Preprint, Institute of Mathematics, AS CR, Prague. 2007-12-18

On Efficient Solution of Linear Systems Arising in *hp*-FEM

Tomáš Vejchodský

Abstract This contribution studies the *static condensation of internal degrees of freedom* which allows for efficient solution of linear algebraic systems arising in higher-order finite element methods. On each element, the static condensation eliminates the degrees of freedom corresponding to the internal (or bubble) basis functions. The elimination is local in elements and can be done in parallel. The resulting Schur complement system is considerably smaller and, moreover, it has less nonzero elements and better condition number in comparison with the original system. This paper focuses on the numerical performace of the static condensation and shows its CPU time efficiency.

1 Introduction and Higher-Order Finite Elements

In the standard finite element method (FEM) or more precisely in its *h*-version (*h*-FEM), the decrease of the discretization error is achieved by successive refinement of the mesh. The method converges if the size of the elements tends to zero, and the rate of this convergence is proved to be algebraic. In an alternative approach called the *p*-version (*p*-FEM), the geometry of the mesh is fixed and the polynomial degrees of the elements vary. The convergence is achieved by increasing the polynomial degrees and the convergence rate is exponential if the exact solution is C^∞-smooth. A combination of these two approaches is known as the *hp*-version (*hp*-FEM), see, e.g., [2, 4, 5, 7, 8]. To decrease the discretization error in the *hp*-FEM we either refine the elements or we increase their polynomial degrees or we both refine the elements and redistribute the polynomial degrees on the subelements in a suitable way. If this *hp*-refinement is done in a correct way, then the *hp*-FEM converges exponentially fast even in the presence of singularities.

Tomáš Vejchodský
Institute of Mathematics, Czech Academy of Sciences, Žitná 25, 115 67 Prague 1, Czech Republic
e-mail: vejchod@math.cas.cz

The higher-order FEM leads to linear algebraic systems with a special structure. This special structure can be utilized to design efficient algebraic solvers. In particular, a characteristic feature of the higher-order FEM is the presence of the so-called bubble (or internal) basis functions that are supported in a single element only. The static condensation of internal degrees of freedom (DOFs) eliminates these bubble functions from the whole system by a local (element-by-element) procedure. After this elimination, we obtain a reduced system of linear algebraic equations – the Schur complement system. From this system, we compute the other (non-internal) DOFs which correspond to vertices, edges, and faces of the elements. The number of the internal DOFs grows with the polynomial degree p by an order of magnitude faster than the number of the non-internal DOFs. Thus, for higher values of p, the number of the internal DOFs dominates and their static condensation leads to a significant decrease of the size of the linear algebraic system.

The technique of the static condensation of the internal DOFs is described in Section 2. The core of this paper lies in Section 3, where the performance of the static condensation is tested by various numerical experiments. Brief conclusions are given in Section 4.

2 Static Condensation of Internal Degrees of Freedom

To simplify the exposition, we only consider 2D elliptic problems discretized by triangular finite elements of an arbitrary order. However, the static condensation of the internal DOFs can be used in any dimension, for much wider class of problems, and for various types of higher-order finite elements.

Let $\Omega \subset \mathbb{R}^2$ be a polygon. We consider a problem whose weak formulation reads: find $u \in V$ such that

$$a(u,v) = \mathscr{F}(v) \quad \forall v \in V, \tag{1}$$

where V is a suitable Hilbert space, $a : V \times V \to \mathbb{R}$ is a continuous V-elliptic bilinear form, and $\mathscr{F}$ is a continuous linear functional on V. Problem (1) possesses a unique solution due to the Lax-Milgram lemma. For example, if

$$V = H_0^1(\Omega), \quad a(u,v) = \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x, \quad \text{and} \quad \mathscr{F}(v) = \int_\Omega f v \,\mathrm{d}x, \tag{2}$$

then (1) corresponds to the Poisson problem with homogeneous Dirichlet boundary conditions.

