ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

SOLVING UNILATERAL CONTACT PROBLEMS IN MULTIBODY DYNAMICS USING A PRIMAL-DUAL INTERIOR POINT METHOD

Dario Mangoni[1], **Alessandro Tasora**[1]

[1]Università degli Studi di Parma
Parco Area delle Scienze 181/A 43124 Parma, Italy
e-mail: dario.mangoni@studenti.unipr.it, alessandro.tasora@unipr.it

Keywords: interior-point, multibody, contacts

Abstract. *Rigid unilateral contacts in multibody problems lead to non-smooth dynamics: this requires the development of new types of time integrators that draw on the solution of a variational inequality at each time step. In most settings, this variational inequality is a complementarity problem. In the context of non-smooth multibody dynamics, projected fixed-point, spectral and Nesterov methods have been successfully used to solve those complementarity problems, especially for granular problems or real-time simulations [2] [3], but large scale and ill-conditioned problems still advocate for solvers with better convergence. The promising class of interior-point method has been applied only recently to multibody dynamics [4]. Their implementation is more complicate, but this comes at the benefit of high precision and good convergence, moreover they can be applied to problems with both deformable and rigid bodies [5]. We present an algorithm based on a dual-primal interior point method. The solution is advanced along the central path using the Mehotra predictor-corrector scheme, in a small number of sub steps that leverage on a parallel direct solver for computing the updates from highly sparse saddle-point problems. Although the original method addresses unilateral constraints only, it is possible to restate it in order to fit also equality constraints, as might happen when introducing joints between colliding parts. The algorithm is implemented in C++ language in our open-source simulator. Results show that the method regularly converges to small residuals even in cases (for instance, odd mass ratios) that degraded the convergence of other methods.*

1 INTRODUCTION

Interior-Point algorithms are born to provide a solution for minimization problems, mainly linear and quadratic, that can arise from many different scenarios. In this paper we would show some improvements related to multibody simulations involving unilateral constraints (contacts) without friction. In this case the solver is called to solve a Quadratic Programming problem (QP from now on) that can be written as:

$$\min \quad q(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T G\mathbf{x} + \mathbf{x}^T\mathbf{c} \tag{1a}$$

$$\text{s.t.} \quad A\mathbf{x} \geq b \tag{1b}$$

where $G \in \mathbb{R}^{n \times n}$ is a square, symmetric, positive definite matrix and $A \in \mathbb{R}^{m \times n}$ is the constraints matrix. Although we will show only the case when only inequality constraints hold, it is possible to restate the equality constraints to fit this formulation.

As demonstrated in [1], there are some necessary and sufficient conditions for which a vector $\mathbf{x}^*$ is solution (i.e. feasible minimizer) of the QP problem. These conditions, known as *Karush-Kuhn-Tucker conditions*, are here reported:

$$G\mathbf{x} - A^T\boldsymbol{\lambda} + \mathbf{c} = 0 \tag{2a}$$

$$A\mathbf{x} - vectb \geq 0 \tag{2b}$$

$$(A\mathbf{x} - \mathbf{b}) \circ \boldsymbol{\lambda} = 0 \tag{2c}$$

$$\boldsymbol{\lambda} \geq 0 \tag{2d}$$

where the $\circ$ operator stands for the Hadamard (or component-wise) product. The condition 2c can also be seen as an orthogonality condition $\mathbf{y} \perp \boldsymbol{\lambda}$ where the operator $\perp$ is applied component-wise.
Introducing *slack* variables in $A\mathbf{x} - \mathbf{b} \geq 0$ let us move the inequality constraints over a variable (namely $\mathbf{y}$).

$$G\mathbf{x} - A^T\boldsymbol{\lambda} + \mathbf{c} = 0 \tag{3a}$$

$$A\mathbf{x} - \mathbf{y} - \mathbf{b} = 0 \tag{3b}$$

$$\mathbf{y} \circ \boldsymbol{\lambda} = 0 \tag{3c}$$

$$\boldsymbol{\lambda} \geq 0 \tag{3d}$$

$$\mathbf{y} \geq 0 \tag{3e}$$

If a set of variables $(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda})$ respects the conditions of eq.2 (or, that is the same, eq.3) then $\mathbf{x}$ is a minimizer for eq.1a that respects eq.1b i.e. is solution for the QP problem. This conditions in fact are not only necessary but, if G is positive semidefinite as in this case, are also sufficient.

2 INTERIOR-POINT BASICS

Since the algorithm we are going to show is based on a *path-following* method we introduce the *perturbed* KKT system:

$$F(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}; \sigma) = \begin{bmatrix} G\mathbf{x} - A^T\boldsymbol{\lambda} + \mathbf{c} \\ A\mathbf{x} - \mathbf{y} - \mathbf{b} \\ \mathbf{y} \circ \boldsymbol{\lambda} - \sigma\mu\mathbf{e} \end{bmatrix}_{(\mathbf{y}, \boldsymbol{\lambda}) \geq 0} \tag{4}$$

where $\mathbf{e} = [1, \dots, 1] \in \mathbb{R}^n$ and σ, the *centering* parameter and μ the *duality measure* that will define in a moment.

