INTERIOR POINT ALGORITHMS FOR 3D CONTACT PROBLEMS

Radek Kučera[*], Jitka Machalová, Pavel Ženčák

1. Introduction

We shall be concerned with solving

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\, x^\top A x - x^\top b, \\ \text{subject to} & x_{1,i} \geq l_i,\ x_{2,i}^2 + x_{3,i}^2 \leq g_i^2,\ i = 1, \ldots, m, \\ & x = (x_1^\top, x_2^\top, x_3^\top)^\top \in \mathbb{R}^n, \end{array} \tag{1}$$

where $|x_1| = |x_2| = |x_3| = m$, $n = 3m$, $A \in \mathbb{R}^{n \times n}$ is the symmetric, positive definite Hessian matrix, $b \in \mathbb{R}^n$, and $l, g \in \mathbb{R}^m$. This problem arises, e.g., in duality based methods for the solution of 3D contact problems of linear elasticity with Tresca friction. As a widely used approach of contact problems with (more realistic) Coulomb friction is based on a sequence of Tresca friction problems [2], an efficient solver for (1) is of crucial importance. In this contribution we shall test algorithms based on an "interior point" idea.

2. Description of algorithms

The solution to (1) exists, and it is necessarily unique. We denote it by x^*. It is well-known [1] that x^* is fully determined by the Karush-Kuhn-Tucker (KKT) conditions. The basic idea of interior point methods consists in applying Newton iterations to equalities in the KKT conditions while inequalities are satisfied strictly by damping Newton steps.

Let us introduce the Lagrangian $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{2m} \mapsto \mathbb{R}$ associated with (1) by

$$\mathcal{L}(x, \lambda, \mu) = \frac{1}{2} x^\top A x - x^\top b + \lambda^\top (l - x_1) + \mu^\top (X_2^2 + X_3^2 - G^2) e,$$

where $X_2, X_3, G \in \mathbb{R}^{m \times m}$ are defined by $X_2 = diag(x_2)$, $X_3 = diag(x_3)$, $G = diag(g)$, and $e = (1, \ldots, 1)^\top \in \mathbb{R}^m$. There is $y^* := (\lambda^*, s^*, \mu^*, d^*) \in \mathbb{R}^{4m}$ so that the pair (x^*, y^*) is the unique solution to the following system:

$$\frac{\partial \mathcal{L}}{\partial x}(x, \lambda, \mu) = 0,\ \frac{\partial \mathcal{L}}{\partial \lambda}(x, \lambda, \mu) + s = 0,\ \lambda^\top s = 0,\ \frac{\partial \mathcal{L}}{\partial \mu}(x, \lambda, \mu) + d = 0,\ \mu^\top d = 0, \tag{2}$$

$$\lambda \geq 0,\ s \geq 0,\ \mu \geq 0,\ d \geq 0. \tag{3}$$

[*] Supported by grants GAČR 101/08/0574, and MSM6198910027.

Here, λ, and μ are the *Lagrange multipliers* while s, and d are the *slack variables*. The classical KKT conditions can be derived from (2), (3) by eliminating the slack variables.

Let us divide A, b into blocks A_{ij}, b_i, $i, j \in \{1, 2, 3\}$ consistently with the partition of x onto x_1, x_2, x_3. We can equivalently rewrite (2), (3) as

$$F(x, y) = 0, \quad y \geq 0, \tag{4}$$

where $y := (\lambda^\top, s^\top, \mu^\top, d^\top)^\top$, and $F : \mathbb{R}^{n+4m} \mapsto \mathbb{R}^{n+4m}$,

$$F(x, y) := \left(\begin{array}{l} A_{11}x_1 + A_{12}x_2 + A_{13}x_3 - \lambda - b_1 \\ A_{21}x_1 + (A_{22} + 2M)x_2 + A_{23}x_3 - b_2 \\ A_{31}x_1 + A_{32}x_2 + (A_{33} + 2M)x_3 - b_3 \\ \hline -x_1 + s + l \\ \Lambda S e \\ (X_2^2 + X_3^2 - G^2)e + d \\ MDe \end{array} \right)$$

