UTILISING OPTIMIZATION METHODS FOR COMPUTING OF NORMAL VECTOR TO CONTACT SURFACE

J. Kopačka, D. Gabriel, J. Plešek, [1] **M. Ulbin** [2]

Summary: *The stability of the contact algorithm using the penalty method is significantly affected by choosing of the penalty function. The penalty function is usually defined like a magnitude of the penetration vector multiplied by the users-defined constant - the penalty parameter. The penetration vector is obtained by solution of the minimum distance problem between the node/Gaussian integration point and the segment of the element. For a general quadrilateral contact segment this task leads to the system of two nonlinear equations. It is shown that the popular Newton-Raphson method is inadvisable for this problem. In this paper, alternative methods like quasi-Newton methods, gradient methods and the simplex method are presented. Especial attention is put on the line-search method that is crucial for a general success of quasi-Newton methods as well as gradient methods. All mentioned methods are tested by means of numerical example, which involves bending of two rectangular plates over a cylinder.*

1. Introduction

The essential part of the solution of a contact problem in the finite element method is to locate probable contact areas reliably and efficiently. Most contact searching algorithms are based on the definition of master and slave contact surfaces, when the slave nodes or integration points are checked on against master segments for penetration. The local search, which consists in the determination of the exact position of the slave node or integration point with respect to a given master segment, is a challenging and time consuming task. Unfortunately, the analytical solution for finding the distance between the node/point and the segment does not exist for a general quadrilateral contact segment. Furthermore, a lack of uniqueness in the computation of the shortest distance to the master surface is manifested in the areas with high or discontinuous curvature. The Newton-Raphson iteration scheme is often used for the solution of this minimization problem (Benson and Hallquist, 1990). However, the method is expensive and can even diverge unless the initial guess is quite accurate. Therefore, various numerical methods are tested in this paper.

[1] Ing. Ján Kopačka, Ing. Dušan Gabriel, Ph.D., Ing. Jiří Plešek, CSc., Institute of Thermomechanics AS CR, v.v.i., Dolejškova 5, 182 00 Praha 8, tel.: +420 266 053 442, e-mail: kopacka@it.cas.cz

[2] Doc. Dr. Miran Ulbin, Faculty of Mechanical Engineering, University of Maribor, Smetanova 17, 2000 Maribor, tel: +386 2 220 7705, e-mail: ulbin@uni-mb.si

2. Local Contact Search

Let us consider the contact searching algorithm has already found contact pairs containing the slave Gaussian integration points and their corresponding master segments (for detail see Gabriel, Pleek and Ulbin (2004)). The aim of the local contact search is to compute the exact position of the point $\mathbf{r}^*$ on the master segment γ (see Fig. 1). The normal $\mathbf{n}$ constructed in the point $\mathbf{r}^*$ have to direct toward the given slave Gaussian integration point $\mathbf{r}_{IG}$. Then the point $\mathbf{r}^*$ is assumed to come in contact with the Gauss point $\mathbf{r}_{IG}$ and the normal vector $\mathbf{n}$ is used for the evaluation of the penetration $\mathbf{d}$ by the contact searching algorithm.

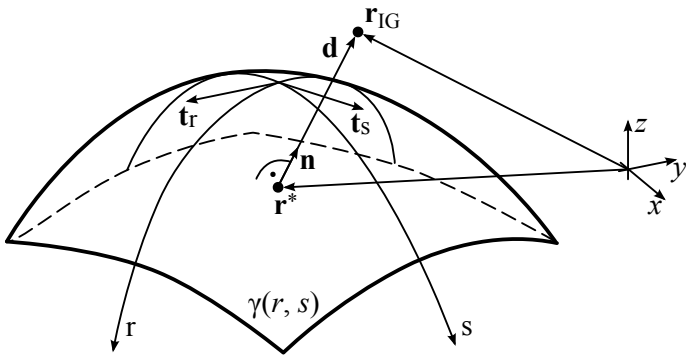

Figure 1: The formulation of the minimum distance problem.

Thus, let us consider the slave Gaussian integration point $\mathbf{r}_{IG} \in \mathbb{E}^3$ and the isoparametric master segment $\gamma\,(r,s) \in \mathbb{E}^3$ for which $r, s \in [-1, 1]$. Then the point $\mathbf{r}^*$ has to satisfy

$$\|\mathbf{r}_{IG} - \mathbf{r}^*\| \leq \|\mathbf{r}_{IG} - \mathbf{r}\|\,, \text{ for}\, \forall \mathbf{r} \in \gamma. \tag{1}$$

For this purpose, the so-called square-distance function is defined as

$$f\,(r,\ s) = [x_{IG} - x\,(r,s)]^2 + [y_{IG} - y\,(r,s)]^2 - [z_{IG} - z\,(r,s)]^2\,. \tag{2}$$

Apparently, the desired point $\mathbf{r}^*\,(r,s)$ has to satisfy the necessary conditions for local extremum of (2)

$$\begin{aligned} \frac{\partial x}{\partial r} \cdot (x_{IG} - x) + \frac{\partial y}{\partial r} \cdot (y_{IG} - y) + \frac{\partial z}{\partial r} \cdot (z_{IG} - z) = 0, \\ \frac{\partial x}{\partial s} \cdot (x_{IG} - x) + \frac{\partial y}{\partial s} \cdot (y_{IG} - y) + \frac{\partial z}{\partial s} \cdot (z_{IG} - z) = 0. \end{aligned} \tag{3}$$

The minimization of the square-distance function (2) leads to the problem of unconstrained optimization.

