Abel Soares Siqueira

Controle Dinâmico de Infactibilidade para Programação Não Linear

CAMPINAS

2013

Universidade Estadual de Campinas

Instituto de Matemática, Estatística
e Computação Científica

Abel Soares Siqueira

Controle Dinâmico de Infactibilidade para Programação Não Linear

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em matemática aplicada.

Orientador: Francisco de Assis Magalhães Gomes Neto

Este exemplar corresponde à versão final da tese defendida pelo aluno Abel Soares Siqueira, e orientada pelo Prof. Dr. Francisco de Assis Magalhães Gomes Neto.

Assinatura do Orientador

_____

Campinas

2013

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Maria Fabiana Bezerra Muller - CRB 8/6162

Si75c

Siqueira, Abel Soares, 1986-
Controle dinâmico de infactibilidade para programação não linear / Abel Soares Siqueira. – Campinas, SP : [s.n.], 2013.

Orientador: Francisco de Assis Magalhães Gomes Neto.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

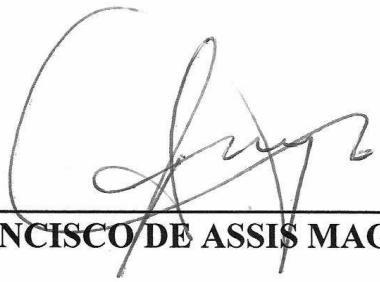
1. Programação não-linear. 2. Otimização matemática. 3. Otimização com restrições. 4. Métodos de pontos interiores. I. Gomes Neto, Francisco de Assis Magalhães,1964-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Dynamic control of infeasibility for nonlinear programming
Palavras-chave em inglês:
Nonlinear Programming
Mathematical Optimization
Constrained optimization
Interior-point methods
Área de concentração: Matemática Aplicada
Titulação: Doutor em Matemática Aplicada
Banca examinadora:
Francisco de Assis Magalhães Gomes Neto [Orientador]
José Mario Martínez Pérez
Roberto Andreani
Ernesto Julián Goldberg Birgin
Elizabeth Wegner Karas
Data de defesa: 02-12-2013
Programa de Pós-Graduação: Matemática Aplicada

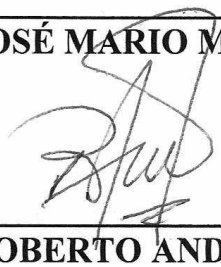
Tese de Doutorado defendida em 02 de dezembro de 2013 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.



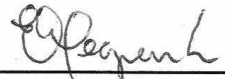
Prof(a). Dr(a). FRANCISCO DE ASSIS MAGALHÃES GOMES NETO

Prof(a). Dr(a). JOSÉ MARIO MARTÍNEZ PÉREZ



Prof(a). Dr(a). ROBERTO ANDREANI

Prof(a). Dr(a). ERNESTO JULIÁN GOLDBERG BIRGIN



Prof(a). Dr(a). ELIZABETH WEGNER KARAS

Abstract

One way to solve general nonlinear programming problems is the composite-step strategies. These strategies usually combine a step tangent to the constraints and a normal step, alternating between reducing the objective function value and the norm of the infeasibility. This kind of method requires the control of the steps or the iterates, in order to prevent one step from destroying the progress of another. We will present an extension of the Dynamic Control of Infeasibility method, which utilizes a strategy to control the steps known as Trust Cylinders. This method was originally designed for problems with equality contraints only, and our extension will handle general constraints. We'll show numerical experiments comparing our method with another composite-step method.

Keywords: Nonlinear Programming, Dynamic Control of Infeasibility, Composite-step Methods, Numerical Experiments.

Resumo

Uma maneira de resolver problemas gerais de programação não linear é utilizar estratégias de passos compostos. Essas estratégias normalmente combinam um passo tangente às restrições e um passo normal, alternando entre a diminuição da função objetivo e da norma da infactibilidade. Esse tipo de método exige o controle dos passos ou dos iterandos, para que não se perca o progresso de um

passo no outro. Apresentaremos uma extensão do método de Controle Dinâmico da Infactibilidade, que utiliza uma estratégia de controle de passos chamado de Cilindros de Confiança. Esse método foi desenvolvido para problemas com restrições apenas de igualdade, e nossa extensão lida com restrições gerais. Mostraremos testes numéricos comparando nosso método com um método do mesmo tipo.

Palavras-chave Programação Não-linear, Controle Dinâmico da Infactibilidade, Métodos de Passos Compostos, Experimentos Numéricos.

Conteúdo

Agradecimentos

Aos meus pais, pela educação e apoio toda a minha vida. Sem eles, nada disso seria capaz.

À Kally Chung, por estar ao meu lado por todos estes anos, e por me incentivar nas minhas escolhas.

Ao meu orientador Francisco de Assis, pela sua dedicação ao nosso trabalho, e por ter sido um ótimo professor desde a minha graduação.

Ao Raniere Gaia pela ajuda com o Linux, e ao Leandro Prudente, pela ajuda com o Fortran.

Aos meus colegas e amigos, que me ajudaram muito em diversas situações, especialmente Wanderson Luiz, Douglas Gonçalves e Thadeu Senne.

Ao meus professores, que me deram o conhecimento para chegar até aqui.

À toda minha família, em especial aos meus irmãos.

À Fundação de Amparo à Pesquisa do Estado de São Paulo (Fapesp) pelo apoio financeiro. (Processo 2009/17273-4)

Capítulo 1

Introdução

Vários problemas práticos podem ser formulados como problemas de otimização não linear. Das várias áreas de engenharia às teorias físicas, muitos problemas requerem a obtenção de um minimizador ou maximizador. Em muitos casos, é possível considerar propriedades específicas do problema e reduzir a complexidade do modelo, obtendo um problema de otimização com características especiais, como são os problemas de programação linear, inteira, quadrática; os problemas de quadrados mínimos, lineares e não lineares; os problemas convexos, irrestritos; e assim por diante. No entanto, quando não podemos fazer essa simplificação, é necessário enfrentar o problema de otimização não linear geral diretamente. Nesta tese, iremos apresentar um novo método para a solução desse problema. Nosso método é uma extensão do Método de Controle Dinâmico da Infactibilidade [5], originalmente desenvolvido para o problema com restrições apenas de igualdade. O método CDI (do inglês "Dynamic Control of Infeasibility") obtém seus iterandos através de passos normais e tangentes, utilizando aproximações quadráticas, e técnicas de pontos interiores. Nossa estratégia de globalização é baseada no que chamamos de *cilindros de confiança*, que são regiões ao redor do conjunto factível, com raio proporcional à norma do gradiente projetado. Esses cilindros limitam a infactibilidade dos iterandos. A cada iteração definimos um cilindro pequeno de raio ρ e um cilindro grande de raio 2ρ. Definimos o passo tangente de modo a diminuir o valor da norma do gradiente projetado, limitando o iterando ao cilindro grande. Prosseguimos então com um ou mais passos normais, até que o iterando fique dentro do cilindro pequeno.

Foi preciso decidir como lidar com as desigualdades do problema com cuidado. A maneira mais

direta, considerando o cilindro como uma penalidade das restrições, não herdava as propriedades do método. Escolhemos a estratégia de pontos interiores, e ela se mostrou muito eficaz. A convergência global do método não teve muitas modificações em relação ao original com essa estratégia. Para a convergência local, foi preciso considerar apenas as restrições ativas na solução. Isso gerou algumas mudanças nas hipóteses e nos teoremas, mas conseguimos obter os mesmos resultados que o método original. Uma grande parte deste trabalho foi a implementação do método. Sempre trabalhamos para suplantar o método original na robustez e na eficiência. Implementamos nossas estruturas de dados e interfaces tentando aproveitar ao máximo as ferramentas utilizadas.

Iniciaremos este trabalho com uma revisão do problema de programação não linear, e de estratégias clássicas no Capítulo 2. Os detalhes dos passos tangentes e normais serão mostrados no Capítulo 3, assim como o pseudo-código para o algoritmo. A teoria de convergência global é mostrada na seção 4.1, e a de convergência local está na seção 4.2. Também vamos falar sobre o comportamento do algoritmo com problemas infactíveis na Seção 4.3. No Capítulo 5 são mostrados detalhes da implementação do algoritmo, no Capítulo 6 mostramos os resultados computacionais obtidos com essa implementação, e no Capítulo 7 apresentamos nossas considerações finais.

Capítulo 2

Conceitos e Métodos de Programação Não Linear

Neste capítulo vamos apresentar alguns conceitos básicos de programação não linear e analisar alguns métodos tradicionais para o problema geral de otimização não linear.

2.1 Condições de Otimalidade

Na maior parte deste trabalho, iremos considerar o problema de programação não linear na forma

$$\begin{aligned} \min \quad & f(x) \\ \text{suj. a} \quad & c_E(x) = 0, \\ & c_I(x) \geq 0, \end{aligned} \tag{2.1}$$

com $f : \mathbb{R}^n \to \mathbb{R}$, $c_E : \mathbb{R}^n \to \mathbb{R}^{m_E}$, $c_I : \mathbb{R}^n \to \mathbb{R}^{m_I}$ continuamente diferenciáveis até segunda ordem. Definimos $m = m_E + m_I$ e a função $c : \mathbb{R}^n \to \mathbb{R}^m$,

$$c(x) = \begin{bmatrix} c_E(x) \\ c_I(x) \end{bmatrix}.$$

A função Lagrangeana para este problema é

$$\mathcal{L}(x, \lambda) = f(x) + c_E(x)^T \lambda_E + c_I(x)^T \lambda_I = f(x) + c(x)^T \lambda, \tag{2.2}$$

onde $\lambda \in \mathbb{R}^m$ são ditos multiplicadores de Lagrange. O problema de otimização não linear contínua consiste em encontrar um minimizador local do problema (2.1), isto é, um ponto x^* tal que $f(x^*) \leq f(x)$ para todo x numa vizinhança factível de x^*.

Definição 2.1 (Regularidade)**.** *Um ponto viável x é dito regular se os gradientes das restrições ativas em x são linearmente independentes, isto é, se o conjunto $\{\nabla c_i(x) : c_i(x) = 0\}$ é linearmente independente.*

Sob a condição de regularidade, podemos mostrar que um minimizador local do problema dado deve satisfazer as condições abaixo.

Teorema 2.1 (KKT)**.** *Se x^* é um minimizador local do problema* (2.1)*, então existe $\lambda^* \in \mathbb{R}^m$, separado em λ_E^* e λ_I^*, de acordo com as igualdades e desigualdades, tal que*

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) + \nabla c(x^*)^T \lambda^* = 0, \tag{2.3a}$$

$$c_E(x^*) = 0, \tag{2.3b}$$

$$c_I(x^*) \geq 0, \tag{2.3c}$$

$$c_{I_i}(x^*)\lambda_{I_i}^* = 0, \qquad i = 1, \ldots, m_I, \tag{2.3d}$$

$$\lambda_I^* \leq 0. \tag{2.3e}$$

As condições acima são ditas condições de KKT.

A condição (2.3a) é dita factibilidade dual. As condições (2.3b) e (2.3c) são ditas factibilidade primal. A condição (2.3d) é dita complementaridade e (2.3e) é uma condição sobre o sinal dos multiplicadores de desigualdade.

Vários métodos e estratégias foram criados para tentar resolver o problema (2.1). Apresentaremos agora alguns dos métodos mais tradicionais ou conhecidos, e vamos ainda comentar sobre estratégias de globalização. Alguns dos métodos citados resultaram em pacotes computacionais. Quando for possível, iremos mencionar quais são esses pacotes.

2.2 Métodos de Penalização

Os métodos de penalização definem uma função de medida para infactibilidade, e procuram resolver o problema irrestrito proveniente da combinação linear da função objetivo e da penalização. Por exemplo, para o problema (2.1), definimos a função de penalidade quadrática como

$$P(x) = \frac{1}{2}\sum_{i=1}^{m_E} c_{E_i}(x)^2 + \frac{1}{2}\sum_{i=1}^{m_I} \min\{0, c_{I_i}(x)\}^2.$$

Com essa função, definimos o problema irrestrito

$$\min \varphi(x, \rho) = f(x) + \rho P(x),$$

onde ρ é um parâmetro real positivo. Os métodos de penalização tentam minimizar essa função irrestrita, e aumentar ρ sucessivamente até obter um ponto factível. Pode-se mostrar que, dada uma sequência crescente $\{\rho^k\}$ tendendo ao infinito, e definindo como x^k o minimizador global da função $\varphi(x, \rho^k)$, a sequência $\{x^k\}$ converge para x^*, minimizador global do problema (2.1). Dentre os métodos de penalização, talvez o mais famoso seja o Lagrageano Aumentado [33], que define a função de Penalidade como

$$P(x, \lambda) = \frac{1}{2} \sum_{i=1}^{m_E} \left[c_{E_i}(x) + \frac{\lambda_{E_i}}{\rho} \right]^2 + \frac{1}{2} \sum_{i=1}^{m_I} \left[\min \left(0, c_{I_i}(x) + \frac{\lambda_{I_i}}{\rho} \right) \right]^2$$

utilizando o parâmetro λ, com $\lambda_I \leq 0$, dito multiplicador. Esse multiplicador é atualizado, usualmente tomando

$$\lambda_{E_i}^{k+1} = P_\lambda(\lambda_{E_i}^k + \rho^k c_{E_i}(x^k)), \qquad \lambda_{I_i}^{k+1} = P_\mu(\min\{0, \lambda_{I_i}^k + \rho^k c_{I_i}(x^k)\}),$$

onde P_λ e P_μ são as projetores nos intervalos $[\lambda_{\min}, \lambda_{\max}]$ e $[\mu_{\min}, \mu_{\max}]$, respectivamente. Sob certas condições, esses multiplicadores convergem para os Multiplicadores de Langrange. Esse método, com uma penalidade deslocada, tem propriedades de convergência melhores que a penalidade anterior, dita penalidade pura. Algumas implementações notáveis incluem o pacote LANCELOT, de Conn et al. [9], e o ALGENCAN, de Andreani et al. [3, 4].

2.3 Programação Quadrática Sequencial

Consideremos o problema com restrições de igualdade

$$\begin{aligned} \min \quad & f(x) \\ \text{suj. a} \quad & c_E(x) = 0. \end{aligned}$$

Uma estratégia para resolver esse problema consiste em fazer aproximações quadráticas sucessivas da função Lagrangeana, nesse caso definida como

$$\mathcal{L}(x, \lambda) = f(x) + c_E(x)^T \lambda_E.$$

Uma interpretação para as aproximações quadráticas é a aplicação do Método de Newton para o sistema não linear

$$F(x,\lambda) = \nabla\mathcal{L}(x,\lambda) = \begin{bmatrix} \nabla f(x) + \nabla c_E(x)^T\lambda \\ c_E(x) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

A Jacobiana de F é

$$F'(x,\lambda) = \begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x,\lambda) & \nabla c_E(x)^T \\ \nabla c_E(x) & 0 \end{bmatrix}.$$

O Método de Newton a partir do iterando (x^k,λ^k) será

$$\begin{bmatrix} x^{k+1} \\ \lambda^{k+1} \end{bmatrix} = \begin{bmatrix} x^k \\ \lambda^k \end{bmatrix} + \begin{bmatrix} p_x \\ p_\lambda \end{bmatrix},$$

onde p_x e p_λ satisfazem o sistema

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x^k,\lambda^k) & \nabla c_E(x^k)^T \\ \nabla c_E(x^k) & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla f(x^k) - \nabla c_E(x^k)^T\lambda^k \\ -c_E(x^k) \end{bmatrix}.$$

Este sistema está relacionado ao problema

$$\begin{aligned} \min_p \quad & \frac{1}{2}p^T\nabla^T_{xx}\mathcal{L}(x^k,\lambda^k)p + \nabla f(x^k)^T p + f(x^k) \\ \text{suj. a} \quad & \nabla c_E(x^k)p + c_E(x^k) = 0, \end{aligned}$$

que é a minimização da aproximação quadrática da função $\mathcal{L}(x^k+p_x,\lambda_k)$ com a aproximação linear da restrição $c_E(x^k+p_x)=0$. Nesse desenvolvimento, normalmente introduzimos uma região de confiança, de modo que o problema é reescrito como

$$\begin{aligned} \min_p \quad & \tfrac{1}{2}p^T\nabla^T_{xx}\mathcal{L}(x^k,\lambda^k)p + \nabla\mathcal{L}(x^k,\lambda_k)^T p + f(x^k) \\ \text{suj. a} \quad & \nabla c_E(x^k)p + c_E(x^k) = 0, \\ & \|p\| \leq \Delta. \end{aligned} \tag{2.4}$$

Infelizmente, nem sempre podemos garantir que esse subproblema é factível. Uma alternativa usada frequentemente [5–7, 11, 12, 17, 18, 20, 26, 27, 31] é a divisão desse passo em dois, normalmente ditos tangente e normal. O passo tangente normalmente envolve a substituição da restrição do

problema (2.4) por alguma restrição com garantia de factibilidade. Uma ideia é calcular passos no espaço tangente dado pela Jacobiana no iterando atual, representado pelo problema

$$\begin{aligned} \min_{p} \quad & \frac{1}{2}p^T\nabla^T_{xx}\mathcal{L}(x^k,\lambda^k)p + \nabla\mathcal{L}(x^k,\lambda_k)^T p + f(x^k) \\ \text{suj. a} \quad & \nabla c_E(x^k)d = 0, \\ & \|d\| \leq \Delta. \end{aligned}$$

A partir desse passo, fazemos um passo normal que procura diminuir o valor da medida de infactibilidade $\|c_E(x)\|^2$. Outra estratégia é obter primeiro um passo normal, minimizando aproximadamente $\|c_E(x^k + d)\|$, obtendo um passo d_N. Daí, tentamos resolver o problema

$$\begin{aligned} \min_{p} \quad & \frac{1}{2}p^T\nabla^T_{xx}\mathcal{L}(x^k,\lambda^k)p + \nabla\mathcal{L}(x^k,\lambda_k)^T p + f(x^k) \\ \text{suj. a} \quad & \nabla c_E(x^k)d = \nabla c_E(x^k)d_N, \\ & \|d\| \leq \Delta. \end{aligned}$$

Algumas implementações conhecidas são SNOPT, de Gill et al. [24], FILTERSQP, de Fletcher and Leyffer [21], e o GMM, de Gomes et al. [26], Gomes [27].

2.4 Métodos de Restrições Ativas

A estratégia de Restrições Ativas consiste em separar, a cada iteração, as restrições que devem ser tratadas como ativas. Feito isso, o método considera apenas as restrições ativas para determinar uma direção. A ideia provém das condições KKT (2.3), que podem ser reescritas como

$$\nabla f(x^*) + \sum_{i\in A} \nabla c_i(x^*)^T\lambda_i^* = 0, \tag{2.5}$$

$$c_i(x^*) = 0, \qquad i \in A, \tag{2.6}$$

$$c_i(x^*) > 0, \qquad i \notin A, \tag{2.7}$$

$$\lambda_i^* \leq 0, \qquad i \in A\backslash E, \tag{2.8}$$

$$\lambda_i^* = 0, \qquad i \notin A, \tag{2.9}$$

onde $A = \{i : c_i(x^*) = 0\} = E \cup \{I_i : c_{I_i} = 0\}$ é o conjunto dos índices das restrições ativas na solução. Se o conjunto A fosse conhecido a priori, poderíamos resolver o problema de igualdade

$$\min f(x) \qquad \text{suj. a} \qquad c_i(x) = 0, \qquad i \in A,$$

e a partir da solução deste, obter o valor das outras restrições e dos multiplicadores adicionais. Se os sinais estiverem corretos, a solução obtida corresponde à solução do problema original.

A ideia do método consiste então de chutar um conjunto de restrições ativas A, e procurar pela solução do problema com restrições de igualdade correspondente. Durante o processo, que inicia de um ponto factível, as direções são feitas de modo a não violar a factibilidade. Caso uma outra restrição seja encontrada no caminho, ela é incorporada ao conjunto de restrições ativas. Ao encontrar uma solução, se o sinal dos multiplicadores não estiver correto, o conjunto de restrições ativas é atualizado retirando a restrição correspondente.

Esse método é muito utilizado para problemas com restrições lineares, devido à facilidade de se movimentar na superfície gerada pelas restrições ativas. Para aplicar o método a restrições não lineares, é necessário projetar o passo na superfície ativa, além de se obter decréscimo suficiente da função objetivo.

2.5 Métodos de Pontos Interiores

Para a estratégia de Pontos Interiores, vamos considerar o problema (2.1) no formato

$$\begin{aligned} \min \quad & f(x) & \\ & c_E(x) &= 0, \\ \text{suj. a} \quad & c_I(x) - s &= 0, \\ & s &\geq 0. \end{aligned}$$

Daqui, definimos o problema de barreira

$$\begin{aligned} \min \quad & f(x) - \mu \sum_{i=1}^{m_I} \ln s_i & \\ & c_E(x) &= 0, \\ \text{suj. a} \quad & c_I(x) - s &= 0, \end{aligned}$$

onde $\mu > 0$. A condição $s > 0$ fica embutida na função objetivo desse novo problema. A estratégia de Pontos Interiores consiste em resolver esse problema aproximadamente, e reduzir gradativamente o parâmetro μ. Espera-se que a solução quando $\mu \to 0$ convirja para a solução do problema original. Classicamente, o método de Newton é aplicado ao sistema KKT desse subproblema, e μ é reduzido seguindo algum esquema de atualização. Outra maneira de resolver esse problema é utilizar técnicas de Programação Quadrática Sequencial ou até mesmo a combinação de penalização para as restrições de igualdade. Nessa linha destacam-se os trabalhos Byrd et al. [6, 7], Curtis et al. [12], Forsgren et al. [22], Wächter and Biegler [42]. Os pacotes LOQO, de Shanno and Vanderbei [36, 37] e o IPOPT, de Curtis et al. [12], Wächter [41], Wächter and Biegler [42], são algumas implementações tradicionais.

Nosso método utilizará as estratégias de pontos interiores e programação quadrática sequencial, fazendo a separação dos passos tangente e normal.

2.6 Convergência Global

Os algoritmos de Programação Não Linear precisam de algum controle das iterações para garantir que o método tenha convergência global, isto é, que gere uma sequência convergente a um ponto estacionário, ou ao menos com pontos de acumulação estacionário, partindo de um ponto inicial qualquer. Existem várias maneiras de se obter convergência global, sendo a maioria delas alguma extensão ou modificação do uso de busca linear, regiões de confiança, funções de mérito e filtro. Recomendamos a leitura de [10, 32, 34] para um aprofundamento nas técnicas. Aqui apresentaremos apenas a ideia básica dessas estratégias.

