High-accuracy computation of rolling friction contact problems

Vincent Acary, Paul Armand, Hoang Minh Nguyen

High-accuracy computation of rolling friction contact problems

Vincent Acary
Univ. Grenoble Alpes, Inria
CNRS, Grenoble INP
Institute of Engineering, LJK
Grenoble, France
vincent.acary@inria.fr

Paul Armand
Univ. de Limoges,
Laboratoire XLIM
UMR CNRS n°7252
Limoges, France
paul.armand@unilim.fr

Hoang Minh Nguyen
Univ. Grenoble Alpes, Inria
CNRS, Grenoble INP
Institute of Engineering, LJK
Grenoble, France
hoang-minh.nguyen@inria.fr

***Abstract*—Our aim is to perform numerical solutions of an optimization model derived from the problem of unilateral contact between solid bodies with rolling friction. The model is an optimization problem with a strictly convex quadratic objective function and a second-order cone of constraints that is not self-dual. The solver is an implementation of a primal-dual interior-point algorithm with the predictor-corrector scheme of Mehrotra extended to second-order cone problem. We focused on the analysis of the limits of numerical computation and propose some treatments to achieve optimal solutions with ten significant digits precision.**

***Keywords*— optimization, Jordan algebra, second-order cone optimization, Nesterov and Todd scaling, rolling friction contact**

I. Introduction

Acary and Bourrier presented in [1] a general model for the problem of unilateral contact between solid bodies with rolling friction. Let $n, m \in \mathbb{N}$ be the number of contact points and degrees of freedom. Let $d = 5$ be the dimension of the velocity and reaction vectors at a contact point. In this paper, we consider only a convex relaxation of this model as

$$\begin{aligned} &Mv + f = H^\top r, \\ &Hv + w = u, \\ &\mathcal{F}^* \ni u \perp r \in \mathcal{F}, \end{aligned} \tag{1}$$

where $M \in \mathbb{R}^{m\times m}$ is symmetric and positive-definite, $f \in \mathbb{R}^m$, $H \in \mathbb{R}^{nd\times m}$, $w \in \mathbb{R}^{nd}$. The vector $r \in \mathbb{R}^{nd}$ is the reaction force at contact, $u \in \mathbb{R}^{nd}$ is the velocity of the contact point and $v \in \mathbb{R}^m$ is the velocity of the system. The *rolling friction cone* is defined by

$$\mathcal{F} = \prod_{i=1}^{n} \mathcal{R}_i := \left\{ r_i = \begin{bmatrix} r_{i_0} \\ \bar{r}_i \\ \tilde{r}_i \end{bmatrix} \in \mathbb{R}^d : r_{i_0} \geq \max\{\|\bar{r}_i\|, \|\tilde{r}_i\|\} \right\}.$$

The dual cone is then

$$\mathcal{F}^* = \prod_{i=1}^{n} \mathcal{R}_i^* := \left\{ u_i = \begin{bmatrix} u_{i_0} \\ \bar{u}_i \\ \tilde{u}_i \end{bmatrix} \in \mathbb{R}^d : u_{i_0} \geq \|\bar{u}_i\| + \|\tilde{u}_i\| \right\}.$$

This cone is not self-dual since $\mathcal{F} \neq \mathcal{F}^*$, a significant difference with the frictional contact (no rolling) framework [3]. The system (1) coincides with the first order optimality conditions of the primal-dual pair of convex problems:

$$\begin{array}{ll} \min\limits_{v,u} & \frac{1}{2} v^\top M v + f^\top v \\ \text{s.t.} & Hv + w = u, \\ & u \in \mathcal{F}^*, \end{array} \qquad \begin{array}{ll} \max\limits_{v,r} & -\frac{1}{2} v^\top M v - w^\top r \\ \text{s.t.} & H^\top r - Mv = f, \\ & r \in \mathcal{F}. \end{array} \tag{2}$$

After the introduction of artificial variables and a reformulation of the constraints, we will show that these problems fall in the class of second-order cone optimization (SOCO) problems. Thus, a primal-dual interior-point method (IPM) is an appropriate choice to efficiently solve them. This paper proposes an adapted IPM for the solution of (2), in order to accurately solve (1). We do it by means of a `C/C++` implementation of the algorithm in SICONOS[1] and perform tests on FCLIB [3], a collection of discrete rolling frictional contact problems[2].