We discretize problem (1) by the hp-FEM. Let $\mathscr{T}_{hp}$ be a triangulation of Ω, let p_K stand for the polynomial degree assigned to the element $K \in \mathscr{T}_{hp}$, and let

$$V_{hp} = \{v_{hp} \in V : v_{hp}|_K \in P^{p_K}(K),\ K \in \mathscr{T}_{hp}\}$$

be the finite element space, where $P^{p_K}(K)$ denotes the space of polynomials of degree at most p_K on the triangle K. The hp-FEM solution $u_{hp} \in V_{hp}$ is defined by

$$a(u_{hp}, v_{hp}) = \mathscr{F}(v_{hp}) \quad \forall v_{hp} \in V_{hp}. \tag{3}$$

We consider a standard *hp*-FEM basis $\varphi_1, \varphi_2, \dots, \varphi_N$ of V_{hp}, where $N = \dim(V_{hp})$, see, e.g., [2, 4, 5, 8]. These basis functions are constructed element by element as

$$\varphi_i|_K = \varphi^K_{\iota_K^{-1}(i)}, \quad i = 1, 2, \dots, N,$$

where φ^K_m, $m = 1, 2, \dots, N^K$, denote the *shape* functions that only are supported in the single element K and $\iota_K : \{1, 2, \dots, N^K\} \to \{1, 2, \dots, N\}$ is the standard connectivity mapping, see [8, 6] for more details and Fig. 1 for an illustration. Notice that if $i \notin \mathrm{Dom}(\iota_K)$, i.e., if $\iota_K^{-1}(i)$ is not defined, then $\varphi^K_{\iota_K^{-1}(i)}$ is considered to be zero.

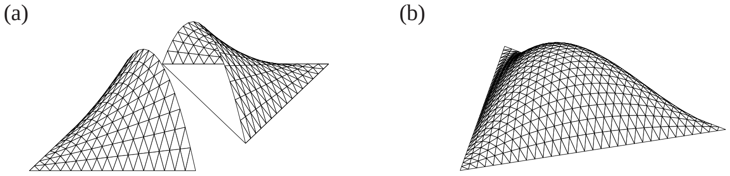

Fig. 1 (a) Two edge shape functions on two neighbouring elements form an edge basis function. (b) The bubble shape function coincides with the bubble basis function.

Problem (3) is equivalent to the system of linear algebraic equations

$$\mathbb{A}\mathbb{Y} = \mathbb{F}, \quad \mathbb{A}_{ij} = a(\varphi_j, \varphi_i), \quad \mathbb{F}_i = \mathscr{F}(\varphi_i), \quad i, j = 1, 2, \dots, N, \tag{4}$$

where $\mathbb{A} \in \mathbb{R}^{N \times N}$ and $\mathbb{F} \in \mathbb{R}^N$ are the (global) stiffness matrix and the (global) load vector, respectively. The vector $\mathbb{Y} \in \mathbb{R}^N$ contains the expansion coefficients of u_{hp} in the finite element basis.

The global stiffness matrix and the global load vector are assembled from the local stiffness matrices $\mathbb{A}^K \in \mathbb{R}^{N^K \times N^K}$ and from the local load vectors $\mathbb{F}^K \in \mathbb{R}^{N^K}$, $K \in \mathscr{T}_{hp}$. These local matrices and vectors are defined by

$$\mathbb{A}^K_{\ell m} = a_K(\varphi_{\iota_K(m)}, \varphi_{\iota_K(\ell)}) \quad \text{and} \quad \mathbb{F}^K_\ell = \mathscr{F}_K(\varphi_{\iota_K(\ell)}), \quad \ell, m = 1, 2, \dots, N^K,$$

where the local bilinear form $a_K(\cdot,\cdot)$ and the local linear functional $\mathscr{F}_K$ satisfy

$$a(\varphi_j, \varphi_i) = \sum_{K \in \mathscr{T}_{hp}} a_K(\varphi_j, \varphi_i) \quad \text{and} \quad \mathscr{F}(\varphi_i) = \sum_{K \in \mathscr{T}_{hp}} \mathscr{F}_K(\varphi_i), \quad i, j = 1, 2, \dots, N.$$