Uma maneira de avaliar o progresso do algoritmo é o uso de funções de mérito. Uma função de mérito é uma função que serve de medida para o progresso do algoritmo, levando em conta o valor da função objetivo e a infactibilidade do ponto atual. De um modo geral, podemos escrever uma função de mérito como

$$\phi(x; \mu) = f(x) + \mu P(x).$$

As escolhas para a função P são, tradicionalmente, as mesmas que as escolhas para penalização. Dado um iterando x^k e uma direção de descida d^k, fazemos uma busca linear para encontrar um

comprimento de passo α_k tal que $x^{k+1} = x^k + \alpha_k d^k$ resulta em um decréscimo suficiente para a função. Uma maneira de fazer essa busca é o chamado *backtracking*, em que se começa com $\alpha_k = 1$ e reduz-se α_k gradativamente até que alguma condição de decréscimo é satisfeita. Uma das condições mais conhecidas é a Condição de Armijo, onde α_k deve ser tal que

$$\phi(x^k + \alpha_k d^k; \mu) \leq \phi(x^k; \mu) + \eta \alpha_k D\phi(x^k; \mu; d^k),$$

onde $\eta \in (0, 1)$ e $D\phi(x; \mu; d)$ é a derivada direcional no ponto x, na direção d.

Outra maneira de escolher os passos é a utilização de filtros, uma ideia baseada em otimização multiobjetivo. Nessa estratégia, também definimos uma função de medida de infactibilidade, denominada por P, e consideramos o problema

$$\min_x f(x) \qquad \text{e} \qquad \min_x P(x).$$

Esses objetivos são considerados separadamente, e nos interessam os pontos que conseguem obter alguma melhora em ao menos um dos dois objetivos. Para cada ponto x, definimos um par (f, P), com $f = f(x)$ e $P = P(x)$. Dizemos que um par (f_a, P_a) domina (f_b, P_b) se $f_a \leq f_b$, $P_a \leq P_b$, e ao menos uma das desigualdades é estrita.

A estratégia de filtro consiste em manter um conjunto de pontos $\mathcal{F}$, onde nenhum dos pontos domina outro. Os iterandos são aceitos pela estratégia se o ponto obtido não é dominado por nenhum dos outros. Nesse caso, ele é adicionado ao conjunto, e os pontos que ele domina são retirados. Na prática, é comum utilizar algum decréscimo suficiente para a dominância também, para evitar que os passos sejam muito pequenos. A Figura 2.1 representa um possível filtro. O trabalho de Gonzaga, Karas, and Vanti [28] apresenta um método com filtro que utiliza uma estratégia de passos compostos.

Outra estratégia é a utilização de regiões de confiança, na qual escolhemos um raio Δ, e definimos a região $\mathcal{B}_k = \{x : \|x - x^k\| \leq \Delta\}$, de modo que a direção de descida d^k satisfaça $x^k + d^k \in \mathcal{B}_k$, isto é, que $\|d^k\| \leq \Delta$. Em geral, o passo é obtido a partir de um modelo de φ em torno de x^k, denotado por q. Esse passo é aceito se o decréscimo real da função, em comparação com o decréscimo previsto pelo modelo, for suficientemente pequeno. Definimos o decréscimo previsto, Pred, como

$$\text{Pred}(d) = q(0) - q(d),$$

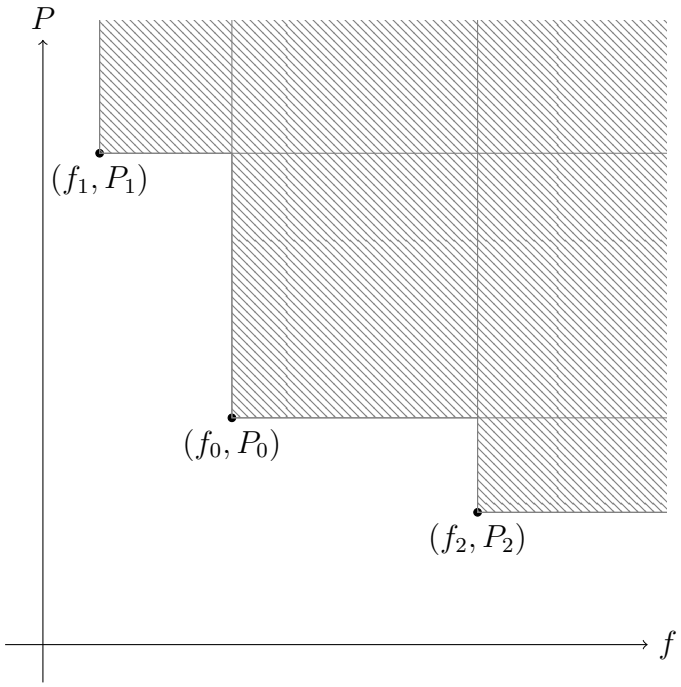


Figura 2.1: Pontos que compõem o filtro. A área hachurada representa a região dos pontos dominados

e o decréscimo real, Ared, como

$$\text{Ared}(d) = \phi(x^k; \mu) - \phi(x^k + d; \mu).$$

Em seguida, definimos a razão

$$\rho^k = \frac{\text{Ared}(d^k)}{\text{Pred}(d^k)},$$

e aceitamos ou rejeitamos o passo d^k de acordo com o valor de ρ^k. Por exemplo, se $\rho^k < \frac{1}{4}$, rejeitamos o passo d^k, reduzimos Δ, e calculamos um novo passo. Senão, aceitamos o passo. Se, além disso, $\rho^k > \frac{3}{4}$, então aumentamos o raio da região de confiança Δ.

O algoritmo apresentado por Bielschowsky and Gomes [5] define uma nova maneira de se obter convergência global, chamada de Cilindros de Confiança, que utilizamos nesta extensão do método. Com esses cilindros, conseguimos limitar a distância entre os iterandos e o conjunto factível, e permitir longos passos tangentes. Os detalhes dessa estratégia serão apresentados no capítulo seguinte.

Capítulo 3

O Método de Controle Dinâmico da Infactibilidade

O método de Controle Dinâmico da Infactibilidade (CDI) foi desenvolvido originalmente para problemas com restrições de igualdade em [5]. Nosso trabalho estende o algoritmo para lidar com inequações e limitantes nas variáveis. Para desenvolver o algoritmo, nós consideramos o problema no formato

$$\begin{array}{rlrcl} \min & f(x) & & & \\ \text{suj. a} & & c_E(x) & = & 0, \\ & c_L & \leq \; c_I(x) & \leq & c_U, \\ & b_L & \leq \quad x & \leq & b_U, \end{array} \tag{3.1}$$

onde $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $c_E : \mathbb{R}^n \longrightarrow \mathbb{R}^{m_E}$, $c_I : \mathbb{R}^n \longrightarrow \mathbb{R}^{m_I}$ são continuamente diferenciáveis até segunda ordem, $c_{L_i} \in \mathbb{R} \cup \{-\infty\}, c_{U_i} \in \mathbb{R} \cup \{\infty\}, i = 1, \ldots, m_I$, e $b_{L_i} \in \mathbb{R} \cup \{-\infty\}, b_{U_i} \in \mathbb{R} \cup \{\infty\}, i = 1, \ldots, n$. No entanto, para facilitar o desenvolvimento da teoria do método, vamos considerar o problema na forma mais simples, e tradicional, que apresentamos na seção anterior:

$$\begin{array}{rl} \min & f(x) \\ \text{suj. a} & c_E(x) = 0, \\ & c_I(x) \geq 0. \end{array} \tag{2.1}$$

Introduzindo uma variável de folga s nesse problema, obtemos

$$\begin{array}{rrcl} \min & f(x) & & \\ \text{suj. a} & c_E(x) & = & 0, \\ & c_I(x) - s & = & 0, \\ & s & \geq & 0. \end{array} \tag{3.2}$$

Definindo a variável

$$z = \begin{bmatrix} x \\ s \end{bmatrix},$$

e a função

$$h(z) = \begin{bmatrix} c_E(x) \\ c_I(x) - s \end{bmatrix}, \tag{3.3}$$

temos o problema com restrições de igualdade e limitantes

$$\begin{array}{rl} \min & f(x) \\ \text{suj. a} & h(z) = 0 \\ & s \geq 0. \end{array}$$

Agora, definimos uma função de barreira $\beta(z)$ que tende ao infinito quando a variável z se aproxima dos limitantes. No caso, como temos apenas $s \geq 0$, definimos

$$\beta(z) = -\sum_{i=1}^{m_I} \ln s_i. \tag{3.4}$$

Discutiremos o caso mais geral para a função de barreira na Seção 5.3. Finalmente, definimos a função

$$\varphi(z, \mu) = f(x) + \mu\beta(z), \tag{3.5}$$

e obtemos o problema

$$\begin{array}{rrcl} \min & \varphi(z, \mu) & & \\ \text{suj. a} & h(z) & = & 0. \end{array} \tag{3.6}$$

O Lagrangeano do problema (3.6) é dado por

$$L(z, \lambda, \mu) = \varphi(z, \mu) + \lambda^T h(z). \tag{3.7}$$

Como mencionamos no capítulo anterior, nosso método separa o progresso do algoritmo em passos tangentes e normais, utilizando aproximações quadráticas da função Lagrangeana e lineares das restrições. A ideia por trás de nosso passo tangente é tentar resolver aproximadamente o problema

$$\begin{aligned} \min_{d} \quad & L(z_c + d, \lambda, \mu) \\ \text{suj. a} \quad & h(z_c + d) = h(z_c), \end{aligned} \tag{3.8}$$

onde z_c é o ponto obtido na iteração anterior. Infelizmente, o problema acima pode ser mal condicionado devido às derivadas da função objetivo. Isso se deve à presença da inversa da matriz diagonal $S = \text{diag}(s_1, \ldots, s_{m_I})$ nas derivadas de φ, como mostrado a seguir:

$$\nabla_z \varphi(z, \mu) = \nabla_z f(x) + \mu \nabla \beta(z) = \begin{bmatrix} \nabla f(x) \\ -\mu S^{-1} e \end{bmatrix},$$

$$\nabla^2_{zz} \varphi(z, \mu) = \nabla^2_{zz} f(x) + \mu \nabla^2 \beta(z) = \begin{bmatrix} \nabla^2 f(x) & 0 \\ 0 & \mu S^{-2} \end{bmatrix}.$$

Para facilitar a visualização, vamos omitir o índice z das derivadas quando não houver confusão. Desse modo $\nabla \varphi(z, \mu) = \nabla_z \varphi(z, \mu)$ e $\nabla^2 \varphi(z, \mu) = \nabla^2_{zz} \varphi(z, \mu)$. Para evitar esse mal condicionamento, vamos introduzir uma matriz de escalamento $\Lambda(z)$, definida como

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

Assim, em nosso passo tangente, fazemos a substituição $d = \Lambda(z_c)\delta$, melhorando o condicionamento do problema. Definimos o gradiente, a Hessiana de φ e a Hessiana do Lagrangeano escalados,

respectivamente, como

$$g(z,\mu) = \Lambda(z)\nabla\varphi(z,\mu) = \begin{bmatrix} \nabla f(x) \\ -\mu e \end{bmatrix}, \tag{3.9}$$

$$\Gamma(z,\mu) = \Lambda(z)\nabla^2\varphi(z,\mu)\Lambda(z) = \begin{bmatrix} \nabla^2 f(x) & 0 \\ 0 & \mu I \end{bmatrix}, \tag{3.10}$$

$$\begin{aligned} W(z,\lambda,\mu) &= \Lambda(z)\nabla^2_{zz}L(z,\lambda,\mu)\Lambda(z), \\ &= \Gamma(z,\mu) + \Lambda(z)\Big[\sum_{i=1}^{m}\lambda_i\nabla^2 h_i(z)\Big]\Lambda(z) \\ &= \Gamma(z,\mu) + \sum_{i=1}^{m}\lambda_i \begin{bmatrix} \nabla^2 c_i(x) & 0 \\ 0 & 0 \end{bmatrix}, \\ &= \begin{bmatrix} \nabla^2 f(x) + \sum_{i=1}^{m}\lambda_i\nabla^2 c_i(x) & 0 \\ 0 & \mu I \end{bmatrix} \end{aligned} \tag{3.11}$$

onde $m = m_E + m_I$. Também definimos a Jacobiana escalada como

$$\begin{aligned} A(z) &= \nabla h(z)\Lambda(z) = \begin{bmatrix} \nabla c_E(x) & 0 \\ \nabla c_I(x) & -I \end{bmatrix}\Lambda(z) \\ &= \begin{bmatrix} \nabla c_E(x) & 0 \\ \nabla c_I(x) & -S \end{bmatrix}, \end{aligned} \tag{3.12}$$

Nosso método utiliza aproximações para os multiplicadores de Lagrange. Essas aproximações são feitas a partir da estimativa por quadrados mínimos da solução da equação de factibilidade dual escalada

$$\Lambda(z)\nabla L(z,\lambda,\mu) = g(z,\mu) + A(z)^T\lambda = 0.$$

Sendo assim, as estimativas são definidas como

$$\lambda_{LS}(z,\mu) = \arg\min_{\lambda}\Big\{\frac{1}{2}\|A(z)^T\lambda + g(z,\mu)\|^2\Big\}. \tag{3.13}$$

Note que, se $A(z)$ tem posto completo, então

$$\lambda_{LS}(z,\mu) = -[A(z)A(z)^T]^{-1}A(z)g(z,\mu).$$

A fatoração da matriz $A(z)A(z)^T$ é feita para calcular os passos internos, de modo que o custo para encontrar esses multiplicadores não aumenta consideravelmente o custo total da iteração.

Separando $\lambda_{LS}(z,\mu)$ em

$$\lambda_{LS}(z,\mu) = \begin{bmatrix} \lambda_E(z,\mu) \\ \lambda_I(z,\mu) \end{bmatrix},$$

$\lambda_E(z,\mu) \in \mathbb{R}^{m_E}$ e $\lambda_I(z,\mu) \in \mathbb{R}^{m_I}$, definimos o gradiente no ponto z projetado no espaço nulo de $A(z)$ escalado por $\Lambda(z)$ como

$$\begin{aligned} g_p(z,\mu) &= g(z,\mu) + A(z)^T\lambda_{LS}(z,\mu) && (3.14) \\ &= \Lambda(z)\nabla_z L(z,\lambda_{LS}(z,\mu),\mu) \\ &= \begin{bmatrix} \nabla f(x) + \nabla c(x)^T\lambda_{LS}(z,\mu) \\ -\mu e - S\lambda_I(z,\mu) \end{bmatrix}. && (3.15) \end{aligned}$$

Vamos denotar esta função vetorial como o gradiente projetado. Na iteração k do algoritmo, calculamos aproximações λ_k e g_p^k para os valores acima. Os detalhes de como são calculadas essas aproximações serão apresentados na próxima seção.

3.1 O Algoritmo CDI

A base do método é o uso do que chamamos de cilindros de confiança. Esses cilindros são regiões ao redor do conjunto factível, definidas por

$$\mathcal{C}(\rho) = \{z \in \mathbb{R}^n : \|h(z)\| \leq \rho\}. \tag{3.16}$$

O método é dividido em passos normais e tangentes. Os passos normais servem para aproximar os iterandos da região factível e os passos tangentes tentam melhorar o progresso dual, segundo uma medida de otimalidade. A Figura 3.1 esboça o processo. Utilizamos como medida de otimalidade a norma do gradiente projetado aproximado na iteração k, denotado por g_p^k, e escolhemos raios dos cilindros de confiança proporcionais a essa medida, isto é,

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

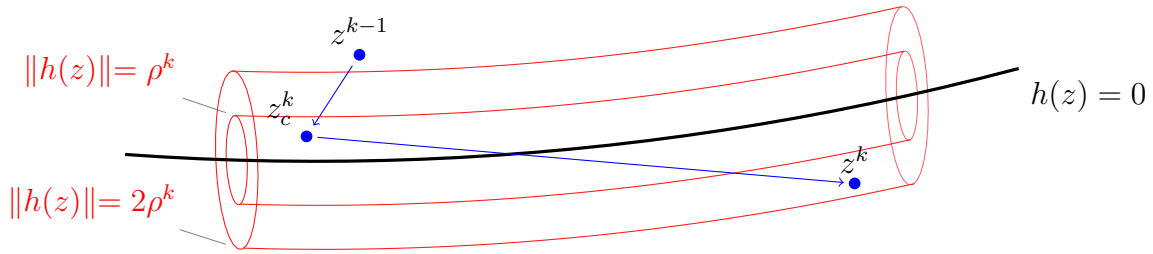


Figura 3.1: A região factível, os cilindros, e os passos do algoritmo. O ponto z_c^k satisfaz $\|h(z_c^k)\| \leq \rho^k$ e z^k satisfaz $\|h(z^k)\| \leq 2\rho^k$.

Nossas iterações devem ser mantidas estritamente factíveis em relação aos limitantes das variáveis, isto é, devemos ter $s > 0$. Para garantir isso, dada a direção d a partir de um ponto z, calculamos um tamanho máximo de passo α, tal que a nova iteração satisfaça

$$s + \alpha d_s \geq \varepsilon_\mu s \tag{3.17}$$

para algum $\varepsilon_\mu > 0$ pequeno dado, onde d_s é a componente do vetor d correspondente às variáveis s. Um esboço do método está descrito no Algoritmo 3.1.

Algoritmo 3.1 Método CDI

1: Parâmetros: $\varepsilon_g > 0$, $\varepsilon_h > 0$, $\varepsilon_a > 0$, $\varepsilon_\mu \in (0,1)$ e $\nu \in [10^{-4}, 1]$.

2: Valores Iniciais: z^0, $\rho^0 > 0$, μ^0, $k = 1$, $\rho_{\max}^0$.

3: Faça a **Etapa Normal**, encontrando z_c^k, λ^k, ρ^k, μ^k e $g_p^k = g(z_c^k, \mu^k) + A(z_c^k)^T \lambda^k$ tais que

$$\|h(z_c^k)\| \leq \rho^k \leq \nu \frac{\|g_p^k\| \rho_{\max}^{k-1}}{\|g(z_c^k, \mu^k)\| + 1}, \qquad \text{e} \qquad s_c^k \geq \varepsilon_\mu s^{k-1}$$

4: **Se** $\|h(z_c^k)\| < \varepsilon_h$, $\|g_p^k\| < \varepsilon_g$ e $\left|(s_c^k)^T \lambda^k\right| < \varepsilon_a$ **Então**

5: PARE com $z^* = z_c^k$.

6: **Fim do Se**

7: Faça a **atualização de** $\rho_{\max}^k$.

8: Faça a **Etapa Tangente**, encontrando z_k com decréscimo suficiente para a função Lagrangeana e tal que

$$\|h(z^k)\| \leq 2\rho^k \qquad \text{e} \qquad s^k \geq \varepsilon_\mu s_c^k.$$

9: Incremente k e volte para o passo 3.

A etapa normal do método é composta do cálculo do passo normal e da atualização dos multiplicadores e do gradiente projetado. Os requerimentos sobre o formato desse passo são poucos, como veremos posteriormente. Escolhemos encontrar um passo normal com uma sequência de iterações de algum método para resolver o problema

$$\begin{aligned} \min \quad & \frac{1}{2}\|h(z)\|^2 \\ \text{suj. a} \quad & s \geq 0. \end{aligned} \tag{3.18}$$

Nossa estratégia consiste em encontrar um z_c dentro do cilindro de raio ρ, atualizar o raio, e verificar se z_c permanece dentro do cilindro. Caso não permaneça, repetimos o processo. Cada aplicação do método inicia no ponto z_c anterior, sendo que o primeiro z_c da iteração k é escolhido como $z_c = z^{k-1}$. Cada passo que leva a um ponto dentro do cilindro é dito um **passo normal interno**. Definimos os subproblemas normais como

$$\begin{aligned} \min \quad & \frac{1}{2}\|h(z_c + d)\|^2 \\ \text{suj. a} \quad & \ell_N \leq d \leq u_N, \end{aligned} \tag{3.19}$$

onde

$$\ell_N = \begin{bmatrix} -\Delta_N e_n \\ -\min\{\Delta_N e_m, (1-\varepsilon_\mu)s^{k-1}\} \end{bmatrix} \quad \text{e} \quad u_N = \Delta_N e_{n+m}, \tag{3.20}$$

com e_n, e_m e e_{n+m} são vetores com todas as componentes iguais a 1, de tamanho n, m e $n+m$, respectivamente. O Algoritmo 3.2 mostra o esboço da etapa normal.

Algoritmo 3.2 Etapa Normal

1: Parâmetros: $\alpha_\rho > 0$ e $\alpha_h > 0$.
2: Valores Iniciais: $z_c = z^{k-1}$.
3: Calcule λ, $g_p = g(z_c, \mu) + A(z_c)^T\lambda$, ρ e μ.
4: **Enquanto** $\|h(z_c)\| > \rho$ **Faça**
5: Encontre z_c tal que $\|h(z_c)\| \leq \rho$ e $s_c \geq \varepsilon_\mu s^{k-1}$ pelo **passo normal interno**.
6: Calcule λ, $g_p = g(z_c, \mu) + A(z_c)^T\lambda$, ρ e μ.
7: **Fim do Enquanto**
8: Defina $\lambda^k = \lambda$, $g_p^k = g_p$ e $\rho^k = \rho$.
9: Defina $\mu^k = \min\left\{\mu^{k-1}, \alpha_\rho\rho, \alpha_\rho\rho^2, \dfrac{(s_c^k)^T \max\{0, -\lambda_I^k\}}{m_I}, \alpha_h h(z^k)\right\}$.

No algoritmo, calculamos λ^k de maneira que fique próximo a $\lambda_{LS}(z_c^k, \mu^k)$, mas garantimos que o sinal dos multiplicadores associados às desigualdades estejam corretos quando μ tende a 0. Esse cálculo é feito como

$$\lambda_i^k = \begin{cases} \lambda_{LS_i}(z_c^k, \mu^k), & \text{se } i \in E \\ \min\{\lambda_{LS_i}(z_c^k, \mu^k), \alpha(\mu_k)^r\}, & \text{se } i \in I \end{cases} \tag{3.21}$$

onde $\alpha > 0$ e $r > 0$ são escolhidos adequadamente. Com esse multiplicador, definimos o gradiente projetado aproximado g_p^k como

$$g_p^k = g(z_c^k, \mu^k) + A(z_c^k)^T \lambda^k = \begin{bmatrix} \nabla_x \mathcal{L}(x_c^k, \lambda^k) \\ -\mu^k e - S_c^{k+1} \lambda_I^k \end{bmatrix}, \tag{3.22}$$

de modo que g_p^k ficará próximo de $g_p(z_c^k, \mu^k)$.