with $\Lambda, S, M, D \in \mathbb{R}^{m \times m}$, $\Lambda = diag(\lambda)$, $S = diag(s)$, $M = diag(\mu)$, $D = diag(d)$. The Jacobi matrix to F reads as follows:

$$J(x, y) = \left(\begin{array}{ccc|cccc} A_{11} & A_{12} & A_{13} & -I & 0 & 0 & 0 \\ A_{21} & A_{22} + 2M & A_{23} & 0 & 0 & 2X_2 & 0 \\ A_{31} & A_{32} & A_{33} + 2M & 0 & 0 & 2X_3 & 0 \\ \hline -I & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & S & \Lambda & 0 & 0 \\ 0 & 2X_2 & 2X_3 & 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 & 0 & D & M \end{array} \right). \tag{5}$$

Let $(x^{(k)}, y^{(k)})$, $y^{(k)} > 0$ be a known approximation of (x^*, y^*). The Newton direction is the solution to the linear system

$$J(x^{(k)}, y^{(k)}) \begin{pmatrix} \Delta x^{(k+1)} \\ \Delta y^{(k+1)} \end{pmatrix} = -F(x^{(k)}, y^{(k)}), \tag{6}$$

and the new iterate is

$$(x^{(k+1)}, y^{(k+1)}) = (x^{(k)}, y^{(k)}) + \alpha^{(k)}(\Delta x^{(k+1)}, \Delta y^{(k+1)}), \tag{7}$$

where $\alpha^{(k)} = \min_{\Delta y_i^{(k+1)} < 0}\{1, -\delta y_i^{(k)}/\Delta y_i^{(k+1)}\}$ with $\delta \in (0, 1]$ providing $y^{(k+1)} > 0$ (typically $\delta = 0.999$).

The computations based on (6), (7) can take short steps before violating $y^{(k+1)} > 0$ so that the convergence rate can be slow. Therefore we use two modifications [7] called *path following method*, and (Mehrotra's) *predictor-corrector method* keeping iterates deeper in the feasible region so that it enables us to perform longer steps.

Let us replace (4) by

$$F(x,y) = (0^\top, 0^\top, 0^\top, 0^\top, \tau e^\top, 0^\top, \tau e^\top)^\top, \quad y > 0, \tag{8}$$

where $\tau > 0$. Solutions (x^τ, y^τ) to (8) define in $\mathbb{R}^n \times \mathbb{R}^{4m}_+$ a curve $\mathcal{C}(\tau)$ called *central path* that leads to (x^*, y^*) as τ tends to zero. The next algorithm combines Newton iterations for the equation in (8) with changes of τ so that the (modified) Newton steps follow $\mathcal{C}(\tau)$.

<u>Algorithm PF</u>: Given $x^{(0)} \in \mathbb{R}^n$, $y^{(0)} \in \mathbb{R}^{4m}_+$, $\sigma_l, \sigma_q, \delta \in [0,1]$, and $\epsilon \geq 0$. Set $k := 0$.

(1) Compute $\beta_l^{(k)} = \lambda^{(k)\top} s^{(k)}/m$, $\beta_q^{(k)} = \mu^{(k)\top} d^{(k)}/m$, and $\tau_l^{(k)} = \sigma_l \beta_l^{(k)}$, $\tau_q^{(k)} = \sigma_q \beta_q^{(k)}$.

Solve

$$J(x^{(k)}, y^{(k)}) \begin{pmatrix} \Delta x^{(k+1)} \\ \Delta y^{(k+1)} \end{pmatrix} = -F(x^{(k)}, y^{(k)}) + (0^\top, 0^\top, 0^\top, 0^\top, \tau_l^{(k)} e^\top, 0^\top, \tau_q^{(k)} e^\top)^\top, \tag{9}$$

and generate $(x^{(k+1)}, y^{(k+1)})$ by (7).

(2) If $\|(\Delta x^{(k+1)}, \Delta y^{(k+1)})\|_{\mathbb{R}^{n+4m}} \leq \epsilon$, return $(\bar{x}, \bar{y}) = (x^{(k+1)}, y^{(k+1)})$, else set $k := k+1$, and go to step (1).