For example, if $a(\cdot,\cdot)$ and $\mathscr{F}$ are given by (2), then the local bilinear form $a_K(\cdot,\cdot)$ and the local linear functional $\mathscr{F}_K$ are defined as

$$a_K(\varphi_j, \varphi_i) = \int_K \nabla\varphi_j \cdot \nabla\varphi_i \, \mathrm{d}x \quad \text{and} \quad \mathscr{F}_K(\varphi_i) = \int_K f \varphi_i \, \mathrm{d}x.$$

With this notation, the standard finite element assembling procedure can be written as

$$\mathbb{A}_{ij} = \sum_{K\in\mathscr{T}_{hp}} \mathbb{A}^K_{\iota_K^{-1}(i),\iota_K^{-1}(j)} \quad \text{and} \quad \mathbb{F}_i = \sum_{K\in\mathscr{T}_{hp}} \mathbb{F}^K_{\iota_K^{-1}(i)}, \quad i,j=1,2,\dots,N. \tag{5}$$

From now on we will consider a special enumeration of the basis function. We enumerate the bubbles first, and then the other basis functions. Hence if M denotes the number of the bubble functions, then $\varphi_1,\dots,\varphi_M$ stand for the bubbles and $\varphi_{M+1},\dots,\varphi_N$ stand for the other basis functions. Similarly, we enumerate the shape functions in all elements. In each element $K\in\mathscr{T}_{hp}$, the first M^K shape functions are the bubbles and the other N^K-M^K shape functions are the non-bubbles. This enumeration splits the global and local stiffness matrices and the global and local load vectors into natural blocks

$$\mathbb{A} = \begin{pmatrix} A & B^T \\ B & D \end{pmatrix}, \quad \mathbb{A}^K = \begin{pmatrix} A^K & (B^K)^T \\ B^K & D^K \end{pmatrix}, \quad \mathbb{F} = \begin{pmatrix} F \\ G \end{pmatrix}, \quad \mathbb{F}^K = \begin{pmatrix} F^K \\ G^K \end{pmatrix}, \tag{6}$$

where $A\in\mathbb{R}^{M\times M}$, $B\in\mathbb{R}^{(N-M)\times M}$, $A^K\in\mathbb{R}^{M^K\times M^K}$, $B^K\in\mathbb{R}^{(N^K-M^K)\times M^K}$, etc.

Since the bubble functions are supported in a single element, the corresponding matrix A is block diagonal with the diagonal blocks being A^K, i.e., $A = \text{blockdiag}\{A^K, K\in\mathscr{T}_{hp}\}$. Thus, the matrix A is easily invertible and this makes the static condensation of the internal DOFs efficient.

The block structure (6) reshapes the global stiffness system (4) as follows

$$\begin{pmatrix} A & B^T \\ B & D \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} F \\ G \end{pmatrix}, \tag{7}$$

where $(\mathbf{x}^T,\mathbf{y}^T) = \mathbb{Y}^T$. The idea of the static condensation is to express $\mathbf{x}\in\mathbb{R}^M$ as

$$\mathbf{x} = A^{-1}(F - B^T\mathbf{y})$$

and substitute this into the second block-row of (7) to obtain the Schur complement system for $\mathbf{y}\in\mathbb{R}^{N-M}$

$$S\mathbf{y} = \widetilde{G}, \quad \text{where} \quad S = D - BA^{-1}B^T \quad \text{and} \quad \widetilde{G} = G - BA^{-1}F. \tag{8}$$

It is shown in [6] that the Schur complement S and the right-hand side $\widetilde{G}$ can be obtained by the standard finite element assembling procedure, cf. (5),

$$S_{ij} = \sum_{K\in\mathscr{T}_{hp}} S^K_{\iota_K^{-1}(M+i),\iota_K^{-1}(M+j)} \quad \text{and} \quad \widetilde{G}_i = \sum_{K\in\mathscr{T}_{hp}} \widetilde{G}^K_{\iota_K^{-1}(M+i)}, \tag{9}$$

$i,j=1,2,\dots,N-M$, where $S^K = D^K - B^K(A^K)^{-1}(B^K)^T$ are the local Schur complements and $\widetilde{G}^K = G^K - B^K(A^K)^{-1}F^K$ are the corresponding local right-hand sides.