3. Numerical Example

For testing of the stability of the local contact search algorithm, the problem of bending two rectangular plates over a cylinder was considered (Gabriel, Pleek and Ulbin, 2004). Two rectangular plates are being bent over a cylinder (see Fig. 2). The plates are loaded by uniformly distributed

surface traction $q = 22.5\,\text{MPa}$. Material properties are: Young modules $E = 2.1 \times 10^5\,\text{MPa}$, Poisson's ratio $\nu = 0.36$ and dimensions: $r = 0.4\,\text{m}$, $l = 2\,\text{m}$, $h = 0.08\,\text{m}$, $b = 0.6\,\text{m}$. The value of the penalty parameter was set to $\xi = 10^{13}\,\text{N} \cdot \text{m}^{-3}$. Due to triple symmetry, only one eighth was modelled using 1168 twenty-node brick elements. The FE mesh is plotted in Fig. 2.

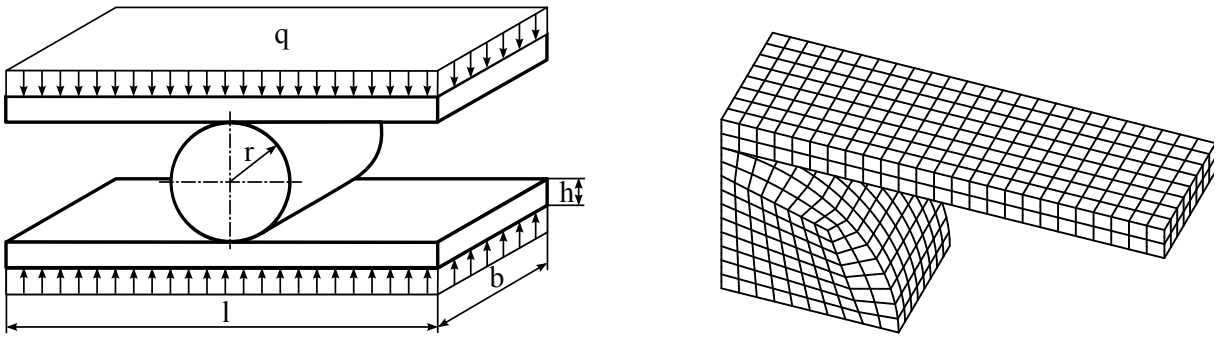

Figure 2: Bending two rectangular plates over a cylinder.

Fig. 3 shows the deformed shape of the FE mesh at the moment the instability of the calculation of the normal vector by the Newton-Raphson method occurred. Due to the regularity of the stiffness matrix, the plate and the cylinder are connected in the centre of symmetry. This connection causes considerable deformation.

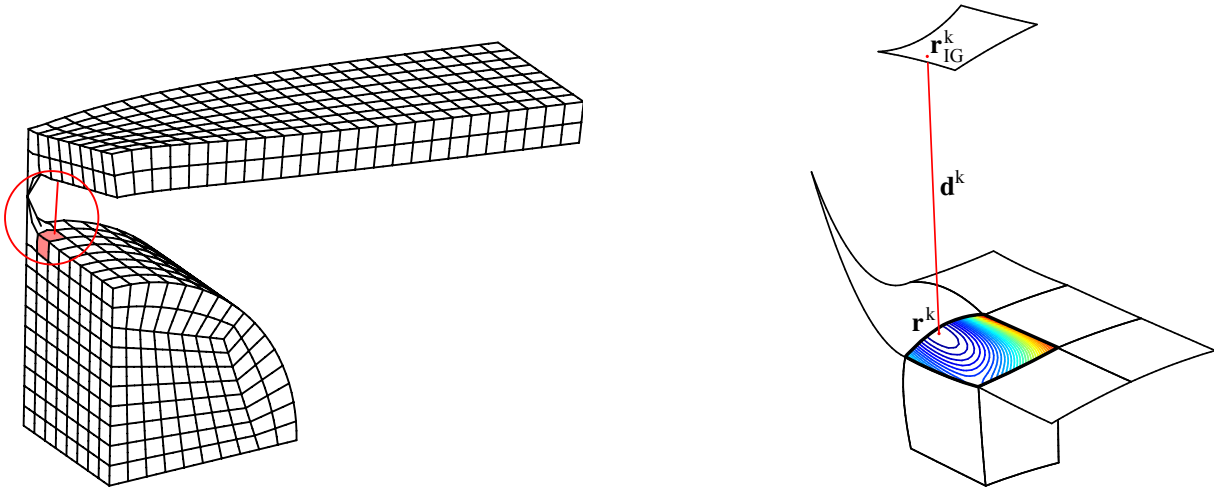

Figure 3: Deformed shape of the plate at that moment the instability of the calculation of the normal vector occurred.

Furthermore, the detail of distorted master segment is plotted in Fig. 3, where contours of the square-distance function (2) are depicted for the isoparametric coordinates $r, s \in [-1, 1]$. The topology of this master segment was used as the benchmark configuration for numerical test. The red abscissa denotes the penetration vector $\mathbf{d}^k$ between the Gaussian integration point $\mathbf{r}^k_{IG}$ and the point $\mathbf{r}^k$ which is located on the master segment.

Fig. 4 depicts the contours of the square-distance function (2) for the isoparametric coordinates $r, s \in [-5, 5]$.

