Dynamic Control of Infeasibility for Nonlinear Programming

Abel Soares Siqueira
Francisco A. M. Gomes

April 22, 2013

Project financed by Fapesp

UNICAMP

Introduction

- Objectives:
  - Extend the Dynamic Control of Infeasibility (DCI) for Equalities to handle inequalities;
  - Implement a C++ software with a CUTEr interface, easily available online;

Introduction

CUTEr problem:

$$
\begin{array}{llccc}
\min & f(x) & & & \\
\text{s.t.} & & c_E(x) & = & 0, \\
& c_L \;\leq & c_I(x) & \leq & c_U, \\
& b_L \;\leq & x & \leq & b_U,
\end{array} \tag{1}
$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $c_E : \mathbb{R}^n \longrightarrow \mathbb{R}^{m_E}$, $c_I : \mathbb{R}^n \longrightarrow \mathbb{R}^{m_I}$,
$f, c_E, c_I \in C^2$, $c_{U_i}, b_{U_i} \in \mathbb{R} \cup \{\infty\}$, $c_{L_i}, b_{L_i} \in \mathbb{R} \cup \{-\infty\}$.

Introduction

$$
\begin{array}{lccccc}
\min & f(x) & & & & \\
\text{s.t.} & & & c_E(x) & = & 0, \\
 & & & c_I(x) - s & = & 0, \\
 & b_L & \leq & x & \leq & b_U\,, \\
 & c_L & \leq & s & \leq & c_U\,,
\end{array}
\tag{2}
$$

Introduction

$$z = \begin{bmatrix} x \\ s \end{bmatrix} \qquad h(z) = \begin{bmatrix} c_E(x) \\ c_I(x) - s \end{bmatrix}$$

$\beta(z)$ boundary barrier

$$\begin{array}{rl} \min & \varphi(z,\mu) = f(x) + \mu\beta(z) \\ \text{s.t.} & h(z) = 0, \end{array} \tag{3}$$

Introduction

$\Lambda(z)$ scaling matrix

$$
\begin{aligned}
g(z,\mu) &= \Lambda(z)\nabla\varphi(z,\mu) \\
A(z) &= \nabla h(z)\Lambda(z), \\
\Gamma(z,\mu) &= \Lambda(z)\nabla^2\varphi(z,\mu)\Lambda(z) \\
W(z,\lambda,\mu) &= \lambda(z)\nabla^2_{zz}L(z,\lambda,\mu)\Lambda(z)
\end{aligned}
$$

UNICAMP

Introduction

$$\mathcal{C}(\rho) = \{z \in \mathbb{R}^N : \|h(z)\| \leq \rho\}$$

$$L(z, \lambda, \mu) = g(z, \mu) + \sum_{i=1}^{m} h_i(z)\lambda_i$$

Introduction

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & c_E(x) & = & 0, \\ & c_I(x) & \geq & 0, \end{array} \tag{4}$$

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & c_E(x) & = & 0, \\ & c_I(x) - s & = & 0, \\ & s & \geq & 0. \end{array} \tag{5}$$

Introduction

$$\Lambda(z) = \begin{bmatrix} I & 0 \\ 0 & S \end{bmatrix}$$

$$\lambda_{LS}(z,\mu) = \arg\min_{\lambda} \left\{ \frac{1}{2} \| g(z,\mu) + A(z)^T \lambda \|^2 \right\}$$

$$\lambda_i^k = \begin{cases} \lambda_{LS}(z_c^k, \mu_c^k)_i, & \text{se } i \in E \\ \min\{\lambda_{LS}(z_c^k, \mu_c^k)_i, \alpha(\mu_c^k)^n\}, & \text{se } i \in I \end{cases}$$

$$g_p(z,\mu) = g(z,\mu) + A(z)^T \lambda_{LS}(z,\mu).$$

$$g_p^k = g(z_c^k, \mu_c^k) + A(z_c^k)^T \lambda^k$$

$$\rho^k = \mathcal{O}(\|g_p^k\|)$$

Method

Horizontal Step

$$\begin{array}{ll} \min & q_k(\delta) = \frac{1}{2}\delta^T B^k \delta + \delta^T g_p(z_c^k, \mu^k), \\ & \nabla h(z_c^k)\delta = 0, \|\delta\| \leq \Delta_H \end{array}$$

