CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHANICS, BIOMECHANICS AND MECHATRONICS

MASTER THESIS

Assessment of numerical procedures for determination of the penalty function in contact problems

Author: Ján Kopačka

Supervisor: Ing. Dušan Gabriel, Ph.D.

Prague, 2009

Annotation list

Author: Ján Kopačka

Title: Assessment of numerical procedures for determination of the penalty function in contact problems

Název práce: Hodnocení numerických procedur pro stanovení pokutové funkce v problémech mechaniky kontaktu

Year: 2009

Branch: Engineering Mechanics and Mechatronics

Department: Department of Mechanics, Biomechanics and Mechatronics

Supervisor: Ing. Dušan Gabriel, Ph.D.

Consultant: Ing. Jiří Plešek, CSc.

Bibliographic data: number of pages 67
number of figures 43
number of tables 10
number of appendixes 0

Keywords: local contact search, line-search methods, normal vector

Klíčová slova: lokální vyhledávání kontaktu, metody line-search, normálový vektor

Abstract In the present work we study methods for the local contact search. The methods are used to the computing of normal vector as the essential part of the penalty function. The inadvisability of the Newton-Raphson method to minimize a non-convex functions is showed on numerical example. Furthermore, the so-called quasi-Newton methods and the simplex method are presented. The efficiency of the mentioned methods is compared in the conclusion.

Anotace V předkládané práci jsou studovány algoritmy pro lokální vyhledávání kontaktu, které slouží k výpočtu normálového vektoru jakožto hlavní součásti pokutová funkce. Na numerickém příkladu je ukázána nevhodnost Newtonovy-Raphsonovy metody pro minimalizaci nekonvexních funkcí. Dále jsou představeny tzv. kvazi-newtonské medoty a simplexová metoda. V závěru porovnáváme efektivitu studovaných metod.

I declare, that I wrote my master thesis autonomously and exclusively by using the sources I have cited. I agree with lending and publishing of this work.

In Prague of the day 11th December 2009 Ján Kopačka

ACKNOWLEDGEMENTS

I am deeply indebted to my supervisor Ing. Dušan Gabriel Ph.D. for inspiration and excellent guidance throughout the creation of my master thesis. I am also very grateful to Ing. Jiří Plešek CSc. for giving me the incentive prompts.

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.

CONTENTS

NOTATION

a	length of simplex edge
$\mathbf{a}, \mathbf{b}$	real vectors
A	polynomial coefficient
b	width
$\mathbf{d}$	penetration vector
D	slope of function
$\mathbf{D}$	secant matrix
E	Young modulus
$\mathbb{E}$	Euclidean space
f	square-distance function/general function
$\mathbf{f}$	body force vector
$\mathbf{F}$	internal force vector
h	height/shape function
$\mathbf{H}$	matrix of shape functions/Hessian matrix
IG	Gauss point counter
i, j	summation counters
$\mathbf{I}$	identity matrix
$\mathbf{J}^S$	surface Jacobian
k	iteration counter
l	length/linear function
L	quadratic form
m	simplex criterion number

$\mathbf{n}$	normal vector
$\mathbf{p}$	search direction
q	surface traction/quadratic function
r	radius
$\mathbf{r}$	radius vector
r, s	isoparametric coordinates
$\mathbf{R}$	prescribed vector of external forces
$\mathbf{R}_c$	equivalent force vector of contact forces
$\mathbb{R}$	real-number space
t	step-length
$\mathbf{t}$	surface traction
$\mathbf{t}_r, \mathbf{t}_s$	tangent vectors
$\mathbf{u}$	displacement vector/unite vector
w	weighing coefficients
x, y, z	global coordinates
$\mathbf{x}$	vector of unknown variables
α, β, δ	real numbers
γ_c	contact surface of an element
Γ	boundary of domain
Γ_c	contact surface
ν	Poisson's ratio
ξ	penalty parameter
ψ	strain energy function
φ	one-dimensional restriction of a function f
Ω	open domain

CHAPTER ONE

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 discontinous curvature. The Newton-Raphson iteration scheme is often used for the solution of this minimization problem [3]. However, the method is expensive and can even diverge unless the initial guess is quite accurate.

In this work, various methods for finding the local and global minimizers of this problem was tested. First, the least-square projection constructed as the linearized Newton-Raphson scheme [3] and the method of steepest descent [9] were applied. Despite of the simplicity both methods the rate of convergence was very low. Next, an alternative approach to the Newton-Raphson method, known as quasi-Newton methods, was discussed. We considered the Broyden's method, the DFP (Davidon-Fletcher-Powell) method [9] and the most popular quasi-Newton solver for finite element applications, the

BFGS (Broyden-Fletcher-Goldfarb-Shanno) method [9, 8]. Finally, the simplex method was used. The effectiveness of different methods was tested in contact analysis using the Gauss point search algorithm [5] implemented in the FE code PMD [1].

1.1 Motivation

For testing of the stability of the local contact search algorithm, the problem of bending two rectangular plates over a cylinder was considered [5]. Two rectangular plates are being bent over a cylinder (see Fig. 1.1). The plates are loaded by uniformly distributed surface traction $q = 22.5\,\mathrm{MPa}$. Material properties are: Young modules $E = 2.1 \times 10^5\,\mathrm{MPa}$, Poisson's ratio $\nu = 0.36$ and dimensions: $r = 0.4\,\mathrm{m}$, $l = 2\,\mathrm{m}$, $h = 0.08\,\mathrm{m}$, $b = 0.6\,\mathrm{m}$. The value of the penalty parameter was set to $\xi = 10^{13}\,\mathrm{N{\cdot}m^{-3}}$. Due to triple symmetry, only one eighth was modelled using 1168 twenty-node brick elements. The FE mesh is plotted in Fig. 1.1.

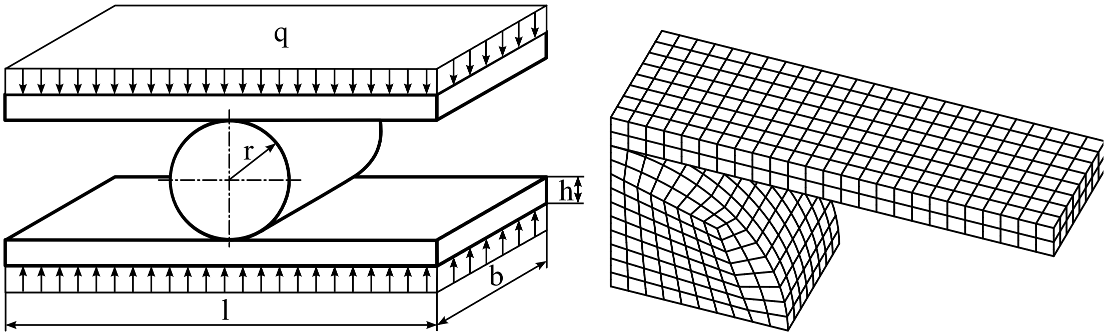


Figure 1.1: Bending two rectangular plates over a cylinder.

As already mentioned, the Newton-Raphson method has been implemented in the current version of the local contact search algorithm. In some cases, the instability of the calculation of the normal contact vector occurred. The topology of this distorted contact segment was used as the benchmark configuration for numerical tests of considered methods.

1.2 Outlines of Master Thesis

The master thesis is based on the previous work and papers. In the seminar work *The Algorithm for Computing of The Normal Contact Vector* (in Czech) the analysis of the current local contact algorithm was made (summer term 2009/2010). Some of the further work have been published in the following papers:

- J. Kopačka, D. Gabriel and J. Plešek. Review of Methods For Local Contact Search. *3rd GACM Colloquium on Computational Mechanics for Young Scientists from Academia and Industry*, Leinbiz Universität Hannover, 2009.
- J. Kopačka, D. Gabriel and J. Plešek. Review of Methods For Local Contact Search. *14th Colloquium - Computing of Constructions by Finite Element Method*, FME Brno University of Technology, 2009, (in Czech).

The master thesis is divided into five chapters. After the introduction in Chapter 1, the brief overview of the contact algorithm is presented in Chapter 2. Special attention is dedicated to the local contact search procedure in Section 2.2.

Chapter 3 is concerned with the applied methods essential for the local contact search. The objective at Section 3.1 is to establish the parametrisation of the contact surfaces. Then, the line-search technique is introduced in Section 3.3. Finally, the numerical methods for unconstrained optimization are presented in Section 3.4

The obtained results are summarised in Chapter 4. The numerical test based on the motivation example is formulated in Section 4.1. Then, achieved results by the optimisation methods are discussed.

In the closing Chapter 5, the most important results are summarized and completed by the discussion of some open questions related to further development of the present methods.

CHAPTER

TWO

OVERVIEW

2.1 Contact Algorithm

The contact algorithm for large displacement frictionless contact, designed in reference [5], was implemented into the FE code PMD [1]. It is based on the pre-discretization contact formulation which enables to perform the contact search on the element faces rather then nodes, preferably at the external Gaussian integration points. The variational formulation with the contact constraint was derived in the form

$$\int_{\Omega^0} \delta\psi\left(\mathbf{u}\right)\mathrm{d}\Omega = \int_{\Omega^0} \mathbf{f}\cdot\delta\mathbf{u}\mathrm{d}\Omega + \int_{\Gamma_S^0} \mathbf{t}\cdot\delta\mathbf{u}\mathrm{d}\Gamma + \int_{\Gamma_{c1}^k} \xi\mathbf{d}_{21}^k\cdot\delta\mathbf{u}\mathrm{d}\Gamma + \int_{\Gamma_{c2}^k} \xi\mathbf{d}_{12}^k\cdot\delta\mathbf{u}\mathrm{d}\Gamma, \quad (2.1)$$

where Ω^0 denotes the initial configuration with the boundary Γ^0, $\psi(\mathbf{u})$ is the strain energy function, $\mathbf{u}$ is the displacement field, $\mathbf{f}$ is the body force vector, $\mathbf{t}$ are the surface tractions prescribed on $\Gamma_S^0 \subset \Gamma^0$ and ξ is the penalty parameter. The iteration counter is denoted by the index k. The penetrations $\mathbf{d}_{21}$, $\mathbf{d}_{12}$ prescribed on contact boundaries Γ_{c1}^k, Γ_{c2}^k will be discussed further.

Applying the finite element method to the variational formulation (2.1), the discrete problem in the form of nonlinear equilibrium equations was obtained as

$$\mathbf{F}\left(\mathbf{u}^k\right) = \mathbf{R}\left(\mathbf{u}^k\right) + \mathbf{R}_{c1}\left(\mathbf{u}^k\right) + \mathbf{R}_{c2}\left(\mathbf{u}^k\right), \quad (2.2)$$

where $\mathbf{F}$ is the vector of internal forces, $\mathbf{R}$ is the vector of externally applied equivalent nodal point forces and

$$\begin{aligned}\mathbf{R}_{c1} &\simeq \sum_{IG=1}^{NIG} \xi_{IG} \mathbf{H}_{IG}^{T} \left[\mathbf{d}_{21}^{k}\right]_{IG} w_{IG} \det \mathbf{J}_{IG}^{S}, \\ \mathbf{R}_{c2} &\simeq \sum_{IG=1}^{NIG} \xi_{IG} \mathbf{H}_{IG}^{T} \left[\mathbf{d}_{12}^{k}\right]_{IG} w_{IG} \det \mathbf{J}_{IG}^{S},\end{aligned} \tag{2.3}$$

are the vectors of contact forces imposing the contact constraint on Γ_{c1}^{k}, Γ_{c2}^{k}, respectively. In this terms $\mathbf{H}_{IG}^{T}$ denotes the matrix of shape functions, that will be discussed in Section 3.1. The subscript IG denotes the Gaussian integration point of evaluation. Further, w_{IG} is the weighing coefficient and $\mathbf{J}_{IG}^{S}$ is the Jacobian matrix of the contact surface.

The system of nonlinear equations (2.2) is solved by the customised BFGS algorithm also proposed in [5]. Since contact boundaries are a priory unknown, they have to be determined as a part of the solution. Therefore, a contact searching procedure has to be employed to the algorithm. The purpose of this procedure is the evaluation of the vectors $\mathbf{R}_{c1}$, $\mathbf{R}_{c2}$ (2.3), specifically the values of penetrations $\mathbf{d}_{12}$, $\mathbf{d}_{21}$ determined at the Gaussian integration points. The scheme of the proposed searching algorithm is possible to find in [4]. In the consequent section we will turn attention to the local contact search, which is part of this procedure.

2.2 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 [4]). 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. 2.1). 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.

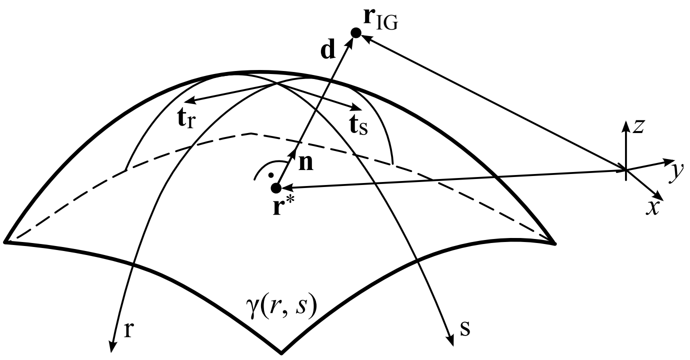


Figure 2.1: The formulation of the minimum distance problem.

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

$$\|\mathbf{r}_{IG} - \mathbf{r}^*\| \leq \|\mathbf{r}_{IG} - \mathbf{r}\|, \text{ for } \forall \mathbf{r} \in \gamma. \tag{2.4}$$

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

$$f(r,\, s) = [x_{IG} - x(r, s)]^2 + [y_{IG} - y(r, s)]^2 - [z_{IG} - z(r, s)]^2 . \tag{2.5}$$

Apparently, the desired point $\mathbf{r}^*(r, s)$ has to satisfy the necessary conditions for local extremum of (2.5)

$$\frac{\partial f}{\partial r} = 0, \qquad \frac{\partial f}{\partial s} = 0, \tag{2.6}$$

that is

$$\begin{aligned} \frac{\partial x}{\partial r} \cdot (x_{IG} - x) + \frac{\partial y}{\partial r} \cdot (y_{IG} - y) + \frac{\partial z}{\partial r} \cdot (z_{IG} - z) = 0, \\ \frac{\partial x}{\partial s} \cdot (x_{IG} - x) + \frac{\partial y}{\partial s} \cdot (y_{IG} - y) + \frac{\partial z}{\partial s} \cdot (z_{IG} - z) = 0. \end{aligned} \tag{2.7}$$

Although the function (2.5) is non-convex, and more local extrema may exist, the only local minimum is assumed in the closed domain defined by $r, s \in [-1; 1]$. The minimization of the square-distance function (2.5) leads to the problem of unconstrained optimization. Therefore, several methods will be presented in Section 3.4.

CHAPTER

THREE

APPLIED METHODS

3.1 Three-dimensional Quadrilateral Element

The geometry of contact surfaces is interpolated by the three-dimensional quadrilateral eight-node element. The coordinate interpolations are

$$x\left(r,s\right)=\sum_{i=1}^{8}h_i\left(r,s\right)x_i \quad y\left(r,s\right)=\sum_{i=1}^{8}h_i\left(r,s\right)y_i \quad z\left(r,s\right)=\sum_{i=1}^{8}h_i\left(r,s\right)z_i, \tag{3.1}$$

where $h_i\left(r,s\right),\ i=1,\ldots,8$ for $r,\ s\in[-1;1]$ are the shape functions and x_i, y_i, z_i are the coordinates of element nodes. The shape functions of eight-node quadrilateral element shown in Fig. 3.1 are expressed in the form [2]

$$\begin{aligned}
h_1 &= 0.25(1-r)(1-s)-0.5h_5-0.5h_8,\\
h_2 &= 0.25(1+r)(1-s)-0.5h_5-0.5h_6,\\
h_3 &= 0.25(1+r)(1+s)-0.5h_6-0.5h_7,\\
h_4 &= 0.25(1-r)(1+s)-0.5h_7-0.5h_8,\\
h_5 &= 0.5(1-r^2)(1-s),\\
h_6 &= 0.5(1-s^2)(1+r),\\
h_7 &= 0.5(1-r^2)(1+s),\\
h_8 &= 0.5(1-s^2)(1-r).
\end{aligned} \tag{3.2}$$

The fundamental property of a shape function h_i is that its value in the natural coordinate system is unity at node i and zero at all other nodes.

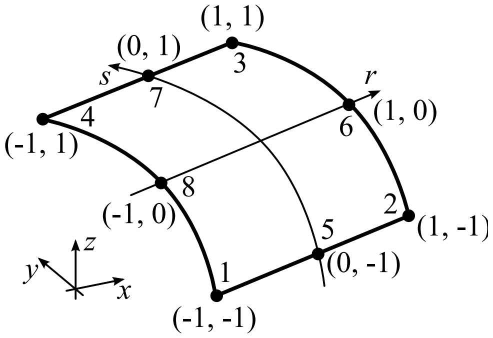


Figure 3.1: A quadrilateral element with eight nodes.

3.2 Quadratic Approximation

In this work, the approach of an approximation of a minimized function by a quadratic function is used. Therefore, in this section we will explain the basic point of this concept. To a general function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n$ with continuous second partial derivatives can be assigned a symmetric matrix

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1 \partial x_1} & \cdots & \frac{\partial f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_n \partial x_1} & \cdots & \frac{\partial f(\mathbf{x})}{\partial x_n \partial x_n} \end{bmatrix}. \tag{3.3}$$

This matrix is known as the Hessian matrix. The function $f(\mathbf{x})$ can be approximated in a point $\mathbf{x}^k$ by quadratic form. It is simply a scalar, quadratic function of a vector with the form

$$L(\mathbf{x}) = \frac{1}{2}\left(\mathbf{x} - \mathbf{x}^k\right)^T \mathbf{H}^k \left(\mathbf{x} - \mathbf{x}^k\right) - \left(\nabla f^k\right)^T \left(\mathbf{x} - \mathbf{x}^k\right) + f^k, \tag{3.4}$$

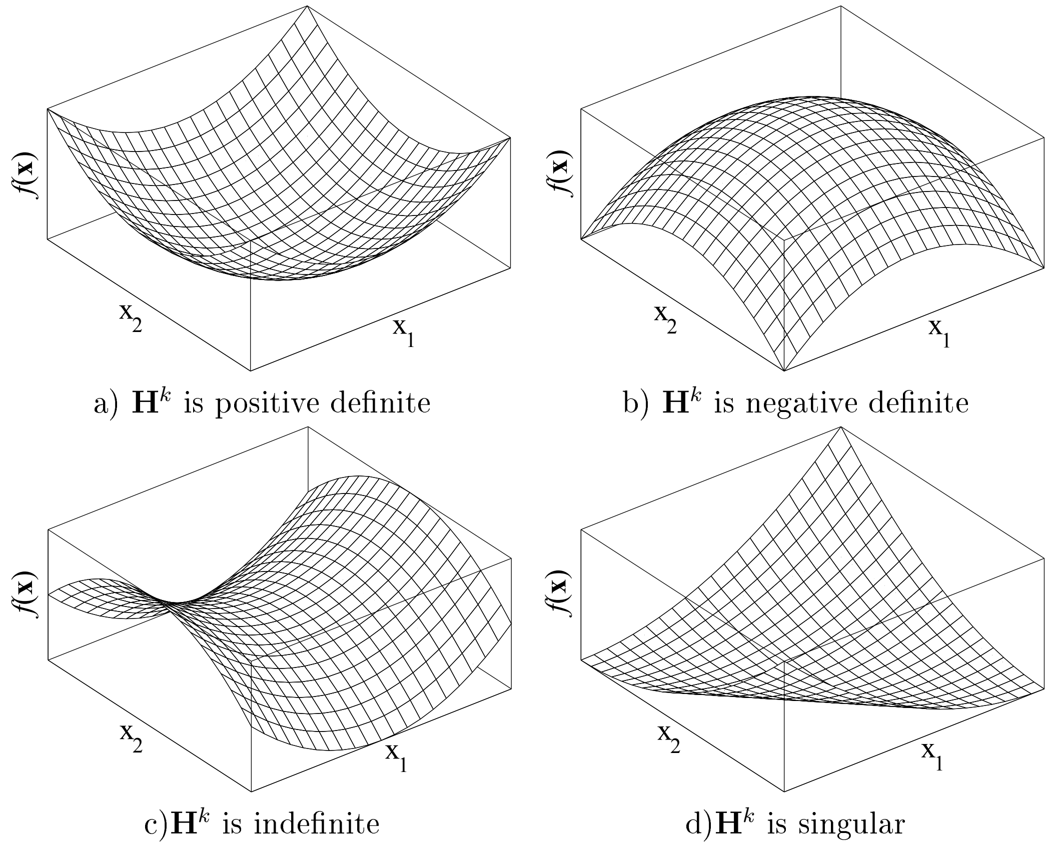
where $\mathbf{H}^k \in \mathbb{R}^{n,n}$ is the Hessian matrix, $\nabla f^k \in \mathbb{R}^n$ and $f^k \in \mathbb{R}$. The stationary point of such a quadratic function has to satisfy the necessary condition for the local extremum

$$\nabla L(\mathbf{x}) = \mathbf{H}^k\left(\mathbf{x} - \mathbf{x}^{\mathbf{k}}\right) - \nabla f^k = 0, \tag{3.5}$$

resulting the simple system of n linear equations is

$$\mathbf{H}^k \left(\mathbf{x} - \mathbf{x}^k\right) = \nabla f^k. \tag{3.6}$$

The sufficient condition for a local extremum is apparent from Fig. 3.2. If $\mathbf{H}^k$ is positive definite, then the quadratic function $L(\mathbf{x})$ has the global minimum in the stationary point. If $\mathbf{H}^k$ is negative definite, then $L(\mathbf{x})$ has the global maximum in the stationary point. If $\mathbf{H}^k$ is indefinite, there is a saddle point in the stationary point. And finally, if $\mathbf{H}^k$ is singular, the system (3.6) has infinitely solutions. The positive definiteness, the negative definiteness and the indefiniteness of the matrix $\mathbf{H}^k$ is directed by the principal minors. For detail see for example [9]. The principal minors for $\mathbf{H}^k \in R^{2,2}$ are H_{11}^k and $\det \mathbf{H}^k$. According to their sign, it is possible to recognize the type of a local extremum (see Tab. 3.1).



a) $\mathbf{H}^k$ is positive definite

b) $\mathbf{H}^k$ is negative definite

c)$\mathbf{H}^k$ is indefinite

d)$\mathbf{H}^k$ is singular

Figure 3.2: The geometrical interpretation of a two-dimensional quadratic form.