The static condensation of the internal DOFs can also be interpreted as an orthogonalization of the non-bubble basis functions with respect to the bubbles. It can be shown that the static condensation and the partial orthogonalization of the basis are just two interpretations of the same arithmetic procedure. Moreover, if the Schur complement system (8) is solved by the ILU-PCG, then this arithmetic procedure is equivalent to the ILU-PCG applied to the original system (4). However, the usage of ILU-PCG for (4) is less efficient than the static condensation because in ILU-PCG we eliminate the internal DOFs superfluously in every iteration while it suffices to do it once. Furthermore, it can be shown that the sparsity patterns of the Schur complement S and of the original block D are identical. Hence, no fill-in appears during the construction of S. Finally, notice that the Schur complement S only depends on the space of the bubbles and not on the particular basis. All these facts are proven in [6], where more technical details can be found.

Another interesting fact, see [3], is that the conditioning of S cannot be worse than the conditioning of $\mathbb{A}$. In practice, however, the condition number of S is observed to be much smaller than the condition number of $\mathbb{A}$.

3 Numerical Performance

This section presents several numerical experiments to compare the performance of the ILU-PCG with and without the static condensation. More precisely, we compare two approaches. First, we use the static condensation and construct the Schur complement system (8), where we explicitely invert the local blocks A^K. The Schur complement system (8) is then solved by ILU-PCG. In the second approach we directly apply the ILU-PCG to system (4). We show in [6] that these approaches are two different implementations of the same arithmetic procedure and hence the number of ILU-PCG iterations N_{iter} is the same in both cases.

For the following tests, we consider the Possion problem

$$-\Delta u = f \quad \text{in } \Omega = (-1,1)^2, \quad u = 0 \quad \text{on } \partial\Omega.$$

The right-hand side $f = u\pi^2/2$ is chosen in agreement with the exact solution $u = \cos(x\pi/2)\cos(y\pi/2)$.

We stress that the static condensation can easily be implemented with the same memory requirements as the standard approach. The memory columns in Tables 1–4 below show the total number of entries in the local stiffness matrices $\mathbb{A}^K$.

The first two experiments illustrate the standard h- and p-version. For the h-FEM we start with the four element mesh with polynomial degrees 4,5,6,7, see Fig. 2(a). Then in every refinement step, we split each triangular element into four similar subtriangles with the same polynomial degree as the parent element has. In Table 1 we present: N, the total number of DOFs (the size of $\mathbb{A}$); $M - N$, the number of DOFs after the elimination of the internal DOFs (the size of S); the memory requirements (specified above); the relative discretization error $\|u - u_h\|/\|u_h\|$ measured in the energy norm; the number of ILU-PCG iterations N_{iter}; and the CPU times needed to solve the stiffness system with and without the static condensation.

Similarly, Table 2 shows the same quantities for the p-FEM. Here we start with the first order elements and increase this order by one in every step. The initial mesh was uniform with 256 elements, see Fig. 2(b). We remark that the values of the relative discretization error for $p = 8$ and $p = 9$ are already polluted by the round-off errors and by the precission of the used numerical quadrature because the discretization error is already close to the machine precission.

The results in Tables 1 and 2 show that for the presented range of polynomial degrees the static condensation of the internal DOFs decreases the solver CPU time up to ten times. We remark that the polynomial degrees higher than ten are rarely used in practice.

Notice the exponential decrease of the error for the p-version in Table 2 and in Fig. 3. This is due to the C^∞-smoothness of the exact solution. However, the number of DOFs grows very rapidly with increasing p. The question is whether the error would decrease if we fix the number of DOFs and increase p only. The answer is given in Table 3. Practically, for a given value of p we construct a uniform triangulation of Ω such that the number of DOFs is more-less fixed. Clearly, the number of elements decreases with growing p. In Table 3 we can observe the decrease of the discretization error as well as the speed-up obtained by the static condensation.