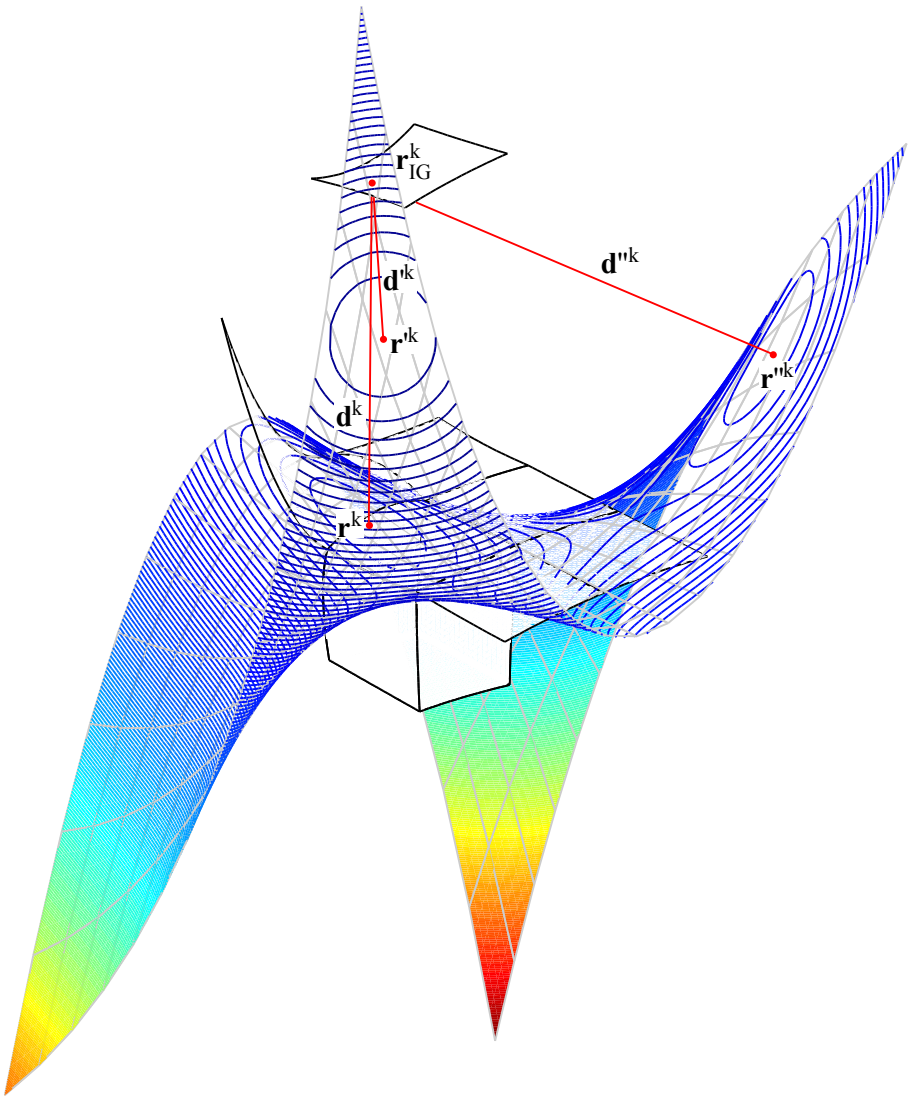

Figure 4: The isoparametric interpolation of the master segment for $r,\ s \in [-5, 5]$ including the contours of square-distance function.

It is clear that the contact area beyond the domain $r,\ s \in [-1, 1]$ is not approximated well. Therefore we are interested only in local minima which both coordinate belong to interval $[-1, 1]$. Fig. 4 shows three possible solutions $\mathbf{r}^k$, $\mathbf{r}'^k$, $\mathbf{r}''^k$ of the nonlinear system (3) plotted in the three dimensional space. Although the square-distance function (2) has the global minimum in the point $\mathbf{r}'^k$, we are interested only in local minimum $\mathbf{r}^k$, which lies in the domain $r \times s$, $r, s \in [-1, 1]$. Note that there are two saddle points, which also fulfils the necessary condition for local extremum (3).

The important part of the assessment of methods for local contact search is a good choice of the initial guess. For the purpose of numerical tests seven initial guesses, depicted in Fig. 5, were considered. Their coordinates are showed in Tab. 1. The first one is the estimation used in the current version of the local contact search algorithm in the FE code PMD. Its coordinates are obtained by one iteration of least-square projection method according to reference Benson and Hallquist (1990). Another four initial guesses were chosen in the corners of the master

segment and the sixth one in the origin of the isoparametric coordinates. The position of the last guess was selected beyond boundaries of the master segment near the local minimum $\mathbf{r}''^k$.