A etapa tangente é responsável por diminuir a infactibilidade dual, calculando um passo tangente que forneça decréscimo suficiente. O passo é obtido utilizando uma aproximação quadrática para o Lagrangeano e regiões de confiança, com a direção escalada pela matriz $\Lambda(z_c^k)$. Lembramos aqui que, para evitar o mal condicionamento, utilizamos uma aproximação quadrática de $L(z_c^k + \Lambda(z_c^k)\delta, \lambda^k, \mu^k)$. Além do decréscimo, o ponto não pode sair do cilindro $\mathcal{C}(2\rho^k)$. O passo tangente δ_t é obtido através da solução aproximada para o problema

$$\begin{aligned} \min_{\delta} \quad & q_k(\delta) = \frac{1}{2}\delta^T B^k \delta + \delta^T g_p^k \\ \text{suj. a} \quad & A(z_c^k)\delta = 0 \\ & \ell_T \leq \Lambda(z_c^k)\delta \leq u_T, \end{aligned} \tag{3.23}$$

onde B^k é uma aproximação para $W(z_c^k, \lambda^k, \mu^k)$, e os limitantes ℓ_T e u_T têm o mesmo formato daqueles definidos em (3.20), mas com uma região de confiança de raio Δ_T, e s_c^k no lugar de s^{k-1}. Pedimos que o passo encontrado seja ao menos melhor que um passo de Cauchy. Então, utilizamos uma implementação que começa com o passo de Cauchy e tenta melhorar o valor da quadrática, até ser forçado a parar pelos limitantes, ou ao obter um minimizador da quadrática. Após o passo tangente, ainda podemos fazer uma correção de segunda ordem, conforme sugerido em [10]. Mostraremos como, e quando, é feita essa correção na Subseção 5.5.2. O ponto obtido pelo passo tangente e pela correção de segunda ordem é o próximo iterando z^k. O algoritmo 3.3 mostra um esboço da etapa tangente.

Algoritmo 3.3 Etapa Tangente

1: Parâmetros: $\eta_1 \in (0, \frac{1}{2}]$, $\eta_2 > \eta_1$, $\alpha_R \in (0, \frac{3}{4}]$, $\alpha_I > 1$.

2: Valores Iniciais: Δ_T, $r = 0$ e $z^+ = z_c^k$.

3: **Enquanto** $\|h(z^+)\| > 2\rho^k$ ou $r < \eta_1$ **Faça**

4: Construa ℓ_T e u_T usando Δ_T.

5: Calcule o Passo de Cauchy $\delta_{CP} = -\alpha_{CP} g_p^k$, onde α_{CP} é solução de

$$\begin{aligned} \min_{\alpha>0} \quad & q(-\alpha g_p^k) \\ \text{suj. a} \quad & \ell_T \leq -\alpha \Lambda(z_c^k) g_p^k \leq u_T. \end{aligned}$$

6: Calcule um **passo tangente interno** δ_t tal que

$$\begin{aligned} q(\delta_t) &\leq q(\delta_{CP}), \\ A(z_c^k)\delta_t &= 0, \\ \ell_T &\leq \Lambda(z_c^k)\delta_t \leq u_T. \end{aligned}$$

7: Se necessário, calcule uma correção de segunda ordem δ_{soc}.

8: $d^+ = \Lambda(z_c^k)(\delta_t + \delta_{soc})$.

9: $z^+ = z_c^k + d^+$

10: $\Delta L_T^k = L(z^+, \lambda^k, \mu^k) - L(z_c^k, \lambda^k, \mu^k)$

11: $r = \dfrac{\Delta L_T^k}{q(\delta_t)}$

12: **Se** $\|h(z^+)\| > 2\rho^k$ OU $r < \eta_1$ **Então**

13: $\Delta_T = \alpha_R \Delta_T$

14: **Senão Se** $r > \eta_2$ **Então**

15: $\Delta_T = \alpha_I \Delta_T$

16: **Fim do Se**

17: **Fim do Enquanto**

18: Defina $z^k = z^+$

Além da norma do gradiente projetado na iteração k, também usamos a variação do Lagrangeano para controlar o tamanho do raio do cilindro. Para explicitarmos a contribuição dos passos

normal e tangente, dividimos a variação do Lagrangeano calculado logo após o passo normal em

$$\Delta L_c^k = L_c^k - L_c^{k-1} = \Delta L_T^{k-1} + \Delta L_N^k, \tag{3.24}$$

onde

$$\begin{aligned} L_c^k &= L(z_c^k, \lambda^k, \mu^k), \\ \Delta L_T^k &= L(z^k, \lambda^k, \mu^k) - L(z_c^k, \lambda^k, \mu^k), \qquad (3.25) \\ \Delta L_N^k &= L(z_c^k, \lambda^k, \mu^k) - L(z^{k-1}, \lambda^{k-1}, \mu^{k-1}). \qquad (3.26) \end{aligned}$$

A maneira que a atualização de $\rho_{\max}$ é feita está descrita no Algoritmo 3.4.

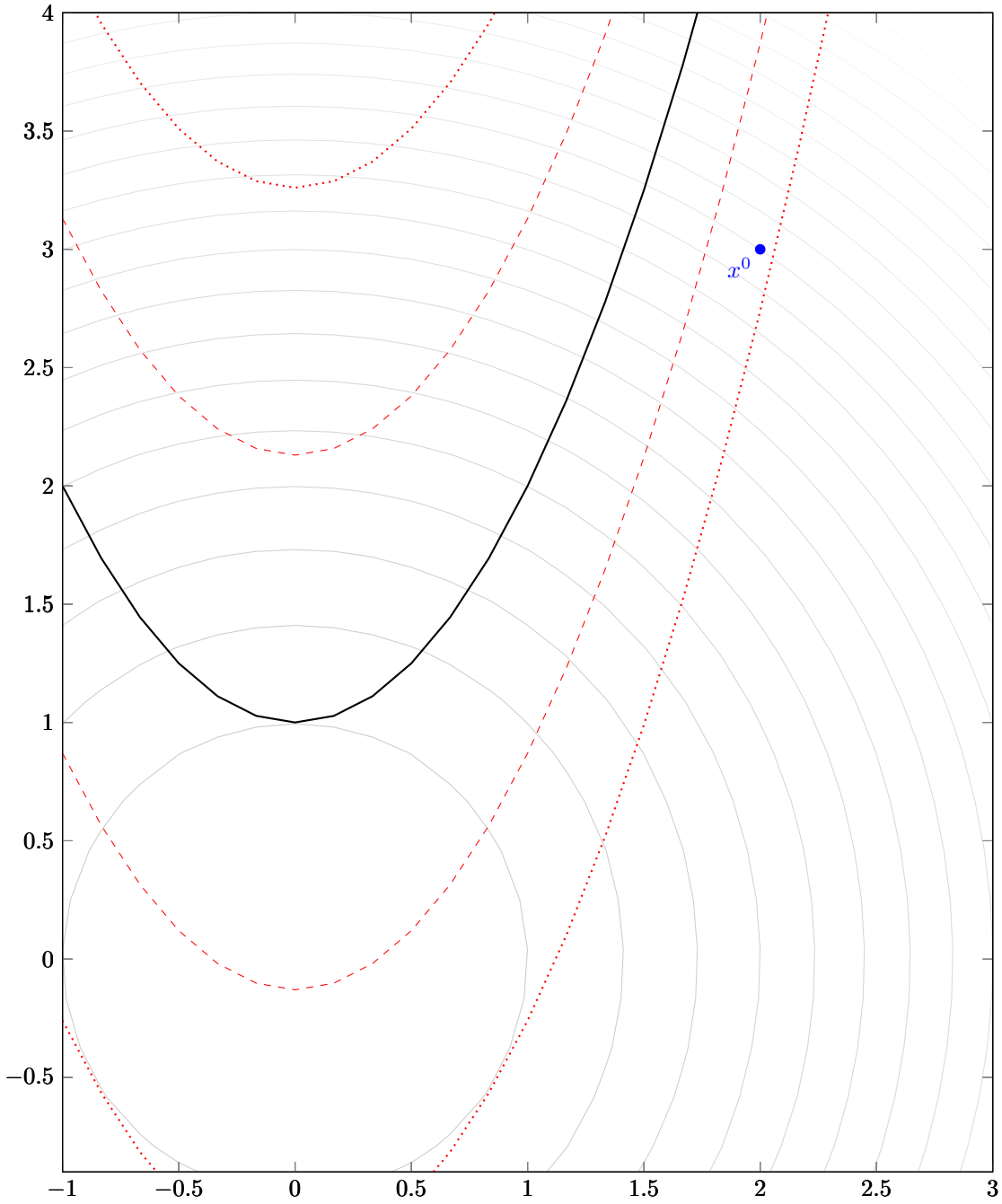
Algoritmo 3.4 Atualização de $\rho_{\max}$

1: Valores Iniciais: L_{ref} definido em iterações anteriores. Inicialmente $L_{ref} = \infty$.
2: $\Delta L_N^k = L(z_c^k, \lambda^k, \mu^k) - L(z^{k-1}, \lambda^{k-1}, \mu^{k-1})$
3: **Se** $\Delta L_N^k \geq \frac{1}{2}[L_{ref} - L(z^{k-1}, \lambda^{k-1}, \mu^{k-1})]$ **Então**
4: $\quad \rho_{\max}^k = \rho_{\max}^{k-1}/2$
5: **Senão**
6: $\quad \rho_{\max}^k = \rho_{\max}^{k-1}$
7: **Fim do Se**
8: **Se** $\Delta L_N^k > -\frac{1}{2}\Delta L_T^{k-1}$ **Então**
9: $\quad L_{ref} = L(z_c^k, \lambda^k, \mu^k)$
10: **Fim do Se**

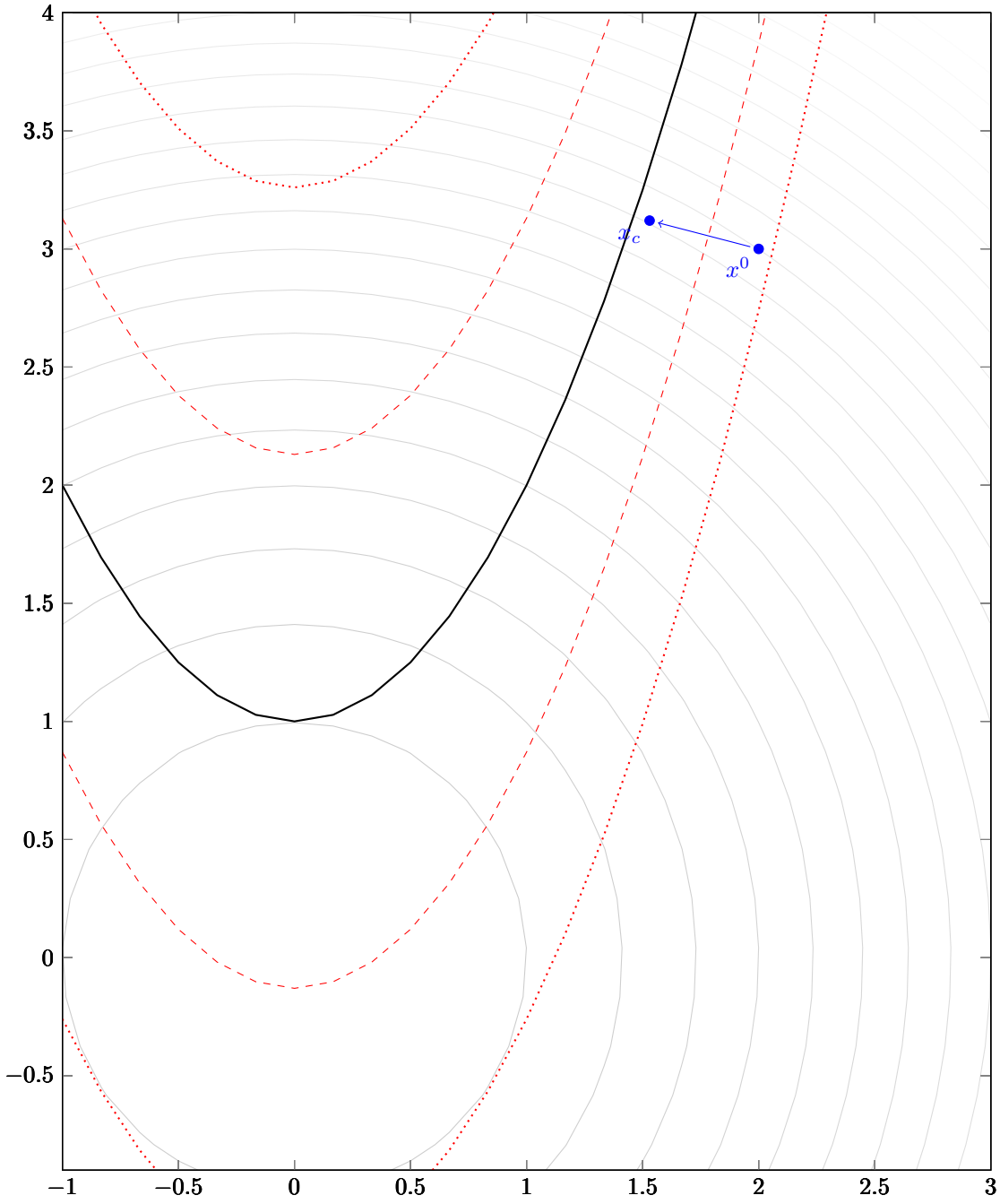
Para deixar mais claro o funcionamento do método, vamos aplicar o algoritmo à dois exemplos. O primeiro é o problema

$$\begin{aligned} \min \quad & f(x) = \tfrac{1}{2}(x_1^2 + x_2^2) \\ \text{s.t} \quad & x_2 = x_1^2 + 1, \end{aligned}$$

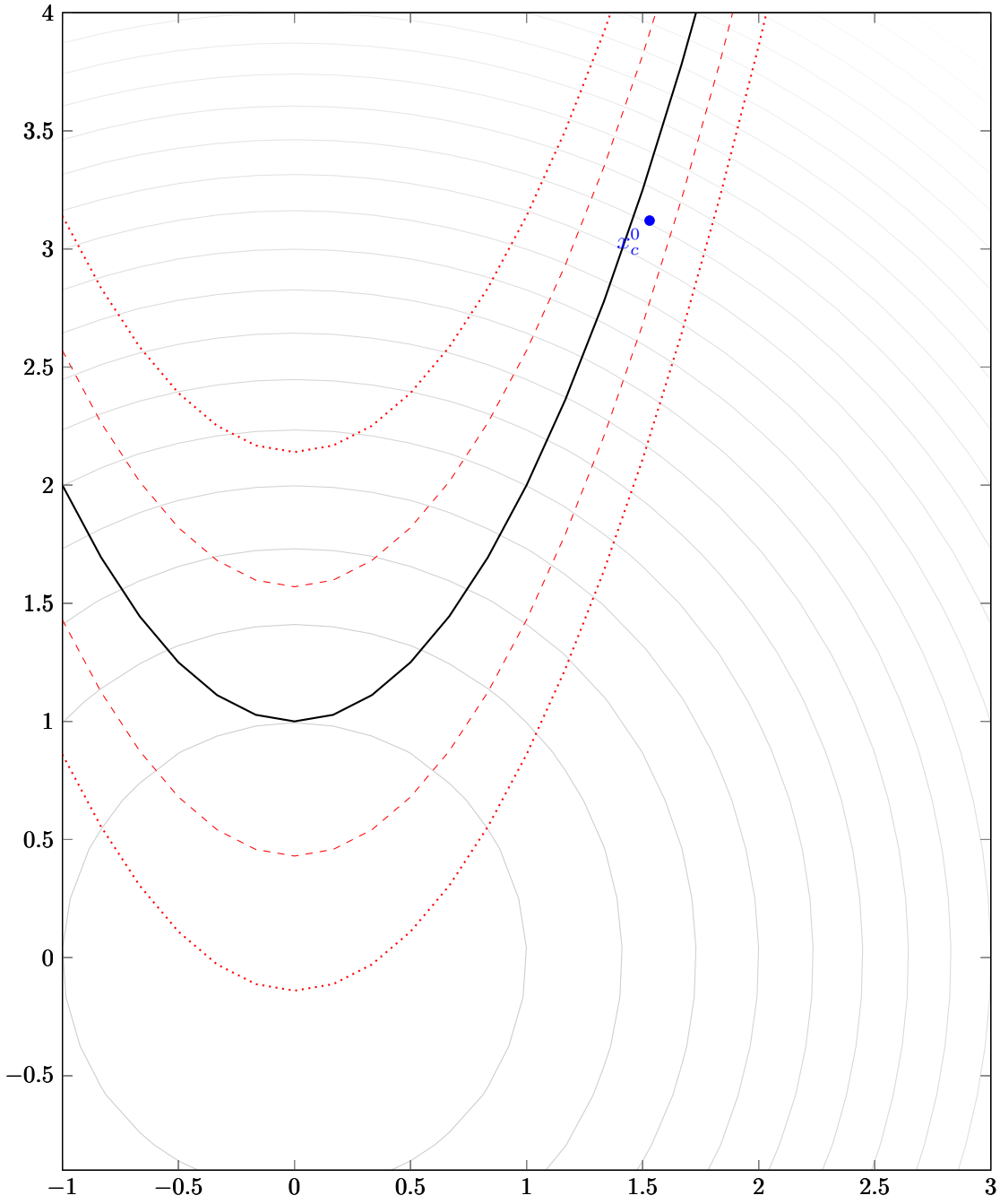
cuja solução é o ponto $(1, 0)$.



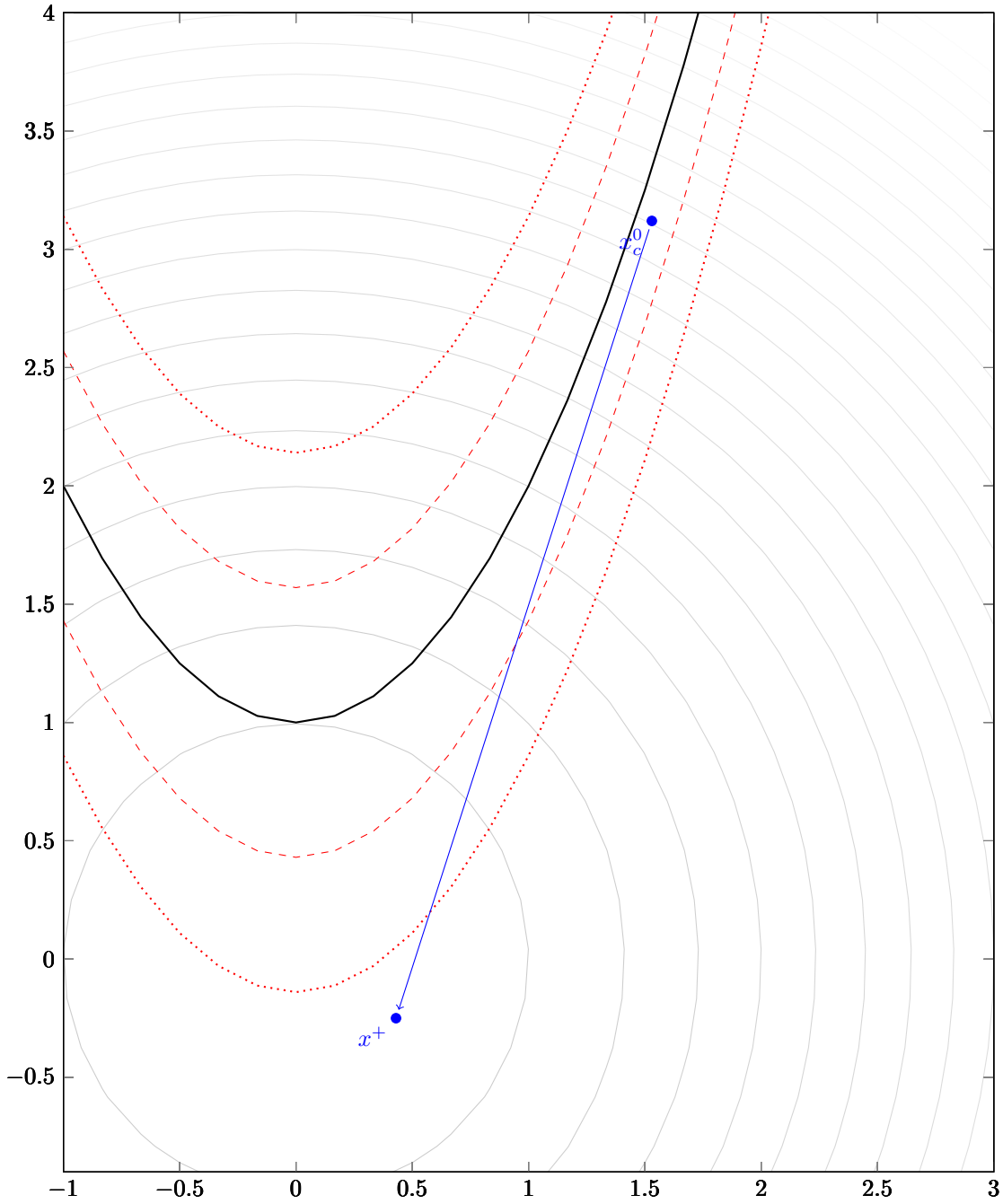
Começamos pelo ponto $x_0 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$, denotado pelo círculo sólido na imagem. A curva sólida representa a região factível, as curvas tracejadas denotam o cilindro menor e as curvas pontilhadas denotam o cilindro maior. As circunferências concêntricas denotam as curvas de nível da função objetivo, cujo menor valor ocorre na origem.