Nevertheless, the memory requirements grow with p even if the number of DOFs is fixed. This is due to the fact that the stiffness matrix $\mathbb{A}$ is more dense for higher polynomial degrees. Hence, we can modify the previous experiment in order to keep the memory requirements fixed. For a given p we construct a uniform triangulation of Ω such that the resulting memory requirements are constant. Table 4 summarizes the results. Interestingly, see also Fig. 3, the number of DOFs decreases quite rapidly but the discretization error decreases as well. However, the rate of the error decrease is not as fast as in the previous cases, which is not surprising.

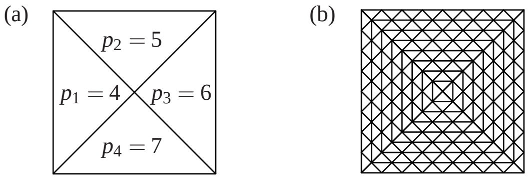

Fig. 2 (a) The initial mesh for the h-FEM. (b) The initial mesh for the p-FEM consits of linear elements ($p = 1$). There are eight elements along each edge of the square.

Table 1 The standard h-FEM.

ref.	N	$N - M$	memory	rel. err.		solver CPU time [s]	
step	(size $\mathbb{A}$)	(size S)	$[\times 10^3]$	[%]	N_{iter}	stat. con.	no conden.
0	50	16	2.7	1.2	3	0.004	0.005
1	225	89	11.0	5.4×10^{-2}	5	0.012	0.017
2	953	409	43.9	3.4×10^{-3}	7	0.049	0.130
3	3921	1745	175.7	2.1×10^{-4}	11	0.389	1.665
4	15905	7201	703.0	1.3×10^{-5}	21	4.697	26.10
5	64065	29249	2 811.9	8.2×10^{-7}	40	71.10	415.5

Table 2 The standard *p*-FEM.

	N	$N-M$	memory	rel. err.		solver CPU time [s]	
p	(size $\mathbb{A}$)	(size S)	[$\times 10^3$]	[%]	N_{iter}	stat. con.	no conden.
1	113	113	2.3	$1.0\times10^{+1}$	8	—	0.007
2	481	481	9.2	5.1×10^{-1}	10	—	0.049
3	1105	849	25.6	1.7×10^{-2}	12	0.102	0.105
4	1985	1217	57.6	4.1×10^{-4}	13	0.171	0.350
5	3121	1585	112.9	7.7×10^{-6}	14	0.302	0.987
6	4513	1953	200.7	1.3×10^{-7}	14	0.504	2.333
7	6161	2321	331.8	1.7×10^{-9}	15	0.808	4.933
8	8065	2689	518.4	2.3×10^{-11}	16	1.218	9.582
9	10225	3057	774.4	1.1×10^{-11}	16	1.773	17.407

Table 3 The *p*-FEM with fixed number of DOFs.

	N	$N-M$	memory	rel. err.		solver CPU time [s]	
p	(size $\mathbb{A}$)	(size S)	[$\times 10^3$]	[%]	N_{iter}	stat. con.	no conden.
1	28561	28561	518	6.9×10^{-1}	82	—	16.9
2	28561	28561	518	9.3×10^{-3}	83	—	28.1
3	28561	22161	640	1.3×10^{-4}	53	26.7	44.4
4	28561	17761	810	2.1×10^{-6}	41	21.2	59.8
5	28561	14737	1 016	3.2×10^{-8}	34	17.6	74.7
6	28561	12561	1 254	5.3×10^{-10}	29	15.0	89.4
7	28085	11221	1 498	8.9×10^{-12}	26	12.8	101.2
8	28561	9661	1 823	4.2×10^{-12}	24	11.7	118.5
9	27145	9663	2 045	1.3×10^{-11}	22	9.6	121.4

Table 4 The *p*-FEM with fixed memory requirements.