The parameters $\beta_l^{(k)}, \beta_q^{(k)}$, and σ_l, σ_q are called *duality measures*, and *centering parameters*, respectively. Let us note that $\sigma_l = \sigma_q = 0$ reduces Algorithm PF to the standard (damped) Newton method. Our choices of σ_l, and σ_q are based on the rule proposed in [4]:

$$\sigma_l = (\min\{2 \cdot 10^{-3}, 5 \cdot 10^{-5}(1 - \xi_l)/\xi_l\})^3,$$

where $\xi_l = \min_{i=1,\dots,m}\{\lambda_i^{(k)} s_i^{(k)}\}/\beta_l^{(k)}$, and analogously for σ_q.

The second algorithm calculates centering parameters adaptively using second order information (curvature) of the central path $\mathcal{C}(\tau)$. First, in the predictor stage, we compute duality measures β_l^P, β_q^P for the longest step of the (standard) Newton direction. Then, in the corrector stage, we set σ_l, σ_q near 0, when the good progress along the predicted direction is made, or near 1 conversely.

<u>Algorithm PC</u>: Given $x^{(0)} \in \mathbb{R}^n$, $y^{(0)} \in \mathbb{R}^{4m}_+$, $\delta \in (0,1)$, and $\epsilon \geq 0$. Set $k := 0$.

(1) Solve

$$J(x^{(k)}, y^{(k)}) \begin{pmatrix} \Delta x^P \\ \Delta y^P \end{pmatrix} = -F(x^{(k)}, y^{(k)}),$$

compute $\alpha^P = \min_{\Delta y_i^P < 0}\{1, -y_i^{(k)}/\Delta y_i^P\}$, and

$$\begin{aligned} \beta_l^P &= (\lambda^{(k)} + \alpha^P \Delta\lambda^P)^\top (s^{(k)} + \alpha^P \Delta s^P)/m, \\ \beta_q^P &= (\mu^{(k)} + \alpha^P \Delta\mu^P)^\top (d^{(k)} + \alpha^P \Delta d^P)/m. \end{aligned}$$

(2) Set $\sigma_l = \left(\beta_l^P/\beta_l^{(k)}\right)^3$, $\sigma_q = \left(\beta_q^P/\beta_q^{(k)}\right)^3$, compute $(\Delta x^{(k+1)}, \Delta y^{(k+1)})$ solving (9) with the right-hand-side replaced by

$$-F(x^{(k)}, y^{(k)}) + (0^\top, 0^\top, 0^\top, 0^\top, -e^\top \Delta\Lambda^P \Delta S^P + \tau_l^{(k)} e^\top, 0^\top, -e^\top \Delta M^P \Delta D^P e + \tau_q^{(k)} e^\top)^\top,$$

and generate $(x^{(k+1)}, y^{(k+1)})$ by (7).

(3) If $\|(\Delta x^{(k+1)}, \Delta y^{(k+1)})\|_{\mathbb{R}^{n+4m}} \leq \epsilon$, return $(\bar{x}, \bar{y}) = (x^{(k+1)}, y^{(k+1)})$, else set $k := k+1$, and go to step (1).

3. Numerical experiments

3.1 Model problem. Let us consider a steel brick in $\mathbb{R}^3$ lying on a rigid foundation. The brick occupies the domain $\Omega = (0,3) \times (0,1) \times (0,1)$, whose boundary $\partial\Omega$ split into three nonempty disjoint parts $\Gamma_u = \{0\} \times (0,1) \times (0,1)$, $\Gamma_c = (0,3) \times (0,1) \times \{0\}$, and $\Gamma_p = \partial\Omega \setminus (\bar{\Gamma}_u \cup \bar{\Gamma}_c)$ with different boundary conditions. The zero displacements are prescribed on Γ_u, whereas the surface tractions act on Γ_p. On Γ_c we consider the contact conditions, i.e., the non-penetration, and the effect of friction. The elastic behavior of the brick is described by Lamé equations that, after finite element discretization, lead to a symmetric positive definite stiffness matrix $K \in \mathbb{R}^{3n_c \times 3n_c}$, and to a load vector $f \in \mathbb{R}^{3n_c}$. Moreover, we introduce full rank matrices $N, T_1, T_2 \in \mathbb{R}^{m \times 3n_c}$ projecting displacements at contact nodes to normal and tangential directions, respectively, and we denote $B = \left(N^\top, T_1^\top, T_2^\top\right)^\top \in \mathbb{R}^{3m \times 3n_c}$. For more details about this model problem we refer to [2].