The paper is organized as follows. Section II shows the rolling friction contact model and the system of equations solved by means of the interior point algorithm described in Section III. The numerical difficulties are analyzed in Section IV. The robustness of our implementation is highlighted by the numerical results presented in Section V. The paper ends with a conclusion in Section VI and some basics on Euclidean Jordan algebra as well as the notation in Appendix.

II. Rolling friction contact model

The cone appearing in a standard SOCO problem, usually the Lorentz cone, is self-dual. This allows to take advantage of the Jordan algebra. But for the rolling friction problems, this is not the case, the cone of constraints is not self-dual. There is no Jordan product such that $\mathcal{R}_i^*$ is a cone of squares, an important property to extend IPM from linear optimization to SOCO [2]. To deal with this difficulty, we propose to transfom the conical constraints of Problem (2) into the product of Lorentz cones by means of artificial variables. Knowing that, for a triplet (a, b, c) of real numbers, $a \geq b + c$ if and only if there exists $t, t' \in \mathbb{R}$ such that $t \geq b, t' \geq c$ and $a = t + t'$. We apply this statement by setting $u_{i,0} = t_i + t'_i$ for all $i \in \{1, \ldots, n\}$. Then, the primal-dual Problem (2) can be reformulated as

$$\begin{array}{ll} \min\limits_{v,z} & \frac{1}{2} v^\top M v + f^\top v \\ \text{s.t.} & Hv + w = Jz, \\ & z \in \mathcal{L}^{2n} \end{array} \qquad \begin{array}{ll} \max\limits_{v,r} & -\frac{1}{2} v^\top M v - w^\top r \\ \text{s.t.} & H^\top r - Mv = f \\ & J^\top r \in \mathcal{L}^{2n}, \end{array} \tag{3}$$

where $z \in \mathbb{R}^{n\times(d+1)}$ is a vector with n components of the form $(t_i; \bar{u}_i; t'_i; \tilde{u}_i)$, $i \in \{1, \ldots, n\}$ and

$$J = \bigoplus_{i=1}^{n} \jmath, \text{ with } \jmath = \begin{bmatrix} 1 & & 1 \\ & I & \\ & & I \end{bmatrix} \in \mathbb{R}^{d\times(d+1)}.$$

By virtue of [2, Lemma 15], the orthogonality condition of (1) becomes $z \circ J^\top r = 0$, with $z \in (\mathcal{L}^{2n})^* = \mathcal{L}^{2n}$ and $J^\top r \in \mathcal{L}^{2n}$. This representation of (1) provides a square system of equalities.

[1] https://nonsmooth.gricad-pages.univ-grenoble-alpes.fr/siconos

[2] https://github.com/FrictionalContactLibrary/fclib-library

It is assumed that the Slater hypothesis holds, i.e., there exists $v \in \mathbb{R}^m$ such that $Hv + w \in \text{int}(\mathcal{F}^*)$. Under this assumption, $(r, u) \in \mathbb{R}^{2nd}$ is a primal-dual optimal solution of Problem (2) if and only if there exists $(t, t') \in \mathbb{R}^{2n}$ such that the Karush-Kuhn-Tucker (KKT) are satisfied:

$$\begin{aligned} &Mv + f = H^\top r, \\ &Hv + w = Jz, \\ &z \circ J^\top r = \mathbf{0}_{n(d+1)}, \\ &(z, J^\top r) \in \mathcal{L}^{4n}. \end{aligned} \tag{4}$$

In practice, the Newton method is applied to a perturbation of (4), such that the solution is maintained in the interior of the cone of constraints. The perturbation is to replace the complementarity conditions by $z \circ J^\top r = 2\mu \boldsymbol{e}$, where the perturbation parameter μ is the *duality measure* defined by

$$\mu = \frac{z^\top J^\top r}{n}.$$

The perturbed KKT system associated to Problem (3) is then

$$\begin{aligned} &Mv + f = H^\top r, \\ &Hv + w = Jz, \\ &z \circ J^\top r = 2\mu \boldsymbol{e}, \\ &(z, J^\top r) \in \text{int}(\mathcal{L}^{4n}). \end{aligned} \tag{5}$$

It can be shown that the system (5) has a unique solution $(z(\mu), J^\top r(\mu)) \in \text{int}(\mathcal{L}^{4n})$ for all $\mu > 0$. The corresponding curve, parametrized by μ, is called the central path. It can be shown that under the Slater hypothesis, when $\mu \downarrow 0$, the central path converges to a relative interior point of the set of primal-dual solutions of (3).