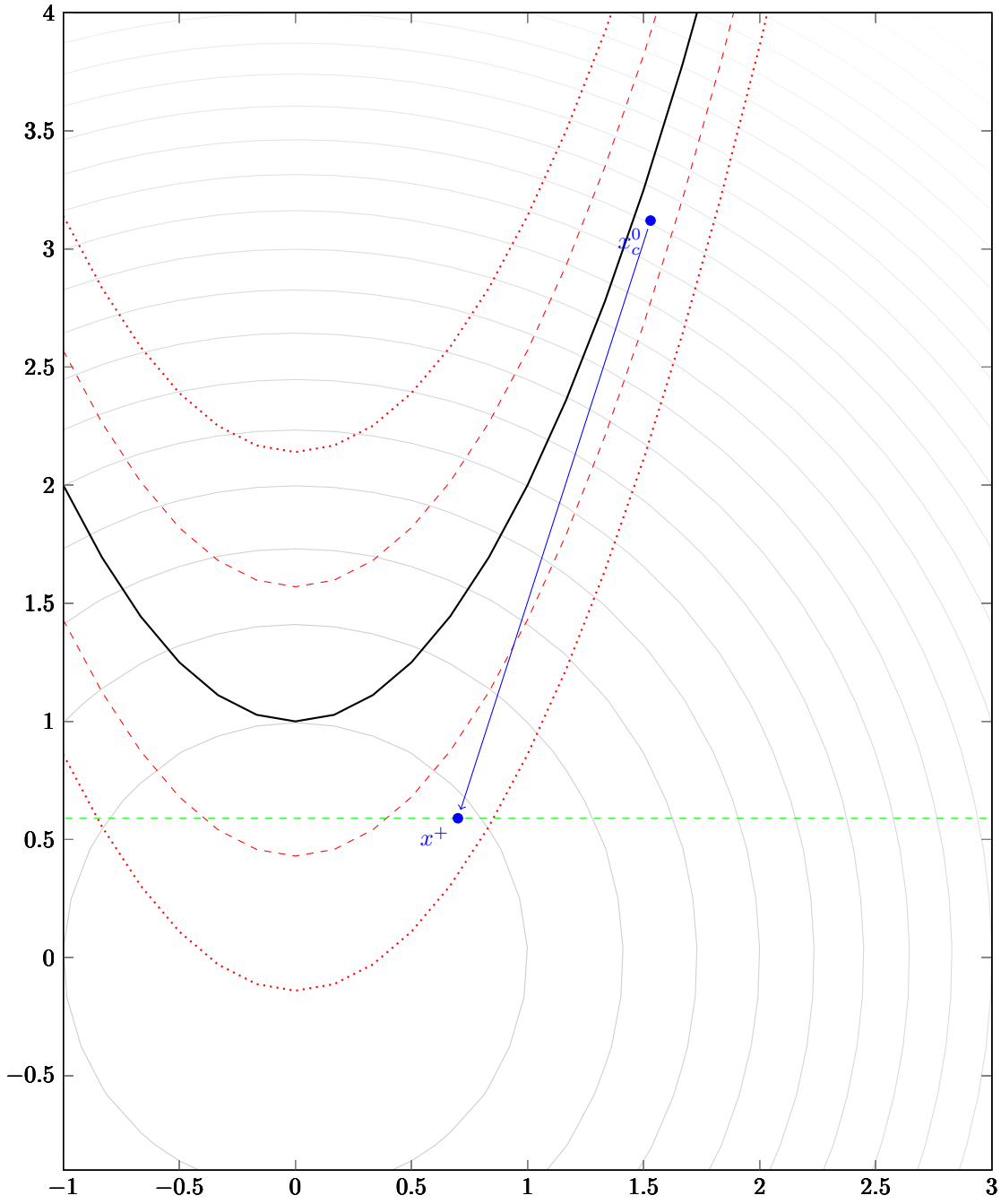
Iniciamos notando que o ponto não está dentro do cilindro menor. Então fazemos um passo normal. O ponto obtido, denotado x_c, é a tentativa de iterando. Como o ponto já está dentro do cilindro menor, o passo é interrompido.



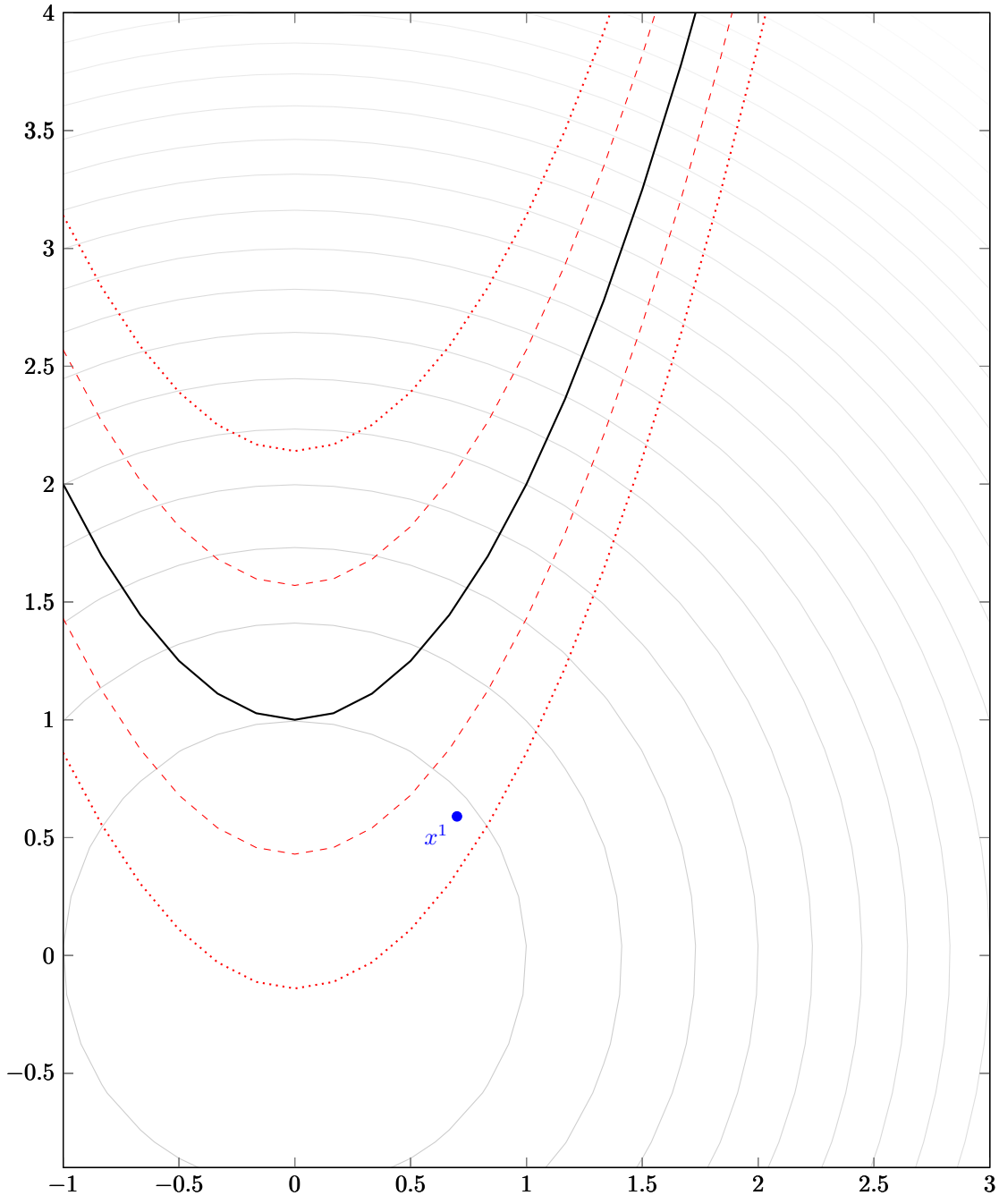
Agora, atualizamos o raio dos cilindros. Como o x_c obtido continuou dentro do cilindro menor após o raio deste ter sido atualizado, definimos x_c^0 como este iterando normal.



A partir deste ponto, tentamos realizar um passo tangente. O raio da região de confiança, não ilustrado na imagem, é grande o suficiente para permitir que o passo seja tomado completamente. O passo aqui é o minimizador da aproximação quadrática do Lagrangeano. x^+. Nesse caso, como esse ponto está fora do cilindro maior, rejeitamos o passo e reduzimos a região de confiança.



Calculamos um novo passo tangente, desta vez limitado pela região de confiança, indicada pelo segmento tracejado. Note que, na verdade, este segmento faz parte da caixa que é a região de confiança. O iterando x^+ obtido está dentro do cilindro maior e fornece decréscimo suficiente, sendo portanto aceito como o próximo iterando.



x^1 obtido pelo passo tangente.

Para verificar a aplicação do método no caso com inequações, vamos apresentar outro exemplo. Note, no entanto, que um problema de duas variáveis e uma inequação precisaria ser analisado em 3 dimensões. Decidimos, para facilitar a visualização, mostrar apenas as variáveis originais do problema, e utilizar a definição da variável de folga para mostrar a restrição deslocada. Pelo mesmo motivo, vamos mostrar apenas uma parte dos cilindros de confiança.

O problema que consideramos é

$$\begin{aligned} \min \quad & f(x) = \tfrac{1}{2}[x_1^2 + (x_2+1)^2] \\ \text{suj. a} \quad & x_2 \geq x_1^2, \\ & x_1 + x_2 = 1, \end{aligned}$$

cuja solução é o ponto $(\frac{-1+\sqrt{5}}{2}, \frac{3-\sqrt{5}}{2}) \approx (0.618, 0.382)$. Ao adicionarmos a variável de folga s, obtemos o problema

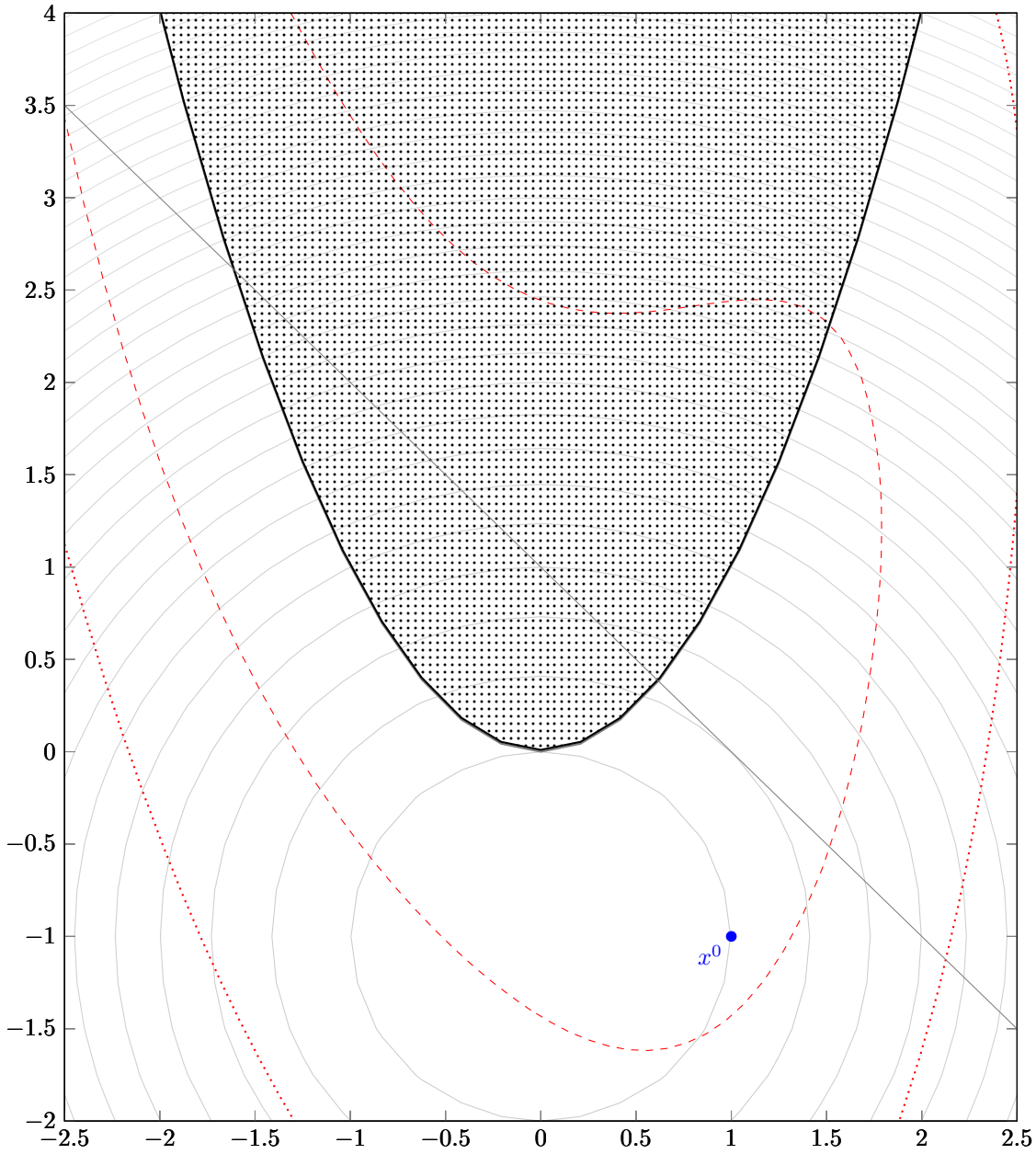
$$\begin{aligned} \min \quad & f(x) = \tfrac{1}{2}[x_1^2 + (x_2+1)^2] \\ \text{suj. a} \quad & x_2 - x_1^2 = s, \\ & x_1 + x_2 = 1, \\ & s \geq 0. \end{aligned}$$

Na solução, o valor de s é 0.

Os cilindros de confiança para esse problema serão os conjuntos da forma

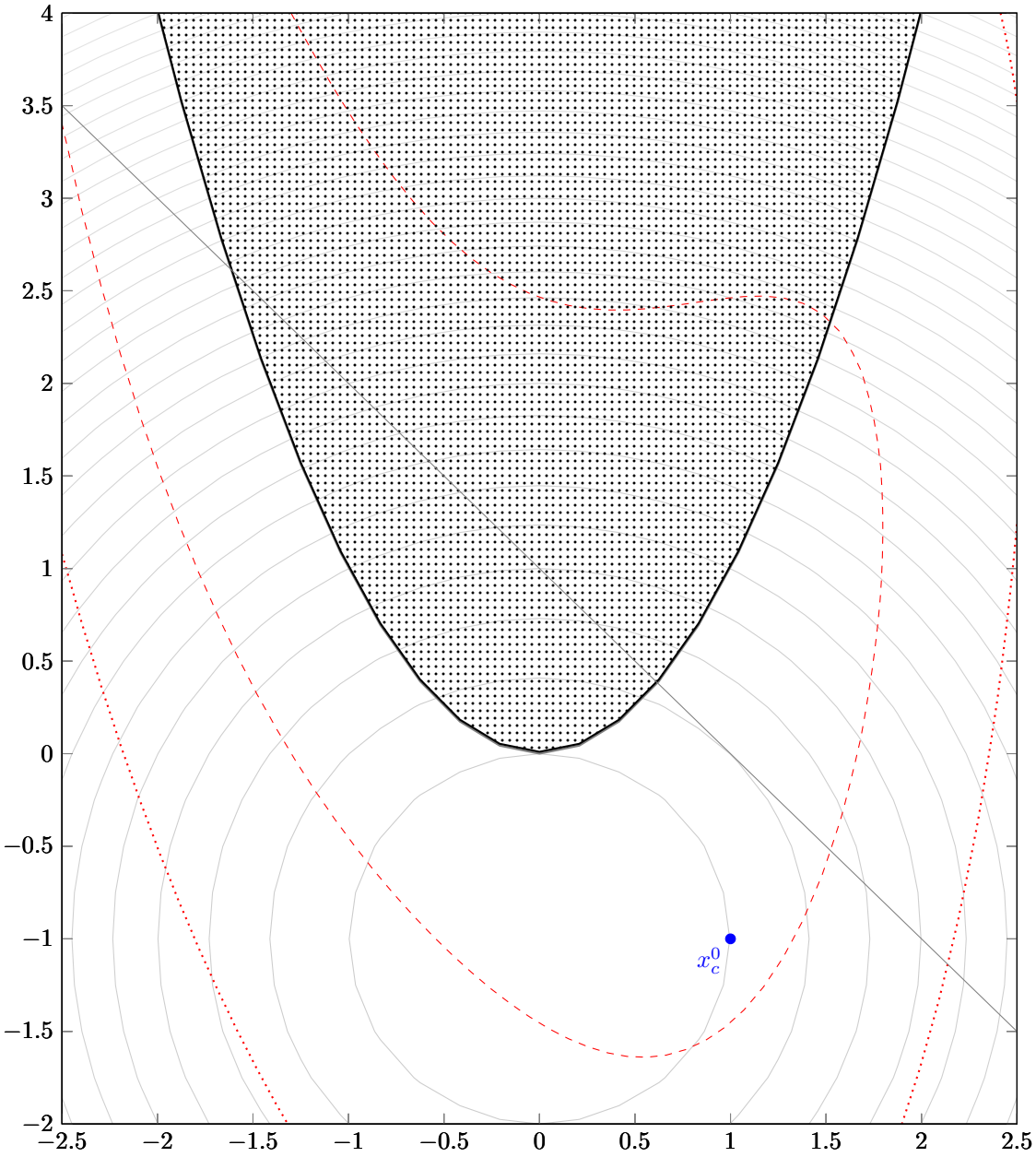
$$\mathcal{C}(\rho) = \{(x, s) \in \mathbb{R}^3 : (x_2 - x_1^2 - s)^2 + (x_1 + x_2 - 1)^2 \leq \rho^2\}.$$

Para mostrar alguma informação dessa região no plano original do problema, decidimos tomar o caso com s fixo. Dessa maneira, podemos ter alguma informação do cilindro para as variáveis x. Note que essa visualização do cilindro pode mudar de uma iteração para outra mesmo que o raio do cilindro permaneça o mesmo.



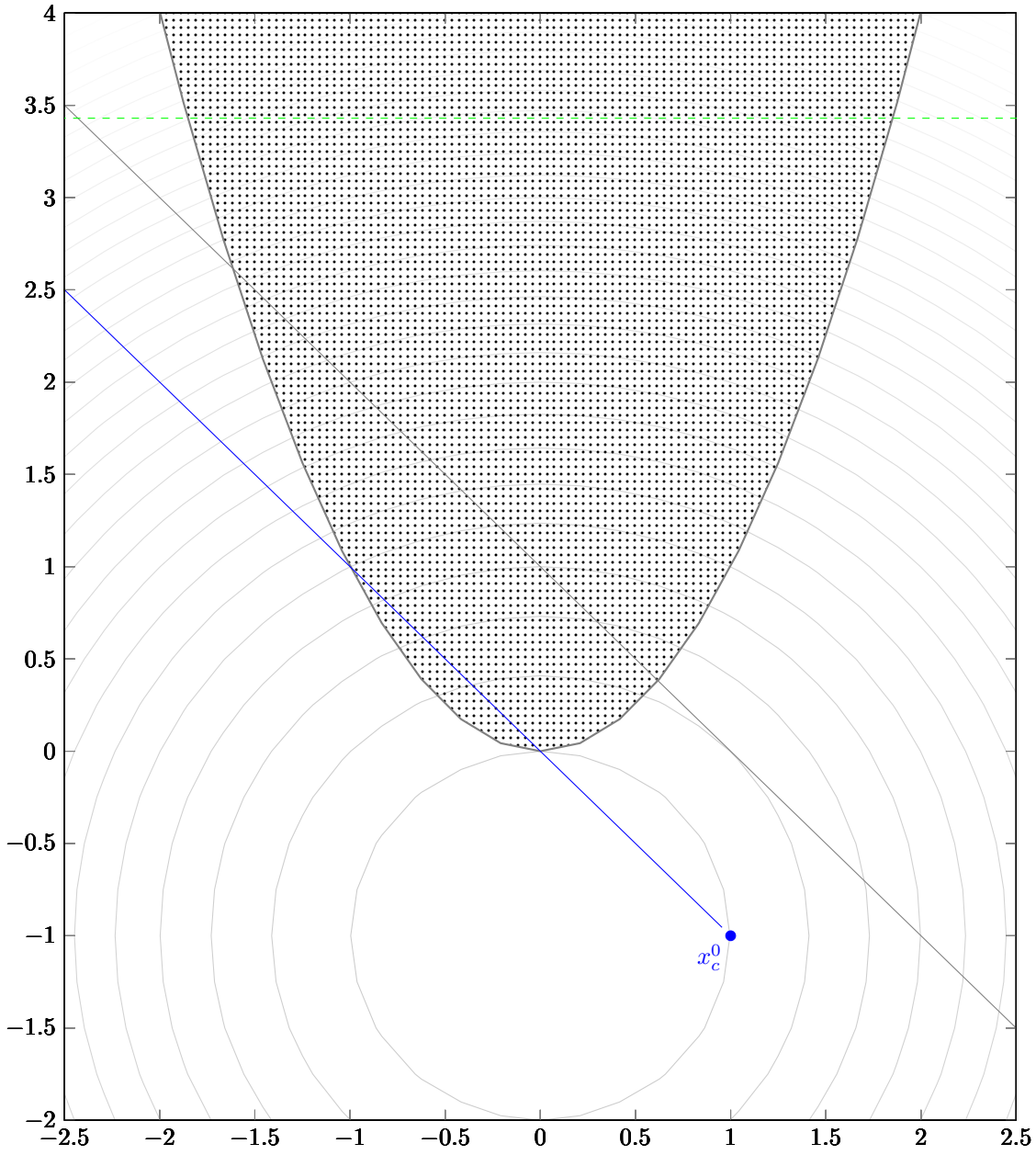
$s^0 = 0.01$

Começamos este exemplo pelo ponto $x^0 = (1, -1)$, e $s^0 = 0.01$, um valor pequeno e positivo. No gráfico, a reta corresponde à segunda equação; a região hachurada correspondente ao conjunto factível da desigualdade; e a parábola sólida corresponde à equação $x_2 - x_1^2 = s^0$. A curva tracejada denota o corte do cilindro pequeno em $s = s^0$, e a pontilhada o corte do cilindro grande pelo mesmo plano.



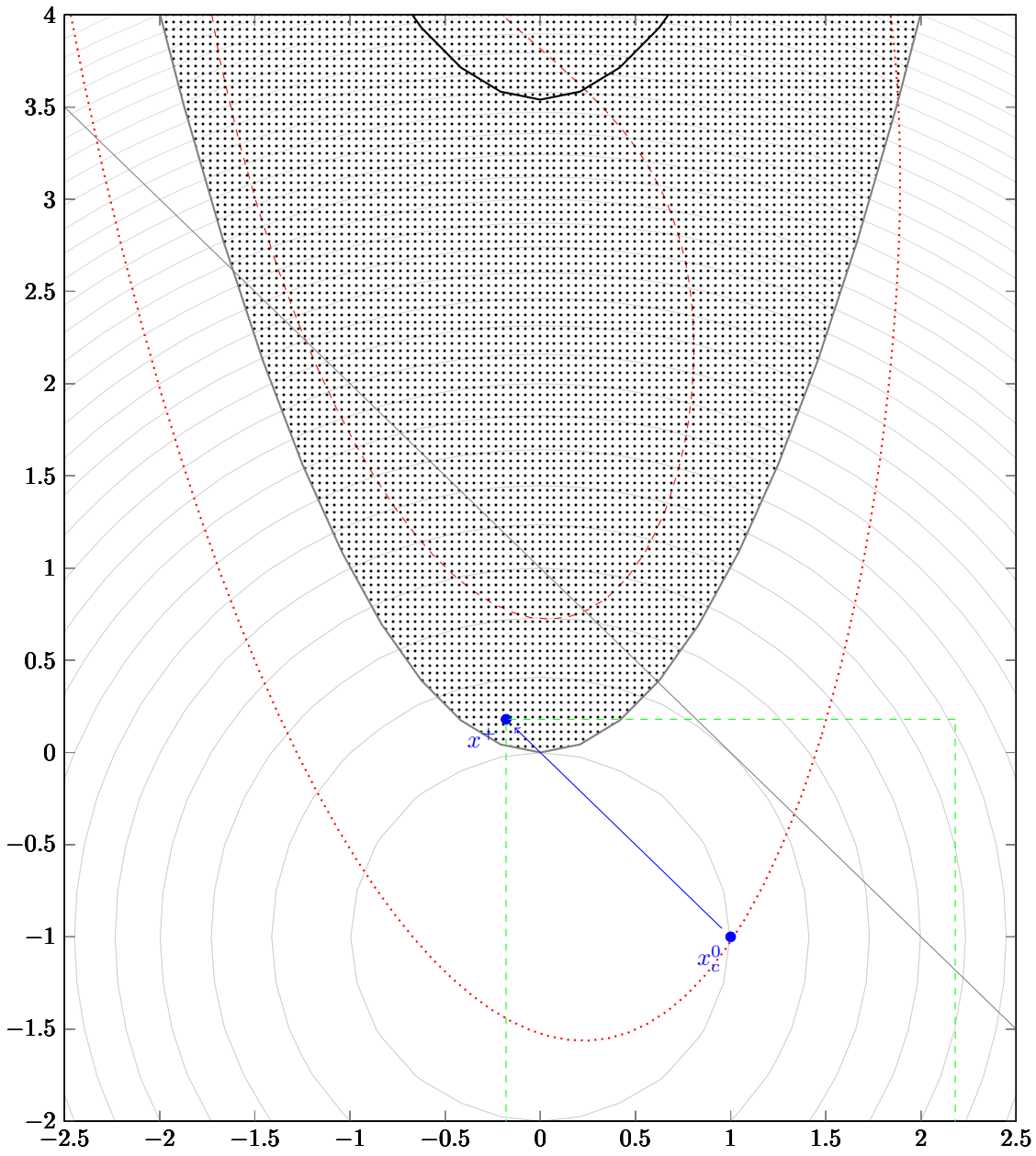
$s_c^0 = 0.01$

O ponto inicial já está dentro do cilindro pequeno, então o ponto é aceito sem necessidade de calcular nenhum passo normal.