Here, we shall use the dual formulation in terms of contact stresses. Considering only *Tresca friction*, our model problem reduces directly to (1), where $A = BK^{-1}B^\top$, $b = BK^{-1}f$, $l = 0$, and $g_i \geq 0$ are given slip bound values at contact nodes. Let us note that unknowns x_1, and x_2, x_3 represent the normal, and tangential contact stresses, respectively.

3.2 Inner solver. Our algorithms require repeatedly to solve the linear systems

$$J(x,y) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} r_x \\ r_y \end{pmatrix} \tag{10}$$

with the Jacobi matrix given by (5), and $r_x \in \mathbb{R}^n$, $r_y = (r_\lambda^\top, r_s^\top, r_\mu^\top, r_d^\top)^\top \in \mathbb{R}^{4m}$, $\Delta y = (\Delta\lambda^\top, \Delta s^\top, \Delta\mu^\top, \Delta d^\top)^\top \in \mathbb{R}^{4m}$. First we compute the solution to the reduced system arising from (10) by eliminating increments with respect to the slack variables:

$$J_R(x,y) \begin{pmatrix} \Delta x \\ \Delta z \end{pmatrix} = \begin{pmatrix} r_x \\ r_z \end{pmatrix}, \tag{11}$$

where

$$J_R(x,y) = \left(\begin{array}{ccc|cc} A_{11} & A_{12} & A_{13} & -I & 0 \\ A_{21} & A_{22} + 2M & A_{23} & 0 & 2X_2 \\ A_{31} & A_{32} & A_{33} + 2M & 0 & 2X_3 \\ \hline -I & 0 & 0 & -\Lambda^{-1}S & 0 \\ 0 & 2X_2 & 2X_3 & 0 & -M^{-1}D \end{array}\right),$$

and $r_z = (r_\lambda^\top - r_s^\top \Lambda^{-1}, r_\mu^\top - r_d^\top M^{-1})^\top \in \mathbb{R}^{2m}$, $\Delta z = (\Delta\lambda^\top, \Delta\mu^\top)^\top \in \mathbb{R}^{2m}$. Then we obtain the eliminated components by $\Delta s = \Lambda^{-1}(r_s - S\Delta\lambda)$, and $\Delta d = M^{-1}(r_d - D\Delta\mu)$. It is easy to prove that the Schur complement with respect to the second diagonal block in

$J_R(x, y)$ is positive definite provided $y > 0$. Therefore $J_R(x, y)$ is non-singular but indefinite. In order to solve (11), we apply the conjugate gradient method with an appropriate preconditioning [5, 6].

3.3 Tests. We compare the algorithms PF, and PC with the one presented in [3], here denoted by QPC. For various numbers of the primal, and dual degrees of freedoms ($3n_c/3m$), we report the computational time (*time*), the total number of the matrix-vector multiplications (n_A), and, in case of the interior point algorithms, the number of the outer iterations (*out*), and the number of the full steps (*full*), i.e. the steps with $\alpha^{(k)} = 1$. All computations are performed by Matlab 7 on Pentium 4, 2.8 GHz with 1GB RAM.

The first experiments in Table 3.1 demonstrate the computational strategy in which the Hessian matrix A is assembled (only in PF, and PC). As the time consumed by assembling A predominates for larger $3n_c/3m$, this strategy seems to be non-acceptable form more realistic contact problems.