$\det \mathbf{H}^k > 0$		$\det \mathbf{H}^k < 0$	$\det \mathbf{H}^k = 0$
$H_{11}^k > 0$	$H_{11}^k < 0$	$H_{11}^k \in R$	$H_{11}^k \in R$
positive definite	negative definite	indefinite	singular
minimum	maximum	saddle point	–

Table 3.1: The positive definiteness, the negative definiteness, the indefiniteness and the singularity of the matrix $\mathbf{H}^k \in R^{2,2}$.

Apparently, unlike the quadratic form (3.2), if the Hessian matrix is evaluated in the stationary point of a general function $f(\mathbf{x})$, no global extremum, but only local one can be determined.

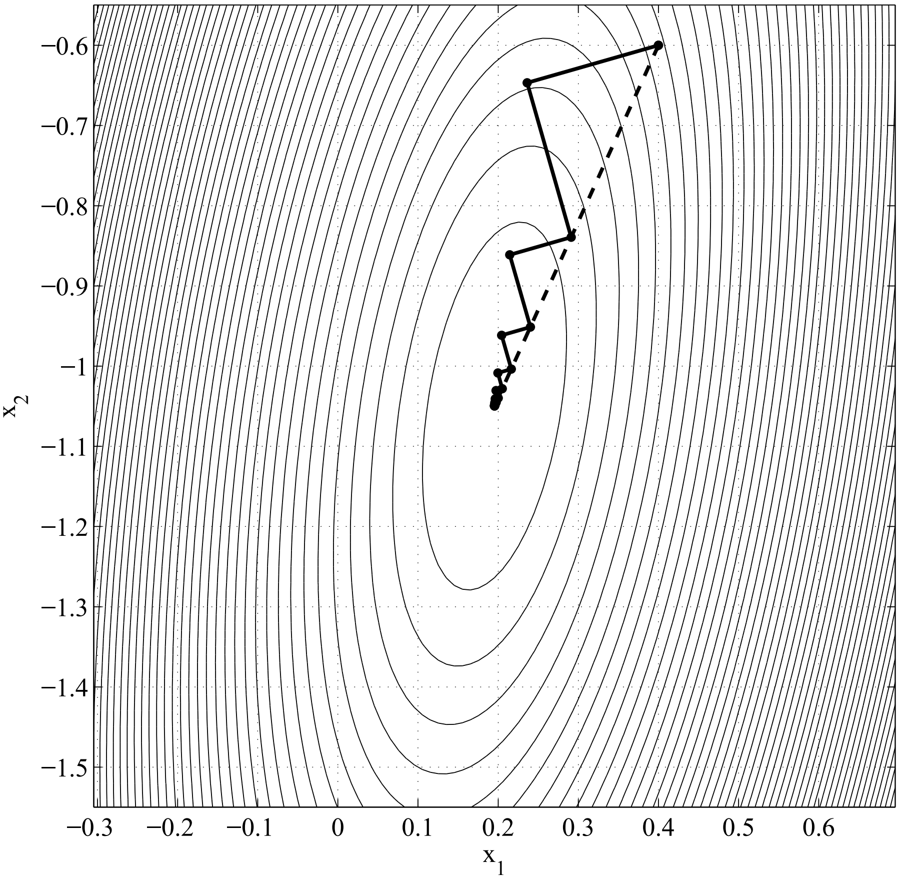


Figure 3.3: Minimization of a quadratic form by the Newton-Raphson method and the steepest descent method.

For illustration, Fig. 3.3 shows minimization of a quadratic form using the Newton-Raphson method and the steepest descent method, which will be discussed in Section 3.4. Since the steepest descent method works only with first derivatives, it has no information about curvature of a function. Note the zigzag path of the steepest descent, which appears because each gradient is orthogonal to the previous gradient [9, p. 101]. In comparison with this method, the Newton-Raphson method minimizes a quadratic form in one step (see the dash line in Fig. 3.3) because it directly solves the linear system (3.6).

3.3 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

$$\mathbf{x}^{k+1} = \mathbf{x}^k + t^k \mathbf{p}^k, \tag{3.7}$$

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

$$\mathbf{p}^k = -\mathbf{D}^k \nabla f^k, \tag{3.8}$$

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 (3.8) by ∇f^k arise

$$\mathbf{p}^k \cdot \nabla f^k = -\left(\mathbf{D}^k \nabla f^k\right) \cdot \nabla f^k < 0. \tag{3.9}$$

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 Section 3.4. We now give attention to the choice of the step-length parameter t^k. Its computation is based on the restriction of a minimized function $f(\mathbf{x})$ to the ray from a point $\mathbf{x}^k$ in the search direction $\mathbf{p}^k$

$$\varphi(t) = f\left(\mathbf{x}^k + t\mathbf{p}^k\right), \quad t > 0. \tag{3.10}$$

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 [8], more sophisticated strategies are mentioned to perform an inexact line-search to identify a step-length that achieves reductions in $f(\mathbf{x})$.

We assume that $\mathbf{p}^k$ is a descent direction, that is, $\varphi'(0) < 0$. Then, the square-distance function (2.5) can be approximated by the quadratic form (3.4). Then, the step-length can be computed analytically by the one-dimensional minimizer of the quadratic function along the ray $\mathbf{x}^k + t\mathbf{p}^k$

$$t^k = -\frac{\left(\nabla f^k\right)^T \mathbf{p}^k}{\left(\mathbf{p}^k\right)^T \mathbf{H}^k \mathbf{p}^k}. \tag{3.11}$$

Note that for a general nonlinear function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n$, especially of higher dimension n, this approach could be expensive, because the exact Hessian matrix is required in the relation (3.11). However, since the square-distance function (2.5) is the polynomial (see Section 4.1.1), the Hessian matrix is easily evaluated.

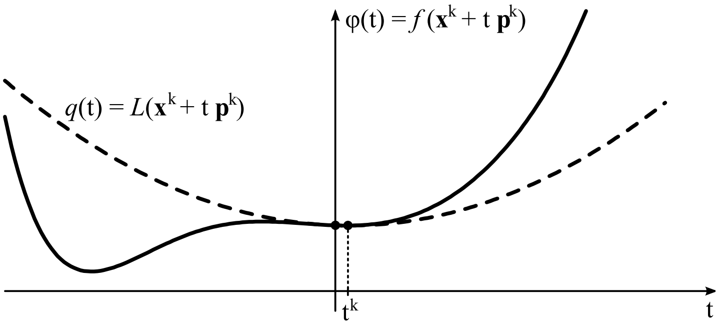


Figure 3.4: The quadratic interpolation of $\varphi(t)$.

3.4 Numerical Methods for Unconstrained Optimization

3.4.1 Newton-Raphson Method

This method has been implemented in the current version of the local contact searching algorithm [5]. The numerical scheme of this method is based on the first-order Taylor's series expansion of (2.6) about $\mathbf{x}^k$

$$\begin{aligned}
\frac{\partial f\left(r,\, s\right)}{\partial r} &\simeq \left.\frac{\partial f}{\partial r}\right|_{\mathbf{x}^k} + \left.\frac{\partial^2 f}{\partial r^2}\right|_{\mathbf{x}^k} \left(r^{k+1} - r^k\right) + \left.\frac{\partial^2 f}{\partial r \partial s}\right|_{\mathbf{x}^k} \left(s^{k+1} - s^k\right) = 0, \\
\frac{\partial f\left(r,\, s\right)}{\partial s} &\simeq \left.\frac{\partial f}{\partial s}\right|_{\mathbf{x}^k} + \left.\frac{\partial^2 f}{\partial r \partial s}\right|_{\mathbf{x}^k} \left(r^{k+1} - r^k\right) + \left.\frac{\partial^2 f}{\partial s^2}\right|_{\mathbf{x}^k} \left(s^{k+1} - s^k\right) = 0.
\end{aligned} \tag{3.12}$$

Written in matrix form, this becomes

$$\begin{bmatrix} \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r^2} & \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r \partial s} \\ \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r \partial s} & \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial s^2} \end{bmatrix} \cdot \begin{bmatrix} r^{k+1} - r^k \\ s^{k+1} - s^k \end{bmatrix} = - \begin{bmatrix} \dfrac{\partial f\left(r^k,\, s^k\right)}{\partial r} \\ \dfrac{\partial f\left(r^k,\, s^k\right)}{\partial s} \end{bmatrix}. \tag{3.13}$$

Denote the Hessian matrix

$$\mathbf{H}^k = \begin{bmatrix} \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r^2} & \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r \partial s} \\ \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial r \partial s} & \dfrac{\partial^2 f\left(r^k,\, s^k\right)}{\partial s^2} \end{bmatrix}, \tag{3.14}$$

the vector of isoparametric coordinates

$$\mathbf{x}^k = \begin{bmatrix} r^k \\ s^k \end{bmatrix}, \qquad \mathbf{x}^{k+1} = \begin{bmatrix} r^{k+1} \\ s^{k+1} \end{bmatrix} \tag{3.15}$$

and the gradient

$$\nabla f^k = \begin{bmatrix} \dfrac{\partial f\left(r^k,\, s^k\right)}{\partial r} \\ \dfrac{\partial f\left(r^k,\, s^k\right)}{\partial s} \end{bmatrix}. \tag{3.16}$$

Finally, the numerical scheme of the Newton-Raphson method can be written in the form

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \left(\mathbf{H}^k\right)^{-1} \nabla f^k. \tag{3.17}$$

A Hessian matrix $\mathbf{H}^k$ could be substituted into the quadratic form (3.4). The consequent iteration $\mathbf{x}^{k+1}$ lies in the computation of stationary point of this quadratic form. In fact, the scheme of the Newton-Raphson method is also possible to obtain by differentiation of the relation (3.4).

3.4.2 Least-square Projection

Whereas the Newton-Raphson method is based on the linear approximation of equations (2.7), the least-square projection (LSP) method constructs the tangent plane τ in the point $\mathbf{x}^k$ (see Fig. 3.5), i.e. linear approximation of equations (3.1)

$$\begin{aligned}
x(r,s) &\simeq x|_{\mathbf{x}^k} + \left.\frac{\partial x}{\partial r}\right|_{\mathbf{x}^k} \left(r^{k+1} - r^k\right) + \left.\frac{\partial x}{\partial s}\right|_{\mathbf{x}^k} \left(s^{k+1} - s^k\right), \\
y(r,s) &\simeq y|_{\mathbf{x}^k} + \left.\frac{\partial y}{\partial r}\right|_{\mathbf{x}^k} \left(r^{k+1} - r^k\right) + \left.\frac{\partial y}{\partial s}\right|_{\mathbf{x}^k} \left(s^{k+1} - s^k\right), \qquad (3.18) \\
z(r,s) &\simeq z|_{\mathbf{x}^k} + \left.\frac{\partial z}{\partial r}\right|_{\mathbf{x}^k} \left(r^{k+1} - r^k\right) + \left.\frac{\partial z}{\partial s}\right|_{\mathbf{x}^k} \left(s^{k+1} - s^k\right).
\end{aligned}$$

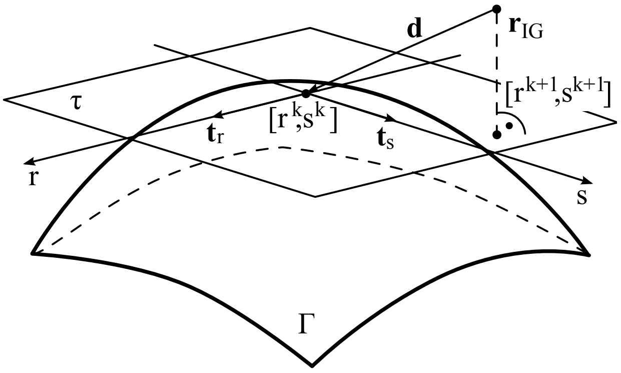


Figure 3.5: The principle of the LSP method.

The minimum distance problem between the Gaussian integration point $\mathbf{r}_{IG}$ and the tangent plane τ leads to the same equations like (2.7). Since equations (3.18) are linear, its partial derivations are constants and equations (2.7) are linear now as well. Let us introduce vectors

$$\begin{aligned}
\mathbf{d}^k &= \left.\{x_{IG}-x,\, y_{IG}-y,\, z_{IG}-z\}\right|_{\mathbf{x}^k}, \\
\mathbf{t}_r^k &= \left.\left\{\frac{\partial x}{\partial r}, \frac{\partial y}{\partial r}, \frac{\partial z}{\partial r}\right\}\right|_{\mathbf{x}^k}, \\
\mathbf{t}_s^k &= \left.\left\{\frac{\partial x}{\partial s}, \frac{\partial y}{\partial s}, \frac{\partial z}{\partial s}\right\}\right|_{\mathbf{x}^k}.
\end{aligned} \tag{3.19}$$

Substituting relations (3.18) into (2.7) and using vectors (3.19) we obtain the numerical scheme

$$\begin{bmatrix} \mathbf{t}_r^k \cdot \mathbf{t}_r^k & \mathbf{t}_r^k \cdot \mathbf{t}_s^k \\ \\ \mathbf{t}_r^k \cdot \mathbf{t}_s^k & \mathbf{t}_s^k \cdot \mathbf{t}_s^k \end{bmatrix} \cdot \begin{bmatrix} r^{k+1} \\ \\ s^{k+1} \end{bmatrix} = - \begin{bmatrix} \mathbf{t}_r^k \cdot \mathbf{d}^k \\ \\ \mathbf{t}_s^k \cdot \mathbf{d}^k \end{bmatrix}. \tag{3.20}$$

3.4.3 Method of Steepest Descent

The principle of this method is a simple consequence of the chain rule [10, p. 105]. Let us restrict the function $f(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ to the ray from a given point $\mathbf{x}^0 \in \mathbb{R}^n$ in a given direction of unit vector $\mathbf{u} \in \mathbb{R}^n$

$$f(t) = f\left(\mathbf{x}^0 + t\mathbf{u}\right), \quad t \geq 0. \tag{3.21}$$

By the chain rule

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = \nabla f\left(\mathbf{x}^0 + t\mathbf{u}\right) \cdot \mathbf{u}, \tag{3.22}$$

so that

$$\frac{\mathrm{d}f(0)}{\mathrm{d}t} = \nabla f\left(\mathbf{x}^0\right) \cdot \mathbf{u} \tag{3.23}$$

measures the rate of change of $f(\mathbf{x})$ at $\mathbf{x}^0$ in the direction of $\mathbf{u}$. For this reason, $\frac{\mathrm{d}f(0)}{\mathrm{d}t}$ is usually called the directional derivative of $f(\mathbf{x})$ at $\mathbf{x}^0$ in the direction $\mathbf{u}$ and is denoted by

$$\nabla_{\boldsymbol{u}} f\left(\mathbf{x}^0\right). \tag{3.24}$$

It is apparent from the name of the method, that the steepest descent of $f(\mathbf{x})$ in the direction $\mathbf{u}$ is demanded. By the Cauchy-Schwartz Inequality [9, p. 98]

$$\begin{aligned} -\left\|\nabla f\left(\mathbf{x}^0\right)\right\| &= -\left\|\nabla f\left(\mathbf{x}^0\right)\right\| \cdot \|\mathbf{u}\| \leq \nabla f\left(\mathbf{x}^0\right) \cdot \mathbf{u} = \nabla f_{\mathbf{u}}\left(\mathbf{x}^0\right) \cdot \mathbf{u} \\ &\leq \left\|\nabla f\left(\mathbf{x}^0\right)\right\| \cdot \|\mathbf{u}\| = \left\|\nabla f\left(\mathbf{x}^0\right)\right\|. \end{aligned} \tag{3.25}$$

Consequently, the directional derivative $\nabla_{\boldsymbol{u}} f\left(\mathbf{x}^0\right)$ is as negative as possible when there is equality in the leftmost inequality in (3.25). This happens when $\mathbf{u}$ is the unit vector

$$\mathbf{u} = -\frac{\nabla f\left(\mathbf{x}^0\right)}{\|\nabla f\left(\mathbf{x}^0\right)\|}. \tag{3.26}$$

Thus, the vector $\mathbf{u}$ is identical with the normalized negative gradient $\nabla f(\mathbf{x})$. Now, the method of steepest descent can be formulated as follows. Suppose that $f(\mathbf{x})$ is a function with continuous first partial derivatives on $\mathbb{R}^n$ and that $\mathbf{x}^0 \in \mathbb{R}^n$. Then the steepest descent sequence $\left\{\mathbf{x}^k\right\}$ with the initial point $\mathbf{x}^0$ for minimizing $f(\mathbf{x})$ is defined by the recurrence formula

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \nabla f^k, \tag{3.27}$$

where the step-length t^k is computed by the line-search procedure (see Section 3.3). In the context of the relations (3.7) and (3.8), the steepest descent method is the easiest line-search method for which $\mathbf{D}^k$ is the identity matrix and the search direction $\mathbf{p}^k$ is the negative gradient.

3.4.4 Broyden's Method

Broyden's method is a generalization of the secant method to multiple dimensions as well as the Newton-Raphson is a generalization of Newton's tangent method. Therefore, let us begin with the secant method. The principle is depicted in Fig. 3.6.

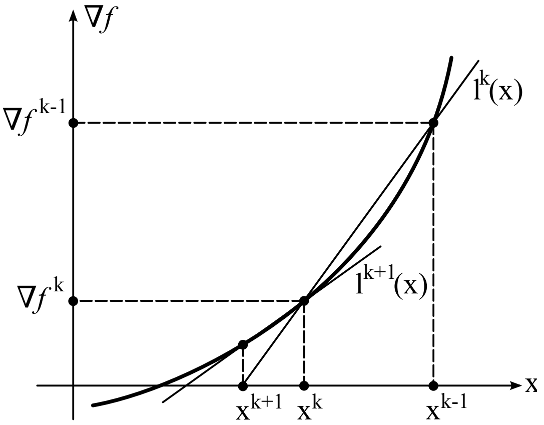


Figure 3.6: The geometrical interpretation of the secant method.