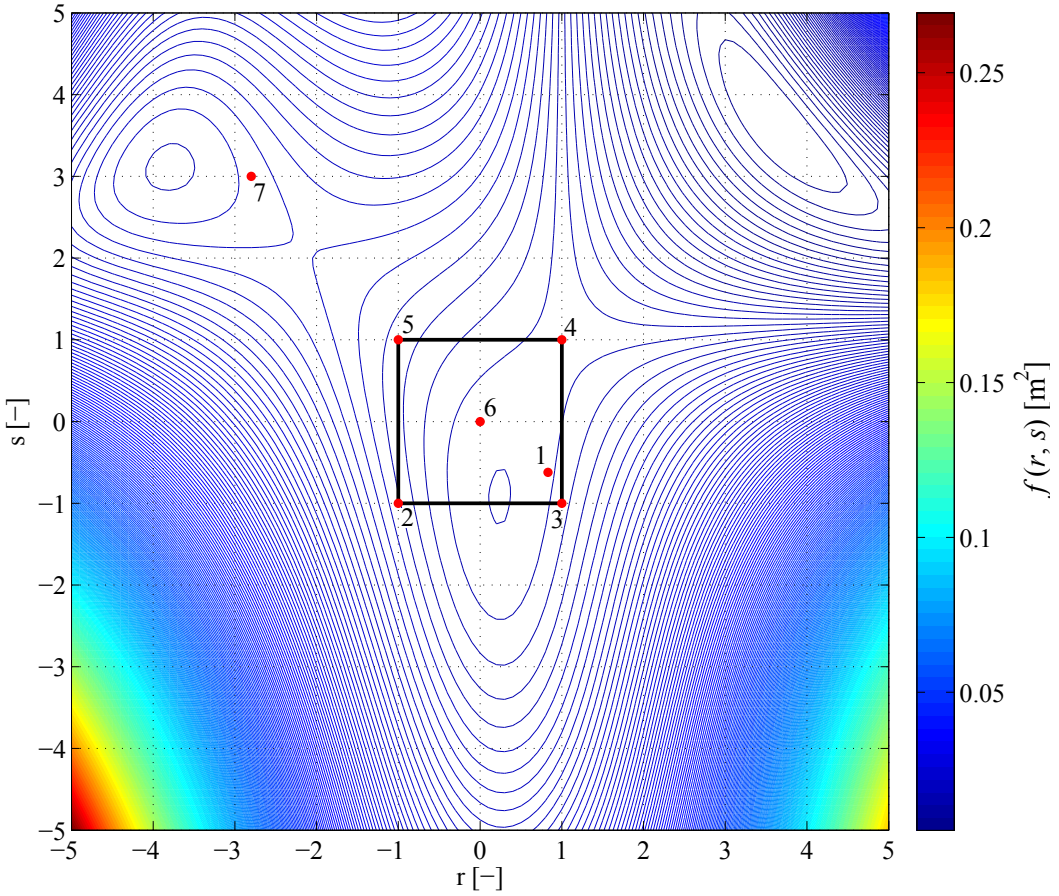

Figure 5: The square-distance function and chosen initial guesses.

Table 1: Isoparametric coordinates of chosen initial guesses.

n	1	2	3	4	5	6	7
Coor.	$(0.83, -0.62)$	$(-1, -1)$	$(1, -1)$	$(1, 1)$	$(-1, 1)$	$(0, 0)$	$(-2.8, 3)$

4. Newton-Raphson method

The Newton-Raphson method is implemented in the current version of the local contact search algorithm in the FE code PMD. The numerical scheme of this method is based on the first-order Taylor's series expansion of (3) about $\mathbf{x}^k$. The numerical scheme of the Newton-Raphson method can be written in the form

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \left(\mathbf{H}^k\right)^{-1} \nabla f^k, \tag{4}$$

where $\mathbf{H}^k \in \mathbb{R}^{n,n}$ is the Hessian matrix, $\mathbf{x}^k$ and $\mathbf{x}^{k+1} \in \mathbb{R}^n$ are vectors of unknown variables and $\nabla f^k \in \mathbb{R}^n$ is the gradient of the minimized function.

The iteration process for the initial guess 1 form Tab. 1 is depicted in Fig. 6. The Newton-Raphson method computes an exact Hessian matrix $\mathbf{H}^k$ in a point $\mathbf{x}^k$. Since the Hessian matrix $\mathbf{H}^0$ and $\mathbf{H}^{10}$ are indefinite, the approximations of solution $\mathbf{x}^1$ and $\mathbf{x}^{11}$ jump far from the previous iterations randomly (see Fig. 7). The Newton-Raphson converges to the stationary point $(0.241, -0.926)$ in 28 iterations. (The criterion of convergence is $\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| \leq 1e^{-10}$.)

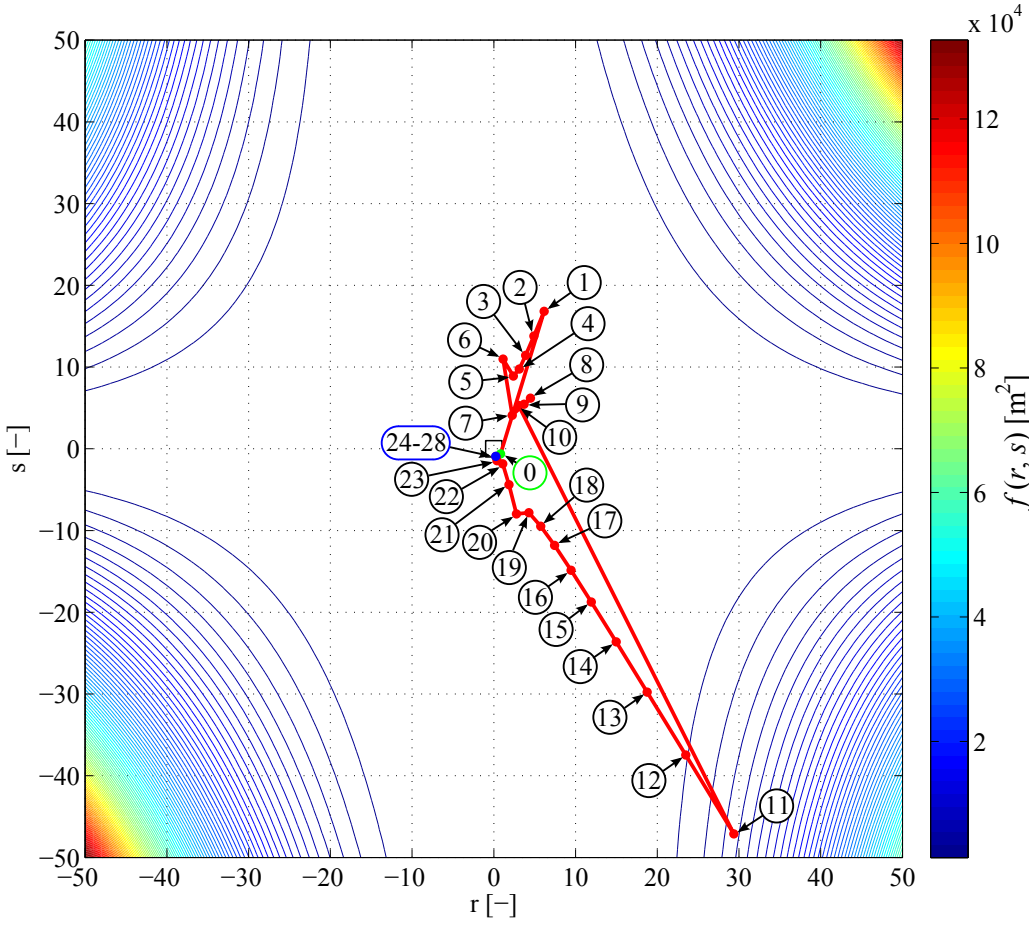

Figure 6: Iterations of the Newton-Raphson methods for the initial guess $(0.83, -0.62)$.