III. Primal-dual Interior-point approach

Basically, an interior point algorithm consists of successive linearizations of (5), while driving μ to zero and keeping the iterates z and $J^\top r$ in the interior of the Lorentz cones. The linearization of equations (5) leads to the following linear system

$$\begin{bmatrix} M & -H^\top & 0 \\ -H & 0 & J \\ 0 & ZJ^\top & R \end{bmatrix} \begin{bmatrix} d^v \\ d^r \\ d^z \end{bmatrix} = - \begin{bmatrix} Mv + f - H^\top r \\ Jz - Hv - w \\ z \circ J^\top r - 2\mu \boldsymbol{e} \end{bmatrix}, \tag{6}$$

where $Z = \text{Arw}(z)$ and $R = \text{Arw}(J^\top r)$. Unfortunately, this system can be singular, even if the matrices Z and R are positive definite, see [8]. This is due to the fact that the arrow matrices do not commute, except if they share a common set of eigenvectors. To overcome this drawback, the idea is to make a local change of variables, which keeps invariant the Lorentz cone as well as its interior and so that, in the new space, the arrow matrices commute. An extensive literature has discussed several choices of variable changes and it is commonly agreed that the best one for solving a SOCO problem is the Nesterov and Todd (NT) scaling scheme [7]. It is defined as follows. Let $\bar{p}, \tilde{p} \in \mathbb{R}^{nd}$ be the NT directions defined with n conic components of the following form: for $i \in \{1, \ldots, n\}$,

$$\begin{aligned} \bar{p}_i &= \left[Q_{(t_i;\bar{u}_i)^{1/2}} \left(Q_{(t_i;\bar{u}_i)^{1/2}} (r_{i_0}; \bar{r}_i) \right)^{-1/2} \right]^{-1/2}, \\ \tilde{p}_i &= \left[Q_{(t'_i;\bar{u}_i)^{1/2}} \left(Q_{(t'_i;\bar{u}_i)^{1/2}} (r_{i_0}; \tilde{i}_i) \right)^{-1/2} \right]^{-1/2}. \end{aligned} \tag{7}$$

Let us define the block-diagonal matrix $Q_p = \bigoplus_{i=1}^n (Q_{\bar{p}_i} \oplus Q_{\tilde{p}_i})$. The change of variables is then

$$\hat{z} := Q_p z \quad \text{and} \quad \check{y} := Q_{p^{-1}} J^\top r.$$

A fundamental property of the NT scaling is that $\hat{z} = \check{y}$, see [2, p.42], which implies that the corresponding arrow matrices commute.

After the change of variables, we obtain the corresponding primal-dual problems

$$\begin{aligned} \min_{v,\hat{z}} \ & \frac{1}{2} v^\top M v + f^\top v \\ \text{s.t.} \ & Hv + w = JQ_{p^{-1}} \hat{z}, \\ & \hat{z} \in \mathcal{L}^{2n}, \end{aligned} \qquad \begin{aligned} \max_{v,\check{y}} \ & -\frac{1}{2} v^\top M v - w^\top K Q_p \check{y} \\ \text{s.t.} \ & H^\top K Q_p \check{y} - Mv = f, \\ & \check{y} \in \mathcal{L}^{2n}, \end{aligned} \tag{8}$$

where

$$K = \bigoplus_{i=1}^n k, \text{ with } k = \begin{bmatrix} 1 & & 0 \\ & I & \\ 0 & & I \end{bmatrix} \in \mathbb{R}^{d \times (d+1)}.$$

Note that $KJ^\top = I$. After some obvious simplifications, the new pertubed KKT system associated to (8) is given by

$$\begin{aligned} &Mv + f = H^\top r, \\ &Hv + w = Jz, \\ &\hat{z} \circ \check{y} = 2\mu \boldsymbol{e}, \\ &(\hat{z}, \check{y}) \in \text{int}(\mathcal{L}^{4n}). \end{aligned} \tag{9}$$