The slope D^k of the secant in the kth iteration can be computed with the aid of the 1D secant condition

$$D^k \left(x^k - x^{k-1} \right) = \nabla f^k - \nabla f^{k-1}. \tag{3.28}$$

Then, the linear approximation

$$l^k \left(x \right) = \nabla f^k + D^k \left(x - x^k \right) \tag{3.29}$$

passes through $\left(x^{k-1}, \nabla f^{k-1} \right)$ and $\left(x^k, \nabla f^k \right)$, and the new point x^{k+1} is obtained from the solution of $l^k \left(x \right) = 0$. Next, we construct new secant D^{k+1} and the process is repeated. This basic feature of the secant method in one dimension is retained for Broyden's method in multiple dimensions. We insist that the matrix $\mathbf{D}^{k+1}$ is selected so that the nD secant condition

$$\mathbf{D}^{k+1} \left(\mathbf{x}^{k+1} - \mathbf{x}^k \right) = \nabla f^{k+1} - \nabla f^k \tag{3.30}$$

is satisfied. Note that this equation is under determined in more than one dimension, since it prescribes only one image vector for the matrix transformation $\mathbf{D}^{k+1}$ on $\mathbb{R}^n$. The intention of Broyden's method is to make the matrix $\mathbf{D}^{k+1}$ as simple to compute as possible. One way to achieve the desired simplicity is to define this matrix in such a way that it is completely defined by prescribing two vectors in $\mathbb{R}^n$. The update of $\mathbf{D}^k$ was chosen to be rank-one matrix of the form

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \mathbf{a}^k \otimes \mathbf{b}^k, \tag{3.31}$$

where the symbol $\otimes$ denotes the outer product or the tensor product. Before the determination of the vectors $\mathbf{a}^k$ and $\mathbf{b}^k$ let us recall the identity [9, p. 115]

$$\left(\mathbf{a}^k \otimes \mathbf{b}^k\right) \mathbf{x}^k = \left(\mathbf{b}^k \cdot \mathbf{x}^k\right) \mathbf{a}^k, \tag{3.32}$$

where $\mathbf{a}^k$, $\mathbf{b}^k$ and $\mathbf{x}^k$ are column vectors. The substitution of (3.31) into the secant condition (3.30) yields

$$\left(\mathbf{D}^k + \mathbf{a}^k \otimes \mathbf{b}^k\right) \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right) = \nabla f^{k+1} - \nabla f^k. \tag{3.33}$$

Rewrite this equation with the aid of (3.32) as

$$\left[\mathbf{b}^k \cdot \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right)\right] \mathbf{a}^k = \nabla f^{k+1} - \nabla f^k - \mathbf{D}^k \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right). \tag{3.34}$$

It follows that once $\mathbf{b}^k$ has been chosen, then $\mathbf{a}^k$ is determined by the equation

$$\mathbf{a}^k = \frac{\nabla f^{k+1} - \nabla f^k - \mathbf{D}^k \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right)}{\mathbf{b}^k \cdot \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right)}. \tag{3.35}$$

In the reference [9, p. 116], it can be found the relation which presents the difference between the linear approximations (3.29)

$$l^{k+1}(\mathbf{x}) - l^k(\mathbf{x}) = \left[\mathbf{b}^k \cdot \left(\mathbf{x} - \mathbf{x}^k\right)\right] \mathbf{a}^k. \tag{3.36}$$

Broyden designed that whenever the vector $\mathbf{x} - \mathbf{x}^k$ is orthogonal to $\mathbf{x}^{k+1} - \mathbf{x}^k$, then

$$l^{k+1}(\mathbf{x}) - l^k(\mathbf{x}) = 0. \tag{3.37}$$

It can be verified by direct substitution into (3.36) that if

$$\mathbf{b}^k = \mathbf{x}^{k+1} - \mathbf{x}^k, \tag{3.38}$$

then the condition (3.37) is satisfied. Finally, the terms (3.35), (3.38) for $\mathbf{a}^k$, $\mathbf{b}^k$ respectively determine the update (3.31) of $\mathbf{D}^k$.

Suppose that $f(\mathbf{x})$ is a function with continuous second partial derivatives on $\mathbb{R}^n$, that $\mathbf{x}^0 \in \mathbb{R}^n$ and $\mathbf{D}^0 \in \mathbb{R}^{n,n}$. Then the Broyden's method sequence $\left\{\mathbf{x}^k\right\}$ is defined by the recurrence formula

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \left(\mathbf{D}^k\right)^{-1} \nabla f^k, \tag{3.39}$$

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

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\left(\nabla f^{k+1} - \nabla f^k\right) - \mathbf{D}^k \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right)}{\left(\mathbf{x}^{k+1} - \mathbf{x}^k\right) \cdot \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right)} \otimes \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right). \tag{3.40}$$

3.4.5 BFGS Method

In this section we will present very effective minimization method, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. This method requires no evaluation of the Hessian matrix and retains the secant feature of Broyden's method. Moreover, the BFGS method was developed so that the search direction $-\left(\mathbf{D}^k\right)^{-1} \nabla f^k$ at $\mathbf{x}^k$ is always a descent direction because there is guarantee that $\mathbf{D}^k$ is positive definite.

The rank-one update $\mathbf{a}^k \otimes \mathbf{b}^k$ is not symmetric unless $\mathbf{a}^k$, $\mathbf{b}^k$ are positive multiples of one another, and $\mathbf{a}^k \otimes \mathbf{b}^k$ is indefinite unless $\mathbf{a}^k$, $\mathbf{b}^k$ are positive multiples of one another [9, p. 124]. Thus, if the positive definiteness of $\mathbf{D}^k$ is insisted, then $\mathbf{a}^k = \alpha^k \mathbf{b}^k$ for $\alpha^k > 0$, $\alpha^k \in \mathbb{R}$ must be valid. But now there is no enough flexibility to obtain the secant condition. This difficulty is overcome by rank-two update

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \alpha^k \left(\mathbf{a}^k \otimes \mathbf{a}^k\right) + \beta^k \left(\mathbf{b}^k \otimes \mathbf{b}^k\right), \tag{3.41}$$

where α^k, $\beta^k \in \mathbb{R}$ and $\mathbf{a}^k$, $\mathbf{b}^k \in \mathbb{R}^n$. The secant condition (3.30) forces

$$\left[\mathbf{D}^k + \alpha^k \left(\mathbf{a}^k \otimes \mathbf{a}^k\right) + \beta^k \left(\mathbf{b}^k \otimes \mathbf{b}^k\right)\right] \left(\mathbf{x}^{k+1} - \mathbf{x}^k\right) = \nabla f^{k+1} - \nabla f^k. \tag{3.42}$$

Let us impose the vectors $\mathbf{d}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$, $\mathbf{y}^k = \nabla f^{k+1} - \nabla f^k$ and rewrite the last term with the aid of the identity (3.32) as

$$\mathbf{D}^k \mathbf{d}^k + \alpha^k \left(\mathbf{a}^k \cdot \mathbf{d}^k\right) \mathbf{a}^k + \beta^k \left(\mathbf{b}^k \cdot \mathbf{d}^k\right) \mathbf{b}^k = \mathbf{y}^k. \tag{3.43}$$

If we set $\mathbf{a}^k = \mathbf{y}^k$ and $\mathbf{b}^k = \mathbf{D}^k \mathbf{d}^k$, then the preceding equation yields

$$\mathbf{D}^k\mathbf{d}^k + \alpha^k\left(\mathbf{y}^k \cdot \mathbf{d}^k\right)\mathbf{y}^k + \beta^k\left(\mathbf{D}^k\mathbf{d}^k \cdot \mathbf{d}^k\right)\mathbf{D}^k\mathbf{d}^k = \mathbf{y}^k. \tag{3.44}$$

Apparently, this equation is satisfied if we set

$$\begin{aligned} \alpha^k\left(\mathbf{y}^k \cdot \mathbf{d}^k\right) &= 1, \\ \beta^k\left(\mathbf{D}^k\mathbf{d}^k \cdot \mathbf{d}^k\right) &= -1, \end{aligned} \tag{3.45}$$

that is,

$$\begin{aligned} \alpha^k &= \frac{1}{\mathbf{y}^k \cdot \mathbf{d}^k}, \\ \beta^k &= -\frac{1}{\mathbf{D}^k\mathbf{d}^k \cdot \mathbf{d}^k}. \end{aligned} \tag{3.46}$$

The update (3.41) is completely defined now. Suppose that $f(\mathbf{x})$ is a function with continuous second partial derivatives on $\mathbb{R}^n$. To minimize $f(\mathbf{x})$, select an initial point $\mathbf{x}^0 \in \mathbb{R}^n$ and an initial positive definite matrix $\mathbf{D}^0 \in \mathbb{R}^{n,n}$. Then the BFGS method sequence $\left\{\mathbf{x}^k\right\}$ is defined by the recurrence formula

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k\left(\mathbf{D}^k\right)^{-1}\nabla f^k. \tag{3.47}$$

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

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\mathbf{y}^k \otimes \mathbf{y}^k}{\mathbf{d}^k \cdot \mathbf{y}^k} - \frac{\mathbf{D}^k\mathbf{d}^k \otimes \mathbf{D}^k\mathbf{d}^k}{\mathbf{d}^k \cdot \mathbf{D}^k\mathbf{d}^k}, \tag{3.48}$$

where $\mathbf{d}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$ and $\mathbf{y}^k = \nabla f^{k+1} - \nabla f^k$. Since the inversion of $\mathbf{D}^k$ is required in the recurrence formula (3.47), the update (3.48) is usually rewritten by the Sherman-Morrison-Woodbury formula [8] in form

$$\left(\mathbf{D}^{k+1}\right)^{-1} = \frac{\mathbf{y}^k \otimes \mathbf{y}^k}{\mathbf{d}^k \cdot \mathbf{y}^k} + \left(I - \frac{\mathbf{d}^k \otimes \mathbf{y}^k}{\mathbf{d}^k \cdot \mathbf{y}^k}\right)\left(\mathbf{D}^k\right)^{-1}\left(I - \frac{\mathbf{d}^k \otimes \mathbf{y}^k}{\mathbf{d}^k \cdot \mathbf{y}^k}\right). \tag{3.49}$$

However, there is no problem with the inversion of $\mathbf{D}^k$ in the local contact search, since $\mathbf{D}^k \in \mathbb{R}^{2,2}$. The initial matrix $\mathbf{D}^0$ is often set to some multiple $\beta\mathbf{I}$ of the identity matrix, but there is no good general strategy for choosing β [8]. If $\beta$is too large, then first step $\mathbf{p}^0 = -\beta\nabla f^0$ is too long. Some software asks the user to prescribe a value δ for the norm of the first step, and then set $\mathbf{D}^0 = \delta\left\|\nabla f^0\right\| \cdot \mathbf{I}$. We will use this concept with $\delta = 1$ for all quasi-Newton methods.

3.4.6 DFP Method

Another famous method is the Davidon-Fletche-Powell (DFP) method. Instead of updating the matrices $\mathbf{D}^k$ in

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \left(\mathbf{D}^k\right)^{-1} \nabla f^k \tag{3.50}$$

as in the BFGS method, the DFP method updates their inverses and yet retains the features of a secant method. The inverse secant condition is

$$\mathbf{x}^{k+1} - \mathbf{x}^k = \mathbf{D}^{k+1} \left(\nabla f^{k+1} - \nabla f^k\right). \tag{3.51}$$

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

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \alpha^k \left(\mathbf{a}^k \otimes \mathbf{a}^k\right) + \beta^k \left(\mathbf{b}^k \otimes \mathbf{b}^k\right), \tag{3.52}$$

where α^k, $\beta^k \in \mathbb{R}$ and $\mathbf{a}^k$, $\mathbf{b}^k \in \mathbb{R}^n$ will be determined by the substitution of the last term into the inverse secant condition (3.51)

$$\mathbf{d}^k = \mathbf{D}^k \mathbf{y}^k + \alpha^k \left(\mathbf{a}^k \cdot \mathbf{y}^k\right) \mathbf{a}^k + \beta^k \left(\mathbf{b}^k \cdot \mathbf{y}^k\right) \mathbf{b}^k, \tag{3.53}$$

where $\mathbf{d}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$ and $\mathbf{y}^k = \nabla f^{k+1} - \nabla f^k$. If we set $\mathbf{a}^k = \mathbf{d}^k$ and $\mathbf{b}^k = \mathbf{D}^k \mathbf{y}^k$, then the equation (3.53) is satisfied for

$$\begin{aligned} \alpha^k &= \frac{1}{\mathbf{d}^k \cdot \mathbf{y}^k}, \\ \beta^k &= -\frac{1}{\mathbf{y}^k \cdot \mathbf{D}^k \mathbf{y}^k}. \end{aligned} \tag{3.54}$$

Finally, suppose that $f(\mathbf{x})$ is a function with continuous second partial derivatives on $\mathbb{R}^n$. To minimize $f(\mathbf{x})$, select an initial point $\mathbf{x}^0 \in \mathbb{R}^n$ and an initial positive definite matrix $\mathbf{D}^0 \in \mathbb{R}^{n,n}$. Then the DFP method sequence $\left\{\mathbf{x}^k\right\}$ is defined by the recurrence formula

$$\mathbf{x}^{k+1} = \mathbf{x}^k - t^k \mathbf{D}^k \nabla f^k. \tag{3.55}$$

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

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\mathbf{d}^k \otimes \mathbf{d}^k}{\mathbf{y}^k \cdot \mathbf{d}^k} - \frac{\mathbf{D}^k\mathbf{y}^k \otimes \mathbf{D}^k\mathbf{y}^k}{\mathbf{y}^k \cdot \mathbf{D}^k\mathbf{y}^k}, \tag{3.56}$$

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

3.4.7 Simplex Method

In geometry, a simplex is a generalization of the motion of a triangle or tetrahedron to arbitrary dimension. Specifically, an n-simplex is an n-dimensional polytope with $n+1$ vertices whereas the distance between each of them is equal. Examples of such a simplex is depicted in Fig. 3.7.

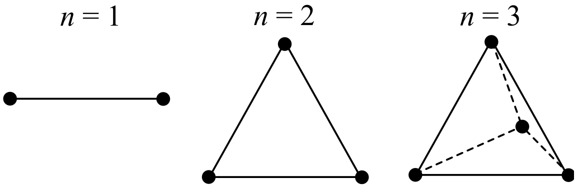


Figure 3.7: Examples of simplexes.

The algorithm of the simplex method consists of three rules [6]. The minimized function is evaluated in all vertices. The first rule says that the vertex with the maximal function value is released. Instead, it is replaced by the new one according to Fig. 3.8

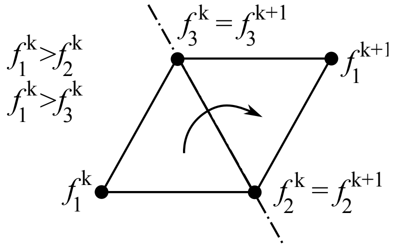


Figure 3.8: First rule of the simplex method.

In case the function value in the new vertex is maximal again, there is the second rule (see Fig. 3.9). 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.

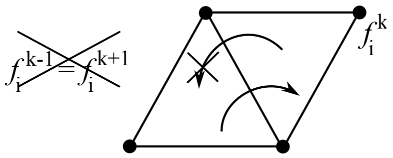


Figure 3.9: Second rule of the simplex method.

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 [6] the number m is recommended to select

$$m = 1.65n + 0.05n^2, \tag{3.57}$$

where n is the dimension. Fig. 3.10 shows such a case for $n = 2$.

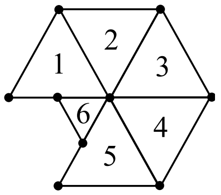


Figure 3.10: Third rule of the simplex method.

In this place, we would like to mention the Nelder-Mead method [7]. The authors proposed several improvement of the simplex method. Whereas in the classic simplex method the distance between each vertices is identical, the Nelder-Mead method performs expansion or contraction of the simplex. It allows an increase of the convergence rate.

CHAPTER

FOUR

RESULTS

4.1 Numerical Example

The effectiveness of the methods, discussed in previous chapter, was tested by means of a numerical test, which involves example introduced in Section 1.1. Fig. 4.1 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.

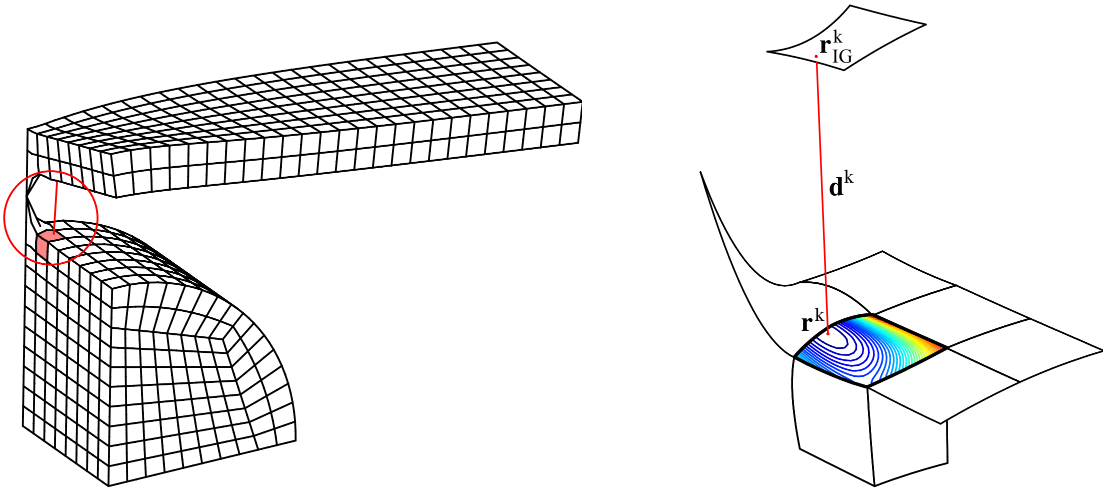


Figure 4.1: Deformed shape of the plate at that moment the instability of the calculation of the normal vector occurred.

Furthermore, the detail of distorted master segment is plotted in Fig. 4.1, where contours of the square-distance function (2.5) 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. 4.2 depicts the contours of the square-distance function (2.5) for the isoparametric coordinates $r, s \in [-5, 5]$.

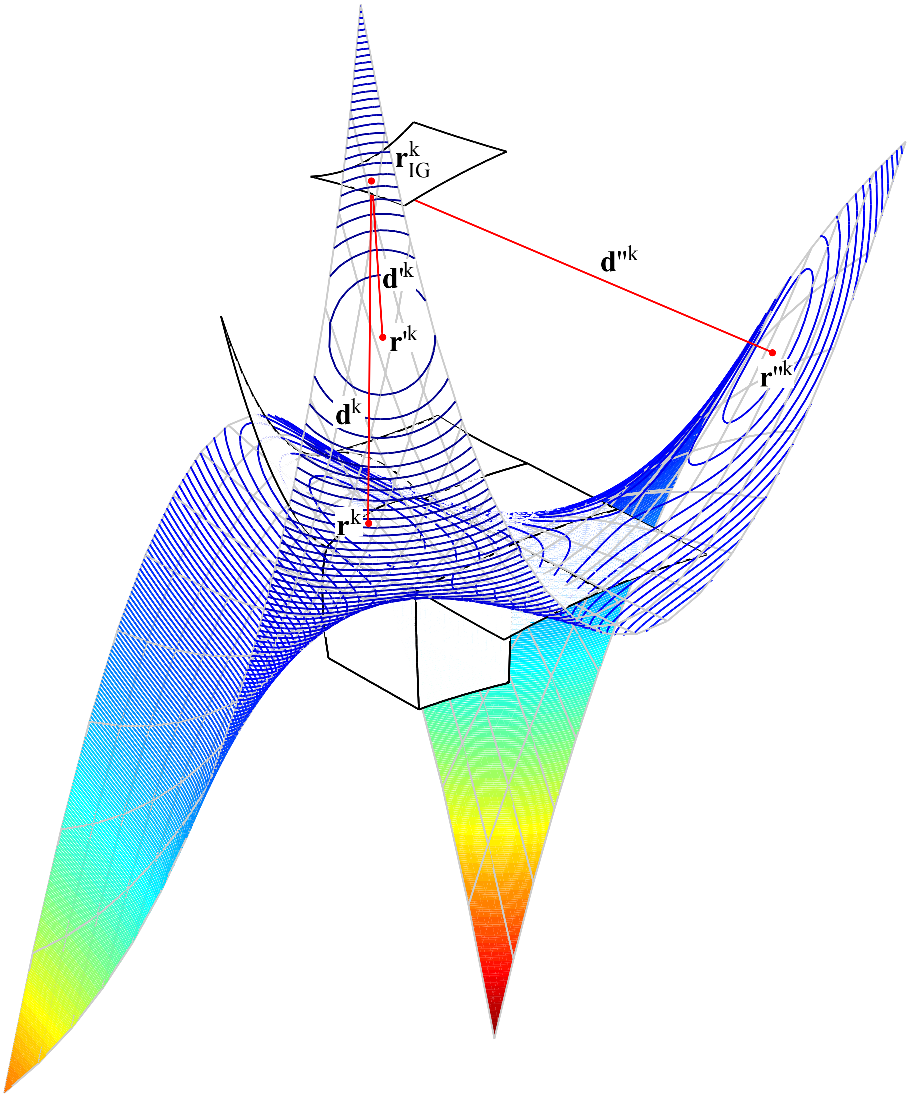


Figure 4.2: The isoparametric interpolation of the master segment for r, s $\in$ $[-5, 5]$ including the contours of square-distance function.

The parametrization (3.1) interpolates the contact surface just on closed domain defined by r, $s \in [-1, 1]$. It is clear that the contact area beyond the domain r, $s \in [-1, 1]$ is not approximated well by the parametrization (3.1) . Therefore we are interested only in local minima which both coordinate belong to interval $[-1, 1]$. Fig. 4.3 shows three possible solutions $\mathbf{r}^k$, $\mathbf{r}'^k$, $\mathbf{r}''^k$ of the nonlinear system (2.7) plotted in the three dimensional space. Although the square-distance function (2.5) 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 $r, s \in [-1, 1]$. Note that there are two saddle points in Fig. 4.3, which also fulfils the necessary condition for local extremum (2.7).