Tab. 2 shows results for all tested initial guesses. For each of them there are the coordinates of solution, the number of iterations (NI), the CPU time and the value of principal minors of the Hessian matrix for the determination of the type of stationary point. Since both principal minors of the Hessian matrix are positive in the point $(0.241, -0.926)$, the square-distance function (2) has the local minimum here.

If the initial guess lies in a sub-domain where the square-distance function is convex, as the starting point 6, the solution converges quadratically. However, the convergence of the Newton-Raphson method is generally difficult to achieve since the Hessian matrix of the function to be minimized (2) is not positive definite. We can conclude that the Newton-Raphson method is not suitable method for the local contact search algorithms.

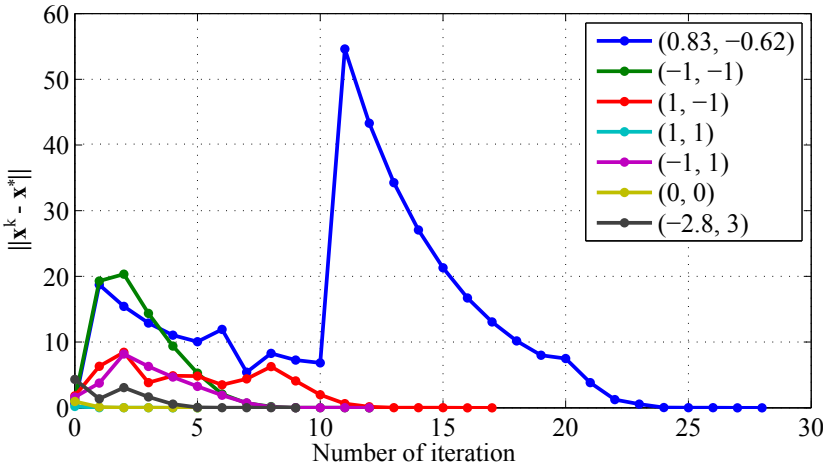

Figure 7: Dependence of $\left\|x^k - x^*\right\|$ on the number of iterations for the Newton-Raphson method.

Table 2: Results of the Newton-Raphson method for all tested initial guesses.

n	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
1	$(0.241, -0.926)$	28	0.0366	1.24e-06	2.82e-03
2	$(0.241, -0.926)$	12	0.0163	1.24e-06	2.82e-03
3	$(0.923, 0.810)$	17	0.0223	-1.76e-06	7.46e-04
4	$(0.923, 0.810)$	5	0.0073	-1.76e-06	7.46e-04
5	$(-2.167, 2.133)$	12	0.0162	-2.68e-06	-2.29e-04
6	$(0.241, -0.926)$	5	0.0074	1.24e-06	2.82e-03
7	$(0.923, 0.810)$	9	0.0130	-1.76e-06	7.46e-04

5. Line-search Technique

Each iteration of a line-search method computes a search direction $\mathbf{p}^k \in \mathbb{R}^n$ and then decides how far to move along that direction. The iteration is given by

$$\mathbf{x}^{k+1} = \mathbf{x}^k + t^k \mathbf{p}^k, \tag{5}$$

where the positive scalar $t^k \in \mathbb{R}$ is called the step-length. The success of a line-search method depends on the effective choices of both the direction $\mathbf{p}^k$ and the step-length parameter t^k. Most line-search algorithms require $\mathbf{p}^k$ to be a descent direction for which $\mathbf{p}^k \cdot \nabla f^k < 0$. The search direction often has the form

$$\mathbf{p}^k = -\mathbf{D}^k \nabla f^k, \tag{6}$$

where $\mathbf{D}^k \in \mathbb{R}^{n,n}$ is a suitable matrix. Let us consider that $\mathbf{D}^k$ is positive definite. From multiplication of (6) by ∇f^k arise

$$\mathbf{p}^k \cdot \nabla f^k = -\left(\mathbf{D}^k \nabla f^k\right) \cdot \nabla f^k < 0. \tag{7}$$

Thus, the positive definiteness of $\mathbf{D}^k$ guarantee a descent direction of $\mathbf{p}^k$. How to compute the matrix $\mathbf{D}^k$ will be discussed in consequence sections. We now give attention to the choice of the

step-length parameter t^k. Its computation is based on the restriction of the minimized function $f(\mathbf{x})$ to the ray from a point $\mathbf{x}^k$ in the search direction $\mathbf{p}^k$

$$\varphi(t) = f\left(\mathbf{x}^k + t\mathbf{p}^k\right), \quad t > 0. \tag{8}$$

Apparently, the exact minimization of this function is computationally expensive. To find even a local minimizer of $\varphi(t)$ generally requires too many evaluations of the minimized function $f(\mathbf{x})$. In the reference Nocedal and Wright (1999), more sophisticated strategies are mentioned to perform an inexact line-search to identify a step-length that achieves reductions in $f(\mathbf{x})$.

5.1. Strong Wolfe Conditions

A suitable step-length t^k should first of all give sufficient decrease in the minimized function $f(\mathbf{x})$. Therefore, we insist that the value of $\varphi(t)$ in the candidate for t^k is less than the value of a linear function $l(t)$. Such an inequality is known as Armijo condition or the sufficient decrease condition that has a form

$$f\left(\mathbf{x}^k + t^k\mathbf{p}^k\right) \leq f\left(\mathbf{x}^k\right) + c_1 t^k \nabla f^k \cdot \mathbf{p}^k, \quad c_1 \in (0,1). \tag{9}$$

The parameter c_1 sets the slope of the linear function $l(t)$ (see Fig. 8). In practise, c_1 is chosen to be quite small. According to Nocedal and Wright (1999), $c_1 = 10^{-4}$. In Fig. 8, there are two intervals, denoted by AC, which fulfill the Armijo condition (9).