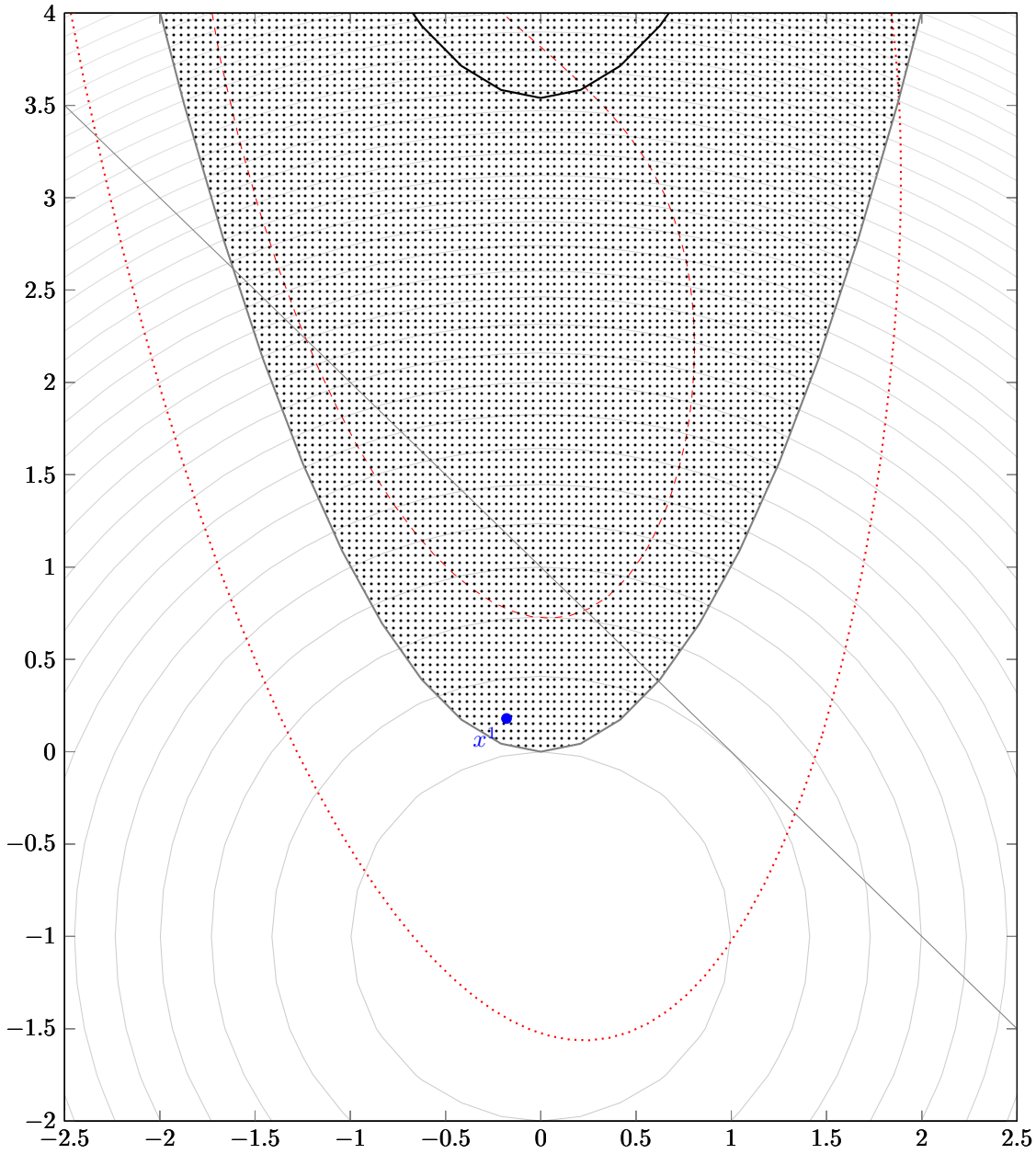
$s^+ = 14.15$

Tentamos agora fazer um passo tangente. O ponto encontrado fica fora do cilindro maior e não obtém decréscimo suficiente. Note que o cilindro não é visualizado, pois o corte é feito com o valor de s encontrado resulta no conjunto vazio. Note que a parábola sólida também não é visualizada por causa do valor de s.



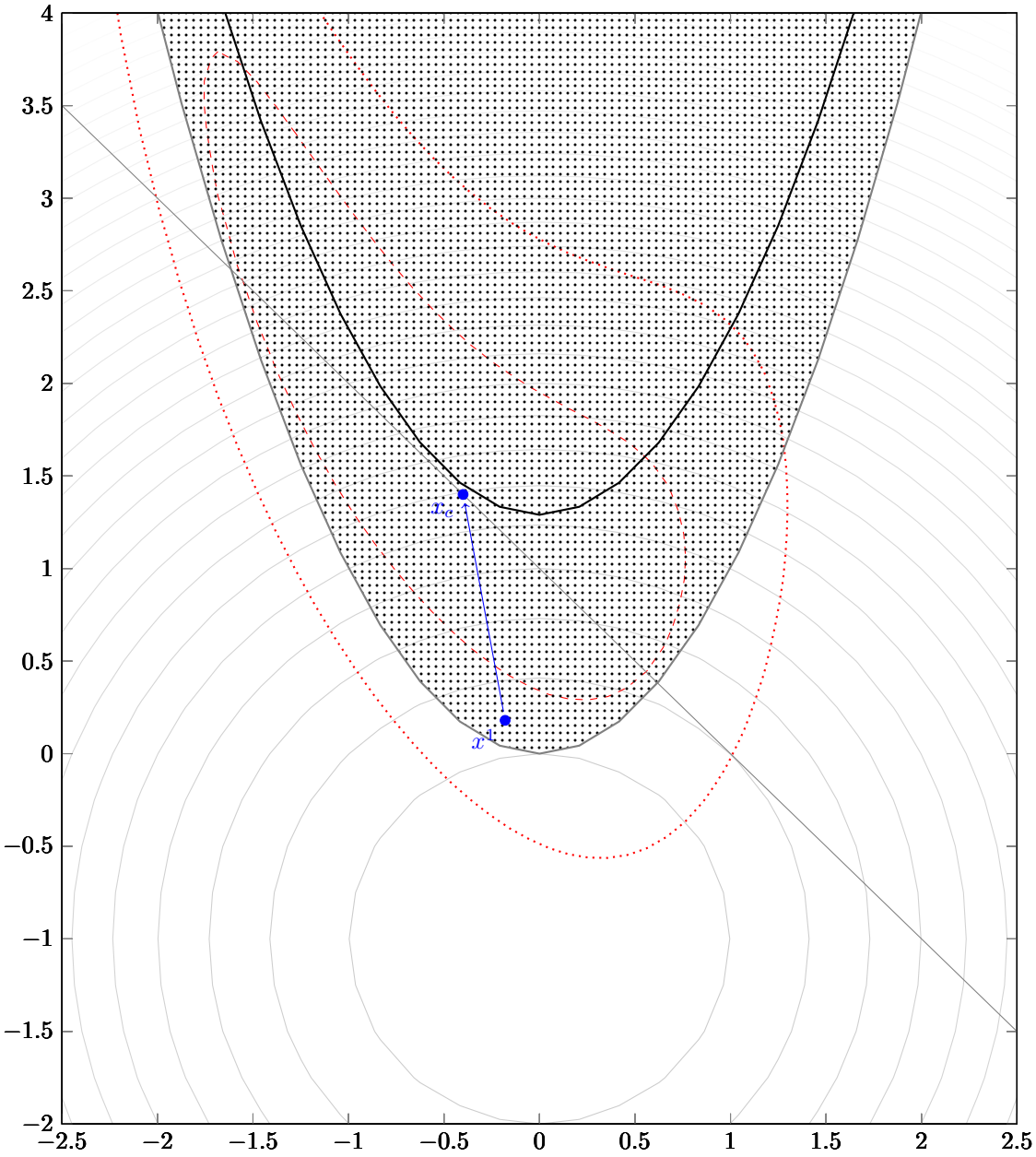
$s^+ = 3.54$

O segundo passo tangente obtém decréscimo suficiente e está dentro do cilindro maior, que agora pode ser visualizado. Note que a parábola sólida também já pode ser visualizada.



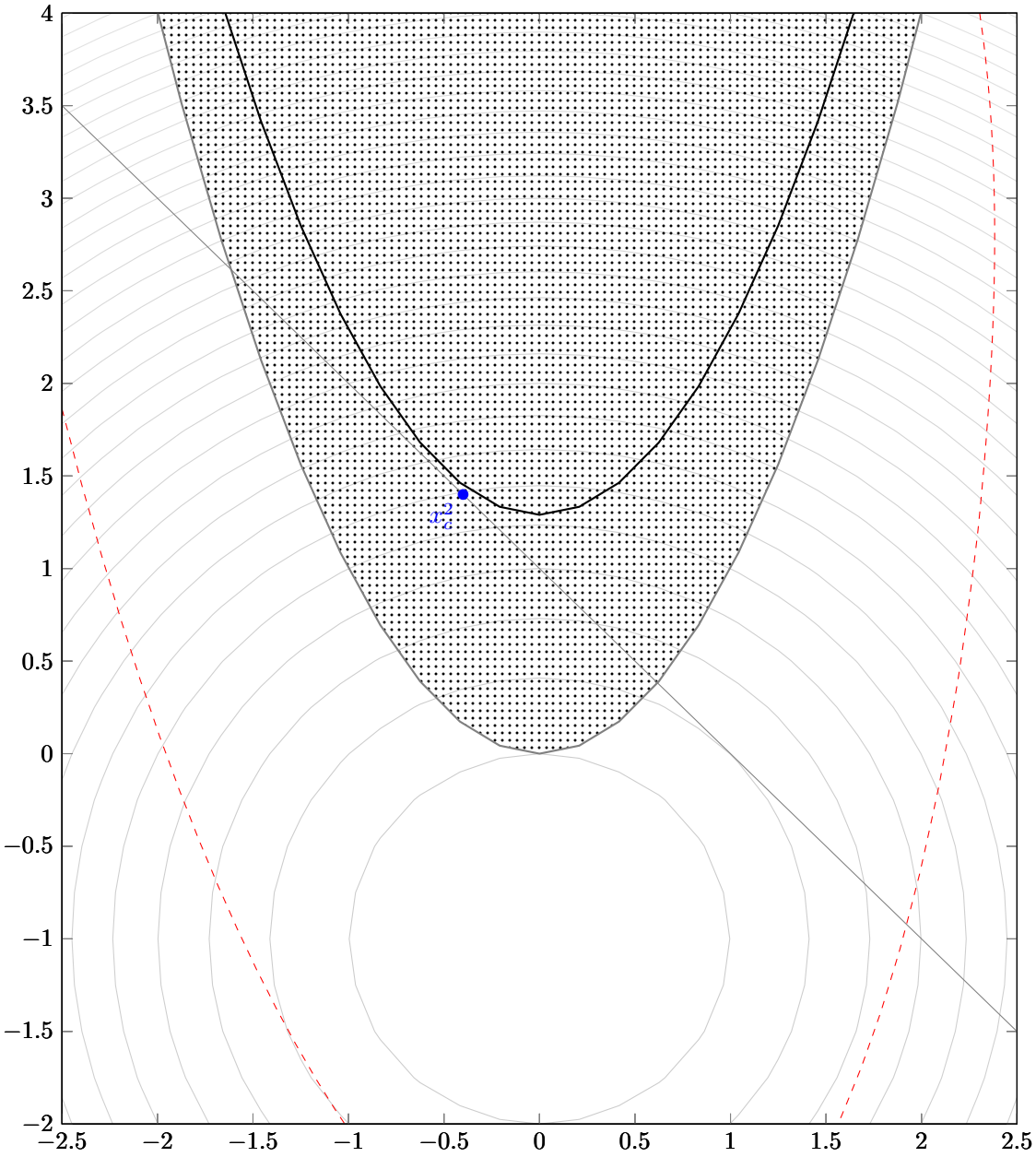
$s^1 = 3.54$

O ponto é aceito como iteração tangente.



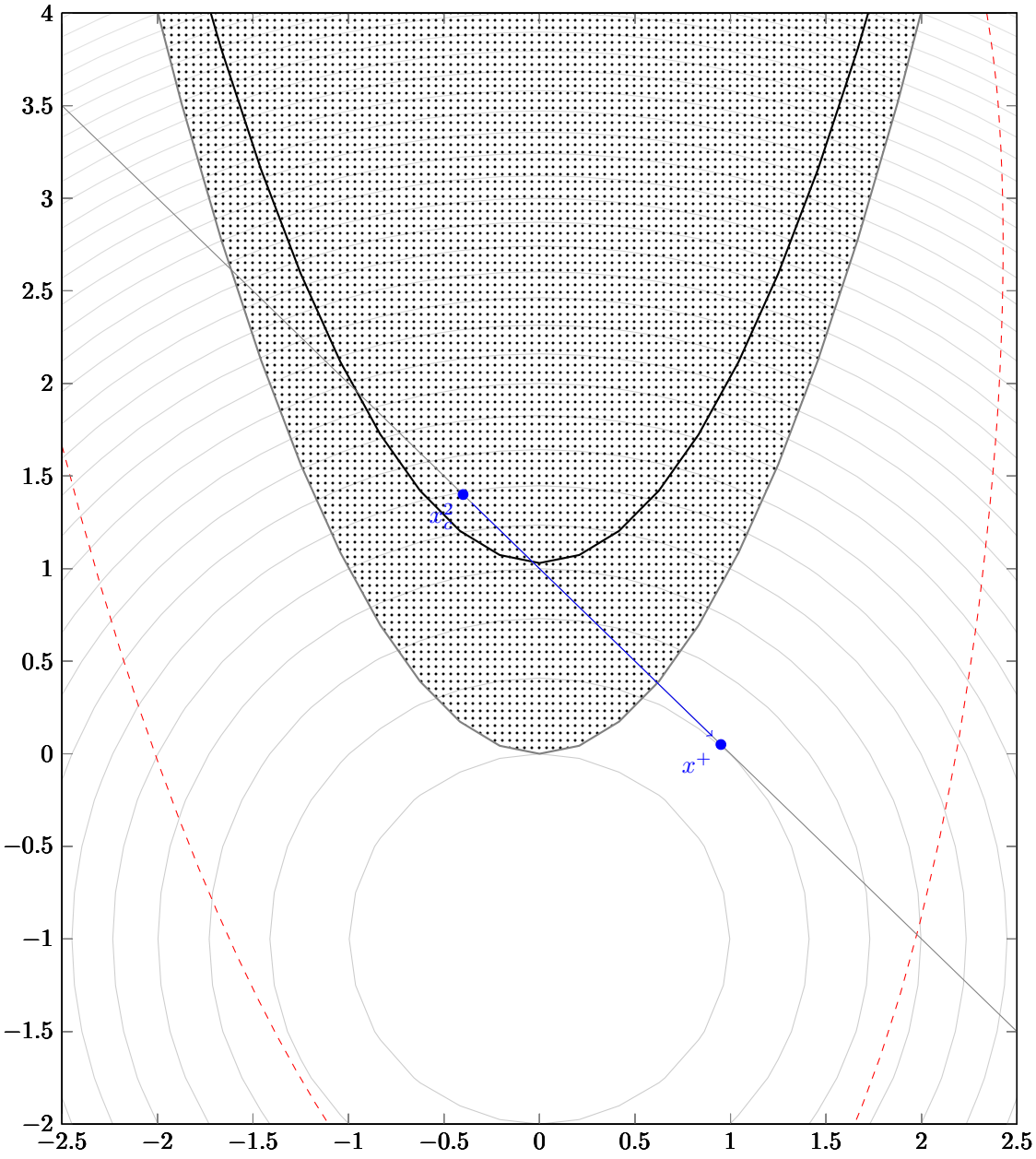
$s_c = 1.29$

Atualizamos o raio do cilindro e fazemos um passo normal, levando o ponto para dentro do cilindro menor.



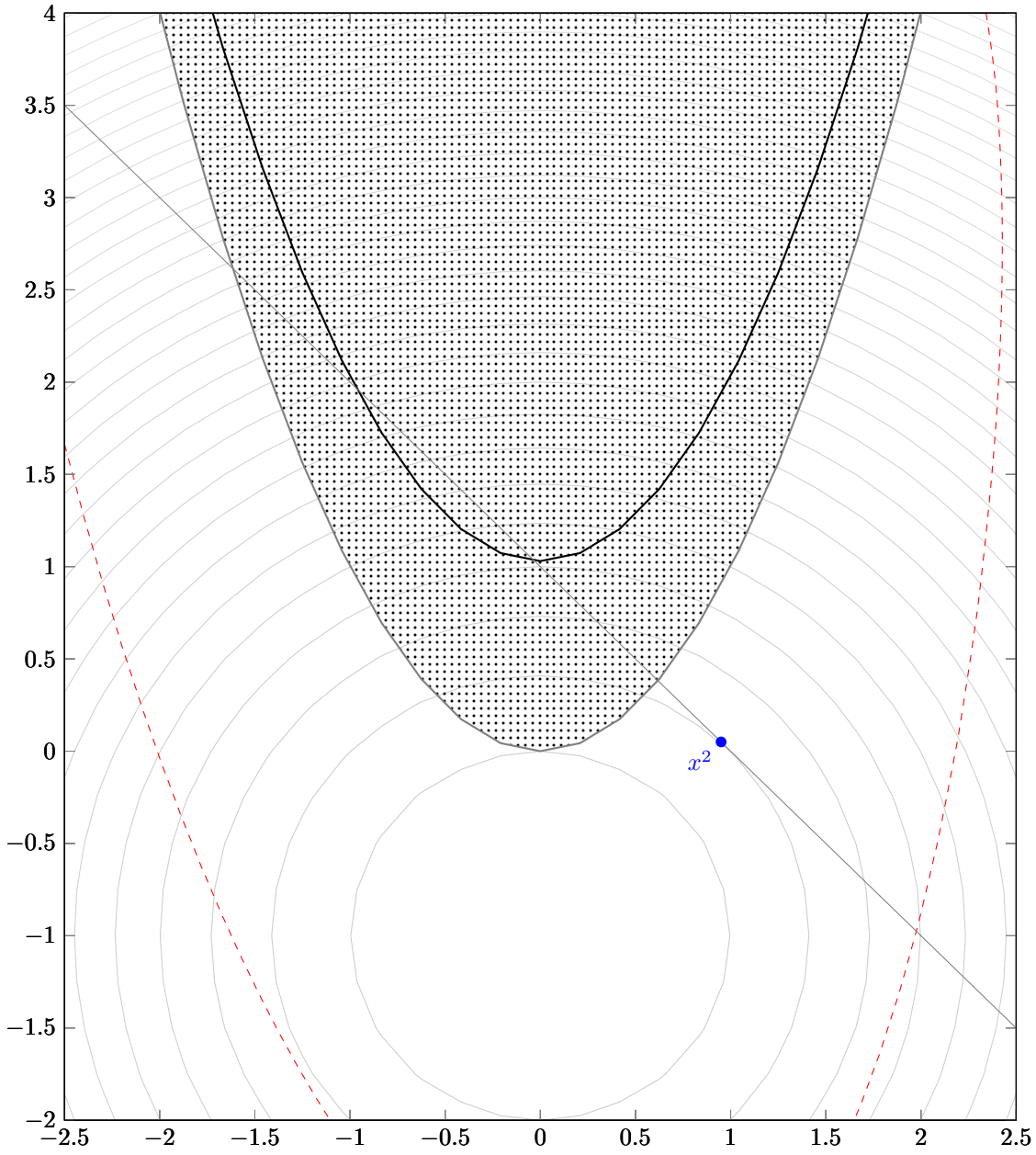
$$s_c^2 = 1.29$$

Atualizamos novamente o raio do cilindro e o ponto encontrado permance dentro do cilindro menor. Portanto encerramos o passo normal.