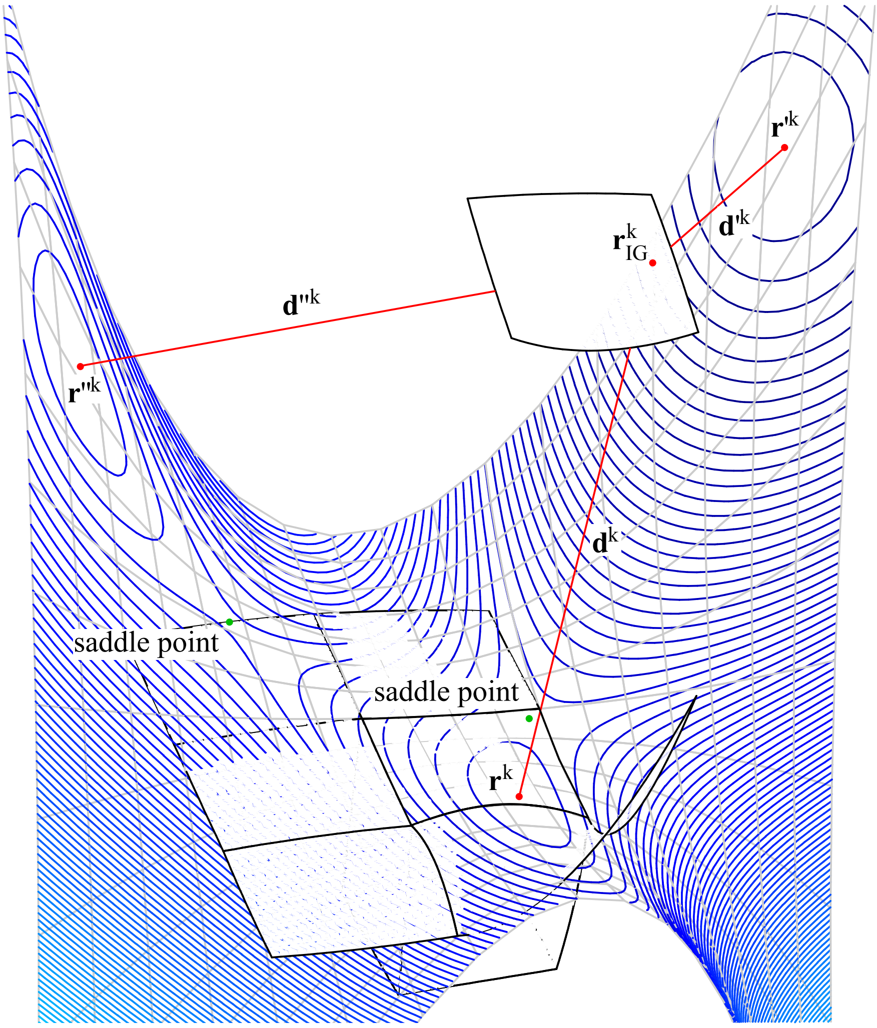


Figure 4.3: Different point of view on the isoparametric interpolation of the master segment for r, $s \in [-5, 5]$.

From above discussion arises these requirements on methods for calculating the normal contact vector:

- convergence to the nearest local minimum,
- neglect of the saddle points,
- fast rate of convergence.

4.1.1 Square-distance Function

In the optimization theory a function to be minimized is called the objective function. Although the methods presented in Chapter 3.4 belong to the class of iterative methods for unconstrained optimization, we will prefer the term square-distance function to the objective function.

If the coordinate interpolations (3.1), the shape functions (3.2) and the coordinates of element nodes are substituted into (2.5), the square-distance function has the following form

$$\begin{aligned} f\left(r,s\right) &= \sum_{i=1}^{3} A_{1,i}r^4s^2 + A_{2,i}r^3s^3 + A_{3,i}r^2s^4 + A_{4,i}r^4s + A_{5,i}r^3s^2 \\ &+ A_{6,i}r^2s^3 + A_{7,i}rs^4 + A_{8,i}r^4 + A_{9,i}r^3s + A_{10,i}r^2s^2 + A_{11,i}rs^3 \\ &+ A_{12,i}s^4 + A_{13,i}r^3 + A_{14,i}r^2s + A_{15,i}rs^2 + A_{16,i}s^3 + A_{17,i}r^2 \\ &+ A_{18,i}rs + A_{19,i}s^2 + A_{20,i}r + A_{21,i}s + A_{22,i}, \end{aligned} \tag{4.1}$$

where $A_{j,i}\left(x_i, y_i, z_i\right)$, $j = 1, \ldots, 22$ are new polynomial coefficients. Note that function (4.1) is a polynomial of sixth order in two variables r, s. Whereas in Figs. 4.2, 4.3 the 3D contours of this function are plotted over a given topology of segment. Fig. 4.4 represents 2D contours of the function (4.1) for the isoparametric coordinates r, s. The boundaries of the master segment are denoted by the black square in this figure.

4.1.2 Initial Guess

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. 4.4, were considered. Their coordinates are showed in Tab. 4.1. 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 LSP method discussed in Section 3.4.2 for $r^0 = s^0 = 0$. 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 $\mathbf{r}''^k$ (see Fig. 4.3). The Hessian matrix is not positive definite in this initial guess.

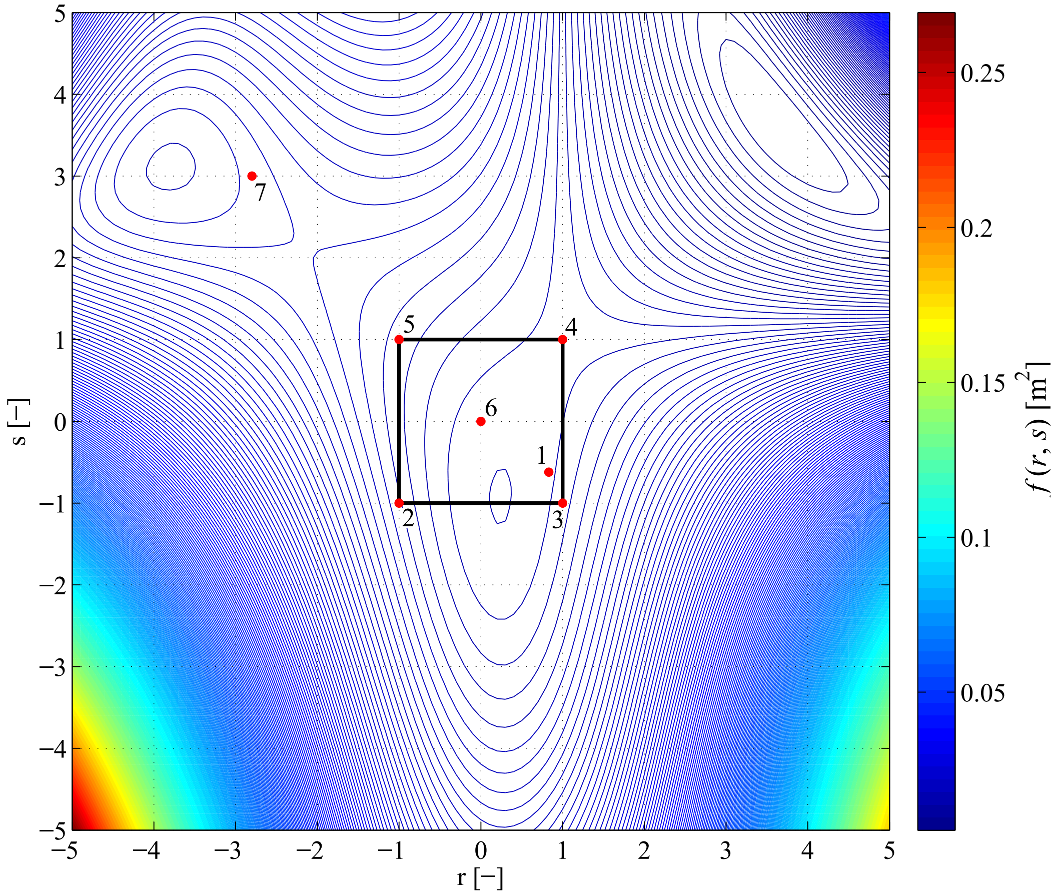


Figure 4.4: The objective function and chosen initial guesses.

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

Table 4.1: Isoparametric coordinates of chosen initial guesses.

4.1.3 Criterion of Convergence

There are more then one way how to check the convergence. Due to simplicity and comparability of the tested method, the Euclidean norm of the vector $\mathbf{x}^k - \mathbf{x}^{k-1}$ was selected. The calculation is finished when

$$\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| \leq 1e^{-10}. \tag{4.2}$$

4.2 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 iteration process for three different initial guesses is depicted in Figs. 4.5, 4.7, 4.8. First initial guess was computed by one iteration of the least-square projection method. Its coordinates are $(0.83, -0.62)$. 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 (see Fig. 3.2), the approximations of solution $\mathbf{x}^1$ and $\mathbf{x}^{11}$ jump far from the previous iterations randomly (see Fig. 4.6). The Newton-Raphson converges to the stationary point $(0.241, -0.926)$ in 28 iterations.

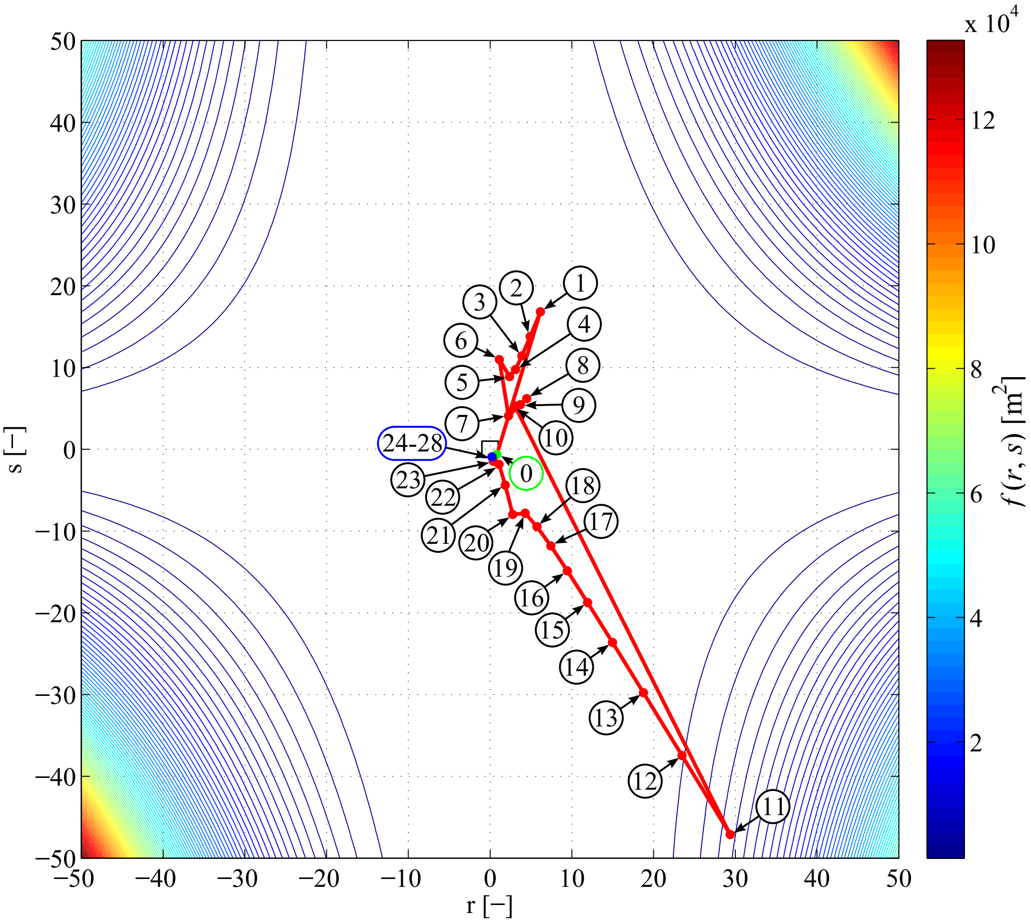


Figure 4.5: Iterations of the Newton-Raphson methods for the initial guess $(0.83, -0.62)$.

Tab. 4.2 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 (see Tab. 3.1). Since both principal minors of the Hessian matrix are positive in the point $(0.241, -0.926)$, the square-distance function (4.1) has the local minimum here.

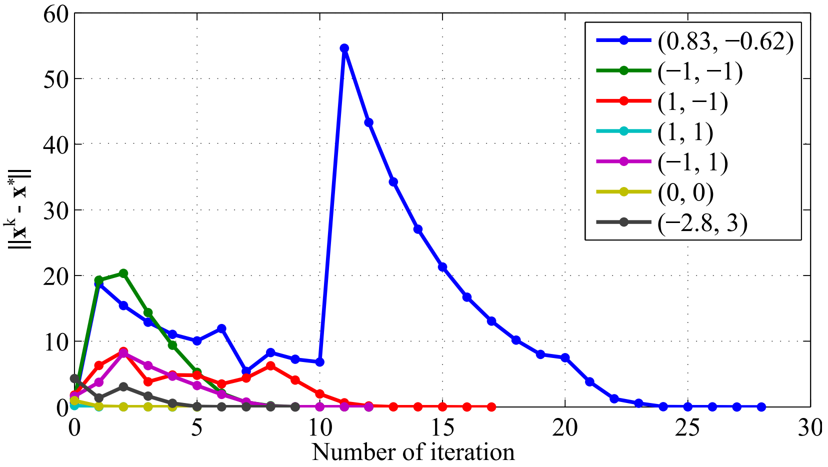


Figure 4.6: Dependence of $\left\|x^k - x^*\right\|$ on the number of iterations for the Newton-Raphson method.

Iterations for the initial guess $(-2.8, 3)$ is depicted in Fig. 4.7. This starting point was chosen intentionally beyond boundaries of the master segment close to other local minimum. The convergence to this local minimum was being expected. Since the Hessian matrix in the initial guess is again indefinite, first two iterations carry the solution far from the expected local minimum. Then the solution converge to the stationary point $(0.923, 0.810)$ in 9 iterations. Since the determinant of the Hessian matrix is negative in this point (see Tab. 4.2), the square-distance function (4.1) has the saddle point here.

The last tested initial guess which is being discussed in detail is the origin of the isoparametric coordinates (see Fig. 4.8). The situation is different at this point, since the square-distance function is convex in a neighbourhood of the origin. The Hessian matrix is positive definite for a convex function and the convergence rate of the Newton-Raphson method is quadratic now. The nearest local minimum is reached in 5 iterations.

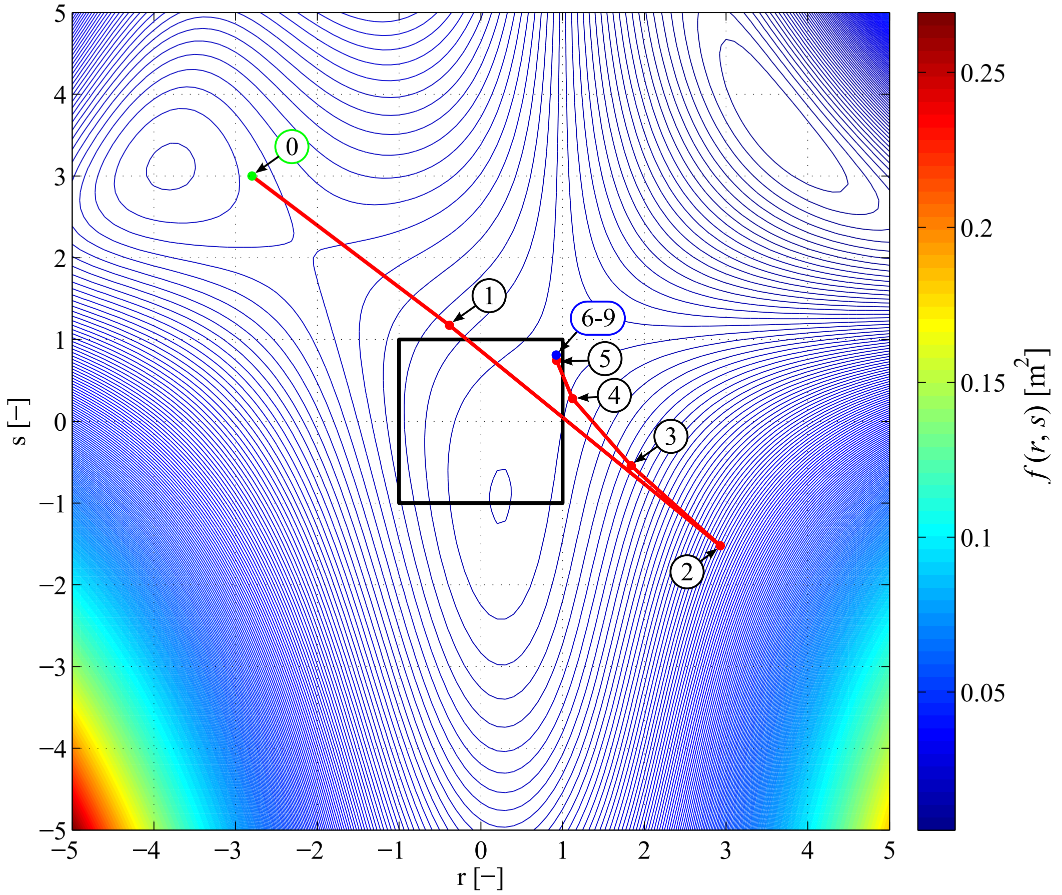


Figure 4.7: Iterations of the Newton-Raphson methods for the initial guess $(-2.8, 3)$.

Finally, we turn attention to the initial approximations lying on the boundary domain of the isoparametric coordinates r, s. Tab. 4.2 shows that solutions for initial guesses $(1, -1)$ and $(1, 1)$ converge to the same saddle point like $(-2.8, 3)$ (see Fig. 4.7). Also the result for the initial guess $(-1, 1)$ is a saddle point. And the last initial guess whose coordinates are $(-1, -1)$ converges to the local minimum $(0.241, -0.926)$.

If the initial guess lies in a sub-domain where the square-distance function is convex, as the starting point in Fig. 4.8, 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 (4.1) is not positive definite. We can conclude that the Newton-Raphson method is not suitable method for the local contact search algorithms.

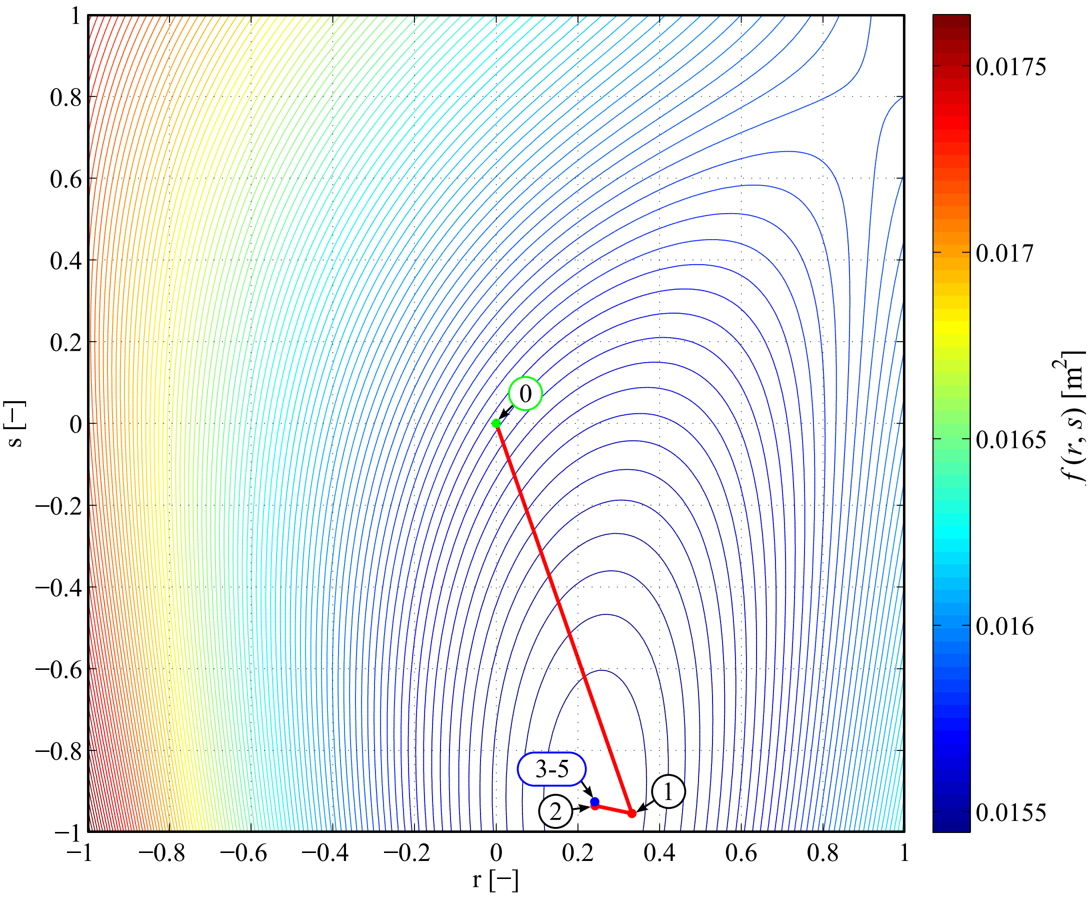


Figure 4.8: Iterations of the Newton-Raphson methods for the initial guess $(0, 0)$.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	28	0.0366	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	12	0.0163	1.24e-06	2.82e-03
$(1; -1)$	$(0.923, 0.810)$	17	0.0223	-1.76e-06	7.46e-04
$(1, 1)$	$(0.923, 0.810)$	5	0.0073	-1.76e-06	7.46e-04
$(-1, 1)$	$(-2.167, 2.133)$	12	0.0162	-2.68e-06	-2.29e-04
$(0, 0)$	$(0.241, -0.926)$	5	0.0074	1.24e-06	2.82e-03
$(-2.8, 3)$	$(0.923, 0.810)$	9	0.0130	-1.76e-06	7.46e-04

Table 4.2: Results of the Newton-Raphson method for all tested initial guesses.