Vertical Step

$$\begin{array}{ll} \min & \frac{1}{2}\|h(z)\|^2 \\ \text{s.t.} & l \leq z \leq u \end{array}$$

UNICAMP

Method

Algorithm 1 Outline of the k-th step of DCICPP

Given z^{k-1}
$z_c^k, \rho^k \longleftarrow$ **VertStep** $\qquad (z_c^k \in \mathcal{C}(\rho^k))$
Compute λ^k and μ^k
if $\|g_p^k\| < \varepsilon$ **and** $\|h(z_c^k)\| < \varepsilon$ **and** $\mu^k < \varepsilon.$ **then**
    STOP with $z^* = z_c^k$.
end if
Update ^{k}.
$\delta_t \longleftarrow$ **HorizStep**
if $z_c^k + \delta_t \notin C(\rho^k)$, **or** No Sufficient Decrease **then**
    Decrease Δ_H and return to the previous step.
end if
Optionally make a Second Order Correction δ_{soc}
Define $z^k = z_c^k + \Lambda(z_c^k)(\delta_t + \delta_{soc})$

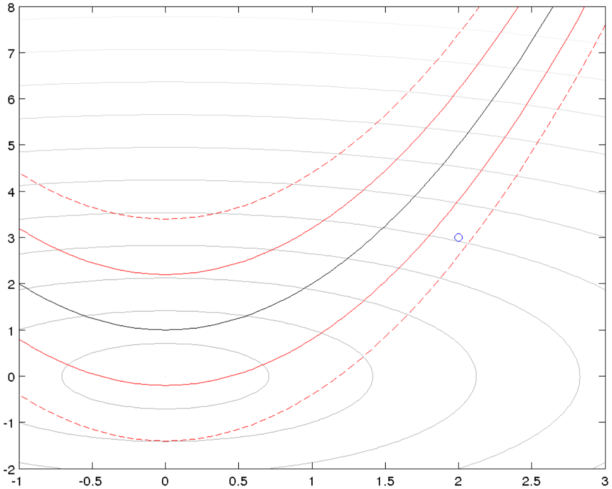
Method Example

Example:

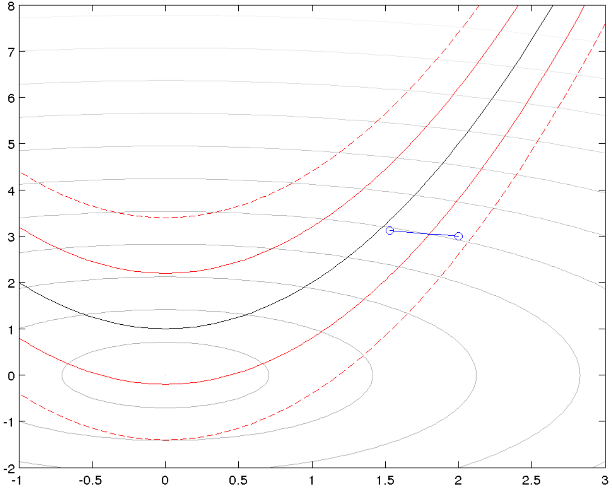
$$\begin{array}{ll} \min & f(x) = \frac{1}{2}(x_1^2 + x_2^2) \\ \text{s.t.} & x_2 = x_1^2 + 1 \end{array}$$