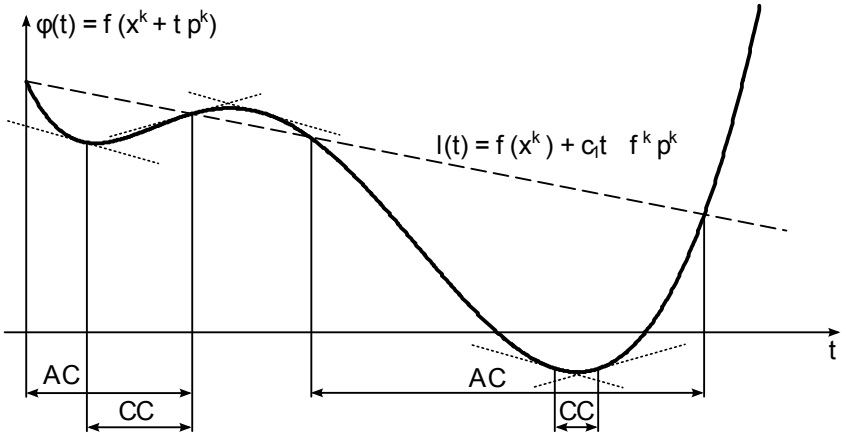

Figure 8: The Wolfe condition (AC - Armijo condition, CC - curvature condition).

The sufficient decrease condition is not enough to ensure that the algorithm makes reasonable progress. It is satisfied for all sufficiently small values of t as can be seen from Fig. 8. Therefore, a second requirement, that is called the curvature condition, is introduced

$$\left|\nabla f\left(\mathbf{x}^k + t^k\mathbf{p}^k\right) \cdot \mathbf{p}^k\right| \leq c_2 \left|\nabla f^k \cdot \mathbf{p}^k\right|, \quad c_2 \in (c_1, 1). \tag{10}$$

It is based on the fact, that the gradient of function is close to zero in a neighborhood of a local extremum. Thus, the curvature condition enforces only a slight slope of $\varphi(t)$ in the candidate for t^k. The acceptable slope is set by the parameter c_2. According to Nocedal and

Wright (1999), typical values are 0.9 when the search direction $\mathbf{p}^k$ is computed by the Newton or the quasi-Newton method, and 0.1 when $\mathbf{p}^k$ is obtained from the gradient methods.

In Fig. 8, the intervals denoted by CC fulfill the curvature condition (10). The Armijo condition (9) and the curvature condition (10) are known as the strong Wolfe conditions.

5.2. Step-length Selection Algorithm

For one-dimensional minimizer of general nonlinear functions is necessary to use an iterative procedure. Our step-length procedure is based on the interpolation of known function $\varphi(t)$ and their derivations. The procedure generates a decreasing sequence of values t^i. The superscript i denotes the iteration counter of the step-length procedure. The initial guess t^0 is simply set to 1. Then the strong Wolfe conditions (9), (10) are checked. If the conditions are satisfied for this step-length, the procedure is terminated. Otherwise, we know that the interval $[0, t^0]$ contains acceptable step-lengths (see Fig. 8). A quadratic approximation $\varphi_q(t)$ to function $\varphi(t)$ can be formed as

$$\varphi_q(t) = \left(\frac{\varphi(t^0) - t^0 \varphi'(0) - \varphi(0)}{(t^0)^2} \right) t^2 + \varphi'(0)\, t + \varphi(0). \tag{11}$$

The value of t^1 is defined as the minimizer of this quadratic function, that is

$$t^1 = -\frac{\varphi'(0)\,(t^0)^2}{2\left[\varphi(t^0) - \varphi(0) - t^0 \varphi'(0)\right]}. \tag{12}$$

If the strong Wolfe conditions (9), (10) are satisfied for t^1, the step-length procedure is terminated. Otherwise, a cubic function is used to interpolate $\varphi(t^{i-1})$, $\varphi'(t^{i-1})$, $\varphi(t^i)$, $\varphi'(t^i)$. In the reference Nocedal and Wright (1999) the minimizer of the cubic function is given by

$$t^{i+1} = t^i - \left(t^i - t^{i-1}\right) \left[\frac{\varphi'(t^i) + d_2 - d_1}{\varphi'(t^i) - \varphi'(t^{i-1}) + 2d_2} \right], \tag{13}$$

with

$$\begin{aligned} d_1 &= \varphi'\left(t^{i-1}\right) + \varphi'\left(t^i\right) - 3\frac{\varphi(t^{i-1}) - \varphi(t^i)}{t^{i-1} - t^i}, \\ d_2 &= \sqrt{d_1^2 - \varphi'(t^{i-1})\,\varphi'(t^i)}. \end{aligned}$$

If the strong Wolfe conditions (9), (10) are satisfied at t^{i+1}, the step-length procedure is terminated. Otherwise, the interpolation process is repeated by discarding the data at one of the step-length and replacing it by $\varphi(t^{i+1})$ and $\varphi'(t^{i+1})$. Then, the repetition of the interpolation process continues until the strong Wolfe conditions are fulfilled.

6. Method of Steepest Descent

The steepest descent method is the easiest line-search method for which $\mathbf{D}^k$ is the identity matrix and so the search direction $\mathbf{p}^k$ is the negative gradient. Thus, the iteration scheme is

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \nabla f^k. \tag{14}$$

Significant feature of the steepest descent is insensibility to the saddle points. It is verify in Tab. 3, where only the positive values of principal minors occur. This is due to the Hessian matrix is not employed in the computation. Initial guess 4 was intended to test each method's behaviour in the vicinity of such a point.