It is easy to observe that the problem stated in 3 can be summarized as $F(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}; 0) = 0$ so we can make the easy assessment that we can operate on such multivariate function using a Newton step (applied to a non-linear equation) in order to find the solution $(\mathbf{x}^*, \mathbf{y}^*, \boldsymbol{\lambda}^*)$ to bring its value down to zero. This operation, also called *affine scaling step*, gives the foundation to the Interior-Point algorithm.

As we will notice in section 2.1, pointing directly to the actual solution is *not* always the smartest option available. In fact, although the first step could provide a big reduction of $q(\mathbf{x})$, it also brings the approximated solution very close to the boundaries of the feasible region so that only very short steps can be taken in the following iteration. Because of this, the Newton step is made towards a different point that lays on the *central path*.

2.1 Basic Newton step

2.1.1 Newton step direction

Since we want $F(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}; 0) = 0$ the most trivial idea is to make a fixed-point iteration like:

$$F_{k+1} = F_k + J_{F_k} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\boldsymbol{\lambda} \end{bmatrix} = 0 \quad \Rightarrow \quad J_{F_k} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\boldsymbol{\lambda} \end{bmatrix} = -F_k \tag{5}$$

where $F_k = F(\mathbf{x}_k, \mathbf{y}_k, \boldsymbol{\lambda}_k; 0)$. Expanding the Jacobian J_{F_k}, that is valid also for the more generic $F_k = F(\mathbf{x}_k, \mathbf{y}_k, \boldsymbol{\lambda}_k; \sigma)$:

$$\begin{bmatrix} G & 0 & A^T \\ A & -I & 0 \\ 0 & \Lambda & Y \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\boldsymbol{\lambda} \end{bmatrix} = - \begin{bmatrix} G\mathbf{x}_k - A^T\boldsymbol{\lambda} + \mathbf{c} \\ A\mathbf{x}_k - \mathbf{y}_k - \mathbf{b} \\ \mathbf{y}_k \circ \boldsymbol{\lambda}_k - \sigma\mu\mathbf{e} \end{bmatrix} \tag{6}$$

since we assume that σ and μ are, in any case, constant parameters. If F were linear, the solution of 6 would have been also the solution of the original KKT system. Unfortunately this is not true with a QP problem, where

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{y}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \\ \boldsymbol{\lambda}_k \end{bmatrix} + \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\boldsymbol{\lambda} \end{bmatrix} \tag{7}$$

is only an approximation. Now that we have the *directions* of the step, the step length(s) must be calculated too.

2.1.2 Newton step length

In order to determine the step length, we use the non-negativity constraints $\mathbf{y} \geq 0$ and $\boldsymbol{\lambda} \geq 0$ for which:

$$\begin{aligned} \mathbf{y}_{k+1} &= \mathbf{y}_k + \alpha_p^{prim} \Delta\mathbf{y}_k \geq 0 \\ \boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k + \alpha^{dual} \Delta\boldsymbol{\lambda}_k \geq 0 \end{aligned} \tag{8}$$

that gives, with simple math:

$$\begin{aligned} \alpha^{prim} &= \min\left(1, \eta \min_{i\,:\,\Delta\mathbf{y}<0} -\frac{\mathbf{y}_i}{\Delta\mathbf{y}_i}\right) \\ \alpha^{dual} &= \min\left(1, \eta \min_{i\,:\,\Delta\boldsymbol{\lambda}<0} -\frac{\boldsymbol{\lambda}_i}{\Delta\boldsymbol{\lambda}_i}\right) \end{aligned} \tag{9}$$

We notice another control parameter η. The standard version considers this value equal to 1, but we will use *also* another update method that improve the robustness. The variation is trivial: it simply takes the minimum step length between the two α parameters and adds that control parameter $\eta \in (0,1)$ that approaches 1 as the iterations come closer to the solution. Also in this case the control parameter is used to soften the constraint.

2.2 The central path

Any iterative solver creates a sequence in where each element is progressively closer to the solution. However, the paths they follow are not unique and give different behaviours in term of convergence speed and robustness. We will focus on the most used one, the *central* path, that is characterized by softening the orthogonality condition while keeping the residuals $\mathbf{r}_p$ and $\mathbf{r}_d$ equal to zero.