The linearization with respect to the variables v, r and z of this new system leads to the following linear system:

$$\begin{bmatrix} M & -H^\top & 0 \\ -H & 0 & J \\ 0 & \hat{Z} Q_{p^{-1}} J^\top & \check{Y} Q_p \end{bmatrix} \begin{bmatrix} d^v \\ d^r \\ d^z \end{bmatrix} = - \begin{bmatrix} Mv + f - H^\top r \\ Jz - Hv - w \\ \hat{z} \circ \check{y} - 2\mu \boldsymbol{e} \end{bmatrix}, \tag{10}$$

where $\hat{Z} := \text{Arw}(\hat{z}) = \text{Arw}(\check{y}) =: \check{Y}$. By left-multiplying the third equation by $Q_p \hat{Z}^{-1}$ and after some simplications, we get the following equivalent symmetric linear system:

$$\begin{bmatrix} M & -H^\top & 0 \\ -H & 0 & J \\ 0 & J^\top & Q_{p^2} \end{bmatrix} \begin{bmatrix} d^v \\ d^r \\ d^z \end{bmatrix} = - \begin{bmatrix} Mv + f - H^\top r \\ Jz - Hv - w \\ y - 2\mu z^{-1} \end{bmatrix}. \tag{11}$$

Algorithm 1 is the implemented algorithm that was used to perform our numerical tests. This is the primal-dual interior point method of Mehrotra [6], extended to the solution of a SOCO problem [7]. At each iteration of Algorithm 1, two linear systems are solved, with two different right-hand-sides (RHS). The first one corresponds to a pure Newton direction, called the *affine scaling direction* and denoted by (d_a^v, d_a^r, d_a^z). It satisfies the linear system (6) (resp. (11)) with $\mu = 0$. The second one corresponds to the computation of a linear combination of the affine scaling direction and a correction step, that is used to force the iterates to get close to the central path. See [5, §14.2] for details. As a result, when solving (6), the third equation is replaced by the following one:

$$ZJ^\top d^r + Rd^z = -z \circ y - d_a^{z\top} (J^\top d_a^r) + 2\sigma\mu \boldsymbol{e}, \tag{12}$$

where $\sigma \in (0, 1)$ is the *centralization parameter*, i.e., the desired reduction factor of the duality measure. When solving (11), this equation becomes

$$J^\top d^r + Q_{p^2} d^z = -y - Q_p(\hat{z}^{-1} \circ (\hat{d}_a^z \circ \check{d}_a^y)) + 2\sigma\mu z^{-1}, \tag{13}$$

where $\hat{d}_a^z = Q_p d_a^z$ and $\check{d}_a^y = Q_{p^{-1}} J^\top d_a^r$.

Once a linear system is solved, a line search is performed so that the next iterate remains within the cone of constraints. Let $x \in \mathcal{L}^{4n}$ be the current iterate and let $d \in \mathcal{L}^{4n}$ be the corresponding solution of the linear system that was solved. The function `get_step_length`(x, d) returns the largest $\alpha \geq 0$ such that $x + \alpha d \in \mathcal{L}^{4n}$. The step length along the affine scaling direction is the largest value $\alpha \in (0, 1]$ such that $x + \alpha d \in \mathcal{L}^{4n}$. This value is used to compute the centralization parameter σ. Next, the step length along the full direction is the largest value $\alpha \in (0, 1]$ such that $x + \alpha d \in (1 - \gamma)x + \mathcal{L}^{4n}$. See the detailed formulas for this

computation in [7]. At the end of the iteration, the variables are updated at Step 26.