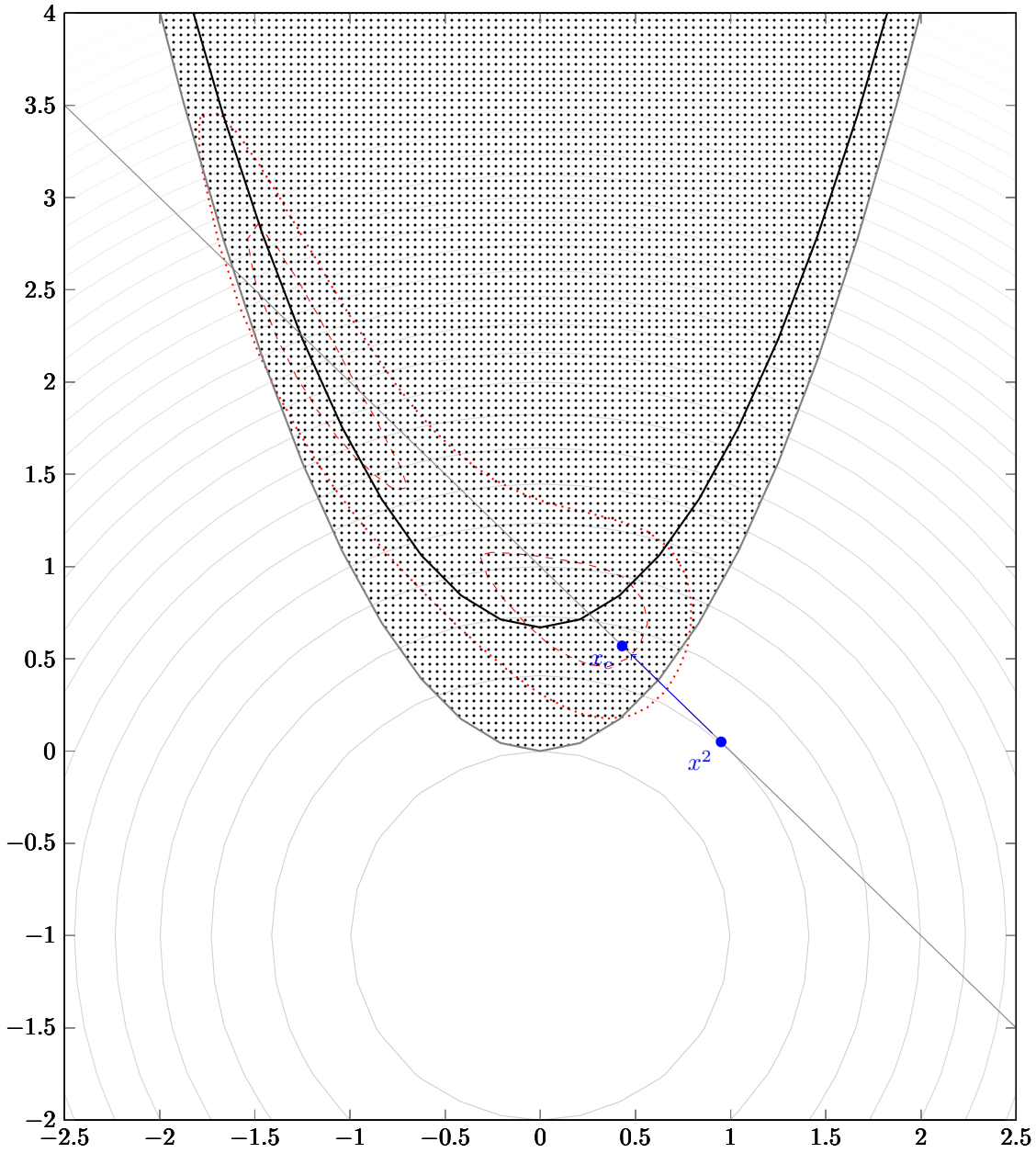
$s^+ = 1.03$

Fazemos um passo tangente, que fica dentro do cilindro e também tem decréscimo suficiente.



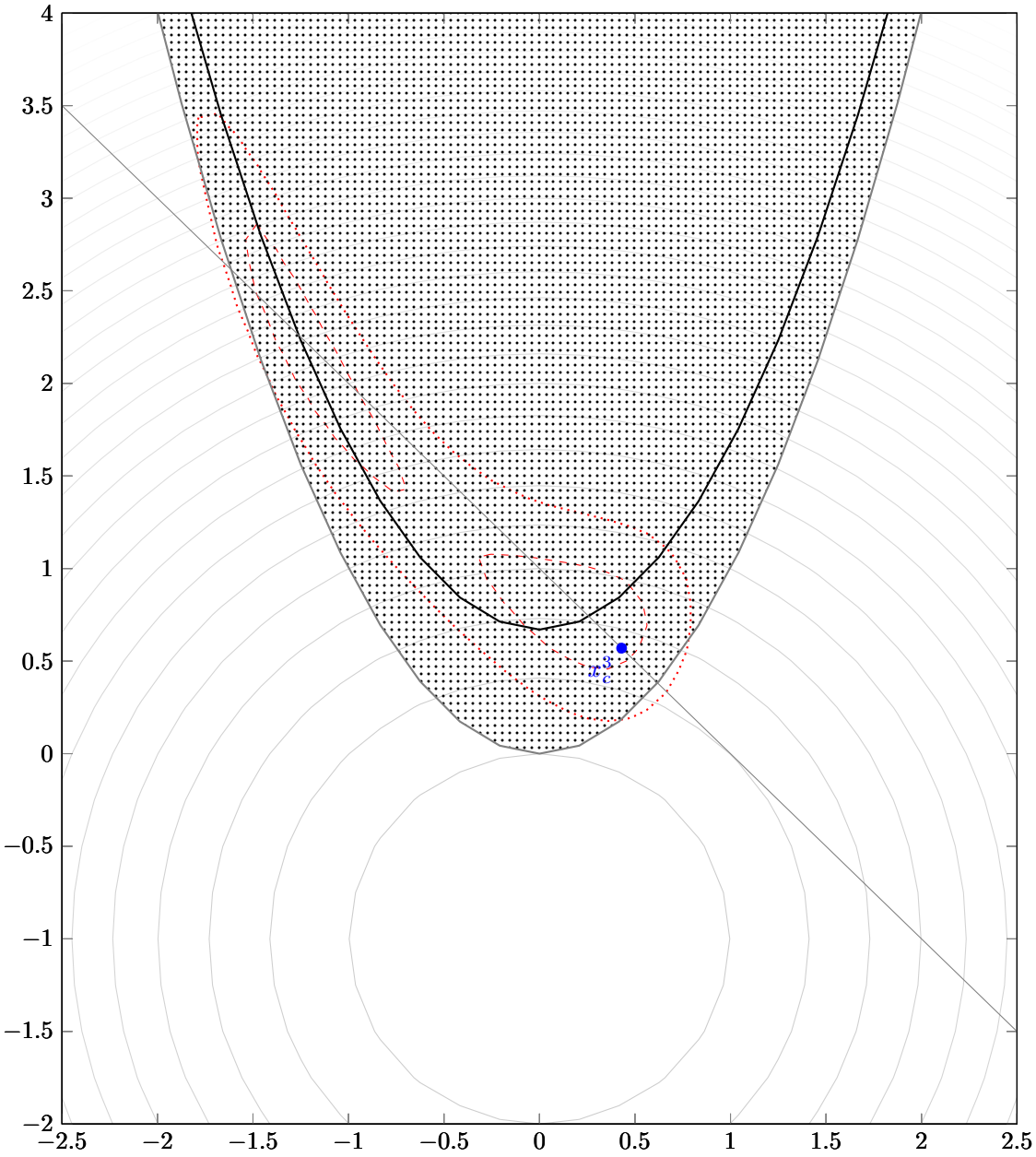
$s^2 = 1.03$

Aceitamos o ponto.



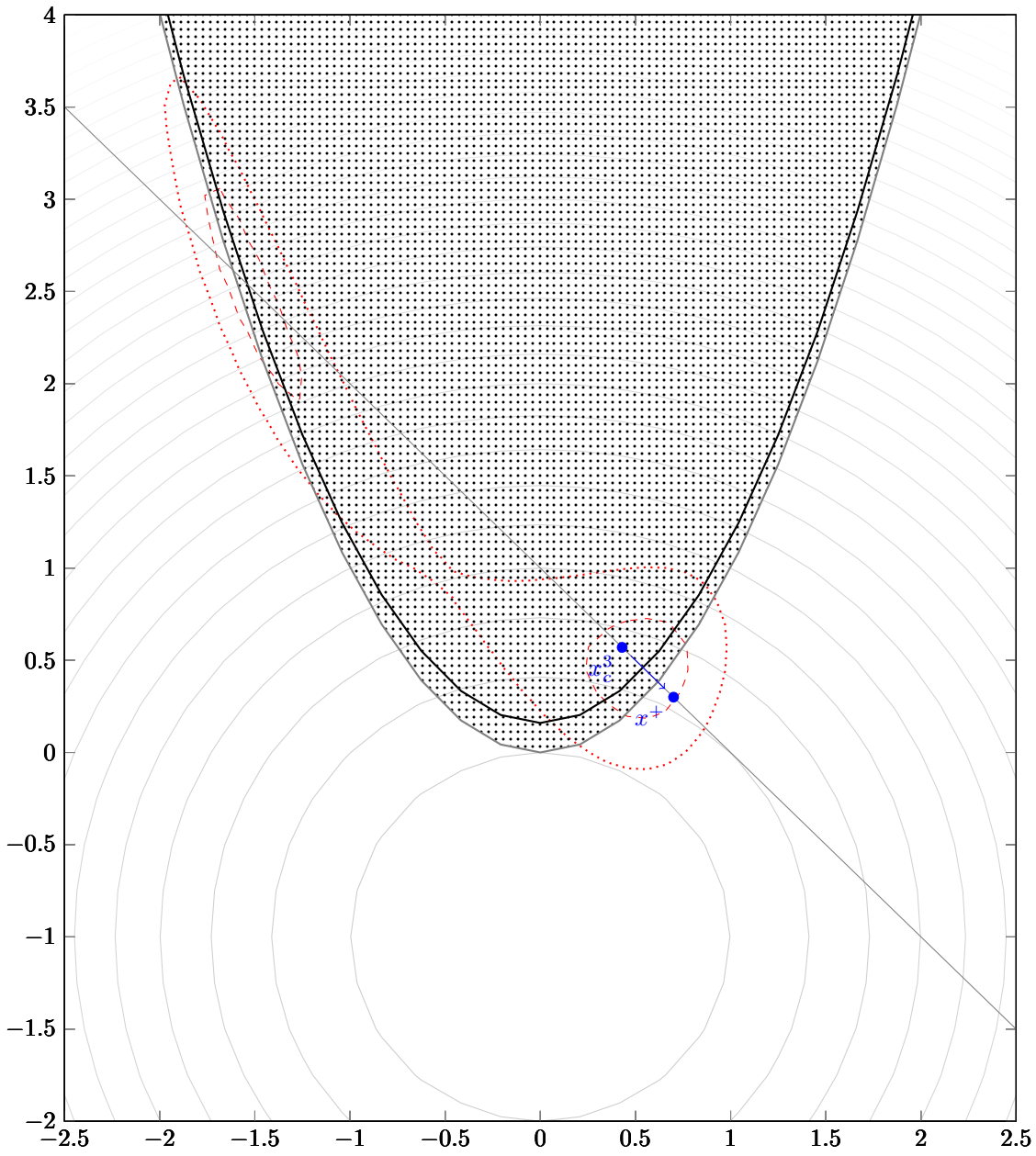
$s_c = 0.67$

Fazemos um passo normal.



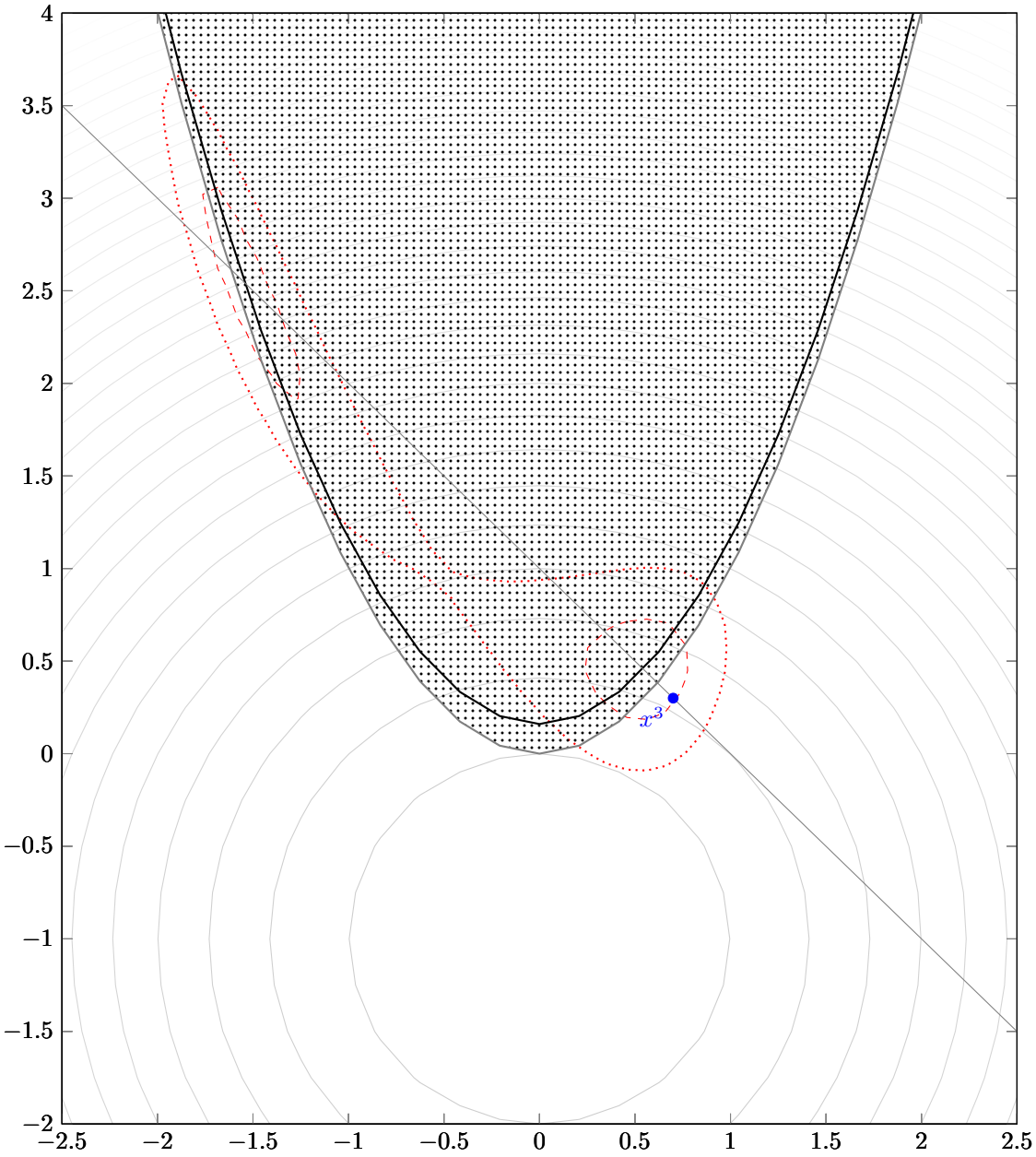
$$s_c^3 = 0.67$$

Aceitamos a iteração normal.



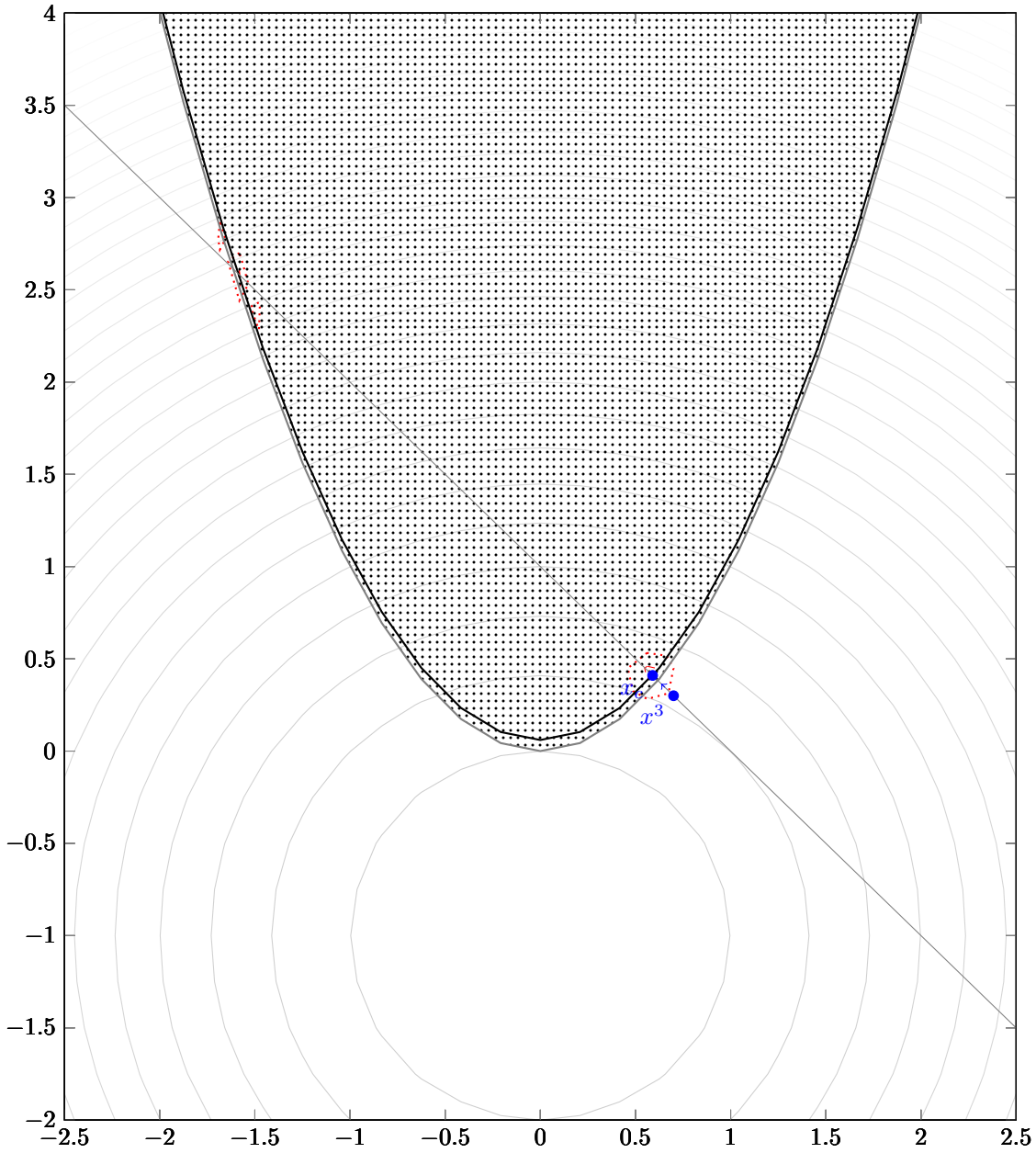
$s^+ = 0.16$

Fazemos um passo tangente.



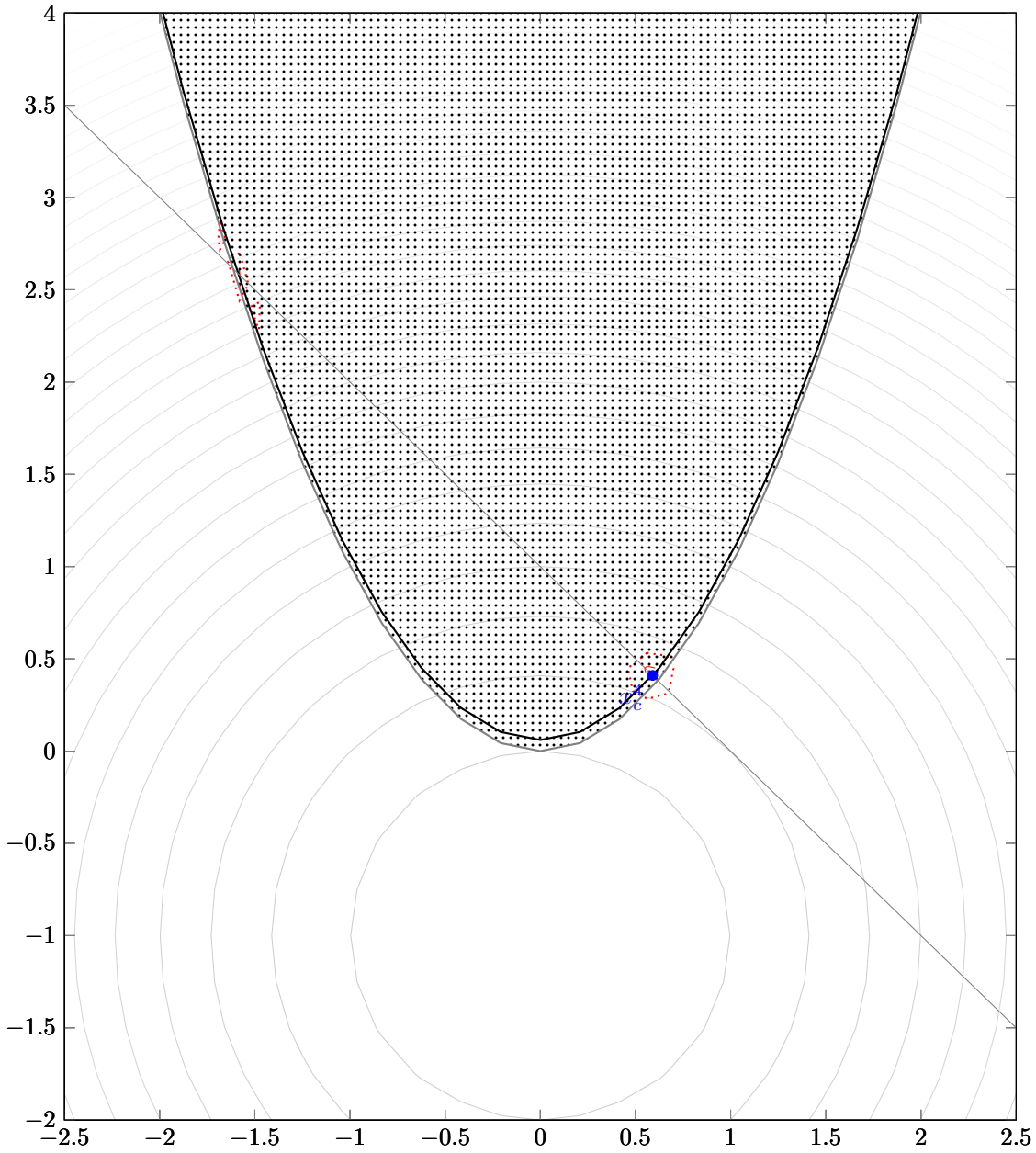
$s^3 = 0.16$

Aceitamos a iteração tangente.



$s_c = 0.06$

Fazemos um passo normal.



$s_c^4 = 0.06$

Aceitamos a iteração normal. O algoritmo oscila mais algumas iterações ao redor da solução até chegar à precisão necessária para declarar convergência.

Capítulo 4

Convergência

Apresentaremos agora os resultados de convergência do método. Inicialmente mostraremos que, com condições pouco restritivas, a sequência gerada pelo método converge para um ponto estacionário. A seguir, mostraremos que, numa vizinhança do ponto estacionário, se este for um minimizador sob condições satisfatórias, o método converge superlinearmente em dois passos. Não obstante, mostraremos ainda que o método converge para pontos estacionários do problema de infactibilidade, de modo que, se o problema for infactível, não ficará iterando infinitamente. Foi possível aproveitar todas as propriedades de convergência do método original [5], sendo necessário apenas alguns ajustes para o caso geral com desigualdades. Para a convergência global, precisamos apenas de algumas modificações para incluir o parâmetro de barreira. Para a convergência local, precisamos modificar um pouco os teoremas para tratar apenas das restrições ativas na solução.