Moreover this path is *strictly feasible* so that it follows the definition of strict feasibility given here:

$$\mathcal{F}_0 = \{(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) \,|\, \mathbf{r}_p = 0, \mathbf{r}_d = 0, (\mathbf{y}, \boldsymbol{\lambda}) > 0\}$$

The equation of the KKT system that is not fully respected is the orthogonality constraint 3c that becomes:

$$\mathbf{y} \circ \boldsymbol{\lambda} = \tau \mathbf{e} \tag{10}$$

giving the *central path* definition:

$$\mathcal{C}(\tau) = \{(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) \in \mathcal{F}_0 \,|\, \mathbf{y} \circ \boldsymbol{\lambda} = \tau \mathbf{e}\} \tag{11}$$

Let imagine the $\mathbb{R}^m$ space in which $\mathbf{y} \circ \boldsymbol{\lambda}$ lays. The inequalities $\boldsymbol{\lambda} \geq 0$ and $\mathbf{y} \geq 0$ still hold so we are restricting our analysis only over the non-negative orthant. The path that equation 11 draws is simply its bisector; because of this is called *central*. As we notice, given τ, the iteration keeps the approximated solution as far as possible from the boundaries. However, this step is not generally the best in terms of residual/error reduction and it will never bring the iteration to the actual solution of the problem unless τ is zero. Actually, this is just want we want to do: as soon as the algorithm advance, τ is progressively reduced to zero, so that the iteration points toward the actual solution of the KKT system.

Another derived problem is how to choose τ. The concept is that it should tend to zero as the approximated solution is closer to the actual one. Since the only part of the error that is not reduced yet is the orthogonality condition, we could take this parameter to drive τ. An useful criteria to understand how far the constraint hold for a given point is the *complementarity measure* (that has the same meaning as the *duality measure* in LP). Given a set $(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda})$ that satisfies both $\boldsymbol{\lambda} \geq 0$ and $\mathbf{y} \geq 0$ we can compute:

$$\mu = \frac{\mathbf{y}^T \boldsymbol{\lambda}}{m} \tag{12}$$

At the optimal solution the complementarity measure should equal to zero so it can be easily adopted to choose τ

$$\tau = \sigma \mu \tag{13}$$

where σ is called *centering* parameter and it is normally chosen as:

$$\sigma = \left(\frac{\mu_p}{\mu_c}\right)^3 \tag{14}$$

as we will describe later.

2.3 Predictor-Corrector Scheme

As we have already defined the concept of *central path* in section 2.2 and the basic Newton iteration in section 2.1 we can assemble the practical algorithm that has been implemented. A more in-depth review of starting point strategies is showed in section 2.4.

2.3.1 Prediction

Basically the prediction phase is based on a bare Newton step on the original eq.3 in the form previously showed in 6 without any perturbation, i.e. $\sigma = 0$. We will name the unknown vector as $[\Delta \mathbf{x}_p \quad \Delta \mathbf{y}_p \quad \Delta \boldsymbol{\lambda}_p]^T$, omitting the subscript k for the sake of clarity. The lengths of this step are obtained from 9 with $\eta = 0$. Considering the new approximation given by eq.8, we obtain the new value of the complementarity measure $\mu_p = \mathbf{y}_p^T \boldsymbol{\lambda}_p / m$. The centering parameter can now be evaluated by mean of the equation 14, remembering that the previous value μ comes directly from the previous iteration.

2.3.2 Correction

From the prediction step we observe that the updated value of the orthogonality condition is, component-wise:

$$\begin{aligned}(\mathbf{y}_i + \Delta \mathbf{y}_i)(\boldsymbol{\lambda}_i + \Delta \boldsymbol{\lambda}_i) &= \mathbf{y}_i \boldsymbol{\lambda}_i + \mathbf{y}_i \Delta \boldsymbol{\lambda}_i + \boldsymbol{\lambda}_i \Delta \mathbf{y}_i + \Delta \boldsymbol{\lambda}_i \Delta \mathbf{y}_i \\ &= \Delta \boldsymbol{\lambda}_i \Delta \mathbf{y}_i\end{aligned} \tag{15}$$

instead of zero. The correction step takes into consideration this residual and aims to delete it, putting it on the right-hand side of the KKT system that now becomes:

$$\begin{bmatrix} G & 0 & A^T \\ A & -I & 0 \\ 0 & \Lambda & Y \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{y} \\ \Delta \boldsymbol{\lambda} \end{bmatrix}_c = - \begin{bmatrix} G\mathbf{x} - A^T \boldsymbol{\lambda} + \mathbf{c} \\ A\mathbf{x} - \mathbf{y} - \mathbf{b} \\ \mathbf{y} \circ \boldsymbol{\lambda} + \Delta \mathbf{y}_p \circ \Delta \boldsymbol{\lambda}_p - \sigma \mu \mathbf{e} \end{bmatrix} \tag{16}$$

where $\mathbf{y}$, $\boldsymbol{\lambda}$ and $\mathbf{x}$ comes from the previous iteration (so they are *not* the updated approximations from the prediction step). It is also true that the entire matrix on the left is kept untouched as well as the upper part of the right-hand side of the equation. So only the product $\Delta \mathbf{y}_p \circ \Delta \boldsymbol{\lambda}_p$ must be effectively evaluated.