4.3 Least-square Projection

In this section, the performance of the least-square projection method is investigated. Despite the excessive number of iterations, which is greater than one hundred for all initial guesses, the CPU time is relatively low. Tab. 4.3 indicates interesting fact that this method always converges to the global minimum. Furthermore, Figs. 4.9, 4.11 show another remarkable property of the method. Although the criterion of convergence is satisfied upon one hundred iterations, the solution is close to the global minimum up to twenty iteration steps. This fact is also confirmed in Fig. 4.10, where residual vectors $\|\mathbf{x}^k - \mathbf{x}^*\|$ for first forty iterations are depicted.

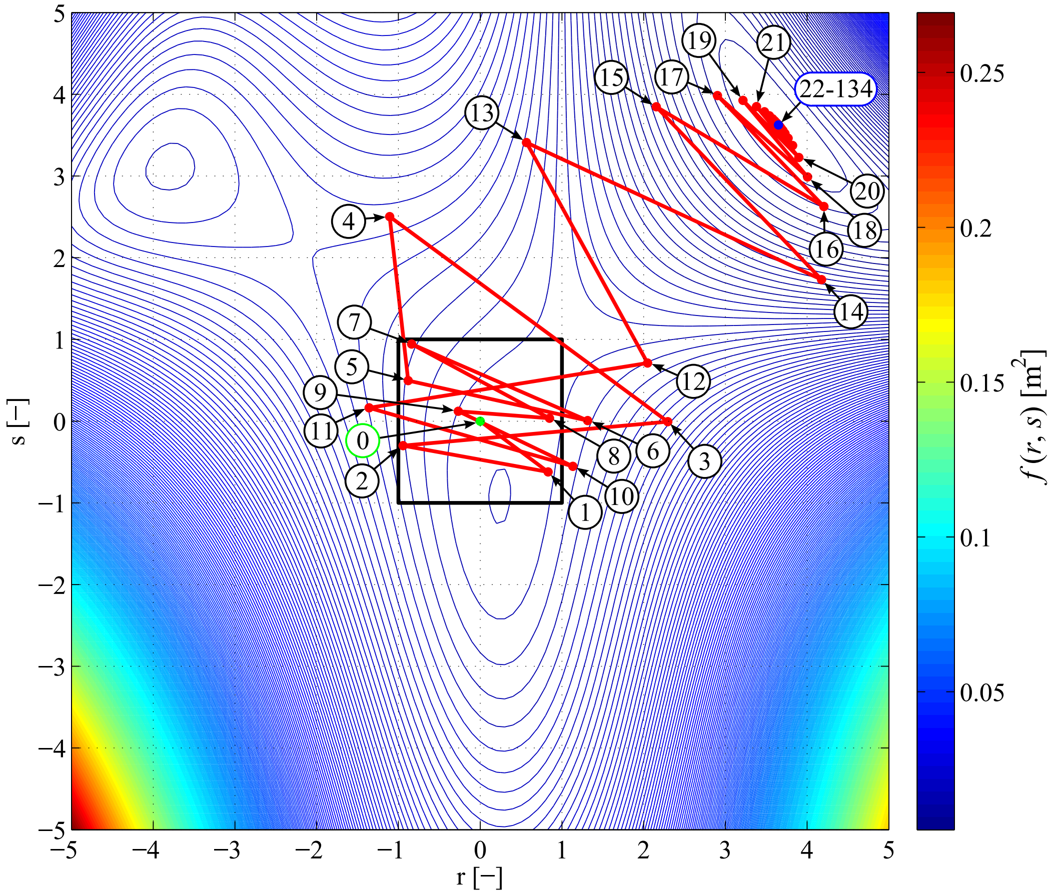


Figure 4.9: Iterations of the least-square projection method for the initial guess (0, 0).

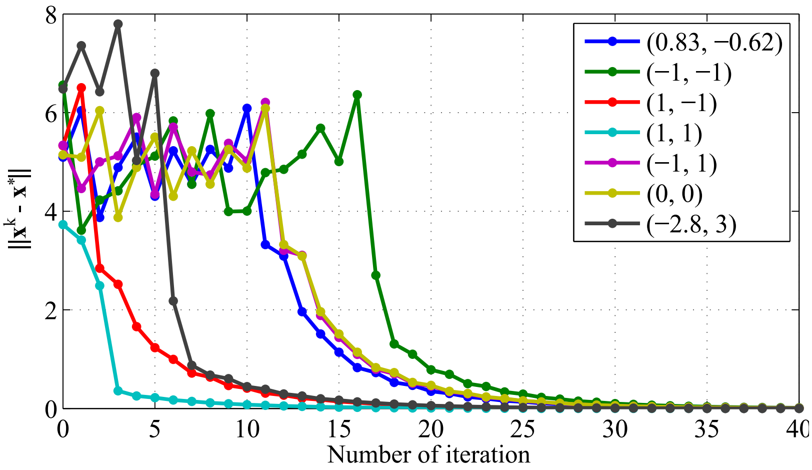


Figure 4.10: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the least-square method.

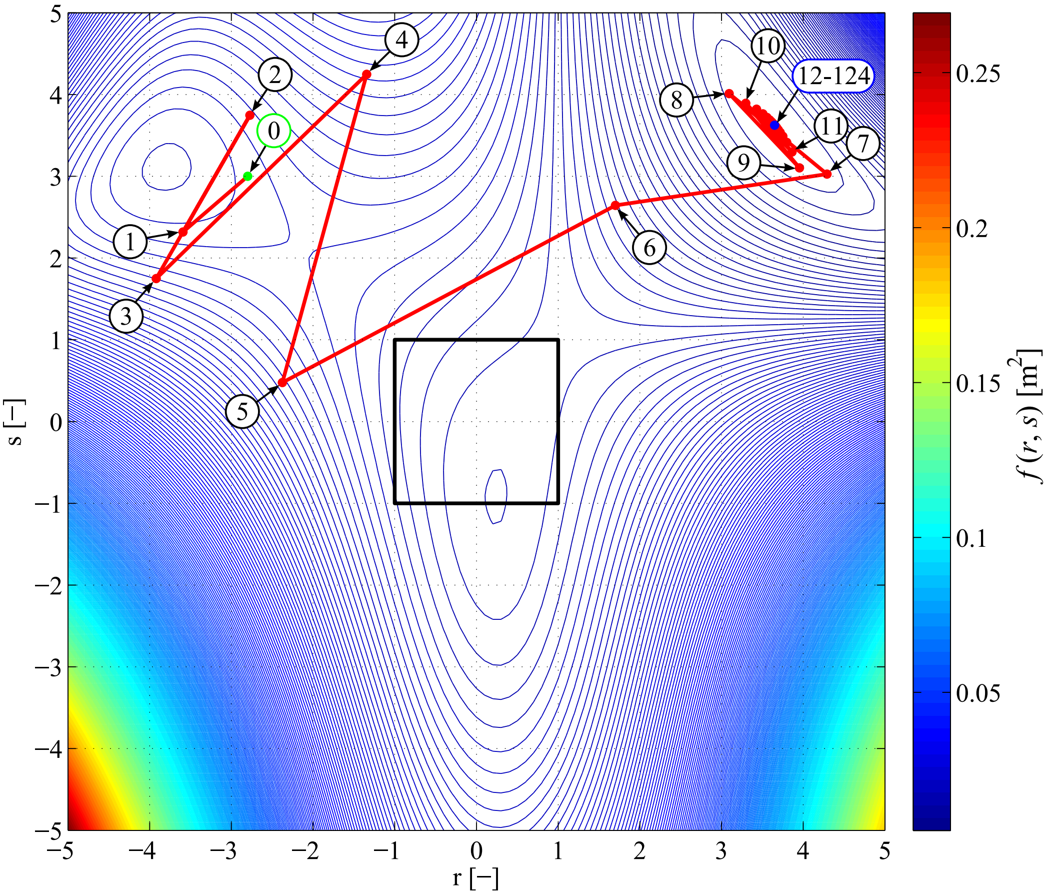


Figure 4.11: Iterations of the least-square projection method for the initial guess $(-2.8, 3)$.

In Fig. 4.3, the radius vector $\mathbf{r}'^k$ denotes the global minimum of the square-distance function (4.1) in three-dimensional space. Unfortunately, this point lies in general out of the master segment domain and the penetration vector $\mathbf{d}'^k$ is not acceptable. Due to this reason the least-square projection method is inadvisable for the local contact search.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(3.649, 3.624)$	133	0.0160	8.03e-03	7.66e-06
$(-1, -1)$	$(3.649, 3.624)$	137	0.0168	8.03e-03	7.66e-06
$(1; -1)$	$(3.649, 3.624)$	124	0.0154	8.03e-03	7.66e-06
$(1, 1)$	$(3.649, 3.624)$	116	0.0143	8.03e-03	7.66e-06
$(-1, 1)$	$(3.649, 3.624)$	134	0.0167	8.03e-03	7.66e-06
$(0, 0)$	$(3.649, 3.624)$	134	0.0165	8.03e-03	7.66e-06
$(-2.8, 3)$	$(3.649, 3.624)$	124	0.0163	8.03e-03	7.66e-06

Table 4.3: Results of the least-square projection method for all tested initial guesses.

4.4 Method of Steepest Descent

The method of steepest descent belongs to a class of methods known as the line-search methods. As it was mentioned in Section 3.4.3, the search direction $\mathbf{p}^k$ is equal to the negative gradient. The step-length is computed by one dimensional quadratic interpolation for fixed point $\mathbf{x}^k$ and direction $\mathbf{p}^k$. The characteristic feature of this method for minimization of a quadratic form is a zigzag trajectory toward converged solution.

Fig. 4.12 depicts typical iteration path toward the local minimum $(0.241, -0.926)$ in 51 iterations. Tab. 4.4 shows results for all tested initial guesses.

Fig. 4.14 shows the iteration path for the initial guess $(0, 0)$ pointing to the local minimum $(0.241, -0.926)$. Comparison of the Newton-Raphson method and the method of steepest descent for a domain where the square-distance function is convex, follows from Figs. 4.8, 4.14 for the initial guess $(0, 0)$. In this case, the Newton-Raphson method converges quadratically. For non-convex functions, the steepest descent guarantees convergence to the nearest local minimum as is shown in Figs. 4.12, 4.14.

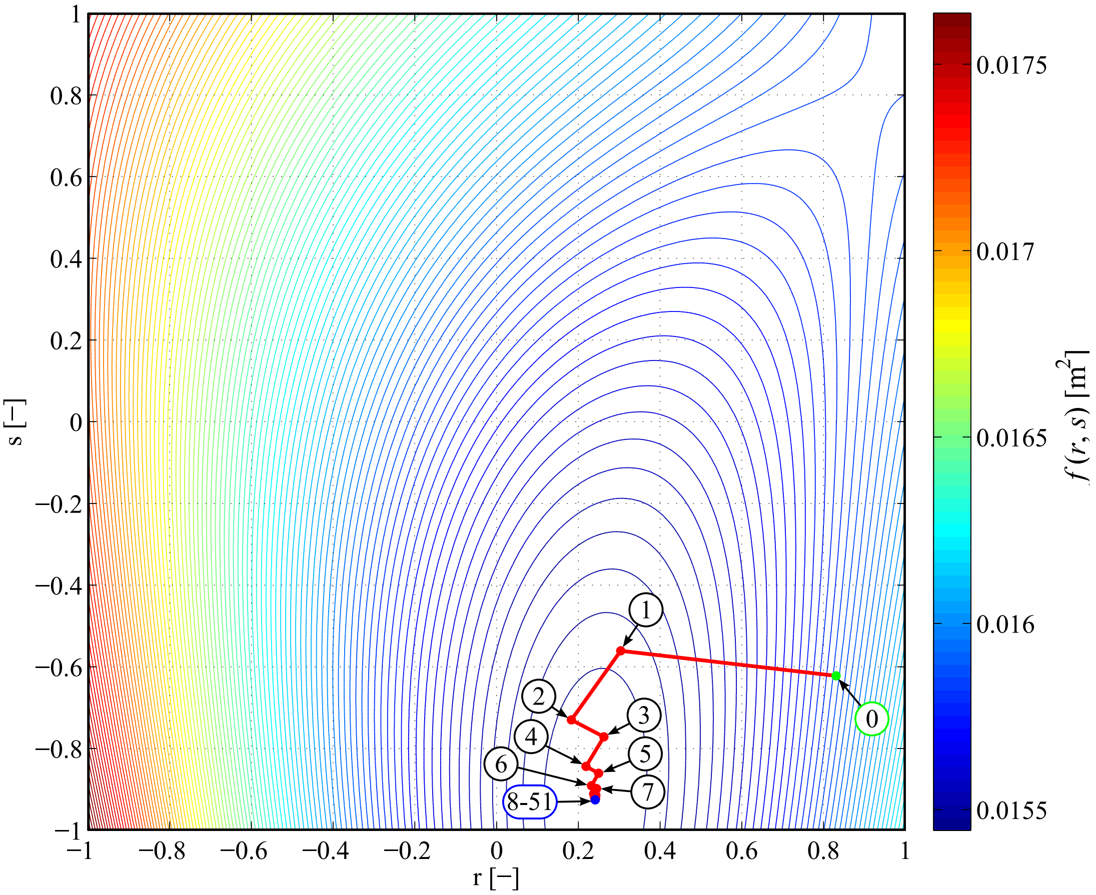


Figure 4.12: Iterations of the method of steepest descent for the initial guess $(0.83, -0.62)$.

Another significant feature of the steepest descent is insensibility to the saddle points. It is verify in Tab. 4.4, where only the positive values of principal minors occur. This is due to the Hessian matrix is not employed in the computation. Initial guess $(1, 1)$ was intended to test each method's behaviour in the vicinity of such a point. The cyan break line in Fig. 4.13 implies certain fluctuation of residual vectors near the saddle point $(0.923, 0.810)$. However, the steepest descent method finally converges to the point $(3.648, 3.624)$, where the square-distance function has its global minimum. Note that other tested quasi-Newton methods do not share this feature.

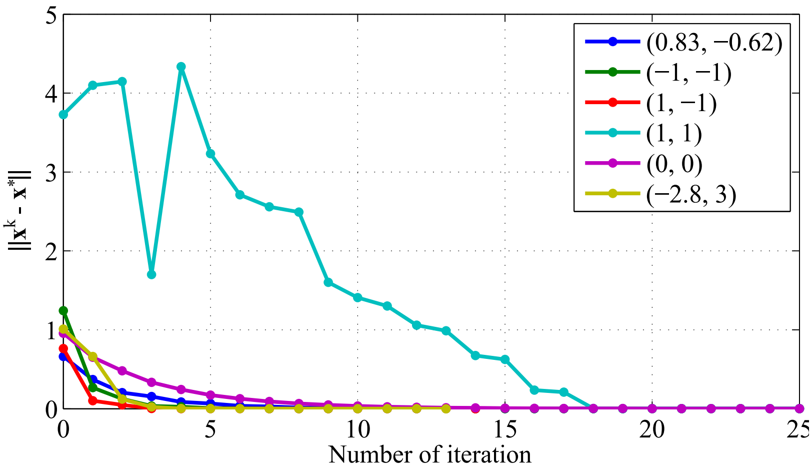


Figure 4.13: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the steepest descent method.

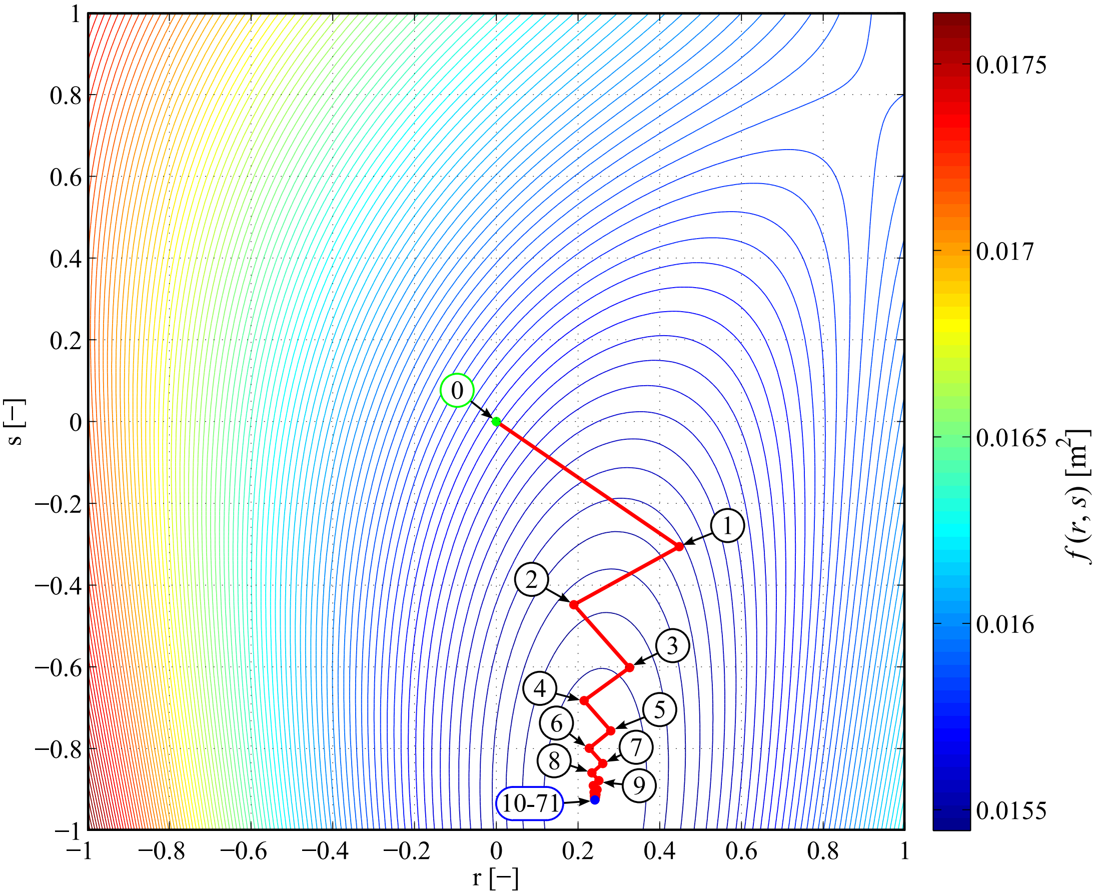


Figure 4.14: Iterations of the method of steepest descent for the initial guess $(0, 0)$.

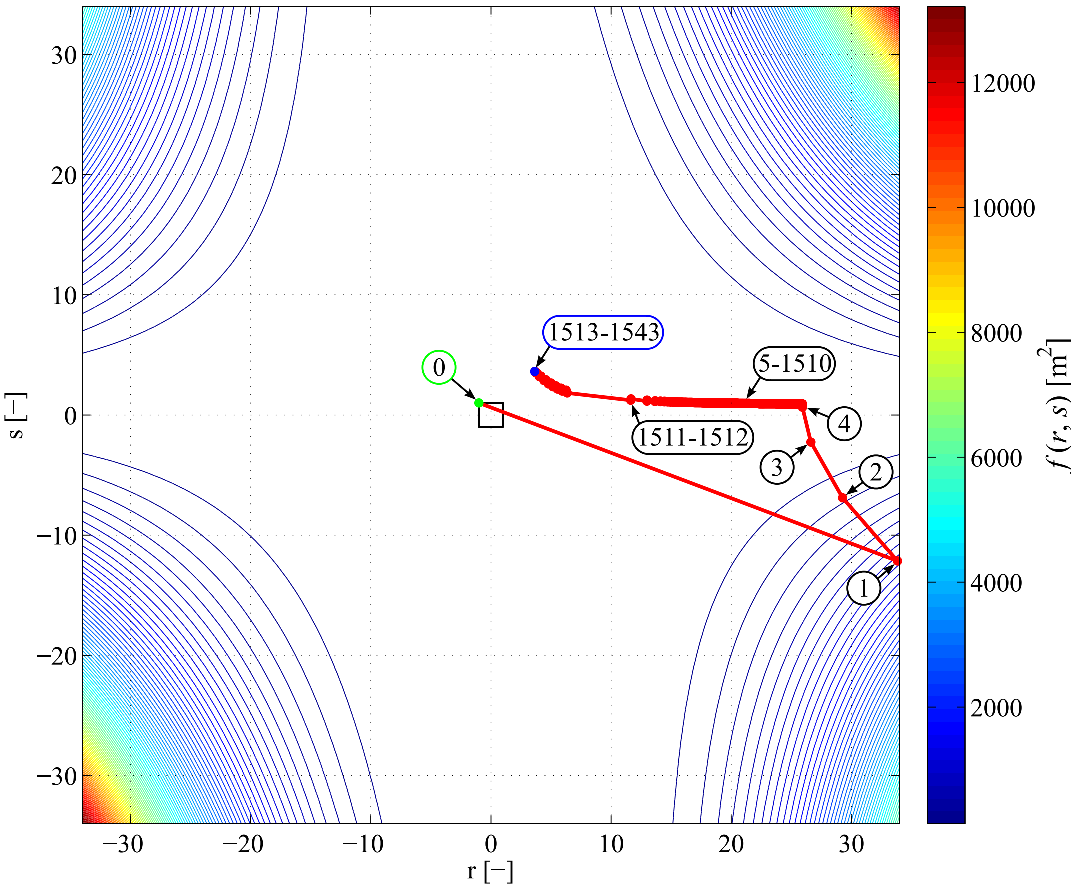


Figure 4.15: Iterations of the steepest descent method for the initial guess $(-1, 1)$.