$$x^{k-1} = \left[\begin{array}{c} 2 \\ 3 \end{array} \right]$$

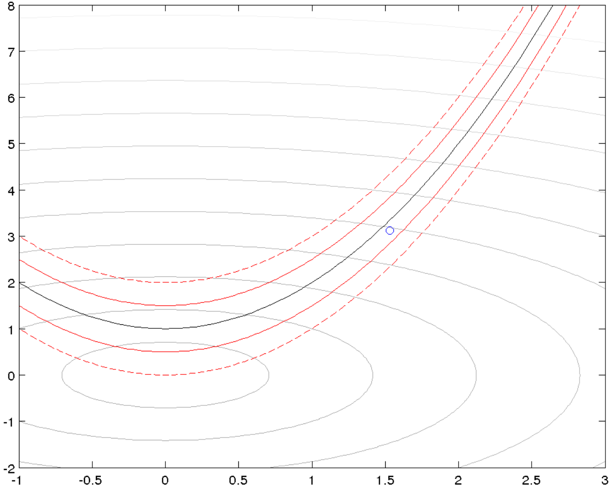
Method Example



Method Example

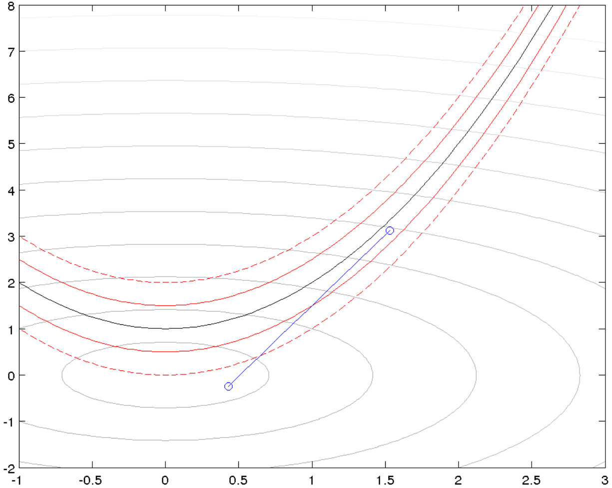


Method Example



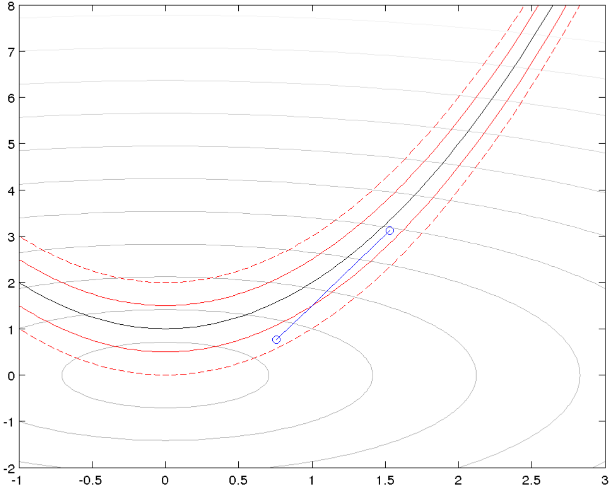
UNICAMP

Method Example

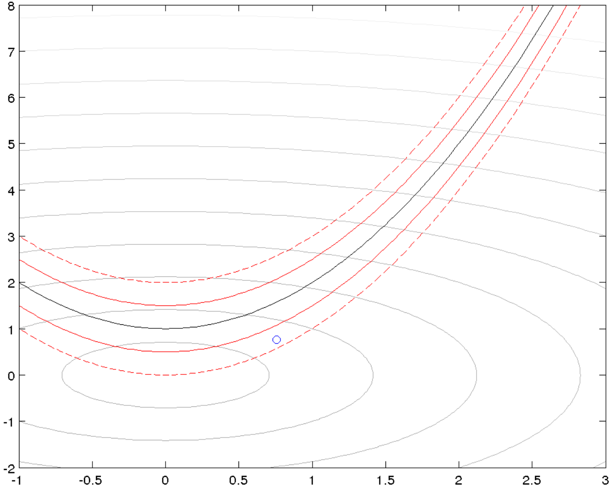


UNICAMP

Method Example



Method Example



UNICAMP

Method

Vertical Step

From z^{k-1}, we make steps that try to solve approximately

$$\min r(z) = \frac{1}{2}\|h(z)\|^2 \qquad \text{s. a} \qquad l \leq z \leq u$$

For this problem, we use a modification of the method proposed by Francisco, Krejić, and Martínez [2]