Algorithm 1 IPM with predictor-corrector scheme

1: Choose a tolerance $\epsilon > 0$. Set $\gamma = 0.9$.
2: Set a starting point $(v, r, z) \in \mathbb{R}^m \times \mathbb{R}^{nd} \times \mathbb{R}^{n(d+1)}$
3: **while** 1 **do**
4: pinfeas $= \dfrac{\|Jz - Hv - w\|}{\max\{\|Jz\|, \|Hv\|, \|w\|\}}$
5: dinfeas $= \dfrac{\|Mv + f - H^\top r\|}{\max\{\|Mv\|, \|f\|, \|H^\top r\|\}}$
6: $\mu = \frac{1}{n} z^\top J^\top r$
7: **if** $\max\{\text{pinfeas}, \text{dinfeas}, \mu\} \leq \epsilon$ **then**
8: BREAK
9: **end if**
10: ——— PREDICTOR Step ———
11: Solve (6) (resp. (11)) with $\mu = 0$ to obtain (d_a^v, d_a^r, d_a^z)
12: $\alpha = \texttt{get\_step\_length}((z, J^\top r), (d_a^z, J^\top d_a^r))$
13: Set $\alpha = \min\{1, \alpha\}$
14: ——— CORRECTOR Step ———
15: $\mu_a = \frac{1}{n}(Jz + \alpha J d_a^z)^\top (r + \alpha d_a^r)$
16: **if** $\mu < 10^{-5}$ **then**
17: $e = \max\{1, 3\alpha\}$
18: **else**
19: $e = 1$
20: **end if**
21: $\sigma = \min\{1, (\mu_a/\mu)^e\}$
22: Solve (6) (resp. (11) with the modified RHS (12) (rep.(13)) to obtain (d^v, d^r, d^z)
23: $\alpha = \texttt{get\_step\_length}((z, J^\top r), (d^z, J^\top d^r))$
24: Set $\alpha = \min\{1, \gamma\alpha\}$
25: Set $\gamma = 0.9 + 0.09\alpha$
26: Set $(v, r, z) = (v, r, z) + \alpha(d^v, d^r, d^z)$
27: **end while**

IV. Computational issues

Although the system (6) may be singular, we have performed numerical experiments with it. The singularity of the matrix is a rather rare event in practice. These first experiments serve as a kind of reference for future experiments, both in terms of number of iterations and CPU time. Since the matrix in (6) is not symmetric, an LU factorization is performed. The advantage of this "no-scaling" strategy is that it is simple to implement and computationally fast, because there is not a bunch of calculations due to the scaling. However, since an LU factorization is more expensive than a symmetric matrix factorization, this advantage should be lowered for large problems.

Our first experiments with the NT scaling scheme have been disastrous. This is due to the ill-conditioning of the matrix Q_p near an optimal solution. This behavior is well known in the community of SOCO and previous experiments with the software SDPT3 [7] warned us about it. We have been searching for a long time for a formulation of the linear system that allows us to obtain suitable results. The form (11) seems to us not too bad, but not enough robust. We propose an alternative equivalent 3×3 symmetric linear system, by left-multiplying the last equation of (11) by $Q_{p^{-1}}$ and setting $\hat{d}^z = Q_p d^z$:

$$\begin{bmatrix} M & -H^\top & 0 \\ -H & 0 & JQ_{p^{-1}} \\ 0 & Q_{p^{-1}}J^\top & I \end{bmatrix} \begin{bmatrix} d^v \\ d^r \\ \hat{d}^z \end{bmatrix} = - \begin{bmatrix} Mv + f - H^\top r \\ Jz - Hv - w \\ \check{y} - 2\mu\hat{z}^{-1} \end{bmatrix}. \quad (14)$$

The matrix of (14) is less ill-conditioned than the one of (11). Although the ill-conditioned matrix Q_p is introduced in the second equation, which leads to a loss of accuracy in the primal feasibility, the use of (14) allowed to solve 97% of the problems with a tolerance $\epsilon = 10^{-10}$.

Since the matrices in (11) and (14) are symmetric, an LDL factorization is done by means of the Harwell subroutine MA57 [9], an efficient code for the solution of sparse symmetric systems. Furthermore, in order to avoid the propagation of computational errors due to the ill-conditioning of Q_p, all calculations of the form $Q_x y$ are performed without explicitly building the vector x and the matrix Q_x. This is done, for example, to compute the vectors $\check{y}$ and $\hat{z}$ in the RHS of (14). We do it by using the formula $Q_x y = 2(x^\top y)x - (\det x)Ry$ and the spectral decompositions of the vectors x and y. In addition, to improve the accuracy, all these calculations are done with the floating-point data type `long double` instead of `double`. Indeed, a difficulty due to the computational limits of processors led us to use this type of data. In experiments with tight tolerances, when the algorithm returns an error, it is usually stopped by a computation involving a NaN (Not a Number). This comes from an invalid arithmetic operation, like a division by zero or the calculation of the square root of a negative number. Despite the fact that z and $J^\top r$ are kept in the interior of the Lorentz cones, when approaching the boundary of the constraints, the second eigenvalue (λ_2) of some conic components becomes almost zero or even equal to zero. This further justifies our choice to use the data type `long double`. However, the use of the `long double` data type may produce a negative λ_2. This is due to the fact that the algorithm is executed on 64-bit processors whose epsilon machine is in double precision, i.e., 16 digit precision [10].

