliary recurrences

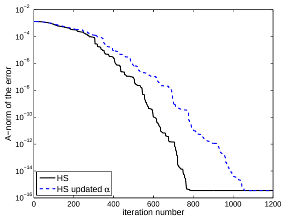


HS CG, modified HS CG with the recursive update

$$Ap_i = Ar_i + \beta_i Ap_{i-1}, \quad \text{i.e.} \quad s_i = Ar_i + \beta_i s_{i-1},$$

and GV CG.

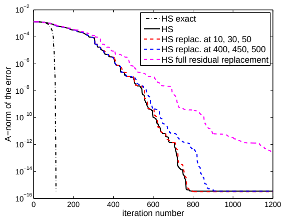
8 Update of the coefficient α_{i-1}



HS CG and the modified HS CG with explicit matrix-vectors multiplications Ar_{i-1}, Ap_{i-1}, and α_{i-1} computed using the relation

$$\frac{1}{\alpha_{i-1}} = \frac{r_{i-1}^T A r_{i-1}}{\|r_{i-1}\|^2} - \frac{\beta_{i-1}}{\alpha_{i-2}}.$$

8 Residual replacement



Residual replacement before and after the linear independence of the computed residual vectors is lost.

8 Adaptivity and stopping criteria? Papež, S (2016)

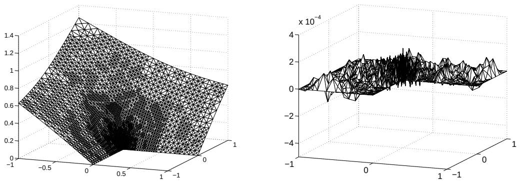
Residual-based a posteriori error bound for the total error that accounts for inexact algebraic computations, for arbitrary $v_h \in V_h$

$$\|\nabla(u-v_h)\|^2 \le 2\,C_1^2\,C_2^2\left(J^2(v_h)+\mathrm{osc}^2\right)+2\,\widetilde{C}_{\mathrm{intp}}^2(u,v_h)\,\|\nabla(u_h-v_h)\|^2\,,$$

where (using the linear FEM discretization basis functions)

$$J_E(v_h) \equiv |E|^{1/2}\left\|\left[\frac{\partial v_h}{\partial n_E}\right]\right\|_E\,, \qquad J(v_h) \equiv \left(\sum_{E\in\mathcal{E}_{\mathrm{int}}} J_E^2(v_h)\right)^{1/2}.$$

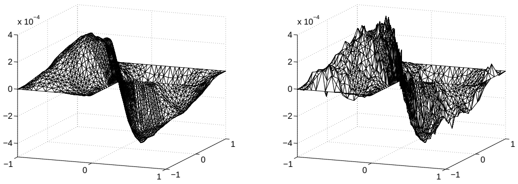
8 L-shape domain, Papež, Liesen, S (2014)



Exact solution u (left) and the discretization error $u - u_h$ (right) in the Poisson model problem, linear FEM, adaptive mesh refinement.

Quasi equilibrated discretization error over the domain.

8 L-shape domain, Papež, Liesen, S (2014)



Algebraic error $u_h - u_h^{(n)}$ (left) and the total error $u - u_h^{(n)}$ (right) after the number of CG iterations guaranteeing

$$\|\nabla(u-u_h)\| \gg \|x-x_n\|_A\,.$$

9. Myths about Krylov subspace methods

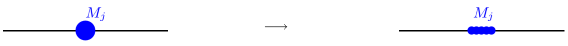
Myth: A belief given uncritical acceptance by the members of a group especially in support of existing or traditional practices and institutions.

Webster's Third New International Dictionary, Enc. Britannica Inc., Chicago (1986)

9 Widespread statements that are misleading or plainly incorrect

1. Minimal polynomials and finite termination property
2. Chebyshev bounds and CG
3. Spectral information and clustering of eigenvalues
4. Operator-based bounds and functional analysis arguments on convergence
5. Finite precision computations can not be seen as a minor modification of the exact considerations
6. Linearization of nonlinear phenomenon without noticing that this eliminates the main principle behind the phenomenon, i.e. the adaptation to the problem
7. Short term recurrences can not guarantee well conditioned basis due to rounding errors. This is true even for symmetric positive definite problems, and it remains true also for nonsymmetric problems
8. Sparsity can have positive as well as negative effects to computations

9 Clustering of eigenvalues in the SPD case



single eigenvalue λ_j $\longrightarrow$ many close eigenvalues $\widehat{\lambda}_{j_1}, \widehat{\lambda}_{j_2}, \dots, \widehat{\lambda}_{j_\ell}$

Replacing a single eigenvalue by a tight cluster can make a substantial difference; Greenbaum (1989); Greenbaum, S (1992); Golub, S (1994).