Method Vertical Step

Algorithm 2 VertStep

1: Given z^{k-1}, define $z_c = z^{k-1}$ and $\rho = \rho^{k-1}$.
2: **while** $\|h(z_c)\| > \rho$ **do**
3: $\quad z_c \longleftarrow$ **InnerVertStep** $\qquad (z_c \in \mathcal{C}(\rho))$
4: $\quad$ Update ρ.
5: $\quad$ **if** $\|h(z_c)\| > \varepsilon$ **and** $\|\nabla h(z_c)^T h(z_c)\| < \varepsilon$ **then**
6: $\qquad$ STOP $z^* = z_c$ an infeasibility stationary point.
7: $\quad$ **end if**
8: **end while**
9: Define $z_c^k = z_c$ and $\rho^k = \rho$.

Method — Vertical Step

Algorithm 3 InnerVertStep

while $\|h(z_c)\| > \rho$ **do**

  Define $m(d) = \frac{1}{2}\|\nabla h(z_c)d + h(z_c)\|^2$

  Compute $g = \nabla m(0) = \nabla h(z_c)^T h(z_c)$

  Define the matrix $D = \text{diag}(v_1, \ldots, v_N)$, where

$$v_i = \begin{cases} (u_i - z_i)^{-1/2}, & \text{se } g_i < 0 \text{ e } u_i < \infty \\ (z_i - l_i)^{-1/2}, & \text{se } g_i > 0 \text{ e } l_i > -\infty \\ 1, & \text{otherwise} \end{cases}$$

  Define $d = -D^{-2}g$

  Define $l_\varepsilon = l + \varepsilon_\mu(z_c - l) - z_c$ and $u_\varepsilon = u - \varepsilon_\mu(u - z_c) - z_c$.

  Define $\beta(d) = \arg\max\{t \geq 0 : l_\varepsilon \leq td \leq u_\varepsilon\}$.

  Compute $\alpha_{CP} = \arg\min_\alpha\{m(\alpha d) : \alpha\|Dd\| \leq \Delta_V\}$

Method — Vertical Step

Define $P(d) = \begin{cases} d, & \text{if } \beta(d) > 1 \\ \max\{\theta, 1 - \|d\|\}\beta(d)d, & \text{otherwise} \end{cases}$

Define $d_{CP} = P(\alpha_{CP}\, d)$.

Define $\rho_C(d) = \frac{m(0)-m(d)}{m(0)-m(d_{CP})}$ e $\rho_h(d) = \frac{r(z_c)-r(z_c+d)}{m(0)-m(d)}$.

Compute $\tilde{d}_N$, approximate solution of $\min_d\{m(d) : \|Dd\| \leq \Delta_V\}$.

Define $d_N = P(\tilde{d}_N)$.

Find $\tilde{d}$ convex combination of d_{CP} and d_N such that $\rho_C(\tilde{d}) \geq \beta_1$.

if $\rho_h(\tilde{d}) \geq \beta_2$ **then**

    $\Delta_V \leftarrow 2\Delta_V$

    $z_c \leftarrow z_c + \tilde{d}$.

else

    $\Delta_V \leftarrow \Delta_V/4$.

end if

end while

Method — Horizontal Step

$$
\begin{aligned}
\min \quad & q(\delta) = \frac{1}{2}\delta^T B^k \delta + \delta^T g(z_c^k) \\
\text{s.t.} \quad & A(z_c^k)\delta = 0, \\
& \tilde{l} \leq \Lambda(z_c^k)\delta \leq \tilde{u},
\end{aligned}
$$

where $B^k \approx W(z_c^k, \lambda^k, \mu_c^k)$ and

$$
\tilde{l}_i = \begin{bmatrix} -\Delta_H e \\ \max\{-\Delta_H e, (\varepsilon_\mu - 1)s_c^k\} \end{bmatrix} \qquad \tilde{u} = \Delta_H e
$$

This method is solved with a modification of Steihaug's method [5].