As before these considerations are useful only to obtain the directions along which the step will be made. The length is evaluated using eq.9 with $\eta \in (0, 1)$ and taking as $\alpha = \min(\alpha^{prim}, \alpha^{dual})$. We used this simple equation to give adaptivity:

$$\eta = \exp(-0.1\mu) + 0.9 \tag{17}$$

The approximated solution is given by eq.8.

At this stage, we would like to update the upper part of the right-hand side of eq.6 — used both for the prediction and correction step — without expensive operations. We will notice that, calling:

$$\begin{aligned}\mathbf{r}_d &= G\mathbf{x} - A^T \boldsymbol{\lambda} + \mathbf{c} \\ \mathbf{r}_p &= A\mathbf{x} - \mathbf{y} - \mathbf{b}\end{aligned} \tag{18}$$

we have:

$$\begin{aligned}\mathbf{r}_{dk+1} &= (1 - \alpha^{dual})\, \mathbf{r}_{dk} + (\alpha^{prim} - \alpha^{dual})\, G\, \Delta \mathbf{x}_k \\ \mathbf{r}_{p_{k+1}} &= (1 - \alpha^{prim})\, \mathbf{r}_{p_k}\end{aligned} \tag{19}$$

Different formulations of eq.6 and of the more general eq.16 exists. One of the most useful is given by substituting $\Delta\mathbf{y} = A\Delta\mathbf{x} + \mathbf{r}_p$ in the last row of these systems, leading to:

$$\begin{bmatrix} G & A^T \\ A & \Lambda^{-1}Y \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\boldsymbol{\lambda} \end{bmatrix}_c = - \begin{bmatrix} \mathbf{r}_d \\ \mathbf{r}_p + \mathbf{y} - \Lambda^{-1}(\Delta\mathbf{y}_p \circ \Delta\boldsymbol{\lambda}_p - \sigma\mu\mathbf{e}) \end{bmatrix} \tag{20}$$

This *augmented* system will be extensively used in our calculations.

2.4 Starting point

One drawback of IP methods is their sensitivity to starting points. In fact, depending on where the algorithm starts, it could converge to the solution in few iterations or it could diverge as well, depending on the initial given value. Because of this particular behaviour the choice of a reliable starting point technique is one of the most critical. Different approaches are shown in the next sections and have been implemented in order to test their performances (see section 4).

2.4.1 Nocedal's starting point

In [1, §16.6] the priority is given to moving away from the boundary of the non-negative orthant while trying to reduce primal-dual residuals and complementarity product. This is done by a classic affine/prediction step (that will compute $\Delta\mathbf{y}$ and $\Delta\boldsymbol{\lambda}$) that will drive the algorithm towards the solution and indirectly will also take the distance from the boundaries.

$$\begin{aligned} \mathbf{x}_0 &= [1 \dots 1]^T \\ \mathbf{y}_0 &= \max(1, |\mathbf{y} + \Delta\mathbf{y}|) \\ \boldsymbol{\lambda}_0 &= \max(1, |\boldsymbol{\lambda} + \Delta\boldsymbol{\lambda}|) \end{aligned} \tag{21}$$

However, also with trivial examples, this technique is not conclusive and some failures have been experienced in our tests. Initial values of $\mathbf{x}_0$, $\mathbf{y}_0$ and $\boldsymbol{\lambda}_0$ cannot be chosen randomly. In fact, changing their value of an order of magnitude could give divergence problems.

2.4.2 D'Apuzzo's starting point

Another possibility is shown in [6]. This method is by far one of the less computational expensive and it grants robustness in a wide range of scenarios. This technique is based on the idea that the the neighbourhood in where the approximated solutions are kept is influenced by the initial residuals/complementarity gap ratio. Because of this, the starting point has to show not only small residuals and a small complementarity product, but they should be, at at the same time, comparable. In our case the strategy leads to this conclusion:

$$\begin{aligned} \mathbf{x}_0 &= [1 \dots 1]^T \\ \mathbf{y}_0 &= \max(1, A\mathbf{x} - \mathbf{b}) \\ \boldsymbol{\lambda}_0 &= 1/\mathbf{y}_0 \end{aligned} \tag{22}$$

where the $\max()$ operator is applied component-wise.

3 MULTI-BODY FORMULATION

In this section we will introduce a multi-body formulation for the non-smooth dynamics of a system subject to an arbitrary number of contacts between multiple parts. The notation in section 3.1 is intentionally kept similar to the references while in section 3.2 we will adapt symbols to the current IP notation.