	N	$N-M$	memory	rel. err.		solver CPU time [s]	
p	(size $\mathbb{A}$)	(size S)	[$\times 10^3$]	[%]	N_{iter}	stat. con.	no conden.
1	28561	28561	518	6.9×10^{-1}	65	—	15.0
2	28561	28561	518	9.3×10^{-3}	62	—	26.8
3	23113	17929	518	1.8×10^{-4}	40	17.8	28.5
4	18241	11329	518	5.1×10^{-6}	28	8.8	23.9
5	14281	7345	510	1.8×10^{-7}	22	4.5	18.5
6	12013	5253	530	7.1×10^{-9}	19	2.8	15.6
7	9661	3661	518	3.6×10^{-10}	17	1.7	12.0
8	8065	2689	518	2.3×10^{-11}	16	1.2	9.6
9	7813	2325	593	1.2×10^{-11}	15	1.1	10.3

4 Conclusions

The presented experiments show that the static condensation of the internal DOFs can lead to a considerable speed-up of the solver. Asymptotically, however, if the polynomial degrees tend to the infinity and the number of elements stays fixed, then the algorithm of the static condensation is close to the computation of the inverse of the (almost) fully populated matrix, which is not efficient. On the other hand, high polynomial degrees are rare in practical computations.

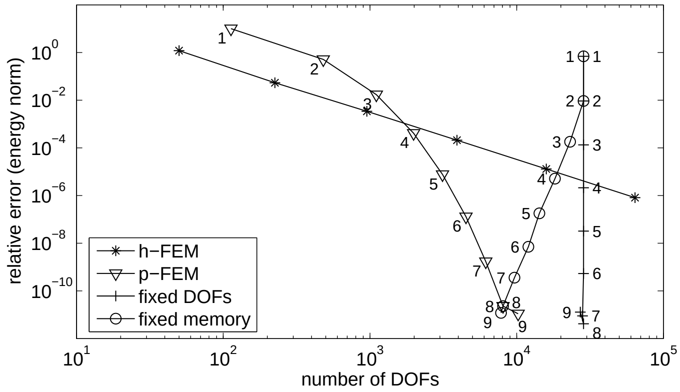

Fig. 3 The error plot in the log-log scale. The numbers indicate the polynomial degrees.

Finally we mention that more elaborate preconditioners than ILU are available for higher-order FEM, see, e.g., [1], where almost optimal preconditioners for the p-FEM are derived. However, even these preconditioners can be implemented either with or without the static condensation. For these preconditioners the static condensation would lead to the same speed-up per iteration as for the ILU preconditioner.

Acknowledgements This research has been supproted by the Grant Agency of the Czech Academy of Sciences, project No. IAA100760702, and by the Czech Academy of Sciences, institutional research plan No. AV0Z10190503. This support is gratefully acknowledged.

References

1. Babuška, I., Craig, A., Mandel, J., Pitkäranta, J.: Efficient preconditioning for the p-version finite element method in two dimensions. SIAM J. Numer. Anal. **28**(3), 624–661 (1991)
2. Demkowicz, L.: Computing with hp-adaptive finite elements. Vol. 1. One and two dimensional elliptic and Maxwell problems. Chapman & Hall/CRC, Boca Raton, FL (2007)
3. Mandel, J.: On block diagonal and Schur complement preconditioning. Numer. Math. **58**(1), 79–93 (1990)
4. Melenk, J.: hp-finite element methods for singular perturbations. Springer-Verlag, Berlin (2002)
5. Schwab, C.: p- and hp-finite element methods. Theory and applications in solid and fluid mechanics. The Clarendon Press, Oxford University Press, New York (1998)
6. Vejchodský, T., Šolín, P.: Static condensation, partial orthogonalization of basis functions, and ILU preconditioning in the hp-FEM. J. Comput. Appl. Math. (2007). DOI 10.1016/j.cam.2007.04.044
7. Šolín, P.: Partial differential equations and the finite element method. Wiley-Interscience, Hoboken, NJ (2006)
8. Šolín, P., Segeth, K., Doležel, I.: Higher-order finite element methods. Chapman & Hall/CRC, Boca Raton, FL (2004)