Method — Horizontal Step

Algorithm 4 Inner Horizontal Step

Given $r^0 = g_p^k$, $p^0 = r^0$, $j = 0$, $\delta^0 = 0$, $\theta^0 = \langle r^0, r^0 \rangle$.
while $\theta^j > \varepsilon$ **e** $\theta^k > \varepsilon\theta^0$ **do**
  if $\langle \delta^j, B^k \delta^j \rangle \leq \varepsilon\theta^j$ **then**
    Define $\delta_t = \delta^j + \nu p^j$ such that $\tilde{l} \leq \Lambda(z_c^k)\delta_t \leq \tilde{u}$ and ν minimizes $q(\delta^j + \nu p^j)$.
  end if
  $\alpha^j = \theta^j / \langle \delta^j, B^k \delta^k \rangle$
  if $\delta^j + \alpha^j p^j < \tilde{l}$ **OR** $\delta^j + \alpha^j p^j > \tilde{u}$ **then**
    Define $\delta_t = \delta^j + \overline{\nu} p^j$, where $\overline{v} = \arg\max\{\nu : \tilde{l} \leq \Lambda(z_c^k)\delta_t \leq \tilde{u}\}$.
  end if
  $\delta^{j+1} = \delta^j + \alpha^j p^j$.
  $r^{j+1} = \text{proj}_{\mathcal{N}(A(z_c^k))}(r^j - \alpha^j B^k p^j)$
  $\theta^{j+1} = \langle r^{j+1}, r^{j+1} \rangle$; $\beta^{k+1} = \theta^{j+1}/\theta^j$; $p^{j+1} = r^{j+1} - \beta^j p^j$.
end while

UNICAMP

Convergence

H1 f, c_E and c_I are C^2.

H2 The sequences $\{z_c^k\}$ and $\{z^k\}$, the approximations B^k and the multipliers $\{\lambda^k\}$ remain uniformly limited.

H3 The restoration never fails and $\mathcal{Z} = \{z_c^k\}$ remains far from the singular set of h, i.e., h is regular in the closure of $\mathcal{Z}$. Furthermore, if the generated sequence $\{z_c^k\}$ is infinity, then

$$\|z_c^{k+1} - z^k\| = \mathcal{O}(\|h(z^k)\|) \tag{6}$$

H4 $\|\delta_{soc}^k\| = \mathcal{O}(\|\delta_t^k\|^2)$

Convergence

Global Convergence

Theorem *Under H0-H4, DCI stops at a stationary point for* (4)*, in a finite number of iterations, or generates a sequence with stationary points in its accummulation set. Furthermore, if the conditions*

- C1 $\|z^k - z_c^k\| = \mathcal{O}(\|g_p(z_c^k, \mu_c^k)\|)$
- C2 $\|\lambda^k - \lambda_{LS}(z_c^k, \mu_c^k)\| = \mathcal{O}(\|g_p(z_c^k, \mu_c^k)\|)$
- C3 $\lambda_{LS}(z_c^{k+1}, \mu_c^{k+1})^T(s_c^{k+1} - s^k) = \mathcal{O}(\|g_p(z_c^k, \mu_c^k)\|\rho^k)$

are satisfied, then every accummulation point of z_c^k *is stationary for* (4)*.*

Convergence Local Convergence

Let $\{z^k\}$ and $\{z_c^k\}$ be generated from the algorithm, converging to z^*, $\{\lambda^k\}$ convergent to $\lambda^* = \lambda_{LS}(z^*, 0)$. From the algorithm, we have

$$\begin{cases} \nabla f(x^*) + \nabla c(x^*)^T \lambda^* & = & 0, \\ c_E(x^*) & = & 0, \\ c_I(x^*) & \geq & 0, \\ c_I(x^*)^T \lambda_I^* & = & 0, \\ \lambda_I^* & \leq & 0. \end{cases}$$

Define $\mathcal{A}(x) = \{i \in E \cup I : c_i(x) = 0\}$, and $\mathcal{A}^* = \mathcal{A}(x^*)$. Define λ_A^k and λ_A^* as the component of λ^k e λ^*, respectivally, corresponding to the active constraints.