4.1 Convergência Global

Apresentamos aqui os resultados de convergência global do método. Veremos que, com condições relativamente fracas, podemos obter resultados satisfatórios de convergência. A seguir estão as nossas hipóteses para a prova de convergência do Algoritmo 3.1.

Hipótese H.1. *As funções f, c_E e c_I são C^2.*

Hipótese H.2. *As sequências* $\{z_c^k\}$ *e* $\{z^k\}$*, as aproximações* B^k *e os multiplicadores* $\{\lambda^k\}$ *permanecem uniformemente limitados.*

Hipótese H.3. *A restauração não falha, no sentido de que o algoritmo normal sempre encontra um ponto dentro do cilindro, e* $\mathcal{Z} = \{z_c^k\}$ *permanece longe do conjunto singular de* h*, no sentido que* h *é regular no fecho de* $\mathcal{Z}$*. Além disso, se a sequência gerada* $\{z_c^k\}$ *é infinita, então*

$$\|z_c^{k+1} - z^k\| = \mathcal{O}(\|h(z^k)\|). \tag{4.1}$$

Hipótese H.4. $\|\delta_{soc}^k\| = \mathcal{O}(\|\delta_t^k\|^2)$

A Hipótese H.1 é natural e esperada, já que canônica, considerando que, no método, utilizamos essas derivadas, ou aproximações para as mesmas. A Hipótese H.2 é necessária pois podemos tomar alguma direção de descida ilimitada. As aproximações podem ser definidas de modo a permanecerem uniformemente limitadas. A Hipótese H.3 é importante pois a restauração pode falhar. Note que na Seção 4.3 comentamos sobre a convergência do método para pontos estacionários da infactibilidade. A Hipótese H.4 é tradicional para os passos de correção de segunda ordem.

Apresentamos agora algumas propriedades provenientes do algoritmo. Começamos por aquelas relacionadas aos cilindros. Pela maneira que calculamos os raios dos cilindros no passo 3 do Algoritmo 3.1, pelas condições dos iterandos nos passos 3 e 8 do Algoritmo 3.1, e pela maneira que atualizamos $\rho_{\max}$ nos passos 3-7 do Algoritmo 3.4, temos

$$\begin{aligned}
\rho^k &\leq \rho_{\max}^{k-1}\|g_p^k\| \leq 2\rho_{\max}^k\|g_p^k\|, && (4.2)\\
\rho_{\max}^k &\leq \rho_{\max}^{k-1} \leq 10^4\rho^k\frac{\|g(z_c^k,\mu^k)\|+1}{\|g_p^k\|}, && (4.3)\\
\|h(z_c^k)\| &\leq \|h(z^{k-1})\| \leq 2\rho^{k-1}. && (4.4)
\end{aligned}$$

Como indicado previamente em (3.17) e visto no algoritmo, nossas iterações seguem uma regra de fração-para-a-fronteira, de modo que são válidas as inequações

$$\begin{aligned}
s_c^k &\geq \varepsilon_\mu s^{k-1},\\
s^k &\geq \varepsilon_\mu s_c^k, && (4.5)\\
s^+ &\geq \varepsilon_\mu s_c^k.
\end{aligned}$$

Alem disso, pela definição de μ^k no passo 9 do Algoritmo 3.2, temos

$$\mu^k \leq \alpha_\rho \min\{\rho^k, (\rho^k)^2\}. \tag{4.6}$$

Deste ponto em diante, supomos que as sequências $\{z_c^k\}$ e $\{z^k\}$, geradas pelo algoritmo, satisfazem H.1-H.4. Denotando por δ_N^k o passo normal e por δ_T^k o passo tangente na iteração k, temos

$$\delta_N^k = z_c^k - z^{k-1} \qquad \text{e} \qquad \delta_T^k = z^k - z_c^k = \delta_t^k + \delta_{soc}^k.$$

As hipóteses H.1-H.4 nos permitem escolher uma constante $\delta_{\max} > 0$, tal que, para todo k,

$$\|\delta_t^k\| + \|\delta_{soc}^k\| + \|\delta_N^k\| \leq \delta_{\max}. \tag{4.7}$$

As hipóteses também nos permitem definir $\xi_0 > 0$, tal que, $\forall k$, se $\|z - z_c^k\| \leq \delta_{\max}$ e $\mu \leq \mu_0$, então

$$\begin{aligned}
\|A_j(z)\| &\leq \xi_0, \qquad j = 1, \ldots, m && (4.8)\\
\|\nabla^2 h_j(z)\| &\leq \xi_0, \qquad j = 1, \ldots, m && (4.9)\\
\|\nabla f(x)\| &\leq \xi_0, && (4.10)\\
\|\nabla^2 f(x)\| &\leq \xi_0, && (4.11)\\
\|g(z, \mu)\| &\leq \xi_0, && (4.12)\\
\|\Gamma(z, \mu)\| &\leq \xi_0, && (4.13)\\
\|B^k\| &\leq \xi_0, && (4.14)\\
\|\lambda^k\| &\leq \xi_0, && (4.15)\\
\|\delta_{soc}^k\| &\leq \xi_0 \|\delta_t^k\|^2 && (4.16)\\
\|\Lambda_c^k\| &\leq \xi_0, && (4.17)
\end{aligned}$$

$$\begin{aligned}
s_c^k &\leq \xi_0 s^{k-1},\\
s^k &\leq \xi_0 s_c^k.
\end{aligned} \tag{4.18}$$

Também supomos que (4.1) pode ser reescrito como

$$\|z_c^{k+1} - z^k\| \leq \xi_0 \|h(z^k)\|. \tag{4.19}$$

Definimos as matrizes $\Lambda_c^k = \Lambda(z_c^k)$, $\Lambda^k = \Lambda(z^k)$ e $\Lambda^+ = \Lambda(z^+)$, para facilitar a notação. Nosso resultado de convergência global é dado no Teorema 4.6. Esse resultado depende dos próximos cinco lemas. O Lema 4.1 a seguir dá um limitante para o aumento da infactibilidade do causada pelo passo tangente em relação ao iterando obtido no passo normal.

Lema 4.1. *A tentativa de iteração z^+ gerada na linha 9 do Algoritmo 3.3 satisfaz*

$$\|h(z^+) - h(z_c^k)\| \leq \overline{\xi}_0 \|\delta_t\|^2, \tag{4.20}$$

em que $\overline{\xi}_0$ é uma constante positiva.

Demonstração: Iremos omitir os índices k nessa demonstração. A iteração é definida por $z^+ = z_c + d^+ = z_c + \Lambda_c(\delta_t + \delta_{soc})$. Usando uma expansão de Taylor, o fato de que $A_j(z_c)\delta_t = 0$ e (4.9), podemos garantir que existe $z_\xi^j = \eta_j z^+ + (1-\eta_j) z_c$, tal que

$$\begin{aligned}
\left|h_j(z^+) - h_j(z_c)\right| &= \left|\nabla h_j(z_c)^T d^+ + \frac{1}{2}(d^+)^T \nabla^2 h_j(z_\xi^j) d^+\right| \\
&= \left|\nabla h_j(z_c)^T \Lambda_c^k(\delta_t + \delta_{soc}) + \frac{1}{2}(\delta_t + \delta_{soc})^T \Lambda_c^k \nabla^2 h_j(z_\xi^j) \Lambda_c^k (\delta_t + \delta_{soc})\right| \\
&\leq \left|A_j(z_c)^T(\delta_t + \delta_{soc})\right| + \frac{1}{2}\|\delta_t + \delta_{soc}\|^2 \|\Lambda_c^k\|^2 \|\nabla^2 h_j(z_\xi^j)\| \\
&\leq \left|A_j(z_c)^T \delta_{soc}\right| + \frac{1}{2}\|\delta_t + \delta_{soc}\|^2 \xi_0^3.
\end{aligned} \tag{4.21}$$

Agora, por (4.8) e (4.16), temos

$$\left|A_j(z_c)^T \delta_{soc}\right| \leq \xi_0^2 \|\delta_t\|^2,$$

e pela desigualdade $\|v+w\|^2 \leq 2(\|v\|^2 + \|w\|^2)$, (4.16) e (4.7),

$$\frac{1}{2}\|\delta_t + \delta_{soc}\|^2 \leq \|\delta_t\|^2 + \|\delta_{soc}\|^2 \leq \|\delta_t\|^2 + \delta_{\max}\xi_0 \|\delta_t\|^2.$$

Substituindo essas duas desigualdades em (4.21), obtemos

$$\begin{aligned}
\left|h_j(z^+) - h_j(z_c)\right| &\leq \xi_0^2\|\delta_t\|^2 + \xi_0^3(\|\delta_t\|^2 + \xi_0 \delta_{\max}\|\delta_t\|^2) \\
&\leq \xi_0^2(1 + \xi_0 + \xi_0^2 \delta_{\max})\|\delta_t\|^2.
\end{aligned}$$

Definindo $\overline{\xi}_0 = \sqrt{m}\xi_0^2(1 + \xi_0 + \xi_0^2\delta_{\max})$, temos o resultado desejado.

■

O lema a seguir mostra que, se as hipóteses H.1-H.4 são satisfeitas, o passo tangente não falha, e obtemos redução suficiente no Lagrangeano.

Lema 4.2. *Se x_c^k não é um ponto estacionário para o problema* (2.1)*, então z^+ é aceito com suficientes iterações. Além disso, podemos definir constantes ξ_1, ξ_2 e ξ_3 tais que, para todo k,*

$$\Delta L_T^+ \leq -\xi_1 \|g_p^k\| \min\{\xi_2 \|g_p^k\|, \xi_3 \sqrt{\rho^k}, 1 - \varepsilon_\mu\}. \tag{4.22}$$

Demonstração: Iremos omitir os índices k nessa demonstração. Seja $z^+ = z_c + d^+ = z_c + \Lambda_c(\delta_t + \delta_{soc})$ um candidato obtido na linha 9 da k-ésima iteração do Algoritmo 3.1.

Utilizando uma expansão de Taylor e (4.20), existe $z_\xi = \eta z^+ + (1-\eta) z_c$, para algum $\eta \in [0,1]$, tal que

$$\begin{aligned} \Delta L_T^+ &= L(z^+, \lambda, \mu) - L(z_c, \lambda, \mu) \\ &= \varphi(z^+, \mu) - \varphi(z_c, \mu) + \lambda^T [h(z^+) - h(z_c)] \\ &\leq \nabla \varphi(z_c, \mu)^T d^+ + \frac{1}{2} (d^+)^T \nabla^2 \varphi(z_\xi, \mu) d^+ + \xi_0 \bar{\xi}_0 \|\delta_t\|^2. \end{aligned} \tag{4.23}$$

Para o primeiro termo, usando (3.9), o fato de que $g(z_c, \mu)^T \delta_t = (g_p^k)^T \delta_t$, a definição (3.23), e as condições (4.16), (4.12) e (4.14), temos

$$\begin{aligned} \nabla \varphi(z_c, \mu)^T \delta^+ &= \nabla \varphi(z_c, \mu)^T \Lambda_c (\delta_t + \delta_{soc}) \\ &= g(z_c, \mu)^T (\delta_t + \delta_{soc}) \\ &= q(\delta_t) - \frac{1}{2} \delta_t^T B \delta_t + g(z_c, \mu)^T \delta_{soc} \\ &\leq q(\delta_t) + \left(\frac{1}{2} \xi_0 + \xi_0^2 \right) \|\delta_t\|^2. \end{aligned} \tag{4.24}$$

Para o segundo termo de (4.23), usando (3.10), (4.13), (4.5), a desigualdade $\|v+w\|^2 \leq 2(\|v\|^2 + \|w\|^2)$,

(4.7) e (4.16), temos

$$\begin{aligned}\frac{1}{2}(\delta^+)^T\nabla^2\varphi(z_\xi,\mu)\delta^+ &= \frac{1}{2}(\delta_t+\delta_{soc})^T\Lambda_c\Lambda(z_\xi)^{-1}\Gamma(z_\xi,\mu)\Lambda(z_\xi)^{-1}\Lambda_c(\delta_t+\delta_{soc})\\ &\leq \frac{1}{2}\|\Gamma(z_\xi,\mu)\|\|\Lambda(z_\xi)^{-1}\Lambda_c\|^2\|\delta_t+\delta_{soc}\|^2\\ &\leq \frac{1}{2}\frac{\xi_0}{\varepsilon_\mu^2}\|\delta_t+\delta_{soc}\|^2\\ &\leq \frac{\xi_0}{\varepsilon_\mu^2}(\|\delta_t\|^2+\|\delta_{soc}\|^2)\\ &\leq \frac{\xi_0}{\varepsilon_\mu^2}(1+\xi_0\delta_{\max})\|\delta_t\|^2. \end{aligned} \tag{4.25}$$

Assim, substituindo (4.24) e (4.25) em (4.23), temos

$$\begin{aligned}\Delta L_T^+ &\leq q(\delta_t)+\left(\frac{\xi_0}{2}+\xi_0^2\right)\|\delta_t\|^2+\frac{\xi_0}{\varepsilon_\mu^2}(1+\xi_0\delta_{\max})\|\delta_t\|^2+\xi_0\overline{\xi_0}\|\delta_t\|^2\\ &= q(\delta_t)+\overline{\xi}_1\|\delta_t\|^2, \end{aligned} \tag{4.26}$$

onde $\overline{\xi}_1 = \frac{\xi_0}{2}+\xi_0^2+\frac{\xi_0}{\varepsilon_\mu^2}(1+\xi_0\delta_{\max})+\xi_0\overline{\xi}_0$.

Pela definição de δ_{CP} na linha 5 do Algoritmo 3.3, temos

$$\|\delta_{CP}\| \geq \min\left\{\frac{\|g_p\|}{\|B\|}, \Delta, 1-\varepsilon_\mu\right\} \geq \min\left\{\frac{\|g_p\|}{\xi_0}, \Delta, 1-\varepsilon_\mu\right\}$$

e

$$q(\delta_{CP}) \leq \frac{1}{2}\delta_{CP}^T g_p \leq -\frac{1}{2}\|g_p\|\min\left\{\frac{\|g_p\|}{\xi_0}, \Delta, 1-\varepsilon_\mu\right\}. \tag{4.27}$$

Vamos mostrar que z^+ é aceito se $\Delta \leq \overline{\Delta}$, onde

$$\overline{\Delta} = \min\left\{\frac{\|g_p\|}{4\overline{\xi}_1}, 1-\varepsilon_\mu, \sqrt{\frac{\rho}{\overline{\xi}_0}}\right\}. \tag{4.28}$$

Note que $\xi_0 < 4\overline{\xi}_1$, logo

$$\overline{\Delta} \leq \frac{\|g_p\|}{\xi_0}. \tag{4.29}$$

Como $\Delta \leq \overline{\Delta}$, usando (4.29), podemos simplificar (4.27) para

$$q(\delta_{CP}) \leq -\frac{1}{2}\|g_p\|\Delta. \tag{4.30}$$

Combinando $\|\delta_t\| \leq \Delta$ e $q(\delta_t) \leq q(\delta_{CP})$ da linha 6 do Algoritmo 3.3, com (4.28) e (4.30) obtemos

$$\begin{aligned} \overline{\xi}_1\|\delta_t\|^2 &\leq \overline{\xi}_1\Delta^2 \leq \overline{\xi}_1\overline{\Delta}\Delta \\ &\leq \frac{1}{4}\|g_p\|\Delta \leq -\frac{1}{2}q(\delta_{CP}) \leq -\frac{1}{2}q(\delta_t). \end{aligned} \tag{4.31}$$

Agora, de (4.26) e (4.31), temos

$$\Delta L_T^+ \leq \frac{1}{2}q(\delta_t) < 0, \tag{4.32}$$

de modo que

$$r = \frac{\Delta L_T^+}{q(\delta_t)} \geq \frac{1}{2} \geq \eta_1. \tag{4.33}$$

Como $\Delta \leq \overline{\Delta}$ e $\|h(z_c)\| \leq \rho$, usando (4.20) e (4.28), temos

$$\|h(z^+)\| \leq \rho + \overline{\xi}_0\|\delta_t\|^2 \leq \rho + \overline{\xi}_0\overline{\Delta}^2 \leq 2\rho.$$

Portanto, ambas condições da linha 3 do Algoritmo 3.3 são satisfeitas e z^+ é aceito.

Para provar a segunda parte, vamos lembrar que cada vez que o passo é rejeitado, multiplicamos o raio Δ por α_R. Então podemos assumir que o raio aceito satisfaz $\Delta \geq \alpha_R\overline{\Delta}$, onde $0 < \alpha_R < 1$. Combinando isto com (4.32), a condição $q(\delta_t) \leq q(\delta_{CP})$, (4.27), (4.29) e (4.28), obtemos

$$\begin{aligned} \Delta L_T^+ &\leq \frac{1}{2}q(\delta_t) \leq \frac{1}{2}q(\delta_{CP}) \\ &\leq -\frac{1}{4}\|g_p\|\min\left\{\frac{\|g_p\|}{\xi_0}, \Delta, 1-\varepsilon_\mu\right\} \\ &\leq -\frac{1}{4}\|g_p\|\alpha_R\overline{\Delta} \\ &\leq -\xi_1\|g_p\|\min\{\xi_2\|g_p\|, \xi_3\sqrt{\rho}, 1-\varepsilon_\mu\}, \end{aligned}$$

com $\xi_1 = \alpha_R/4$, $\xi_2 = 1/(4\overline{\xi}_1)$ e $\xi_3 = 1/\sqrt{\overline{\xi}_0}$.

■

O próximo lema define um limitante superior para a variação normal do Lagrangeano. Note que essa variação pode ser positiva.

Lema 4.3. *Existe uma constante positiva ξ_4 tal que, para k suficientemente grande,*

$$\Delta L_N^{k+1} \leq \xi_4\rho_{\max}^k\|g_p^k\|.$$

Demonstração: Usando o Teorema do Valor Médio, temos

$$\begin{aligned}
\Delta L_N^{k+1} &= L(z_c^{k+1},\lambda^{k+1},\mu^{k+1}) - L(z^k,\lambda^k,\mu^k)\\
&= \varphi(z_c^{k+1},\mu^{k+1}) - \varphi(z^k,\mu^k) + h(z_c^{k+1})^T\lambda^{k+1} - h(z^k)^T\lambda^k\\
&= \varphi(z_c^{k+1},\mu^k) - \varphi(z^k,\mu^k) + h(z_c^{k+1})^T\lambda^{k+1} - h(z^k)^T\lambda^k + (\mu^{k+1}-\mu^k)\beta(z_c^{k+1})\\
&= \nabla\varphi(z_\xi,\mu^k)^T(z_c^{k+1}-z^k) + h(z_c^{k+1})^T\lambda^{k+1} - h(z^k)^T\lambda^k + (\mu^{k+1}-\mu^k)\beta(z_c^{k+1})\\
&= \nabla f(x_\xi)^T(x_c^{k+1}-x^k) + \mu^k\nabla\beta(z_\xi)^T(z_c^{k+1}-z^k) + h(z_c^{k+1})^T\lambda^{k+1} - h(z^k)^T\lambda^k\\
&\quad +(\mu^{k+1}-\mu^k)\beta(z_c^{k+1}),
\end{aligned}$$

com $z_\xi = \eta z^k + (1-\eta)z_c^{k+1}$, para algum $\eta \in [0,1]$. Pela hipótese H.2, existe $M > 0$ tal que $s_i^k, s_{c_i}^k \le M$. Usando isso, (4.10), (4.5), (4.15), (4.19), (4.4), (4.6) e (4.2) temos

$$\begin{aligned}
\Delta L_N^{k+1} &\le \xi_0\|x_c^{k+1}-x^k\| + \mu^k\sum_{i=1}^{m_I}\frac{s_i^k - s_{c_i}^{k+1}}{\eta s_i^k + (1-\eta)s_{c_i}^{k+1}} + \xi_0\|h(z_c^{k+1})\| + \xi_0\|h(z^k)\| + (\mu^k-\mu^{k+1})m_I M\\
&\le \xi_0^2\|h(z^k)\| + \xi_0\|h(z^k)\| + \xi_0\|h(z^k)\| + \mu^k m_I\Big(\frac{1-\varepsilon_\mu}{\eta+(1-\eta)\varepsilon_\mu} + M\Big)\\
&\le (\xi_0^2+2\xi_0)2\rho^k + \rho^k m_I\Big(\frac{1-\varepsilon_\mu}{\eta+(1-\eta)\varepsilon_\mu} + M\Big)\\
&\le \xi_4\rho_{\max}^k\|g_p^k\|,
\end{aligned}$$

onde $\xi_4 = 4(\xi_0^2+2\xi_0) + 2m_I\dfrac{1-\varepsilon_\mu}{\eta+(1-\eta)\varepsilon_\mu} + 2M$.

■

O Lema 4.4 mostra que entre iterações sucessivas em que $\rho_{\max}$ não muda, o Lagrangeano decresce proporcionalmente à variação do Lagrangeano nos passos tangentes.

Lema 4.4. *Se $\rho_{\max}^{k+1} = \rho_{\max}^{k+2} = \cdots = \rho_{\max}^{k+j}$, para $j \ge 1$, então*

$$L_c^{k+j} - L_c^k = \sum_{i=k+1}^{k+j}\Delta L_c^i \le \frac{1}{4}\sum_{i=k}^{k+j-1}\Delta L_T^i + r^k, \tag{4.34}$$

onde $r^k = \frac{1}{2}[L_{ref}^k - L_c^k]$.

Demonstração: Suponha que L_{ref} não muda entre as iterações $k+1$ e $k+j_1-1$, onde $0 < j_1 \le j+1$. Neste caso, por (3.24) e pelo critério da linha 8 do Algoritmo 3.4, temos

$$L_c^{k+j_1-1} - L_c^k = \sum_{i=k+1}^{k+j_1-1}(\Delta L_N^i + \Delta L_T^{i-1}) \le \frac{1}{2}\sum_{i=k}^{k+j_1-2}\Delta L_T^i. \tag{4.35}$$

Por outro lado, se L_{ref} muda na iteração $k + j_1$, então a condição da linha 8 é satisfeita. Neste caso, como $\rho_{\max}$ não muda nesta iteração, de modo que a condição da linha 3 não é satisfeita, e como $\Delta L_T^k \leq 0$ para todo k, temos

$$\begin{aligned} L_c^{k+j_1} - L_c^k &\leq \Delta L_N^{k+j_1} + L(z^{k+j_1-1}, \lambda^{k+j_1-1}, \mu^{k+j_1-1}) - L_{ref}^k + [L_{ref}^k - L_c^k] \\ &\leq \frac{1}{2}[L(z^{k+j_1-1}, \lambda^{k+j_1-1}, \mu^{k+j_1-1}) - L_{ref}^k] + [L_{ref}^k - L_c^k] \\ &\leq \frac{1}{2}[\Delta L_T^{k+j_1-1} + L_c^{k+j_1-1} - L_c^k] + \frac{1}{2}[L_{ref}^k - L_c^k] \\ &\leq \frac{1}{4}\sum_{i=k}^{k+j_1-1} \Delta L_T^i + r^k. \end{aligned} \tag{4.36}$$

Se $j_1 \geq j$, então (4.35) e (4.36) implicam (4.34).