We also tried some experiments with a solution of the linear system in a reduced form, which might have been preferred because of its smaller dimension. By eliminating the variable d^z in the system (11), a reduced version is given by

$$\begin{bmatrix} -M & H^\top \\ H & JQ_{p^{-2}}J^\top \end{bmatrix} \begin{bmatrix} d^v \\ d^r \end{bmatrix} = \begin{bmatrix} Mv - f - H^\top r \\ -Hv - w + 2\mu J y^{-1} \end{bmatrix}. \quad (15)$$

But, like as (11), the matrix of (15) is very ill-conditioned. Another reduction with a smaller condition number is given by

$$\begin{bmatrix} -M & \hat{H}^\top \\ \hat{H} & I \end{bmatrix} \begin{bmatrix} d^v \\ \underline{d}^r \end{bmatrix} = \begin{bmatrix} Mv + f - H^\top r \\ P^{-1}(-Hv - w + 2\mu J y^{-1}) \end{bmatrix}, \quad (16)$$

where $\hat{H} = P^{-1}H$ and $\underline{d}^r = P^\top d^r$. The matrix P is such that $JQ_{p^{-2}}J^\top = PP^\top$. This factorization can be done in several ways, such as Cholesky, LDU, etc. Whatever the formulation used, in addition to the numerical computation problems mentioned above, we again encounter numerous computations to perform at each iteration, which leads to propagating too many errors. Any reduction to a 2×2 form by eliminating the variable d^z always returned worse results.

V. Numerical results

Introduced in [3], three collections of 523 rolling friction problem tests have been carried out by Algorithm 1. Detailed descriptions of these problems are given in Table I.

TABLE I
Size of data set

Collection	Problems	Cones (n)	Degrees of fredom (m)
Chute	155	$[4, 1372]$	$[768, 6528]$
PrimitiveSoup	168	$[37, 2269]$	6000
SpherePile	200	$[2, 542]$	$[24, 1500]$

The performance profiles were introduced by Dolan and Moré [11] to get the benchmarking optimization solvers through a cumulative distribution function $\rho_s(\tau)$. It is the probability that the performance ratio of the solver s to the best solver does not exceed a factor 2^τ. The higher the probability is, the better the solver is. Figures 1, 2 and 3 show the performance profiles of Algorithm 1, with a linear system under the form (6) "No scaling", (11) "Qp2" and (14) "JQinv", with tolerances $\epsilon = 10^{-8}$, $\epsilon = 10^{-9}$ and $\epsilon = 10^{-10}$ respectively.

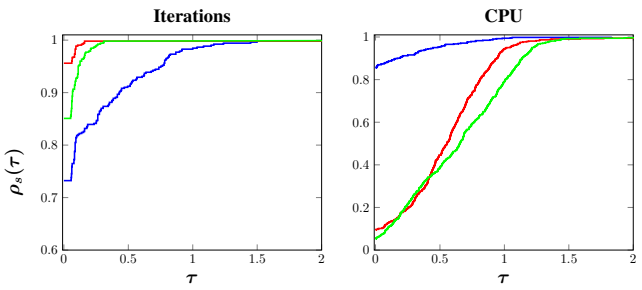

Fig. 1. Performance profile for **tol** $= 10^{-8}$.

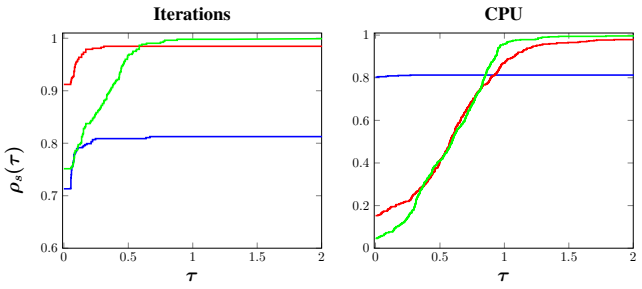

Fig. 2. Performance profile for **tol** $= 10^{-9}$.