3.1 Motion equation and constraints

The non-smooth nature of mechanical systems subject to contacts stems from the fact that contact constraints are unilateral, as the contact force cannot be negative. More in general, in case of frictional contacts, the contact force must satisfy some set-valued law, that cannot be translated in simple equality constraints: for instance for the Coloumb-Amontons friction, the contact force $\mathbf{F}_c$, expressed in the local contact reference must satisfy $\mathbf{F}_c \in \mathsf{F}_c \subset \mathbb{R}^3$ with F_c being a second-order Lorentz cone.

In sake of clarity, in this work we will deal with the simpler friction-less contact case. Therefore, the contact constraint can be expressed with the following Signorini complementarity condition at the position level:

$$F_{c,n} \geq 0 \quad \perp \quad \Phi_{c,n} \geq 0 \tag{23}$$

In the previous constraint, $F_{c,n}$ is the surface-normal component of the force, and $\Phi_{c,n}$ is the normal displacement; the $\perp$ complementarity symbol means:

$$\mathbf{a} \geq \mathbf{0} \quad \perp \quad \mathbf{b} \geq \mathbf{0} \quad \Leftrightarrow \quad a_i \geq 0, \quad b_i \geq 0, \quad a_i b_i = 0 \tag{24}$$

As we will see in the following, this introduces a major difficulty in the formulation, which requires the solution of complementarity problems at discrete time steps, instead of solving trivial linear problems as those required by classical smooth systems with bilateral constraints. In fact, when bilateral constraints are added to a system of ordinary equations (ODE), it turns into a Differential Algebraic Equation (DAE) problem. In our case, the addition of complementarity constraints lead to a Differential Variational Inequality (DVI).

Assume we have multiple contacts and multiple parts. Each $i-th$ part is a rigid body with position $\{\mathbf{x}_i, \rho_i\}$, using three dimensional translation vectors $\mathbf{x}_i \in \mathbb{R}^3$ and four-dimensional unit quaternions $\rho_i \in \mathbb{H}^1$ for rotation in $\mathsf{SO}(\mathbb{R}, 3)$. Velocity is represented by $\{\dot{\mathbf{x}}_i, \omega_i\}$, with $\omega \in \mathbb{R}^3$ angular velocity in local reference of the body. Each body has a mass m_i and a tensor of inertia $I \in \mathbb{R}^{3x3}$.

We can arrange all the states in single vectors $\mathbf{q} = \{\mathbf{x}_1, \mathbf{q}_1, \mathbf{x}_2, \mathbf{q}_2, ..., \mathbf{x}_n, \mathbf{q}_n,\}$ and $\mathbf{v} = \{\dot{\mathbf{x}}_1, \omega_1, \dot{\mathbf{x}}_2, \omega_2, ..., \dot{\mathbf{x}}_n, \omega_n,\}$. Using simple quaternion algebra, there is a linear map relation between $d\mathbf{q}/dt$ and $\mathbf{v}$ as $d\mathbf{q}/dt = \Gamma(\mathbf{q}, \mathbf{v})$. Similarly, all masses can be assembled in a single matrix M, that is very sparse since it includes just m and I values along its diagonal.

Forces and torques can be applied to bodies, we collect them into a $\mathbf{f}_t = \mathbf{f}_e + \mathbf{f}_g$ term, which includes the $\mathbf{f}_g$ gyroscopic term and generic external loads $\mathbf{f}_e$.

For each j-th contact, we have a contact distance Φ_j, where $j \in \mathcal{G}_\mathcal{A}$ given a set $\mathcal{G}_\mathcal{A}$ of active unilateral contacts. We make the assumption that for each system state, the distance Φ_j can be computed as a (smooth, differentiable) [1] function of state $\Phi_j = \Phi_j(\mathbf{q}, t)$, and we assume that for every integration time step an algorithm can update the $\mathcal{G}_u$ set by adding potential contact points that come within a distance threshold η. [2].

[1] This is not always the case when sharp and non-concave shapes come into contact, where some regularization or heuristic workarounds might be needed especially interpenetrating $\Phi_j < 0$, but the rest of the discussion still holds.

[2] This process is handled by the *collision engine*, and can be very CPU-intensive when shapes are numerous and complex. Also, defining a proper threshold η is not trivial, but it is not discussed here.

We relate contact-orthogonal displacements to state changes via jacobians:

$$
\begin{aligned}
d\Phi_j(\mathbf{q},t)/dt &= [\partial\Phi_j/\partial\mathbf{q}]d\mathbf{q}/dt && (25)\\
&= \nabla_{\mathbf{q}}{\Phi_j}^T\Gamma(\mathbf{q},\mathbf{v}) && (26)\\
&= \nabla{\Phi_j}^T\mathbf{v} && (27)
\end{aligned}
$$

We remark that the $\nabla{\Phi_j}^T$ jacobian is a single-row matrix with high sparsity: since the j-th contact depends on two bodies only, the $\nabla{\Phi_j}^T$ jacobian is a row full of zeroes, with only two 1x6 non-zero blocks.