It is interesting to notice that the method suffers from poor convergence for the initial guess $(-1, 1)$. What happened shows Fig. 4.15. The problem is not in the search direction $\mathbf{p}^0$ but in the step-length t^0. The square-distance function $f(t)$ restricted by $\mathbf{p}^0$ is depicted in Fig. 4.16. Since the initial guess $(-1, 1)$ (where $t = 0$) lies close to an inflect point, the second derivative of $f(t)$ is nearly zero. Then, the result of the quadratic interpolation is a parabola with small curvature. As a result, the next iteration in Fig. 4.15 jumps far from the expected solution. Nevertheless, we consider this case as a singular perturbation which can be overcome by a reasonable initial guess in the local contact search algorithm.

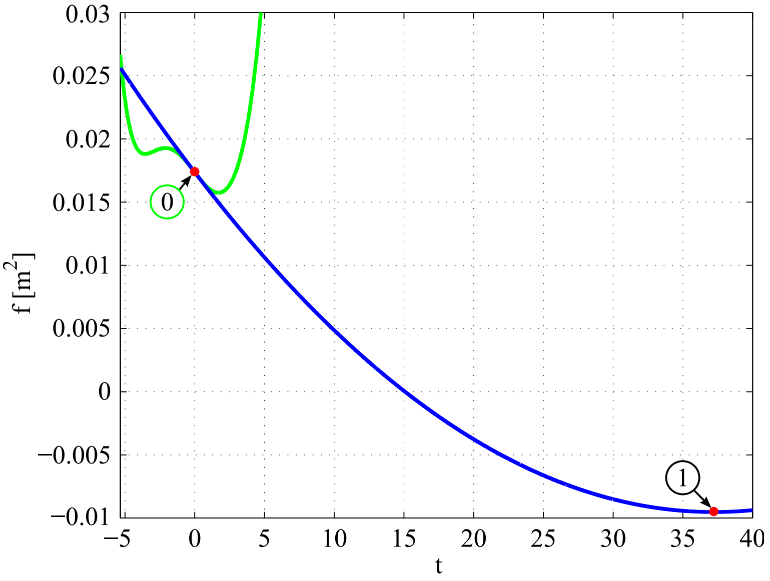


Figure 4.16: The square-distance function restricted in the direction of $\mathbf{p}^0$.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	51	0.1647	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	28	0.0923	1.24e-06	2.82e-03
$(1; -1)$	$(0.241, -0.926)$	14	0.04451	1.24e-06	2.82e-03
$(1, 1)$	$(3.648, 3.624)$	29	0.0951	7.66e-06	6.33e-03
$(-1, 1)$	$(3.648, 3.624)$	1543	5.2326	7.66e-06	6.33e-03
$(0, 0)$	$(0.241, -0.926)$	71	0.2231	1.24e-06	2.82e-03
$(-2.8, 3)$	$(-3.804, 3.111)$	13	0.0414	6.76e-06	2.53e-03

Table 4.4: Results of the method of steepest descent for all tested initial guesses.

Although the initial guess $(-1, 1)$ uncovers an imperfection of the implemented line-search procedure, the remaining initial guess achieve a good convergence rate. Thus, with assumption of a reasonable starting point this method is suitable for the local contact search algorithms. In the next section we turn attention to results obtained by the quasi-Newton methods.

4.5 Broyden's Method

Broyden's method requires the selection of the initial secant matrix $\mathbf{D}^0$. It plays a critical role in computation of the first search direction $\mathbf{p}^0$. A good idea is to select some positive definite matrix ensuring a descent direction of $\mathbf{p}^0$. In Section 4.2 was shown that an exact Hessian matrix $\mathbf{H}^0$ is not always positive definite. But if we put $\mathbf{D}^0$ equal to the identity matrix $\mathbf{I}$, or its multiple, the steepest descent direction of $\mathbf{p}^0$ is guaranteed. Therefore we set $\mathbf{D}^0 = \left\|\nabla f^0\right\| \cdot \mathbf{I}$. Since secant matrices $\mathbf{D}^k$ approximate Hessian matrices in consequent iterations, better rate of convergence than the method of steepest descent is obtained. The improvement of convergence rate is obvious from the comparison of Figs. 4.12, 4.17, Figs. 4.14, 4.18 and Tabs. 4.4, 4.5.

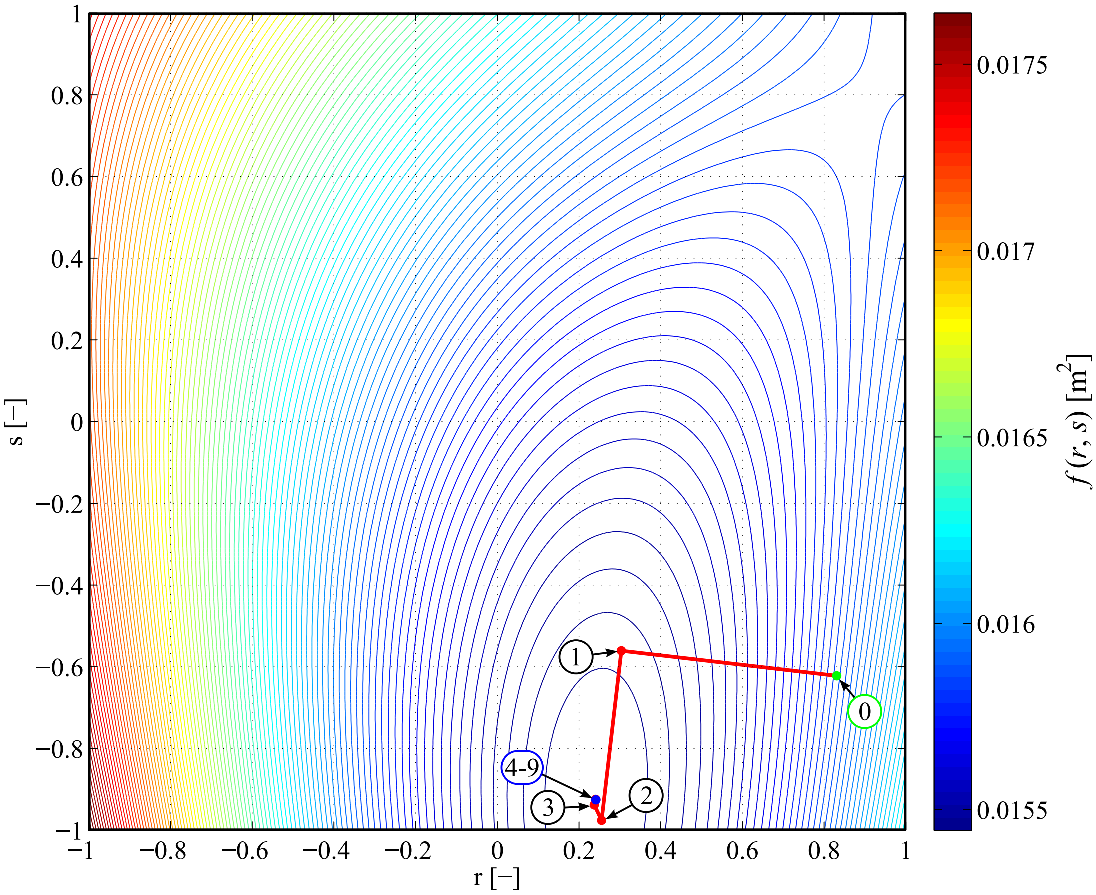


Figure 4.17: Iterations of the Broyden's method for the initial guess (0.83, −0.62).

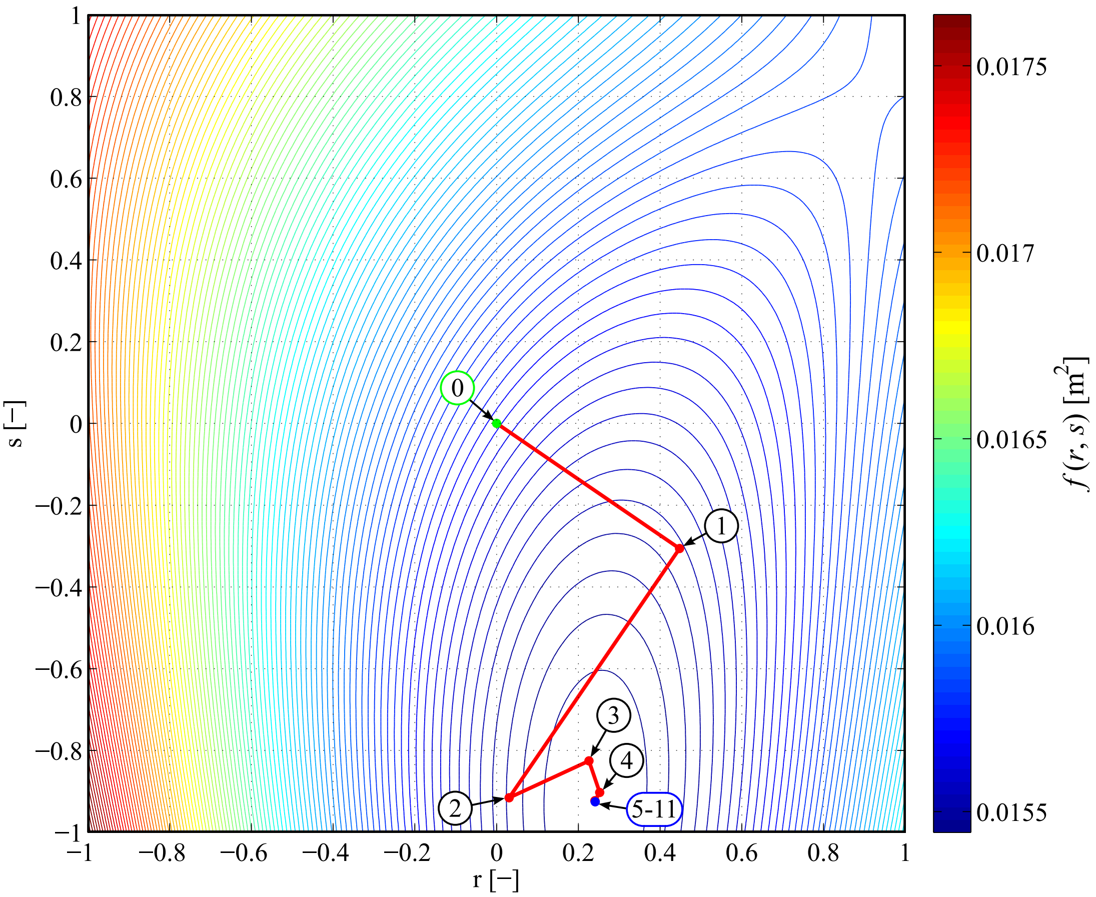


Figure 4.18: Iterations of the Broyden's method for the initial guess $(0, 0)$.

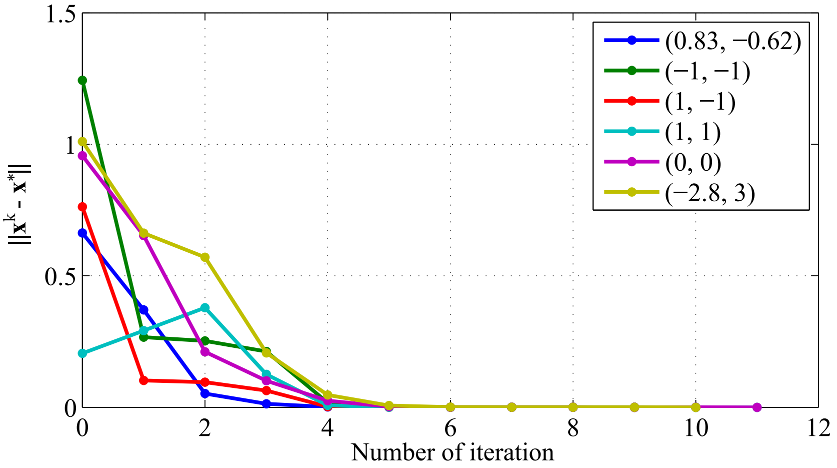


Figure 4.19: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the Broyden's method.

Fig. 4.19 shows, that numbers of iterations are at most equal to eleven for each initial guess with exception of the point $(-1, 1)$. In this case the computation converges in 14900 iterations (see Fig. 4.20). The reason was discussed in Section 4.4.

The solution for initial guess $(1, 1)$ converges to the saddle point $(0.923, 0.810)$ (see Tab. 4.5). This confirms the mentioned fact, that quasi-Newton methods are not generally indifferent to the saddle points.

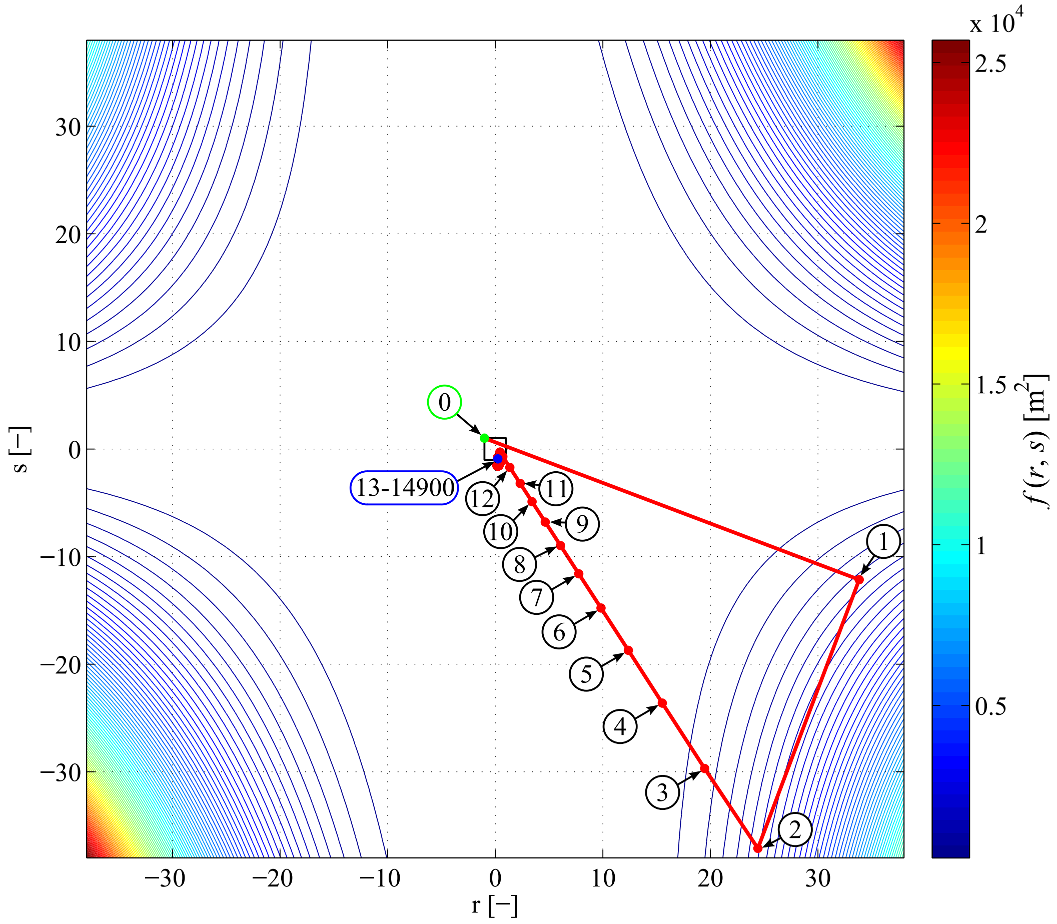


Figure 4.20: Iterations of the Broyden's method for the initial guess $(-1, 1)$.

The Broyden's method was derived to replace the computation of exact Hessian matrices. Since there is no guarantee of positive definite secant matrices, a descent direction is not ensured within the whole computation. In the following section we focus on the most popular quasi-Newton solver, the BFGS method.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	9	0.0176	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	10	0.0187	1.24e-06	2.82e-03
$(1; -1)$	$(0.241, -0.926)$	9	0.0169	1.24e-06	2.82e-03
$(1, 1)$	(0.923, 0.810)	11	0.0203	-1.76e-06	7.46e-04
$(-1, 1)$	$(0.241, -0.926)$	14900	29.981	1.24e-06	2.82e-03
$(0, 0)$	$(0.241, -0.926)$	11	0.0212	1.24e-06	2.82e-03
$(-2.8, 3)$	$(-3.804, 3.111)$	10	0.0193	6.76e-06	2.53e-03

Table 4.5: Results of the Broyden's method for all tested initial guesses.

4.6 BFGS Method

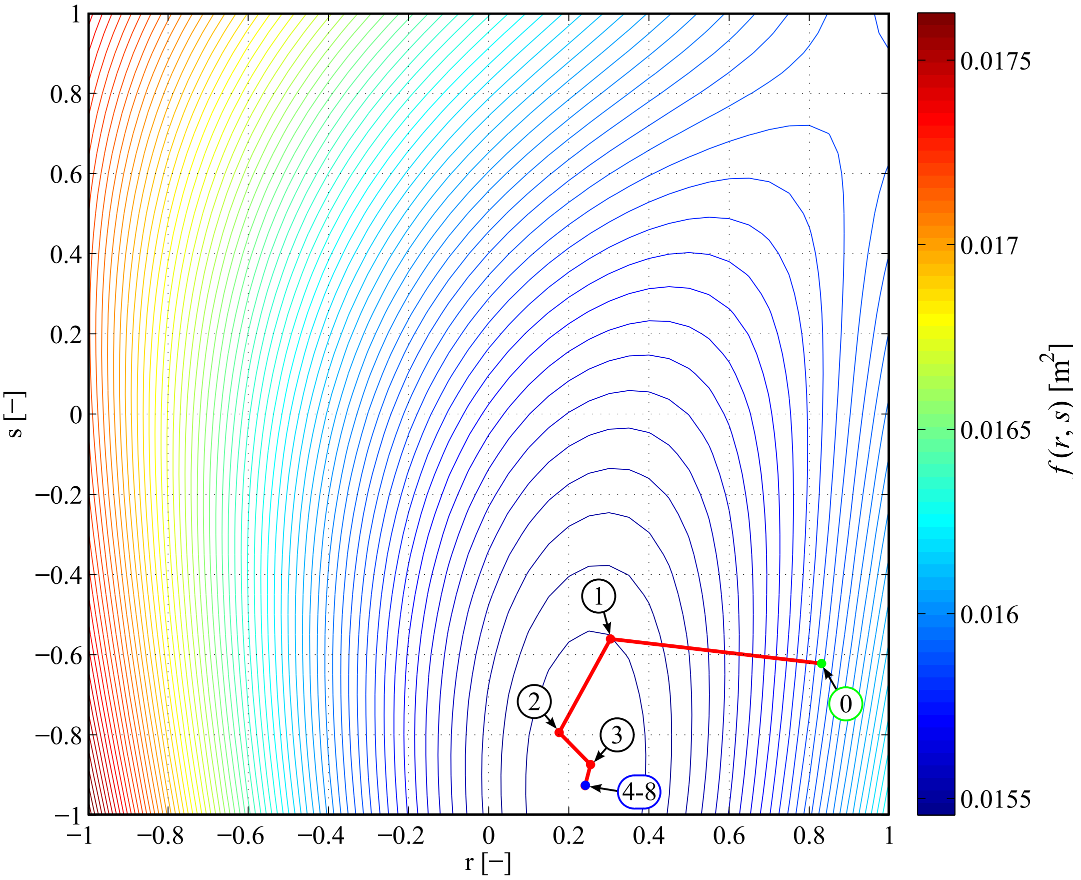


Figure 4.21: Iterations of the BFGS method for the initial guess $(0.83, -0.62)$.

The BFGS method requires no evaluation of the Hessian matrix and retains the secant feature of Broyden's method. Moreover, the descent direction is guaranteed by the secant matrices, that are all positive definite. $\mathbf{D}^0$ is initialized by the scaled identity matrix $\mathbf{I}$ to $\mathbf{D}^0 = \|\nabla f^0\| \cdot \mathbf{I}$. The best rate of convergence was reached by this method. For the initial guess $(0.83, -0.62)$ there are only eight iteration steps necessary to achieve the solution (see Fig. 4.21). Fig. 4.22 shows eight iterations for the initial guess $(0, 0)$. Low numbers of iterations imply low CPU times in Tab. 4.6.