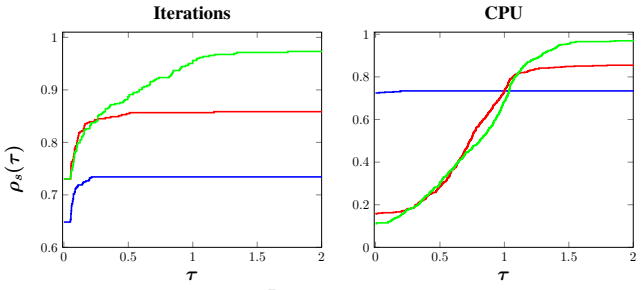

Fig. 3. Performance profile for **tol** $= 10^{-10}$.

In Figure 1, the linear systems have been solved for the tolerance $\epsilon = 10^{-8}$. Within this tolerance, the three solvers are able to solve all problems successfully. The "no-scaling" system (6), represented by the blue curve, has shown it to be faster (better CPU time) but less efficient (larger number of iterations) than the other two. The curves in red and green, corresponding to (11) and (14), do not differ significantly in this assessment. However, when the tolerance is decreased to 10^{-9} and 10^{-10}, the difference becomes more significant between these curves. In Figure 2 and Figure 3, despite still being the fastest one, the blue curve has shown the worst efficiency as the number of favorable results is only approximately 82% and 74% of the total number of experiments. Meanwhile, the red curves in Figure 2 and Figure 3 show that (11) has started to suffer from problems of NaN, so its result has stopped at 98% and 85%. Finally, through the greens, we can see that (14) still retains its robustness. There is no failure with the tolerances 10^{-8}, 10^{-9}. When the tolerance is 10^{-10}, the algorithm is stopped by NaN for only $14/523$ tests, thus 97% performance.

VI. Conclusion

The reformation of the non self-dual cone of constraints of the original problem into a self-dual one of the SOCO problem is an important contribution to the proposed interior-point algorithm. First, a scaling strategy by NT directions to produce a symmetric linear system is performed to overcome the disadvantages that come from the asymmetric linear system. Then, because of the considerable computational volume of this strategy, it is essential to choose suitable alternative formulas, for example the calculation of $Q_x y$. This not only creates efficiency in terms of execution time, but also is to achieve better accuracy of optimal solutions. Finally, the experiments indicate that Algorithm 1 is efficient and robust for solving the rolling friction problem to obtain the optimal solution of 10 digit precision.

References

[1] V. Acary, and F. Bourrier (2021). Coulomb friction with rolling resistance as a cone complementarity problem. European Journal of Mechanics-A/Solids, 85, 104046.

[2] F. Alizadeh, and D. Goldfarb (2003). Second-order cone programming. Mathematical programming 95.1: 3-51.

[3] Vincent Acary, Maurice Brémond, Tomasz Koziara, Franck Pérignon. FCLIB: a collection of discrete 3D Frictional Contact problems. [Technical Report] RT-0444, 2014, pp.34. ⟨hal-00945820v1⟩.

[4] A. Ghannad, D. Orban, and M. A. Saunders (2021). Linear systems arising in interior methods for convex optimization: a symmetric formulation with bounded condition number. Optimization Methods and Software, 1-26.

[5] S. Wright, and J. Nocedal (1999). Numerical optimization. Springer Science, 35(67-68), 7.

[6] S. Mehrotra (1992). On the implementation of a primal-dual interior point method. SIAM Journal on optimization, 2(4), 575-601.

[7] R. H. Tütüncü, K. C. Toh, and M. J. Todd (2003). Solving semidefinite-quadratic-linear programs using SDPT3. Mathematical programming, 95(2), 189-217.

[8] T. Terlaky, C. Roos, and J. Peng (2002). Self-regularity: A New Paradigm for Primal-dual Interior-point Algorithms (Princeton Series in Applied Mathematics). Princeton University Press.

[9] I. S. Duff (2004). MA57—a code for the solution of sparse symmetric definite and indefinite systems. ACM Transactions on Mathematical Software (TOMS), 30(2), 118-144.

[10] V. Rajaraman (2016). IEEE standard for floating point numbers. Resonance, 21(1), 11-30.

[11] E. D. Dolan, and J. J. Moré (2002). Benchmarking optimization software with performance profiles. Mathematical programming, 91(2), 201-213.