The contact-orthogonal force $\mathbf{F}_{c,j}$ can be expressed by introducing a $\hat{\gamma}_j$ scalar multiplier per each contact, thus in absolute coordinates it holds:

$$\mathbf{F}_{c,j} = \nabla\Phi_j\hat{\gamma}_j \tag{28}$$

The full system that expresses the mechanical model subject to unilateral constraints is the following DVI:

$$
\begin{aligned}
M\frac{d\mathbf{v}}{dt} &= \sum_{j\in\mathcal{G}_{\mathcal{A}}}\nabla\Phi_j\hat{\gamma}_j + \mathbf{f}_t(t,\mathbf{q},\mathbf{v}) && (29)\\
\Phi_j(\mathbf{q},t)\geq 0 &\perp \hat{\gamma}_j \geq 0, \quad j\in\mathcal{G}_{\mathcal{A}} && (30)\\
\dot{\mathbf{q}} &= \Gamma(\mathbf{q},\mathbf{v}) && (31)
\end{aligned}
$$

The time integration of the model above can be performed using a stabilized discrete time-stepping method, as presented in [2]. This requires the introduction of a time step h to advance the state from $\mathbf{q}^{(l)},\mathbf{v}^{(l)}$ to $\mathbf{q}^{(l+1)},\mathbf{v}^{(l+1)}$. The DVI problem is discretized at the velocity-level, in terms of measures to include impulsive effects if any, so we define $\gamma_j = h\hat{\gamma}_j$ and we write the following system where the unknowns are the changes in velocities and the contact impulses γ_j:

$$
\begin{aligned}
M(\mathbf{v}^{(l+1)}-\mathbf{v}^{(l)}) &= \sum_{j\in\mathcal{G}_{\mathcal{A}}}\nabla\Phi_j\gamma_j + h\mathbf{f}_t(t,\mathbf{q},\mathbf{v}) && (32)\\
\frac{1}{h}\Phi_j(\mathbf{q},t)+\nabla{\Phi_j}^T\mathbf{v}\geq 0 &\perp \gamma_j\geq 0, \quad j\in\mathcal{G}_{\mathcal{A}} && (33)\\
& && (34)
\end{aligned}
$$

followed by an update $\mathbf{q}^{(l+1)} = \boldsymbol{\lambda}(\mathbf{q}^{(l)},\mathbf{v}^{(l+1)},h)$.

We note that the system above can be expressed as a (mixed) Linear Complementarity System (M-LCP), using a compact matrix notation:

$$\begin{bmatrix} M & D\\ D^T & 0\end{bmatrix}\begin{bmatrix}\mathbf{v}^{(l+1)}\\ \boldsymbol{\gamma}\end{bmatrix} - \begin{bmatrix}h\mathbf{f}_t + M\mathbf{v}^{(l)}\\ -\frac{1}{h}\boldsymbol{\Phi}(\mathbf{q},t)\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ \mathbf{C}\end{bmatrix} \tag{35}$$

$$\mathbf{C}\geq\mathbf{0}\perp\boldsymbol{\gamma}\geq\mathbf{0} \tag{36}$$

where D is a sparse matrix that contains all the jacobians $\nabla\Phi_j$ aligned column-by-column, as $D = \{\nabla\Phi_1,\nabla\Phi_2,...,\nabla\Phi_{n_c}\}$. Also $\boldsymbol{\Phi} = \{\Phi_1,\Phi_2,...,\Phi_{n_c}\}^T$ and $\boldsymbol{\gamma} = \{\gamma_1,\gamma_2,...,\gamma_{n_c}\}^T$.

This M-LCP is also the Karush-Kuhn-Tucker optimality condition of a quadratic program with convex (linear, unilateral) constraints, a class of problems that have been solved successfully with Interior-Point algorithms.

We note that, if the constraints were bilateral, as happens in revolute joints, mating constraints etc., the MLCP would turn into a simple linear problem of the type:

$$\begin{bmatrix} M & D \\ D^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}^{(l+1)} \\ \gamma \end{bmatrix} = \begin{bmatrix} h\mathbf{f}_t + M\mathbf{v}^{(l)} \\ -\frac{1}{h}\boldsymbol{\Phi}(\mathbf{q}, t) \end{bmatrix} \tag{37}$$

as often met in the solution of ODEs and DAEs in smooth multibody dynamics. In such special case, the discussed integration scheme would turn into a linearized implicit Euler integration scheme. The difficulty, in our more general case, is the fact that the solution of a MLCP is by far more complex than the solution of a linear system.