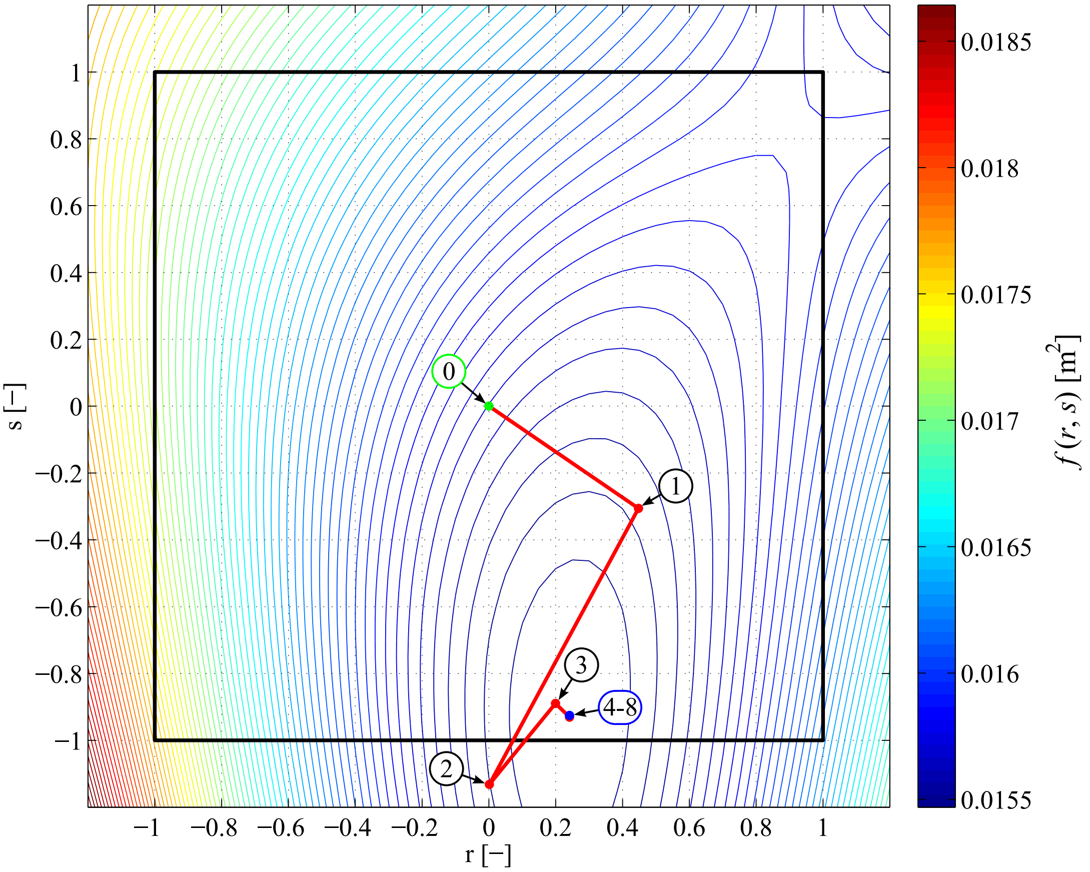


Figure 4.22: Iterations of the BFGS method for the initial guess$(0, 0)$.

It should be noted that the residual norm $\|\mathbf{x}^k - \mathbf{x}^*\|$ for all initial points except $(-1, 1)$ is practically zero within four iterations. (see Fig. 4.23).

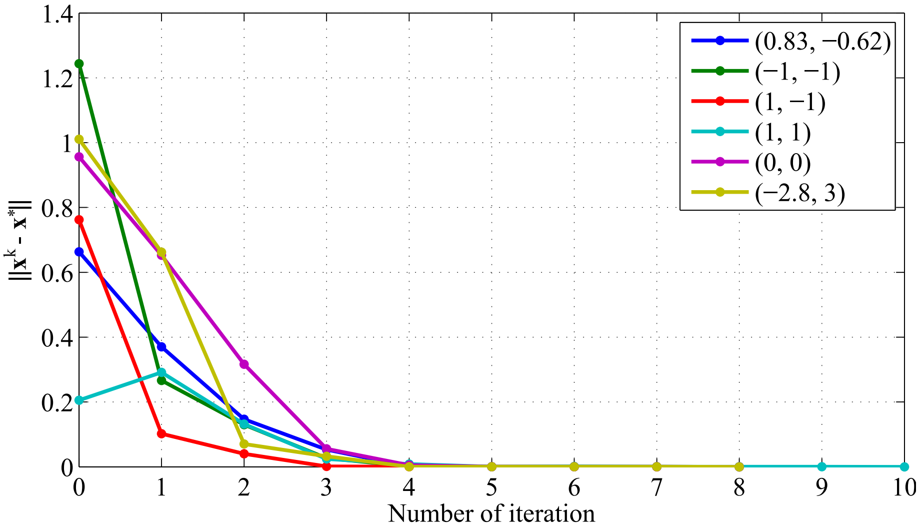


Figure 4.23: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the BFGS method.

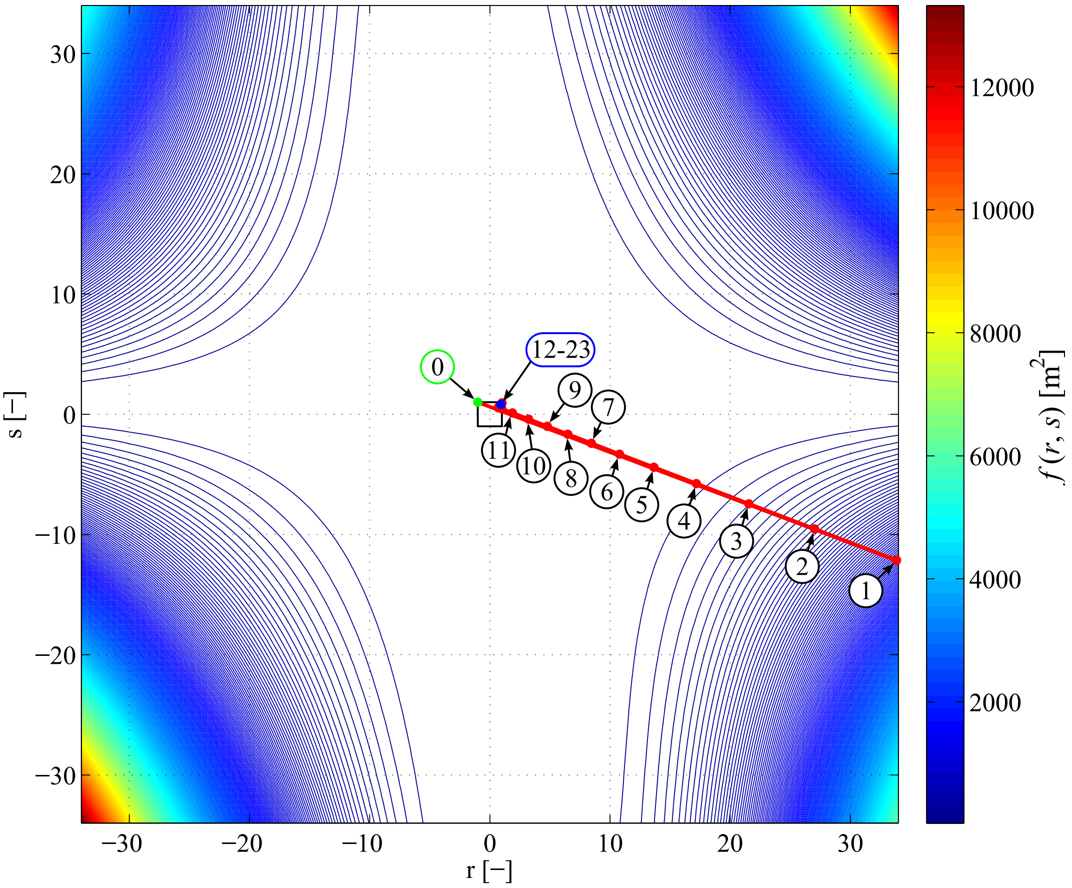


Figure 4.24: Iterations of the BFGS method for the initial guess $(-1, 1)$.

Similarly as in Section 4.4 the problem with the step-length t^0 arises. As it is obvious from Fig. 4.24 where the solution for the initial guess $(-1, 1)$ is depicted, the computation is returned back into the master segment domain within eleven iterations. It converges to the saddle point $(0.923, 0.810)$. In comparison with Broyden's method, 23 iterations represent a considerable improvement of the convergence rate.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	8	0.0136	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	8	0.0149	1.24e-06	2.82e-03
$(1; -1)$	$(0.241, -0.926)$	7	0.0132	1.24e-06	2.82e-03
$(1, 1)$	$(0.923, 0.810)$	9	0.0174	-1.76e-06	7.46e-04
$(-1, 1)$	$(0.923, 0.810)$	23	0.0444	-1.76e-06	7.46e-04
$(0, 0)$	$(0.241, -0.926)$	8	0.0154	1.24e-06	2.82e-03
$(-2.8, 3)$	$(-3.804, 3.111)$	8	0.0149	6.76e-06	2.53e-03

Table 4.6: Results of the BFGS method for all tested initial guesses.

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. 4.6). It is a very good candidate for the local contact search algorithm, especially with the powerful step-length procedure.

4.7 DFP Method

As it was mentioned in Section 3.4.6, one apparent advantage of the DFP method over the BFGS method is that the BFGS search direction (3.8) for the latter must be computed by solving the system $\mathbf{D}^k \mathbf{p}^k = \nabla f^k$ while the DFP search direction can be computed directly.

As it is adduced in the reference [9], it turns out that this advantage is offset by some computational advantages of the BFGS method over the DFP method. For example, although the secant matrices $\mathbf{D}^k$ produced by both methods are positive definite in theory, the DFP method has a tendency to produce $\mathbf{D}^k$ that are not positive definite because of computer round-off error, while the BFGS method does not seem to share this defect. Similarly to the initialization of $\mathbf{D}^0$ in Section 4.6, the initial secant matrix was set to $\mathbf{D}^0 = \left\|\nabla f^0\right\|^{-1} \cdot \mathbf{I}$.

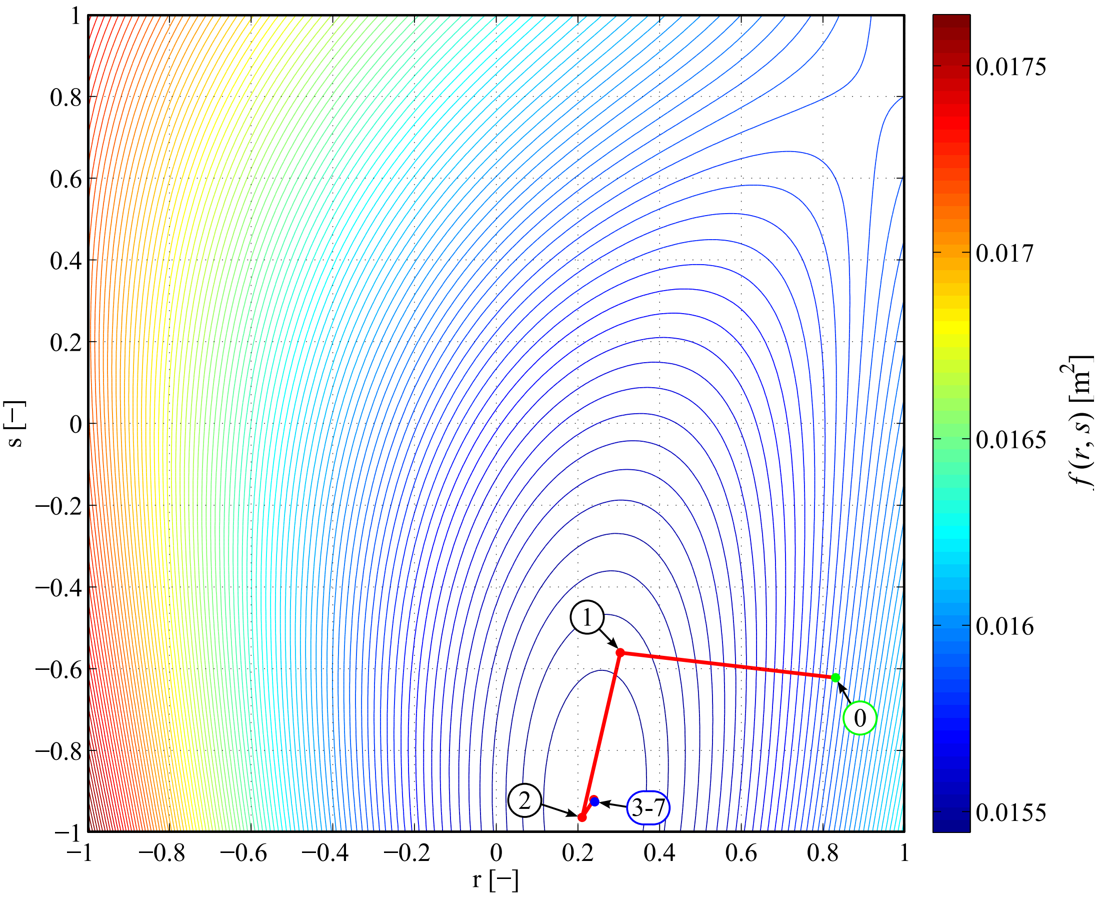


Figure 4.25: Iterations of the DFP method for the initial guess $(0.83, -0.62)$.

Fig. 4.25 and 4.27 denote the DFP iteration paths similar to the BFGS ones. Differences are only in numbers of iterations and CPU times as follow from the comparison of Tabs. 4.6 and 4.7. Since the search direction $\mathbf{p}^k$ can be computed directly without the inverse matrix $\left(\mathbf{D}^k\right)^{-1}$ calculation, the CPU times of the DFP method are slightly less then the CPU times of the BFGS method. Nevertheless, the problematic initial guess $(-1, 1)$ confirms the computational advantages of the BFGS method mentioned in the beginning of this section. Comparison of Figs. 4.24, 4.28 shows that both iteration paths are the same. The increase of the DFP iterations is probably caused by computer round-off errors. Fig. 4.26 shows the residual norms $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ for all initial guesses except the point $(-1, 1)$.

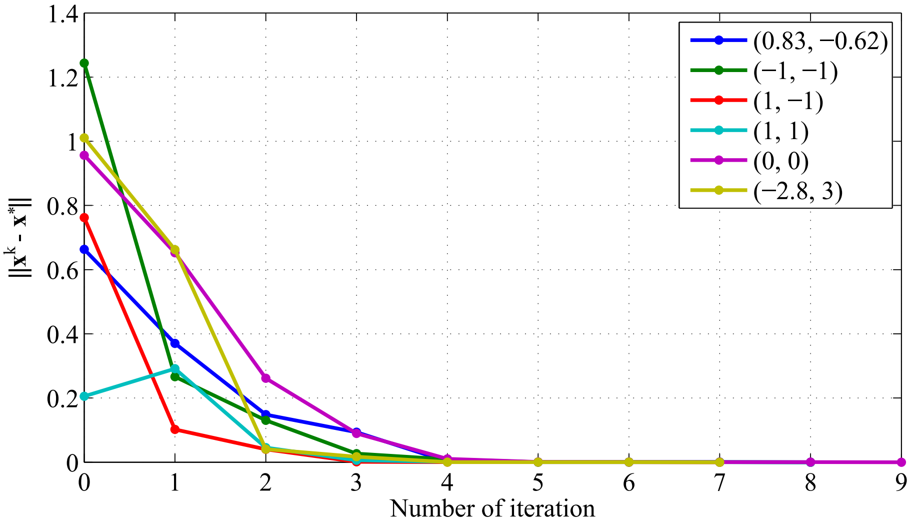


Figure 4.26: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the DFP method.

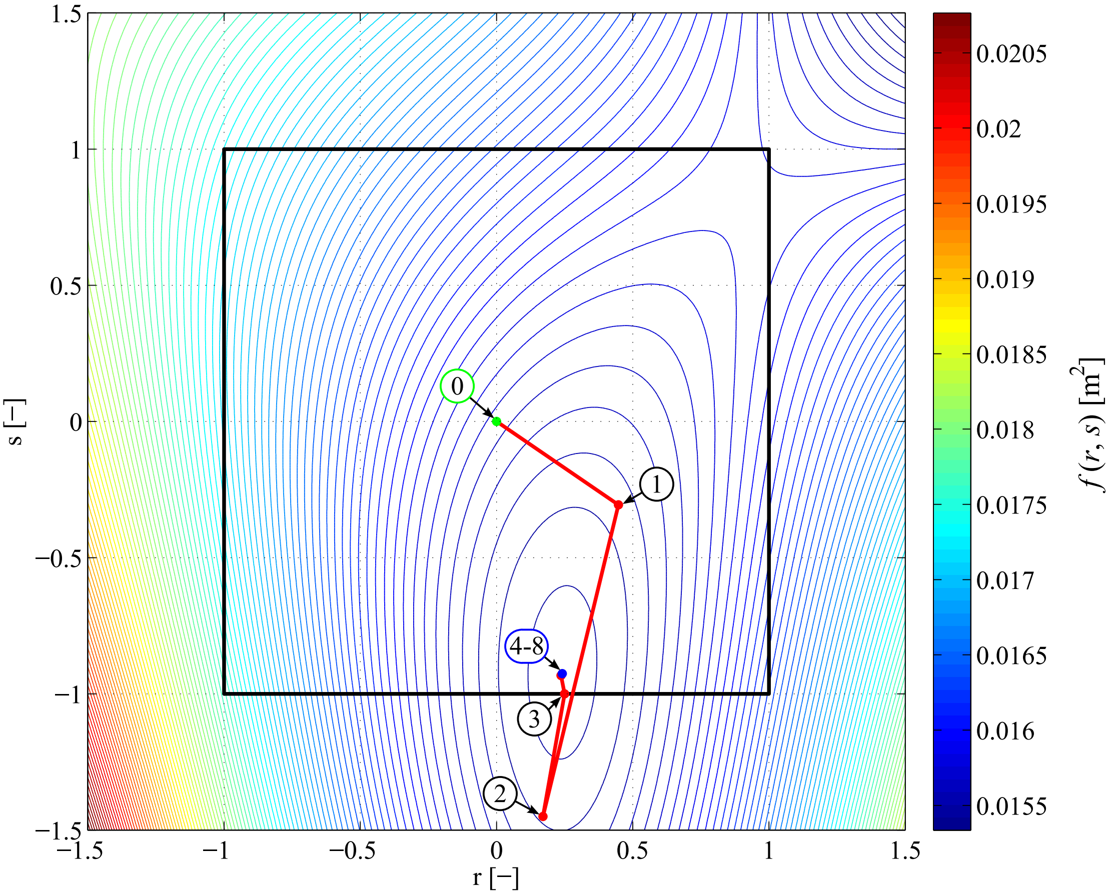


Figure 4.27: Iterations of the DFP method for the initial guess$(0,0)$.

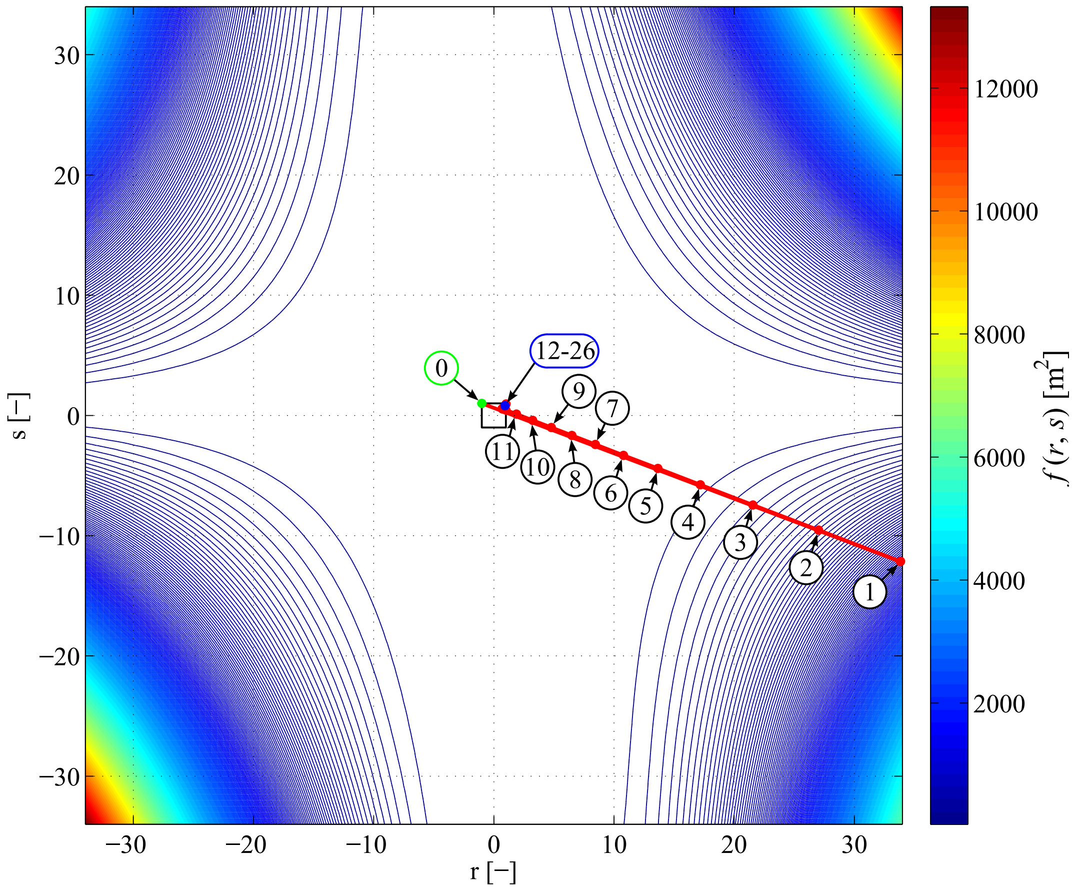


Figure 4.28: Iterations of the DFP method for the initial guess $(-1, 1)$.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	7	0.0127	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	8	0.0148	1.24e-06	2.82e-03
$(1; -1)$	$(0.241, -0.926)$	7	0.0133	1.24e-06	2.82e-03
$(1, 1)$	$(0.923, 0.810)$	9	0.0166	-1.76e-06	7.46e-04
$(-1, 1)$	$(0.923, 0.810)$	26	0.0469	-1.76e-06	7.46e-04
$(0, 0)$	$(0.241, -0.926)$	8	0.0144	1.24e-06	2.82e-03
$(-2.8, 3)$	$(-3.804, 3.111)$	8	0.0150	6.76e-06	2.53e-03

Table 4.7: Results of the DFP method for all tested initial guesses.