Convergence Local Convergence

Suppose that $V = \{\nabla c_i(x^*) : i \in \mathcal{A}^*\}$ is linearly independent and that there is $\theta_1 > 0$, such that

$$y^T \left[\nabla^2 f(x^*) + \sum_{i \in \mathcal{A}^*} \nabla^2 c_i(x^*) \lambda_i^* \right] y \geq \theta_1 \|y\|^2,$$

for $y \in T = \{w : w^T \nabla c_i(x^*) = 0 : i \in E \cup J\}$, where $J = \{i \in I : \lambda_i^* < 0\}$. Define the matrix $\nabla c_A(x)$ whose lines are the vectors of V. In a neighbourhood of x^*, $\nabla c_A(x)$ has full rank. Hence, we can define

$$\begin{aligned}
\lambda_A(x) &= -[\nabla c_A(x) \nabla c_A(x)^T]^{-1} \nabla c_A(x) \nabla f(x), \\
g_A(x) &= \nabla f(x) + \nabla c_A(x)^T \lambda_A(x), \\
H_A(x, \lambda) &= \nabla^2 f(x) + \sum_{i \in \mathcal{A}^*} \nabla^2 c_i(x) \lambda_i \\
P(x) &= I - \nabla c_A(x)^T [\nabla c_A(x) \nabla c_A(x)^T]^{-1} \nabla c_A(x),
\end{aligned}$$

UNICAMP

Convergence — Local Convergence

A1 $\|\lambda^k - \lambda_{LS}(z_c^k, \mu_c^k)\| = \mathcal{O}(\|g_p(z_c^k, \mu_c^k)\|)$,
$\lambda_{LS}(z_c^{k+1}, \mu_c^{k+1})^T(s_c^{k+1} - s^k) = \mathcal{O}(\|g_p(z_c^k, \mu_c^k)\|\rho^k)$

A2 B^k is assimptotically uniformly positive definite on $\mathcal{N}(A(x_c^k))$, that is, in some neighbourhood of z^*, we can define $\theta_2 > 0$ and redefine θ_1 so that

$$\theta_1\|y\|^2 \leq y^T B^k y \leq \theta_2\|y\|^2,$$

for $y \in \mathcal{N}(A(z_c^k))$.

A3 For k sufficiently large,

$$\begin{aligned}
\|g_A(x_c^k)\| &= \Theta(\|g_p^k\|), \\
\|c_A(x_c^k)\| &= \Theta(\|h(z_c^k)\|), \\
\|c_A(x^k)\| &= \Theta(\|h(z^k)\|), \\
\|x_c^{k+1} - x^k\| &= \mathcal{O}(\|c_A(x^k)\|).
\end{aligned}$$

Convergence — Local Convergence

A4 Define the matrix Z_A^k whose columns form an orthonormal basis for the null space of $\nabla c_A(x_c^k)$. Define

$$\begin{aligned} \delta_x^k &= -Z_A^k[(Z_A^k)^T B_x^k Z_A^k]^{-1}(Z_A^k)^T g_A(x_c^k), \\ \delta_s^k &= (S_c^k)^{-1}\nabla c_I(x_c^k)\delta_x^k, \end{aligned}$$

and

$$\delta_A^k = \begin{bmatrix} \delta_x^k \\ \delta_s^k \end{bmatrix}.$$

Note that if $s_{c_i}^k \longrightarrow 0$, that is, $i \in \mathcal{A}^*$, then the corresponding component of δ_s^k is zero, therefore δ_s^k is limited. In addition, we define $s_{\min} > 0$ such that if $i \notin \mathcal{A}^*$, then $s_{c_i}^k \geq s_{\min}$. We assume that δ_A^k is the first step tried by the algorithm whenever $\|\delta_A^k\| \leq \Delta$ and $s_c^k + S_c^k \delta_s^k \geq \varepsilon_\mu s_c^k$. Besides, we assume that

$$P(x_c^k)[B_x^k - H_A(x^*, \lambda^*)]\delta_x^k = o(\|\delta_x^k\|).$$

Convergence — Local Convergence

A5 Each vertical step $\delta_V^{k+1} = z_c^{k+1} - z^k$ is obtained taking one or more steps in the form

$$\delta_V^+ = -J^T(JJ^T)^{-1}h(z_c),$$

where J satisfies

$$\|J - \nabla h(z_c)\| = \mathcal{O}(\|g_p^k\|).$$

Convergence Local Convergence

Theorem *With assumption H1-H4 and A1-A5,* x^k *and* x_c^k *are 2-step superlinearly convergent to* x^**. If a restoration is made at every* x^k*, then* $\{x^k\}$ *converges superlinearly to* x^**.*