Table 3: Results of the method of steepest descent for all tested initial guesses.

n	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
1	$(0.241, -0.926)$	51	0.1647	1.24e-06	2.82e-03
2	$(0.241, -0.926)$	28	0.0923	1.24e-06	2.82e-03
3	$(0.241, -0.926)$	14	0.04451	1.24e-06	2.82e-03
4	$(3.648, 3.624)$	29	0.0951	7.66e-06	6.33e-03
5	$(3.648, 3.624)$	1543	5.2326	7.66e-06	6.33e-03
6	$(0.241, -0.926)$	71	0.2231	1.24e-06	2.82e-03
7	$(-3.804, 3.111)$	13	0.0414	6.76e-06	2.53e-03

7. BFGS Method

The very effective minimization method is the Broyden-Fletcher-Goldfarb-Shenno (BFGS) method. This method requires no evaluation of the Hessian matrix. Moreover, the BFGS method was developed so that the search direction $\mathbf{p}^k$ is always a descent direction because there is guarantee that matrix $\mathbf{D}^k$ is positive definite.

The BFGS iteration scheme is defined by the recurrence formula (Peressini, Sulliven and Uhl, 1988)

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \left(\mathbf{D}^k\right)^{-1} \nabla f^k. \tag{15}$$

The update of $\mathbf{D}^k$ is computed by

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\mathbf{y}^k \otimes \mathbf{y}^k}{\mathbf{d}^k \cdot \mathbf{y}^k} - \frac{\mathbf{D}^k \mathbf{d}^k \otimes \mathbf{D}^k \mathbf{d}^k}{\mathbf{d}^k \cdot \mathbf{D}^k \mathbf{d}^k}, \tag{16}$$

where $\mathbf{d}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$ and $\mathbf{y}^k = \nabla f^{k+1} - \nabla f^k$.

The best rate of convergence was reached by this method. For the initial guess 1 there are only eight iteration steps necessary to achieve the solution. Low numbers of iterations imply low CPU times in Tab. 4. It should be noted that the residual norm $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ for all initial points except initial guess 5 is practically zero within four iterations. (see Fig. 9).

The BFGS is one of the most popular methods to solve unconstrained nonlinear optimization problems. Low CPU times for all tested points confirm the effectiveness of this method (see Tab. 4). It is a very good candidate for the local contact search algorithm.

8. Simplex Method

The algorithm of the simplex method consists of three rules (Lederer, 1988). The minimized function is evaluated in all vertices. The first rule says that the vertex with the maximal function

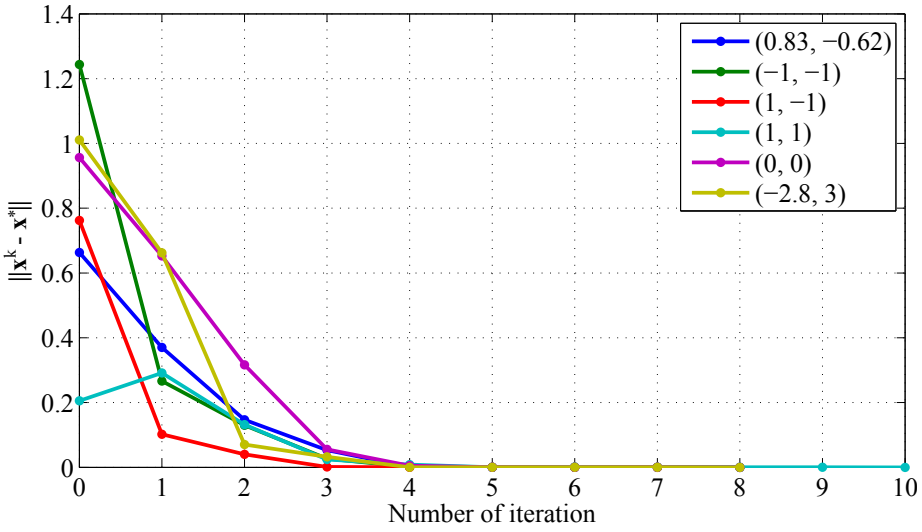

Figure 9: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the BFGS method.

Table 4: Results of the BFGS method for all tested initial guesses.

n	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
1	$(0.241, -0.926)$	8	0.0136	1.24e-06	2.82e-03
2	$(0.241, -0.926)$	8	0.0149	1.24e-06	2.82e-03
3	$(0.241, -0.926)$	7	0.0132	1.24e-06	2.82e-03
4	$(0.923, 0.810)$	9	0.0174	-1.76e-06	7.46e-04
5	$(0.923, 0.810)$	23	0.0444	-1.76e-06	7.46e-04
6	$(0.241, -0.926)$	8	0.0154	1.24e-06	2.82e-03
7	$(-3.804, 3.111)$	8	0.0149	6.76e-06	2.53e-03

value is released. Instead, it is replaced by the new one.

In case the function value in the new vertex is maximal again, there is the second rule. Due to the infinite loop, it is not allowed to return the vertex back in the consequential iteration. Instead, the vertex with second highest function value is released.

And, at last, the third rule treats the case when one of the vertices is still on the same place. This situation indicates that the simplex rotates above a local extremum. Therefore, the simplex edge length a is halved after m iterations. According to (Lederer, 1988) the number m is recommended to select

$$m = 1.65n + 0.05n^2, \tag{17}$$

where n is the dimension. Fig. 10 shows how the simplex method works.

Although there are no evaluations of derivations in this method, CPU times in Tab. 5 are also higher in comparison with other tested methods. This is, among other things, because MATLAB is interpreted programming language. There are many iteration loops and if-else statements in our code. Nevertheless, the implementation to a compiled language like FORTRAN could be suitable for the local contact search.