If it does not, then it means that CG can not adapt to the problem, and it converges almost linearly. **In such cases - is it worth using?**

9 Minimal polynomials, asymptotics

- It is not true that CG (or other Krylov subspace methods used for solving systems of linear algebraic equations with symmetric matrices) applied to a matrix with t distinct well separated tight clusters of eigenvalues produces in general a large error reduction after t steps; see Sections 5.6.5 and 5.9.1 of Liesen, S (2013). This myth has been disproved more than 20 years ago; see Greenbaum (1989); S (1991); Greenbaum, S (1992). Still it is persistently repeated in literature as an obvious fact.
- With no information on the structure of invariant subspaces it is not true that distribution of eigenvalues provides insight into the asymptotic behavior of Krylov subspace methods (such as GMRES) applied to systems with generally nonsymmetric matrices; see Sections 5.7.4, 5.7.6 and 5.11 of Liesen, S (2013). As before, the relevant results Greenbaum, S (1994); Greenbaum, Pták, S (1996) and Arioli, Pták, S (1998) are (almost) twenty years old.

9 How the mathematical myths are created?

- Rutishauser (1959) as well as Lanczos (1952) considered CG principally different in their nature from the method based on the Chebyshev polynomials.
- Daniel (1967) did not identify the CG convergence with the Chebyshev polynomials-based bound. He carefully writes (modifying slightly his notation)

  "assuming only that the spectrum of the matrix A lies inside the interval $[\lambda_1, \lambda_N]$, we can do no better than Theorem 1.2.2."
- That means that the Chebyshev polynomials-based bound holds for any distribution of eigenvalues between λ_1 and λ_1 and for any distribution of the components of the initial residuals in the individual invariant subspaces.
- Why we do not read the original works? They are many times most valuable sources of insight, that can be gradually forgotten and can be overshadowed by commonly accepted myth ...

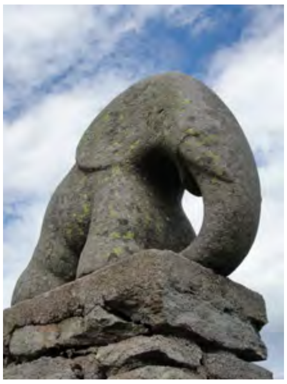
9 Analogy with a priori and a posteriori numerical PDE analysis

- Think of a priori and a posteriori numerical PDE analysis!
- The Chebyshev bound is a typical a priori bound; it uses no a posteriori information.
- A priori bounds are useful for the purpose they have been derived to. They can not take over the role of the a posteriori bounds.

9 Concluding remarks and outlook

- Krylov subspace methods adapt to the problem. Exploiting this adaptation is the key to their efficient use.
- Unlike in nonlinear problems and/or multilevel methods, analysis of Krylov subspace methods can not be based, in general, on contraction arguments.
- Individual steps modeling-analysis-discretization-computation should not be considered separately within isolated disciplines. They form a single problem. Operator preconditioning follows this philosophy.
- Fast HPC computations require handling all involved issues. A posteriori error analysis and stopping criteria are essential ...
- Assumptions must be honored.
- Historia Magistra Vitae

Thank you very much for your kind patience!



Title:	Seminar on Numerical Analysis & Winter School Proceedings of the conference SNA'17 Ostrava January 30 – February 3, 2017
Editors:	Radim Blaheta, Jiří Starý, Dagmar Sysalová
Published by:	Institute of Geonics of the CAS, Ostrava
Printed by:	Tiskárna Kleinwächter, Frýdek-Místek

First edition

Ostrava, 2017

95 copies

ISBN 978-80-86407-64-7