Appendix

We will take a glance at the Euclidean Jordan algebra which is a well-known framework associated with SOCO. See the reference paper [2] for a detailed presentation. We consider column vectors of the form $(x_0; \bar{x}) \in \mathbb{R} \times \mathbb{R}^{N-1}$. For each such vector $x \in \mathbb{R}^N$, we define an *Arrow-shaped* matrix given by

$$\mathrm{Arw}(x) = \begin{bmatrix} x_0 & \bar{x} \\ \bar{x}^\top & x_0 I \end{bmatrix}.$$

Note that $x \in \mathcal{L} := \{v = (v_0; \bar{v}) \in \mathbb{R} \times \mathbb{R}^{N-1} : \|\bar{v}\| \le v_0\}$ (known as Lorentz cone) if and only if $\mathrm{Arw}(x)$ is positive definite. In SOCO, it is useful to introduce the identity vector $e_N = (1; \mathbf{0}) \in \mathbb{R} \times \mathbb{R}^{N-1}$, the reflection matrix

$$R_N = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & -I \end{bmatrix} \in \mathbb{R}^{N \times N},$$

the direct sum of matrices

$$A \oplus B := \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix} \text{ and } \bigoplus_{i=1}^{p} A_i = A_1 \oplus \ldots \oplus A_p.$$

The Jordan product of two vectors x, y is given by

$$x \circ y := \begin{bmatrix} x^\top y \\ x_0 \bar{y} + y_0 \bar{x} \end{bmatrix} = \mathrm{Arw}(x) y = \mathrm{Arw}(x)\mathrm{Arw}(y) e.$$

The Jordan product is commutative, but not associative. The spectral decomposition of any element of this Jordan algebra is given by

$$x = \lambda_1 \boldsymbol{c_1} + \lambda_2 \boldsymbol{c_2},$$

where $\lambda_{1,2} = x_0 \pm \|\bar{x}\|$ and $\boldsymbol{c_{1,2}} = \frac{1}{2}\begin{bmatrix} 1 \\ \pm \frac{\bar{x}}{\|\bar{x}\|} \end{bmatrix}$ are the eigenvalues and eigenvectors of x. The *trace* and *determinant* of x are defined by

$$\mathrm{tr}(x) = \lambda_1 + \lambda_2 \quad \text{and} \quad \det(x) = \lambda_1 \lambda_2.$$

For any arbitrary power $p \in \mathbb{Q}$, we define

$$x^p = \lambda_1^p \boldsymbol{c_1} + \lambda_2^p \boldsymbol{c_2}.$$

For example, a useful formula for the gradient of the logarithm of the determinant is given by

$$\nabla_x \log\det(x) = 2x^{-1} = 2(\lambda_1^{-1} \boldsymbol{c_1} + \lambda_2^{-1} \boldsymbol{c_2}).$$

The *quadratic-representation* associated to x is defined by

$$Q_x = 2\mathrm{Arw}^2(x) - \mathrm{Arw}(x^2) = 2xx^\top - \det(x) R.$$

For $x \in \mathbb{R}^N$, x nonsingular (i.e., $\lambda_2 \neq 0$), $y \in \mathbb{R}^N$ and $t \in \mathbb{Z}$, we have the following properties [2, Theorem 8]

$$\begin{aligned} Q_x y &= 2(x^\top y)x - \det(x) R y, \\ Q_{x^t} &= Q_x^t, \\ (Q_x y)^t &= Q_{x^t} y^t \text{ if } x \text{ and } y \text{ commute.} \end{aligned}$$

For vectors x, y of the form $(v_1; \ldots; v_\ell)$, such that $v_i \in \mathbb{R}^{n_i}$ belongs to the second order cone of dimension n_i, we define

$$x \circ y = (x_1 \circ y_1; \ldots; x_\ell \circ y_\ell), \quad \boldsymbol{e} = (e_{n_1}; \ldots; e_{n_\ell}),$$

$$\mathrm{Arw}(x) = \bigoplus_{i=1}^{\ell} \mathrm{Arw}(x_i), \quad Q_x = \bigoplus_{i=1}^{\ell} Q_{x_i} \quad \text{and} \quad R = \bigoplus_{i=1}^{\ell} R_{n_i}.$$