Convergence — Infeasible Problems

If the problem is infeasible, the restoration phase can't find a feasible point. However, the method will find a stationary point for the infeasibility, that is, for the problem

$$\min \|c_E(x)\|^2 + \|c_I^-(x)\|^2.$$

The method we are using in the vertical step assure us that with the following assumptions:

I1 The sequence generated by the vertical algorithm is limited.

I2 Let L be a convex, open and limited set containing all points tried in the vertical algorithm. Then, for all $x, y \in L$, we have

$$\|\nabla h(x) - \nabla h(y)\| \leq 2\gamma_0 \|x - y\|.$$

UNICAMP

Numerical Experiments

- A C++ implementation of the method, called DCICPP, was created.
- DCICPP was built on top of the Cholesky library.
- GPL licensed, avaible online on Github.
- Used the following libraries
  - CHOLMOD [1] (Cholesky);
  - METIS [4] (permutation library for Cholesky);
  - GotoBLAS2 [6]
  - base_matrices (C++ wrapper for Cholesky);
  - CUTEr [3] (testing);

Numerical Experiments

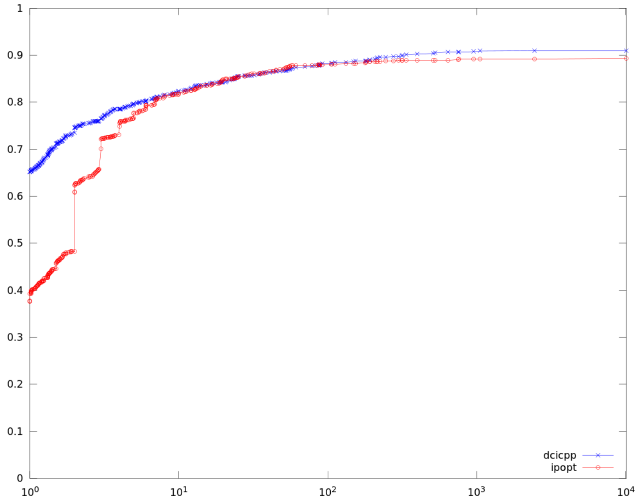
- 767 problems from CUTEr `small` selection
- Problems with fixed constraints were removed

Numerical Experiments

ExitFlag	Total	
	N°	%
Converged	698	91.00
Maximum	17	2.22
small $\rho_{\max}$	21	2.74
Max Time	15	1.96
Infeasible	7	0.91
Unlimited	6	0.78
Other fail	3	0.39
Total	767	100.00

Table: DCICPP results

Numerical Experiments Performance Profile



Next Steps

- Implement fixed variable support;
- Investigate each failed problem for a possible general solution;
- Experiment with singular jacobians;
- Investigate how to make it more efficient.

Bibliografia

[1] Y. Chen, T. A. Davis, W. W. Hager, and S. Raamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software*, 35(3), 887.

[2] J. B. Francisco, N. Krejić, and J. M. Martínez. An interior-point method for solving box-constrained underdetermined nonlinear system. *Journal of Computational and Applied Mathematics*, 177:67–88, 2005.

[3] N. Gould, D. Orban, and Ph. L. Toint. Cuter, a constrained and unconstrained testing environment, revisited. *Transactions of the American Mathematical Society on Mathematical Software*, 29(4):373–394, 2003.

[4] Karypis Lab. Metis - serial graph partitioning and fill-reducing matrix ordering. http://www-users.cs.umn.edi/ karypis/metis.

[5] Trond Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal of Numerical Analysis*, 20(3):626–637, 1983.

[6] TACC: Texas Advanced Computing Center. GotoBLAS2. http://www.tacc.utexas.edu/tacc-projects/gotoblas2/.

UNICAMP