4.8 Simplex Method

Although there are no evaluations of derivations in this method, CPU times in Tab. 4.8 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. The implementation of the code to a compiled language like FORTRAN could improves performance of our simplex algorithm.

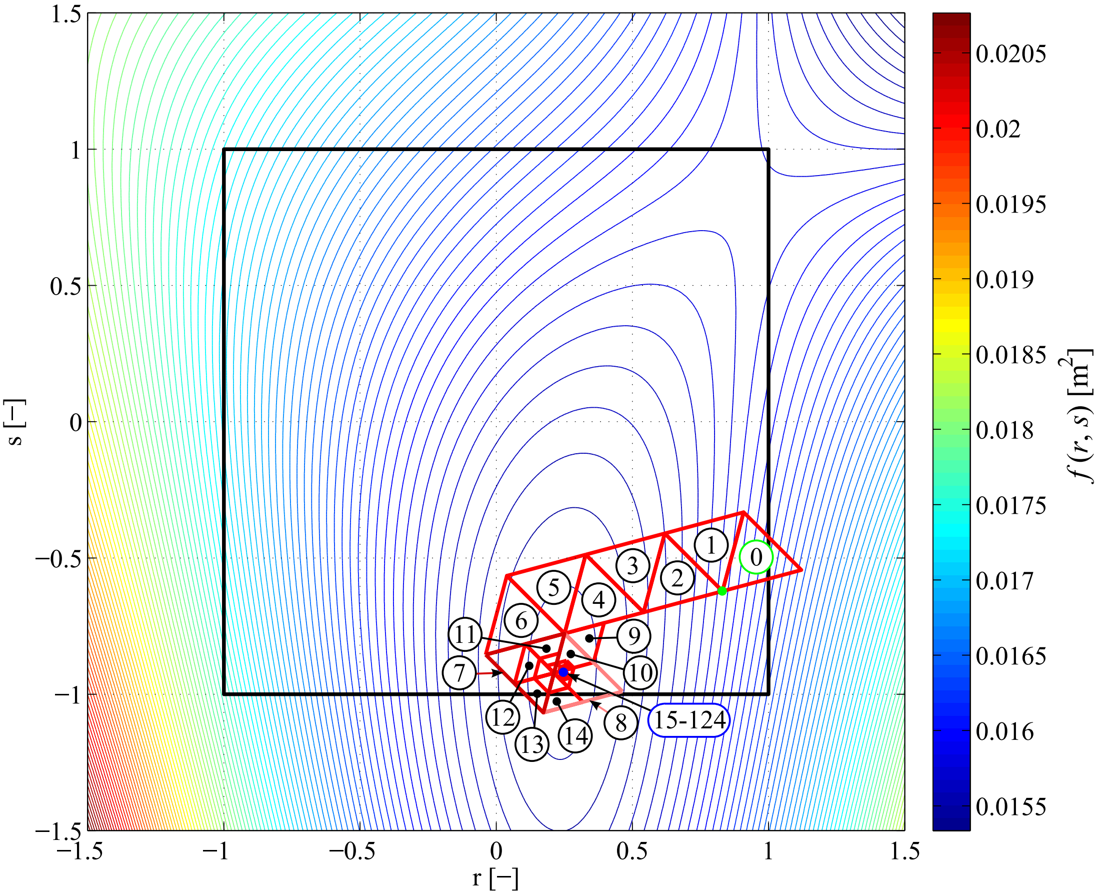


Figure 4.29: Iterations of the simplex method for the initial guess $(0.83, -0.62)$.

Fig. 4.29 shows simplex sliding into the nearest local minimum. At the starting procedure, the length of simplex edge has to be set. A numerical experiment was performed for this purpose. The results are depicted in Fig. 4.30.

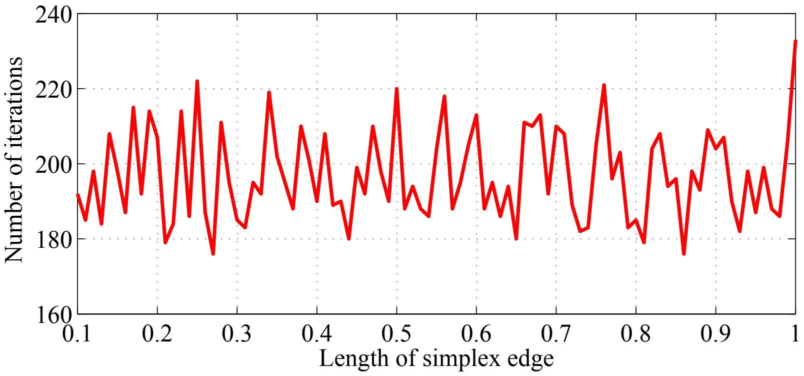


Figure 4.30: Results of the numerical experiment to determine an optimal length of simplex edge for the initial guess (0.83, −0.62).

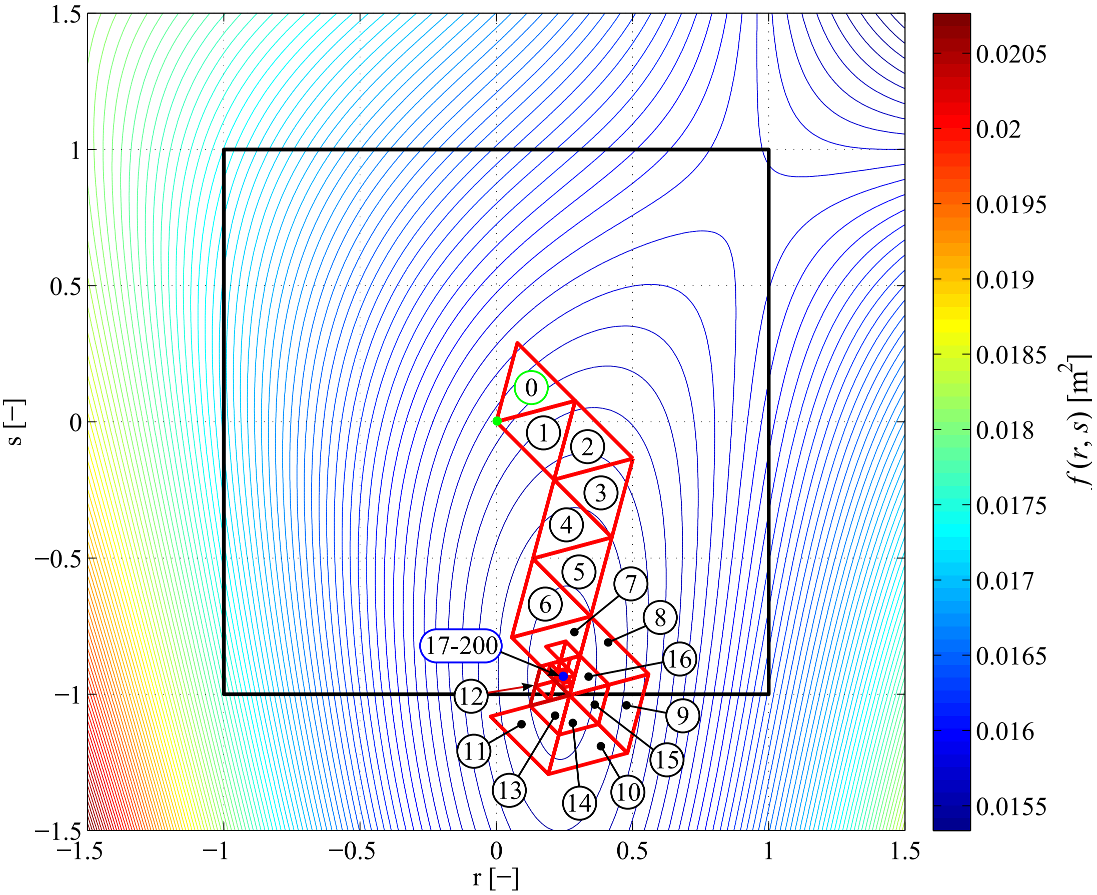


Figure 4.31: Iterations of the simplex method for the initial guess $(0; 0)$.

Since the dependence of number of iterations on the simplex edge length is stochastic, the value was set to 0.43 for all initial guesses. This value was obtained by the arithmetic mean of number of iterations over all simplex edge lengths.

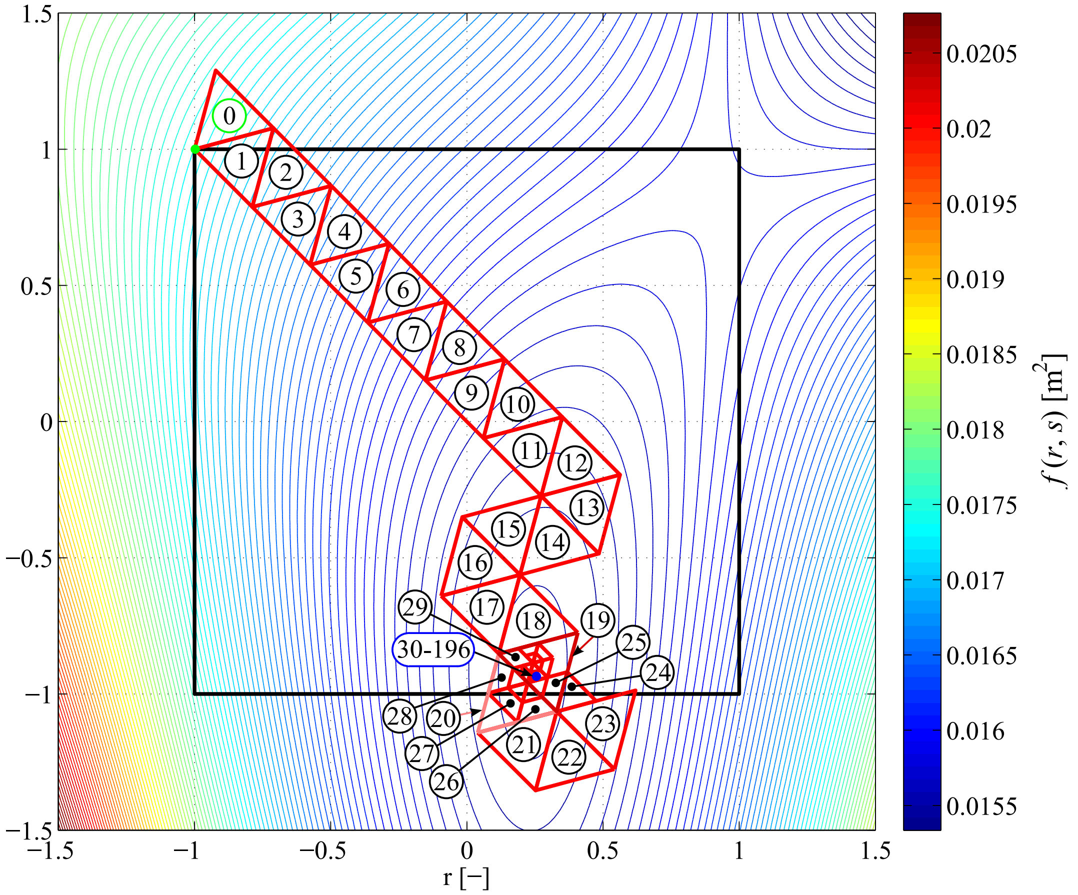


Figure 4.32: Iterations of the simplex method for the initial guess $(-1, 1)$.

The improvement could be observed in Fig. 4.32 for the problematic initial guess $(-1, 1)$, because the step-length selection procedure is not employed in the simplex method. Solutions for all corner and centric starting points (see Fig. 4.4) except point $(1, 1)$ converge to the nearest local minimum $(0.241, -0.926)$. The computation for the initial guess $(1, 1)$ directs toward the global minimum $(3.649, 3.624)$, since the initial position of the simplex is on the opposite sides of the saddle ridge.

The residual norms $\|\mathbf{x}^k - \mathbf{x}^*\|$ for first fifty iterations are depicted in Fig. 4.33. Numbers of iterations in Tab. 4.8 are higher than one hundred even though Fig. 4.33 demonstrates a considerable decrement of $\|\mathbf{x}^k - \mathbf{x}^*\|$ within thirty iteration steps.

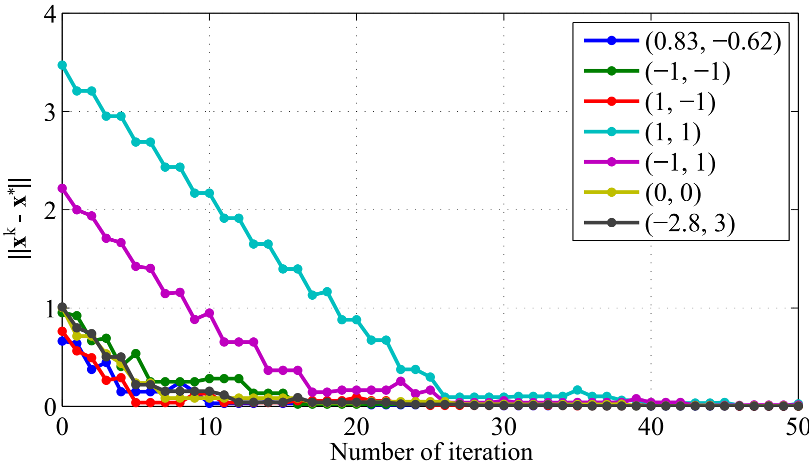


Figure 4.33: Dependence of $\left\|\mathbf{x}^k - \mathbf{x}^*\right\|$ on the number of iterations for the simplex method.

$\mathbf{x}^0$	$\mathbf{x}^*$	NI	CPU time	$\det \mathbf{H}(\mathbf{x}^*)$	$H_{11}(\mathbf{x}^*)$
$(0.83, -0.62)$	$(0.241, -0.926)$	190	0.2472	1.24e-06	2.82e-03
$(-1, -1)$	$(0.241, -0.926)$	192	0.2472	1.24e-06	2.82e-03
$(1; -1)$	$(0.241, -0.926)$	184	0.2504	1.24e-06	2.82e-03
$(1, 1)$	$(3.649, 3.624)$	210	0.2786	7.66e-06	6.33e-03
$(-1, 1)$	$(0.241, -0.926)$	196	0.2530	1.24e-06	2.82e-03
$(0, 0)$	$(0.241, -0.926)$	200	0.2609	1.24e-06	2.82e-03
$(-2.8, 3)$	$(-3.804, 3.111)$	190	0.2381	6.76e-06	2.53e-03

Table 4.8: Results of the simplex method for all tested initial guesses.

Although several good features of the simplex method was showed, higher time consumption handicaps this method. Nevertheless, the implementation to a compiled language like FORTRAN could be suitable for the local contact search.

CHAPTER

FIVE

DISCUSSION, CONCLUSIONS AND FUTURE WORK

In this study, various numerical methods for the local contact search were investigated. The geometry of contact segments was interpolated by the three-dimensional quadrilateral eight-node element based on the isoparametric finite element formulation. The calculation of the normal vector for such a surface leads to the minimization problem of nonlinear function – the square-distance function. Seven 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 important part of the solution is a good choice of the initial guess. For the purpose of numerical tests seven initial guesses were considered. The results are summarized in Tab. 5.1. It shows the numbers of iterations (NI) and the CPU time (t [s]) for each tested method and initial guess.

The Newton-Raphson method is used in the current version of the local contact search algorithm in the FE code PMD. It was shown that this method is not suitable for local contact search due to the strong nonlinearity of the square-distance function. However, if the square-distance function is convex in a neighbourhood of the initial guess and the Hessian matrix is positive definite in this point, then the Newton-Raphson method converges quadratically. This situation occurs for the initial guess $(0, 0)$, where the solution converges in five iterations. Another disadvantage of the Newton-Raphson method as well as the quasi-Newton methods is the convergence to the saddle points. The solutions, which converged to the saddle points are denoted in Tab. 5.1 by suffix sp behind the number of iterations.

The main group of the tested methods involves the quasi-Newton methods. It was shown that a significant increase of stability of the local contact search can be achieved by these methods. Tab. 5.1 shows that the best results has been achieved by the BFGS and the DFP methods.

Broyden's method also improves the convergence properties of the Newton-Raphson. However, there is no guarantee of the positive definiteness of the Hessian matrix, it was overcome by more sophisticated quasi-Newton methods like the BFGS and the DFP methods. Note that well known method of steepest descent was also tested but with results of average convergence rate.

The line-search is crucial for a general success of quasi-Newton and gradient methods. It significantly increases the effectiveness of methods with one exception of point $(-1, 1)$. The treatment of this case will be investigated in further work.

Although the CPU times of the least-square projection method are comparable with the Newton-Raphson, this method is not suitable for the local contact search, since it converges to the global minimum. This interesting feature of the least-square projection method was also observed for other contact segment topologies.

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 line-search algorithm, we selected this method for the local contact search procedure. The implementation of the BFGS method into the FE code PMD will be the objective of further work.

	Initial Guess													
	(0.83, −0.62)		(−1, −1)		(1, −1)		(1, 1)		(−1, 1)		(0, 0)		(−2.8, 3)	
	NI	$t\,[s]$	NI	$t\,[s]$	NI	$t\,[s]$	NI	$t\,[s]$	NI	$t\,[s]$	NI	$t\,[s]$	NI	$t\,[s]$
NR	28	0.0366	12	0.0163	17sp	0.0223	5sp	0.0073	12sp	0.0162	5	0.0074	9sp	0.0130
LSP	133	0.0160	137	0.0168	124	0.0154	116	0.0143	134	0.0167	134	0.0165	124	0.0163
SD	51	0.1647	28	0.0923	14	0.0445	29	0.0951	1543	5.2326	71	0.2231	13	0.0414
Broyden	9	0.0176	10	0.0187	9	0.0169	11sp	0.0203	14900	29.981	11	0.0212	10	0.0193
BFGS	**8**	**0.0136**	**8**	**0.0149**	**7**	**0.0132**	**9sp**	**0.0174**	**23**	**0.0444**	**8**	**0.0154**	**8**	**0.0149**
DFP	**7**	**0.0127**	**8**	**0.0148**	**7**	**0.0133**	**9sp**	**0.0166**	**26**	**0.0469**	**8**	**0.0144**	**8**	**0.0150**
Simplex	190	0.2472	192	0.2472	184	0.2504	210	0.2786	196	0.2530	200	0.2609	190	0.2381

Table 5.1: Efficiency assessment of methods for the local contact search.

BIBLIOGRAPHY

[1] *PMD version f77.9, VAMET/Institute of Thermomechanics*, 2003.

[2] K. J. Bathe. *Finite Element Procedures.* Prentice-Hall, 1996.

[3] D. J. Benson and J. O. Hallquist. A single surface contact algorithm for the post-buckling analysis of shell structures. *Computer Methods in Applied Mechanics and Engineering*, 78:141–163, 1990.

[4] D. Gabriel. *Numerical Solution of Large Displacement Contact Problems by the Finite Element Method.* PhD thesis, Czech Technical University in Prague, 2007.

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

[6] P. Lederer. *Teorie a optimalizace mechanických systémů I.* ČVUT Praha, 1988.

[7] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[8] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, 1999.

[9] A. L. Peressini, F. E. Sulliven, and J. J. Uhl. *The Mathematics in Nonlinear Programming.* Spinger, 1988.

[10] W. Rudin. *Principles of Mathematical Analysis.* McGraw-Hill, 1976.