3.2 Fitting the MB system in Interior-Point algorithm

We would now provide another perspective from which the system 35 can be observed. Its solution can be thought as the minimizer of the QP problem (as stated for example in 3) considering:

$$\begin{aligned}
\mathbf{x} &\Leftrightarrow \mathbf{v}^{(l+1)} \\
\mathbf{c} &\Leftrightarrow -\left(h\mathbf{f}_t + M\mathbf{v}^{(l)}\right) \\
\mathbf{b} &\Leftrightarrow -\frac{1}{h}\boldsymbol{\Phi}(\mathbf{q}, t) \\
\mathbf{y} &\Leftrightarrow \mathbf{C} \\
\boldsymbol{\lambda} &\Leftrightarrow \gamma \\
G &\Leftrightarrow M \\
A &\Leftrightarrow D^T
\end{aligned}$$

Because of this strict similarity, the multibody problem formulated in 35 can be solved through the IP algorithm described in the previous sections.

4 RESULTS

The algorithm presented in section 2 has been implemented in C++ language in order to be tested in a multibody dynamic simulation software, Chrono::Engine. The hardware on which these tests are performed consists in an Intel i7-4710HQ processor, 2x4GB DDR3-1600K RAM, 1TB 2.5” SATA-300 5400RPM. The software is compiled for x64 architecture with MSVC compiler with full optimization. The Intel MKL Pardiso direct linear solver is used for the solution of the linear system showed in 20. The Interior-Point algorithm exits whenever the residual norms and the complementarity product are below a threshold that in the cases below are set as:

$$\frac{\|\mathbf{r}_p\|}{m} < 1 \times 10^{-8} \tag{38a}$$

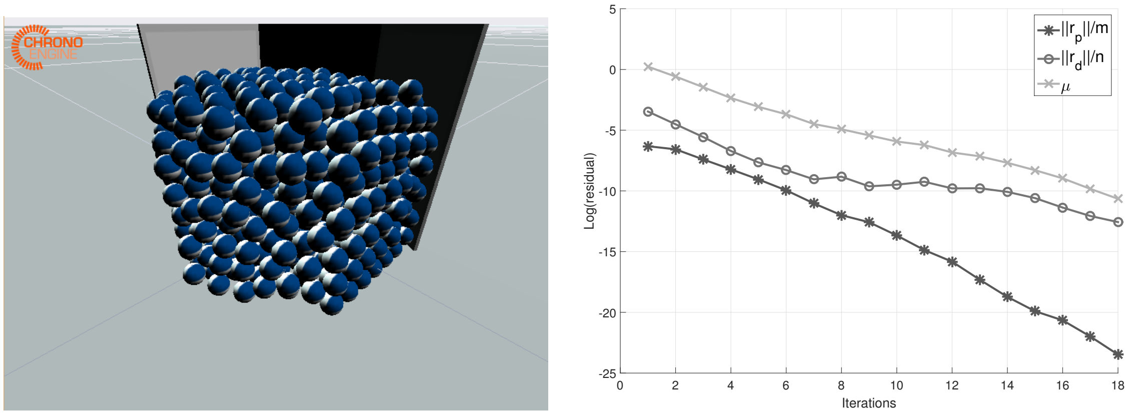
$$\frac{\|\mathbf{r}_d\|}{n} < 1 \times 10^{-8} \tag{38b}$$

$$\mu < 1 \times 10^{-7} \tag{38c}$$

4.1 Convergence and speed results

The first test involves unilateral contacts of spheres that drop inside a box. As we can see in table 1 increasing the number of bodies leads to a higher number of degrees of freedom (bigger mass/stiffness matrices) and higher number of contacts/constraints (bigger Jacobian matrices).

However, the test demonstrate that the number of iterations needed to get the approximated solution is almost invariant. Obviously the increased size negatively affects the time and computational expense of the linear solver.



(a) *Ball in the box test.*

(b) *Convergence result.*

Figure 1: Balls falling in a box.

Matrix size	Mean iterations
13800	18.9
4500	19.4
3200	18.6
1200	16.7

Table 1: Iterations needed at each timestep for different matrix sizes (Ball in the box example).

4.2 Warm start

In many different case-scenario it would be quite useful if the solver could take advantage of warm start. In fact it could happen that the problem matrices are quite similar between consecutive timesteps so it would be obvious to re-use the solution of the previous step as starting point of the following. This choice often improves the convergence speed of many iterative solvers and it would be desirable if also IP methods could manifest this behaviour.

In order to test if this feature is natively supported by the proposed algorithm we set up a couple of trivial examples. The first one is a brick supported by four contact points. Although this problem can be easily solved statically we tested it in a dynamic simulation. This means that the collision detection algorithm will slightly change the constraint equations at each timestep while keeping the mass matrix unchanged.
In the second case the brick is substituted with a ball so that the contacts are no more redundant and the solution is unique.