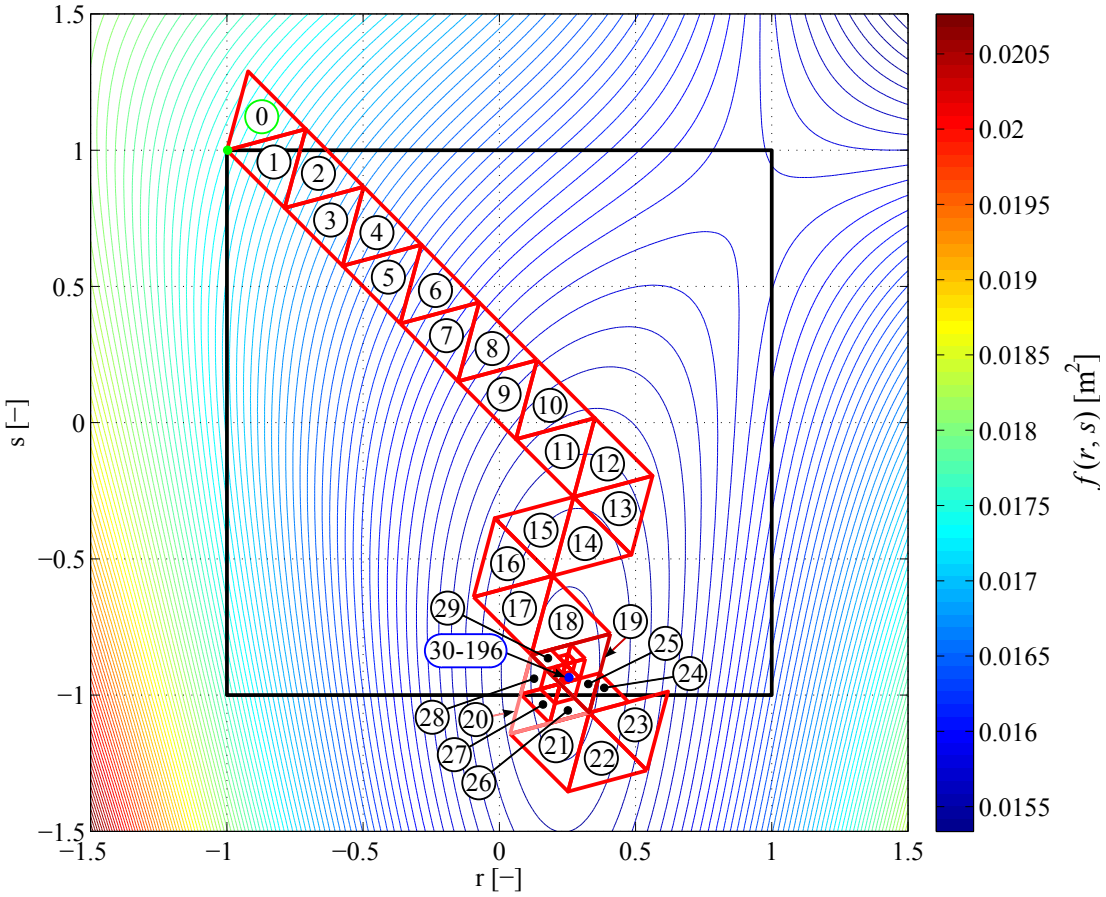

Figure 10: Iterations of the simplex method for the initial guess $(-1, 1)$.

9. Conclusion

Four numerical methods for unconstrained optimization were tested by means of numerical example which involves bending of two rectangular plates over a cylinder. All tested methods were implemented in MATLAB code allowing colour graphical outputs. The results are summarized in Tabs. 2, 3, 4 and 5. They show the numbers of iterations (NI) and the CPU time for each tested method and initial guess.

It was shown that Newton-Raphson method is not suitable for local contact search due to the strong nonlinearity of the square-distance function.

The line-search is crucial for a general success of quasi-Newton and gradient methods. It significantly increases the effectiveness of methods.

It was shown that a significant increase of stability of the local contact search can be achieved by the steepest descent method and especially by the BFGS method.

Due to the principle of the simplex method, a greater number of iterations was expected. Although several good features of this method was showed, higher time consumption handicaps this algorithm. Nevertheless, the implementation to a compiled language like FORTRAN could significantly decrease the CPU time.

In conclusion, on the base of this very good results of the BFGS method with proposed

Table 5: Results of the simplex method for all tested initial guesses.

n	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
1	$(0.241, -0.926)$	190	0.2472	1.24e-06	2.82e-03
2	$(0.241, -0.926)$	192	0.2472	1.24e-06	2.82e-03
3	$(0.241, -0.926)$	184	0.2504	1.24e-06	2.82e-03
4	$(3.649, 3.624)$	210	0.2786	7.66e-06	6.33e-03
5	$(0.241, -0.926)$	196	0.2530	1.24e-06	2.82e-03
6	$(0.241, -0.926)$	200	0.2609	1.24e-06	2.82e-03
7	$(-3.804, 3.111)$	190	0.2381	6.76e-06	2.53e-03

line-search algorithm, we selected this method for the local contact search procedure.

10. Acknowledgment

This work was supported by the Grant Agency of the Czech Republic under grant number GA101/07/1471 and GA101/09/1630 in the framework of AV0Z20760514.

11. References

Benson, D.J. & Hallquist, J.O. 1990: A Single Surface Contact Algorithm for the Post-buckling Analysis of Shell Structures. *Computer Methods in Applied Mechanics and Engineering.* vol. 78, 141-163

Gabriel, D., Plešek, J. & Ulbin, M. 2004: Symmetry preserving algorithm for large displacement frictionless contact by the pre-discetization penalty method *International Journal for Numerical Methods in Engineering.* vol. 61, 2615-2638

Nocedal J. & Wright S.J. 1999: *Numerical Optimization* Springer-Verlag, New York Berlin Heidelberg

Peressini A.L., Sulliven F.E. & Uhl J.J. 1988: *The Mathematics in Nonlinear Programming* Springer-Verlag, New York Berlin Heidelberg

Lederer P. 1988: *Teorie a optimalizace mechanických systémů I* ČVUT, Praha