	QPC		PF		PC	
$3n_c/3m$	*time*	n_A	*time*	$time_A$	*time*	$time_A$
162/54	0.29	203	0.07	0.03 (45%)	0.12	0.03 (26%)
900/180	2.08	311	0.68	0.34 (50%)	1.07	0.34 (31%)
2646/378	12.91	347	5.85	3.46 (59%)	7.00	3.26 (47%)
5832/648	53.4	384	27.1	18.1 (67%)	27.0	15.8 (59%)
10890/990	126.2	408	79.7	58.5 (73%)	90.0	60.5 (67%)
18252/1404	361.9	493	246.2	192.5 (78%)	274.0	184 (67%)
28350/1890	809.4	478	620.5	493.0 (79%)	677.6	493.5 (73%)

Table 3.1: A is assembled in PF, and PC ($time_A$ is consumed by assembling A).

	QPC		PF, precond. 1				PC, precond. 1			
$3n_c/3m$	*time*	n_A	*time*	n_A	*out*	*full*	*time*	n_A	*out*	*full*
162/54	0.29	203	0.25	97	19	11	0.30	195	27	16
900/180	2.08	311	0.94	112	20	12	1.20	154	17	11
2646/378	12.91	347	6.69	139	22	13	6.33	147	13	8
5832/648	53.4	384	29.33	173	25	13	23.8	157	13	7
10890/990	126.2	408	109.5	233	30	13	68.3	159	13	6
18252/1404	361.9	493	265.3	244	31	12	177.7	183	14	7
28350/1890	809.4	478	644.2	282	36	13	420.9	209	16	7

Table 3.2: A is not assembled, the preconditioner 1 is assembled.

In Tables 3.2, and 3.3 we present the computational efficiency of PF, and PC with non-assembled A. We test two preconditioners for (10) based on an approximation of a

	QPC		PF, precond. 2				PC, precond. 2			
$3n_c/3m$	*time*	n_A	*time*	n_A	*out*	*full*	*time*	n_A	*out*	*full*
162/54	0.29	203	0.28	154	27	12	0.23	148	16	12
900/180	2.08	311	0.98	112	20	14	1.05	138	13	8
2646/378	12.91	347	6.61	133	22	13	5.5	122	12	8
5832/648	53.4	384	26.6	150	24	13	22.6	145	13	8
10890/990	126.2	408	105.4	218	30	13	66.4	151	13	7
18252/1404	361.9	493	253.2	224	31	13	169.8	169	14	7
28350/1890	809.4	478	584.7	251	33	12	397.1	190	15	7

Table 3.3: A is not assembled, the preconditioner 2 is assembled.

Schur complement to the stiffness matrix K. While the first preconditioner uses diagonal scaling, the second one requires more expensive computations [6]. Let us note that both preconditioners are assembled.

4. Conclusions

In the contribution we present our first experience with solving contact problems by the interior point algorithms. As the results seem promising many questions are still open, namely the convergence prove of both algorithms. The future work consists also of applying non-assembled preconditioners, and of implementing theoretically supported inner solvers.

References

[1] S. Boyd, L. Vandenberghe: *Convex optimization.* Cambridge University Press, Cambridge 2004.

[2] J. Haslinger, R. Kučera, and Z. Dostál: An algorithm for the numerical realization of 3D contact problems with Coulomb friction. J. Comput. Appl. Math., **164-165** (2004), pp. 387–408.

[3] R. Kučera: Convergence rate of an optimization algorithm for minimizing quadratic functions with separable convex constraints. SIAM J. Optim., **19** (2008), pp. 846–862.

[4] J. Nocedal, A. Wächter, and R. A. Waltz: Adaptive barrier strategies for nonlinear interior methods. TR RC 23563, IBM T.J. Watson Research Center, 2005.

[5] L. Lukšan, C. Matonoha, J. Vlček: Interior point methods for large-scale nonlinear programming. TR No. 917, Academy of Sciences of the Czech Republic 2004.

[6] J. Machalová, P. Ženčák, and R. Kučera: Metody vnitřních bodu pro řešení úlohy nelineárního programování. ODAM 2007, pp. 4–17.

[7] J. Nocedal, S. J. Wright: *Numerical optimization.* Springer, New York 1999.