Classic starting point algorithms usually reset the initial vectors to some default value (usually 1) that are further manipulated in order to reduce the ratio between residuals and complementarity product like in [6] or in other used to make a step away from the non-negative orthant boundary like in [1, §16.6]. Adapting these methods to use a trivial warm start is quite straightforward although it cannot be considered a state-of-the-art solution. The tests performed aim to point out if, at least in a controlled environment, the IP algorithm could take advantage of a solution recycle.

It has to be underlined that, in order to reuse the full solutions from previous timesteps, the matrix must have kept the same dimension and moreover the constraint equations should not have undergone big modifications. However, this is not always true especially when the problem involves unilateral constraint so, in such cases, a *partial* warm start has been applied. In fact, although the Jacobian matrix A has suffered not negligible changes, the mass/stiffness matrix G is usually kept unchanged and because of this, the $\mathbf{x}$ vector keeps the same dimension and it can be warm started. In this case $\mathbf{y}$ and $\boldsymbol{\lambda}$ have to be reinitialized.

Test (IP calls)	None	Partial WarmStart	Warm Start
Balls in Box (50)	935	927	747
Single Brick (100)	499	370	81
Single Ball (100)	720	515	90

Table 2: IP iterations for solving unilateral contact problems.

4.3 Redundant constraints and softened contacts

Formally the IP method needs a full-rank constraint matrix A i.e. the Jacobian. Unfortunately it is a common situation to have redundant constrained bodies for which the Jacobian is rank-deficient and no unique solution exists. Under those circumstances the matrix that is composed to make a Newton step is more prone to become singular as soon as the components of $\mathbf{y}/\boldsymbol{\lambda}$ tend to zero. We investigated this case and we present here some interesting results.

Before trying classical workarounds let us consider how the IP is approaching the contact problem. As we can see in 20 the $\mathbf{y}/\boldsymbol{\lambda}$ vector occupies the diagonal of the south-east corner of the augmented matrix, so that the Lagrangian multipliers i.e. the forces that acts on the bodies will affect the constraints equations. Trying to give a physical meaning to this phenomena we can easily come to the conclusion that these elements acts as contact springs i.e. they give a reaction that tends to push away the bodies that are going to collide. In some way we can say that the hard constraints that were permitted by the DVI approach are now softened, introducing a spring behaviour to the contacts. However, this is also an interesting feature considering the numerical issue given by redundant constraints: the non-zero, although small, elements that are inserted yield to a better conditioned matrix, adding the required linear independence to the possibly rank-deficient A. This approach permitted to get a reliable solution also in such cases where the Jacobian matrix was highy compromised. Equipping the IP algorithm with a direct solver is, in such cases, the best choice since they are less affected by the ill-conditioning of the matrix.

Moreover, it has been tested another useful feature in which a compliance matrix is added in place of the diagonal $\Lambda^{-1}Y$ so that, when the algorithm converges, the matrix still has a non-zero diagonal matrix in its south-east corner:

$$\begin{bmatrix} G & A^T \\ A & \Lambda^{-1}Y + E \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\boldsymbol{\lambda} \end{bmatrix}_c = - \begin{bmatrix} \mathbf{r}_d \\ \mathbf{r}_p + \mathbf{y} - \boldsymbol{\Lambda}^{-1}(\Delta\mathbf{y}_p \circ \Delta\boldsymbol{\lambda}_p - \sigma\mu\mathbf{e}) \end{bmatrix} \tag{39}$$

where E is a diagonal matrix with elements $1/k$ with k stiffness of the contact spring.

The more robust workaround of removing redundant lines with SVD or QR revealing methods is, in our tests, not needed since the IP methods always converged with the required accuracy to the solution.

REFERENCES

[1] J. Nocedal, S. Wright, *Numerical Optimization, 2nd Edition*, Springer New York, 2006.

[2] A. Tasora, M. Anitescu, A Matrix-Free Cone Complementarity Approach for Solving Large-Scale, Nonsmooth, Rigid Body Dynamics *Computer Methods in Applied Mechanics and Engineering*, **200.5**, 439–453, 2011.

[3] T. Heyn, et al. Using Krylov Subspace and Spectral Methods for Solving Complementarity Problems in ManyBody Contact Dynamics Simulation *International Journal for Numerical Methods in Engineering*, **95.7**, 541–561, 2013.

[4] J. Klinert, S. Bernd, M. Obermayr An Inexact Interior Point Method for the Large-Scale Simulation of Granular Material *Computer Methods in Applied Mechanics and Engineering*, **278**, 567–598, 2014.

[5] R. Kučera et al., An Interior-Point Algorithm for the Minimization Arising from 3D Contact Problems with Friction *Optimization Methods and Software*, **28.6**, 1195–1217, 2013.

[6] M. D'Apuzzo, V. De Simone, D. Di Serafino, Starting-Point Strategies for an Infeasible Potential Reduction Method, *Optimization Letters*, **4.1**, 131–146, Springer, 2010.