Por outro lado, se L_{ref} é atualizado nas iterações $k + j_1, \ldots, k + j_s$, onde $0 < j_1 < j_2 < \cdots < j_s \leq j$, então $r^{k+j_1} = r^{k+j_2} = \cdots = r^{k+j_s} = 0$. Portanto, aplicando o mesmo processo descrito acima várias vezes, e definindo $j_0 = 0$, obtemos

$$L_c^{k+j} - L_c^k = \sum_{i=1}^{s}[L_c^{k+j_i} - L_c^{k+j_i-1}] + L_c^{k+j} - L_c^{k+j_s} \leq \frac{1}{4}\sum_{i=k}^{k+j-1} \Delta L_T^i + r^k.$$

■

O próximo lema estabelece a existência de *espaço normal suficiente* nos cilindros de confiança para garantir que o Lagrangeano possa ter decréscimo suficiente. A base desse lema é que a inequação (4.22) garante que, assintoticamente, $\left|\Delta L_H^k\right|$ é maior que uma fração de $\sqrt{\rho^k}$, enquanto que, na demonstração do Lema 4.4, vemos que $\|\Delta L_V^k\| = \mathcal{O}(\rho^k)$. Isso significa que, no limite, a restauração não irá destruir o descréscimo do Lagrangeano, o que evita que sejam feitas alterações excessivas de $\rho_{\max}$.

Lema 4.5. *Se CDI gera uma sequência infinita $\{z^k\}$, então*

(i) *Existem constantes positivas ξ_5 e ξ_6 tais que, se*

$$\rho_{\max}^k < \min\{\xi_5 \|g_p(z_c^k, \mu^k)\|, \xi_6\}, \tag{4.37}$$

então $\rho_{\max}$ não muda na iteração $k + 1$.

(ii) *Além disso, se* $\liminf\|g_p^k\|>0$*, então existe* $k_0>0$ *tal que, para todo* $k\geq k_0$

$$\rho_{\max}^k = \rho_{\max}^{k_0}. \tag{4.38}$$

(iii) *Se o passo tangente e o vetor de multiplicadores satisfazem*

$$\|z^k - z_c^k\| = \mathcal{O}(\|g_p(z_c^k,\mu^k)\|), \tag{4.39}$$

$$\|\lambda^k - \lambda_{LS}(z_c^k,\mu^k)\| = \mathcal{O}(\|g_p(z_c^k,\mu^k)\|), \tag{4.40}$$

$$(\lambda_I^{k+1})^T(s_c^{k+1}-s^k) = \mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k), \tag{4.41}$$

então (4.38) é satisfeito, independentemente do valor de $\liminf\|g_p(z_c^k,\mu^k)\|$*. Em outras palavras,* $\rho_{\max}^k$ *permanece suficientemente afastado de zero.*

Demonstração: (i) Para provar que $\rho_{\max}$ não muda na iteração $k+1$, só precisamos mostrar que $\Delta L_N^{k+1} < -\Delta L_T^k/2$, o que, segundo o Lema 4.2, é obtido quando

$$\Delta L_N^{k+1} \leq \frac{\xi_1}{2}\|g_p(z_c^k,\mu^k)\|\min\{\xi_2\|g_p(z_c^k,\mu^k)\|, \xi_3\sqrt{\rho^k}, 1-\varepsilon_\mu\}. \tag{4.42}$$

Para isso, tomamos

$$\xi_5 \leq \min\left\{\frac{\xi_1\xi_2}{2\xi_4}, \frac{10^{-4}\xi_1^2\xi_3^2}{4\xi_4^2(\xi_0+1)}\right\} \qquad \text{e} \qquad \xi_6 \leq \frac{\xi_1(1-\varepsilon_\mu)}{2\xi_4}, \tag{4.43}$$

onde ξ_4 é a constante definida no Lema 4.3. Assim, a partir do Lema 4.3, de (4.37) e (4.43), obtemos

$$\Delta L_N^{k+1} \leq \xi_4\rho_{\max}^k\|g_p^k\| \leq \xi_4\xi_5\|g_p^k\|^2 \leq \frac{1}{2}\xi_1\xi_2\|g_p^k\|^2. \tag{4.44}$$

Além disso, usando (4.3) e (4.12), temos

$$\sqrt{\rho_{\max}^k} \leq 10^2\sqrt{\rho^k(\xi_0+1)}\|g_p^k\|^{-1/2}. \tag{4.45}$$

Extraindo a raiz quadrada dos dois lados de (4.37) e combinando o resultado com (4.45), obtemos

$$\rho_{\max}^k \leq \sqrt{\xi_5}10^2\sqrt{\xi_0+1}\sqrt{\rho^k}. \tag{4.46}$$

Usando o Lema 4.3, (4.46) e (4.43), temos

$$\Delta L_N^{k+1} \leq \xi_4\rho_{\max}^k\|g_p^k\| \leq \xi_4\sqrt{\xi_5}10^2\sqrt{\xi_0+1}\sqrt{\rho^k}\|g_p^k\| \leq \frac{1}{2}\xi_1\xi_3\|g_p^k\|\sqrt{\rho^k}. \tag{4.47}$$

Finalmente, usando o Lema 4.3, (4.37) e (4.43), temos

$$\Delta L_N^{k+1} \leq \xi_4 \rho_{\max}^k \|g_p^k\| \leq \xi_4 \xi_6 \|g_p^k\| \leq \frac{1}{2}\xi_1(1-\varepsilon_\mu)\|g_p^k\|. \tag{4.48}$$

O resultado segue de (4.44), (4.47) e (4.48).

(ii) Sejam $b = \liminf(\|g_p^k\|)$ e $\overline{k}_0$ um índice tal que $\|g_p^k\| > b/2$, $\forall k \geq \overline{k}_0$. Tome $k \geq \overline{k}_0$. Nesse caso, existem duas situações possíveis: ou existe k tal que $\rho_{\max}^k < \min\{\xi_5 b/2, \xi_6\}$, ou $\rho_{\max}^k \geq \min\{\xi_5 b/2, \xi_6\}, \forall k$. Na primeira situação, pelo item (i), $\rho_{\max}^{k+1} = \rho_{\max}^k$ e, a partir deste k, $\rho_{\max}$ não muda mais. Desta forma, basta definirmos este k como k_0. Na segunda situação, $\rho_{\max}$ permance limitado inferiormente. Portanto, como $\rho_{\max}^k = \rho_{\max}^0 2^{-j}$, para algum $j \in \mathbb{N}$, a partir de uma certa iteração k_0, $\rho_{\max}$ não irá descrescer mais.

(iii) Note que, pelas definições de λ_{LS} e g_p, e pelas hipóteses H.1-H.3, $\lambda_{LS}(z,\mu)$ e $g_p(z,\mu)$ estão bem definidos e são de classe C^1 em uma vizinhança compacta de $\overline{\mathcal{Z}}$, o fecho de $\mathcal{Z} = \{z_c^k\}$. Portanto, λ_{LS} e g_p são Lipschitz contínuas no sentido que

$$\|\lambda_{LS}(z_c^{k+1},\mu) - \lambda_{LS}(z_c^k,\mu)\| = \mathcal{O}(\|z_c^{k+1} - z_c^k\|), \tag{4.49a}$$

$$\|\lambda_{LS}(z_c^k,\mu_1) - \lambda_{LS}(z_c^k,\mu_2)\| = \mathcal{O}(|\mu_1 - \mu_2|), \tag{4.49b}$$

e

$$\|g_p(z_c^{k+1},\mu) - g_p(z_c^k,\mu)\| = \mathcal{O}(\|z_c^{k+1} - z_c^k\|), \tag{4.50a}$$

$$\|g_p(z_c^k,\mu_1) - g_p(z_c^k,\mu_2)\| = \mathcal{O}(|\mu_1 - \mu_2|), \tag{4.50b}$$

De (4.1), (4.39), (4.4) e (4.2), temos

$$\begin{aligned} \|z_c^{k+1} - z_c^k\| &\leq \|z_c^{k+1} - z^k\| + \|z^k - z_c^k\| \\ &= \mathcal{O}(\|h(z^k)\|) + \mathcal{O}(\|g_p(z_c^k,\mu^k)\|) \\ &= \mathcal{O}(\rho^k) + \mathcal{O}(\|g_p(z_c^k,\mu^k)\|) \\ &= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|), \end{aligned} \tag{4.51}$$

e de (4.50a) e (4.51), obtemos

$$\begin{aligned} \|g_p(z_c^{k+1},\mu^k)\| &\leq \|g_p(z_c^k,\mu^k)\| + \mathcal{O}(\|z_c^{k+1} - z_c^k\|) \\ &= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|). \end{aligned} \tag{4.52}$$

Agora, usando (4.50b), (4.6), (4.52) e (4.2), obtemos

$$\begin{aligned}\|g_p(z_c^{k+1},\mu^{k+1})\| &\leq \|g_p(z_c^{k+1},\mu^k)\|+\mathcal{O}(\left|\mu^k-\mu^{k+1}\right|)\\ &= \mathcal{O}(\|g_p(z_c^{k+1},\mu^k)\|)+\mathcal{O}(\rho^k)\\ &= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|). \end{aligned} \tag{4.53}$$

Pelas definições (3.7), (3.5), (3.3) e (2.2), a variação do Lagrangeano no passo normal da iteração k pode ser escrita como

$$\begin{aligned}\Delta L_N^{k+1} &= L(z_c^{k+1},\lambda^{k+1},\mu^{k+1})-L(z^k,\lambda^k,\mu^k)\\ &= f(x_c^{k+1})+\mu^{k+1}\beta(z_c^{k+1})+(\lambda^{k+1})^T h(z_c^{k+1})-\\ &\quad f(x^k)-\mu^k\beta(z^k)-(\lambda^k)^T h(z^k)\\ &= \mathcal{L}(x_c^{k+1},\lambda^{k+1})-\mathcal{L}(x^k,\lambda^{k+1})+\mu^{k+1}\beta(z_c^{k+1})-\mu^k\beta(z^k)+\\ &\quad [\lambda^{k+1}-\lambda^k]^T h(z^k)-(\lambda_I^{k+1})^T(s_c^{k+1}-s^k). \end{aligned} \tag{4.54}$$

Usando uma expansão de Taylor, (3.22), a Hipótese H.3, (3.14), (4.40), (4.53), (4.11), (4.9), (4.15), (4.1), (4.4) e (4.2), garantimos que existe x_ξ tal que

$$\begin{aligned}\mathcal{L}(x_c^{k+1},\lambda^{k+1})-\mathcal{L}(x^k,\lambda^{k+1}) &= \nabla_x\mathcal{L}(x_c^{k+1},\lambda^{k+1})^T(x_c^{k+1}-x^k)+\\ &\quad \frac{1}{2}(x_c^{k+1}-x^k)^T\nabla^2_{xx}\mathcal{L}(x_\xi,\lambda^{k+1})(x_c^{k+1}-x^k)\\ &\leq \|\begin{bmatrix} I & 0\end{bmatrix} g_p^{k+1}\|\|x_c^{k+1}-x^k\|+\frac{1}{2}(\xi_0+m_I\xi_0^2)\|x_c^{k+1}-x^k\|^2\\ &= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|^2) \end{aligned} \tag{4.55}$$

Usando uma expansão de Taylor, (3.4), (4.18), (4.5), (4.6) e (4.2), temos

$$\begin{aligned}\mu^{k+1}\beta(z_c^{k+1})-\mu^k\beta(z^k) &= \mu^{k+1}[\beta(z^k)+\nabla\beta(z_\xi)^T(z_c^{k+1}-z^k)]-\mu^k\beta(z^k)\\ &= (\mu^k-\mu^{k+1})\sum_{i=1}^{m_I}\log(s_i^k)-\\ &\quad \mu^{k+1}e^T[\eta S^k+(1-\eta)S_c^{k+1}]^{-1}(s_c^{k+1}-s^k)\\ &= \mathcal{O}(\mu^k)+\mu^{k+1}\sum_{i=1}^{m_I}\frac{s_i^k-s_{c_i}^{k+1}}{\eta s_i^k+(1-\eta)s_{c_i}^{k+1}}\\ &\leq \mathcal{O}(\mu^k)+\mu^{k+1}m_I\frac{1-\varepsilon_\mu}{\eta+(1-\eta)\varepsilon_\mu}\\ &= \mathcal{O}(\mu^k)=\mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k). \end{aligned} \tag{4.56}$$

Usamos (4.40), (4.53), (4.49b), (4.6), (4.2), (4.49a), (4.51) e (4.4), obtendo

$$\begin{aligned}
[\lambda^{k+1}-\lambda^k]^T h(z^k) &\leq \|\lambda^{k+1}-\lambda_{LS}(z_c^{k+1},\mu^{k+1})\|\|h(z^k)\|+ \\
&\quad \|\lambda_{LS}(z_c^{k+1},\mu^{k+1})-\lambda_{LS}(z_c^{k+1},\mu^k)\|\|h(z^k)\|+ \\
&\quad \|\lambda_{LS}(z_c^{k+1},\mu^k)-\lambda_{LS}(z_c^k,\mu^k)\|\|h(z^k)\|+ \\
&\quad \|\lambda_{LS}(z_c^k,\mu^k)-\lambda^k\|\|h(z^k)\| \\
&= \mathcal{O}(\|g_p(z_c^{k+1},\mu^{k+1})\|\rho^k)+\mathcal{O}(\mu^k\rho^k)+\mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k) \\
&= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k). \qquad (4.57)
\end{aligned}$$

Assim, substituindo (4.55), (4.56) e (4.57) em (4.54), obtemos

$$\begin{aligned}
\Delta L_N^{k+1} &= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k)+\mathcal{O}(\rho^{k^2}) \\
&= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k)+\mathcal{O}(\rho_{\max}^k\|g_p(z_c^k,\mu^k)\|\rho^k) \\
&= \mathcal{O}(\|g_p(z_c^k,\mu^k)\|\rho^k)=\mathcal{O}(\|g_p(z_c^k,\mu^k)\|^2\rho_{\max}^k).
\end{aligned}$$

Portanto, existe $\xi_7>0$ tal que

$$\Delta L_N^{k+1}\leq \xi_7\rho_{\max}^k\|g_p(z_c^k,\mu^k)\|^2. \qquad (4.58)$$

Vamos, agora, mostrar que, também no caso (iii), (4.42) é satisfeito. Para tanto, definimos $\overline{\xi}_7=\xi_0+\mu_{\max}m_I+(n+m_I)\xi_0^2$ e

$$\overline{\rho}_{\max}=\min\left\{\frac{\xi_1\xi_2}{2\xi_7},\frac{10^{-4}}{4\overline{\xi}_7(\xi_0+1)}\left(\frac{\xi_1\xi_3}{\xi_7}\right)^2,\frac{\xi_1(1-\varepsilon_\mu)}{2\xi_7\overline{\xi}_7}\right\}. \qquad (4.59)$$

Assim, tomando $k\geq k_0$ tal que $\rho_{\max}^{k_0}\leq\overline{\rho}_{\max}$, e usando (4.58) e (4.59), temos

$$\begin{aligned}
\Delta L_N^{k+1} &< \xi_7\left(\frac{\xi_1\xi_2}{2\xi_7}\right)\|g_p(z_c^k,\mu^k)\|^2 \\
&= \frac{\xi_1\xi_2}{2}\|g_p(z_c^k,\mu^k)\|^2.
\end{aligned}$$

Além disso, note que, por (4.12), (4.8) e (4.15), temos

$$\begin{aligned}
\|g_p(z_c^k,\mu^k)\| &= \|g(z_c^k)+A(z_c^k)^T\lambda_{LS}(z_c^k,\mu^k)\| \\
&\leq \|g(z_c^k)\|+\sum_{i=1}^{n+m_I}\|\lambda_{LS_i}(z_c^k,\mu^k)A_i(z_c^k)\| \\
&\leq \xi_0+\mu m_I+(n+m_I)\xi_0^2 \\
&\leq \xi_0+\mu_{\max}m_I+(n+m_I)\xi_0^2=\overline{\xi}_7.
\end{aligned}$$

Assim, usando (4.58), (4.59), (4.3) e (4.12), temos

$$\begin{aligned}
\Delta L_N^{k+1} &= \xi_7\sqrt{\rho_{\max}^k}\sqrt{\rho_{\max}^k}\|g_p(z_c^k,\mu^k)\|^2 \\
&\leq \xi_7\Big(\frac{\xi_1\xi_3}{\xi_7}\frac{10^{-2}\|g_p(z_c^k,\mu^k)\|^2}{2\sqrt{\bar{\xi}_7(\xi_0+1)}}\Big)\Big(10^2\sqrt{\rho^k}\frac{\sqrt{\|g(z_c^k,\mu^k)\|+1}}{\sqrt{\|g_p^k\|}}\Big) \\
&\leq \frac{\xi_1\xi_3}{2}\sqrt{\rho^k}\frac{\|g_p(z_c^k,\mu^k)\|^{3/2}}{\sqrt{\bar{\xi}_7}} \\
&\leq \frac{\xi_1\xi_3}{2}\sqrt{\rho^k}\|g_p(z_c^k,\mu^k)\|.
\end{aligned}$$

Finalmente, usando (4.58) e (4.59), temos

$$\begin{aligned}
\Delta L_N^{k+1} &< \xi_7\frac{\xi_1(1-\varepsilon_\mu)}{2\xi_7\bar{\xi}_7}\|g_p(z_c^k,\mu^k)\|^2 \\
&\leq \frac{\xi_1(1-\varepsilon_\mu)}{2}\|g_p(z_c^k,\mu^k)\|.
\end{aligned}$$

Desse modo, temos $\Delta L_N^{k+1} < -\frac{1}{2}\Delta L_T^k$. Logo $\rho_{\max}^k$ não muda depois de k_0.

■

Apresentamos agora o teorema de convergência global de nosso algoritmo.

Teorema 4.6. *Sob as Hipóteses H.1-H.4, se o método CDI gera uma sequência infinita então existe uma subsequência convergente a um ponto estacionário para* (2.1). *Se as condições (4.39), (4.40) e (4.41) também são satisfeitas, então toda subsequência convergente de* $\{x_c^k\}$ *tem ponto limite estacionário para (2.1).*

Demonstração: Suponha, por contradição, que $\liminf(\|g_p(z_c^k,\mu^k)\|) = 2b > 0$. Seja $\overline{k}_0 \in \mathbb{N}$ tal que $\|g_p(z_c^k,\mu^k)\| > b$ para qualquer $k > \overline{k}_0$. Assim, pelo item (ii) do Lema 4.5, existe $k_0 \geq \overline{k}_0$ tal que, para todo $k \geq k_0$, $\rho_{\max}^k = \rho_{\max}^{k_0}$. Junto com (4.3) e (4.12), isto implica que

$$\rho^k \geq 10^{-4}\frac{\rho_{\max}^{k_0} b}{\xi_0+1},$$

e, portanto, para qualquer $i \geq k_0$, usando (4.22), temos $\Delta L_T^i \leq -\theta$, onde

$$\theta = \xi_1 b \min\Big\{\xi_2 b, 10^{-2}\xi_3\sqrt{\frac{\rho_{\max}^{k_0} b}{\xi_0+1}}, 1-\varepsilon_\mu\Big\} > 0. \tag{4.60}$$

Agora, usando (4.34) e (4.60), podemos garantir que, para $k > k_0$,

$$\begin{aligned} L(z_c^k, \lambda^k, \mu^k) - L(z_c^{k_0}, \lambda^{k_0}, \mu^{k_0}) &= \sum_{i=k_0+1}^{k} \Delta L_c^i \\ &\leq \frac{1}{4} \sum_{i=k_0}^{k-1} \Delta L_T^i + r^{k_0} \\ &\leq -\frac{1}{4}(k - k_0)\theta + r^{k_0} \longrightarrow -\infty. \end{aligned}$$

Isso implica que f é descontínua, ou h é descontínua, ou uma das sequências, $\{z_c^k\}$ ou $\{\lambda^k\}$, não é limitada, contrariando H.1-H.2. Portanto, $\liminf(\|g_p(z_c^k, \mu^k)\|) = 0$.

Para a segunda parte do teorema, suponhamos válidas (4.39), (4.40) e (4.41). Então, pelo Lema 4.5, existe k_0 tal que $\rho_{\max}^k = \rho_{\max}^{k_0}$, para todo $k \geq k_0$.

Suponha, por absurdo, que $\|g_p(z_c^{k_l}, \mu^{k_l})\| \geq b > 0$ para uma subsequência infinita $\{k_l\}$. Neste caso, usando (4.34) e (4.60) novamente, e fazendo $n_k \longrightarrow \infty$, temos

$$L(z_c^k, \lambda^k, \mu^k) - L(z_c^{k_0}, \lambda^{k_0}, \mu^{k_0}) \leq -\frac{1}{4} n_k \theta + r^{k_0} \longrightarrow -\infty,$$

onde θ é dado por (4.60). Isso também contradiz H.1-H.2, implicando que $\|g_p(z_c^{k_l}, \mu^{k_l})\| \longrightarrow 0$ para toda subsequência de $\{z_c^k\}$.

Seja $\{k_l\}$ uma subsequência convergente tal que $\|g_p(z_c^{k_l}, \mu^{k_l})\| \longrightarrow 0$. Pelas propriedades do algoritmo descritas em (4.2)-(4.4) e (4.6), temos $\rho^{k_l} \longrightarrow 0$, $\|h(z_c^{k_l})\| \longrightarrow 0$ e $\mu^{k_l} \longrightarrow 0$. Pela definição de $g_p(z_c^{k_l}, \mu^{k_l})$ temos $-\mu^{k_l} e - S_c^{k_l} \lambda_I^{k_l} \longrightarrow 0$, de modo que $S_c^{k_l} \lambda_I^{k_l} \longrightarrow 0$, e $\nabla f(x_c^{k_l}) + \nabla c(x_c^{k_l})^T \lambda^{k_l} \longrightarrow 0$. Além disso, $\lambda_I^{k_l} \leq \alpha(\mu^{k_l})^n$, de modo que $\lim \lambda_I^{k_l} \leq 0$. Portanto, o ponto limite dessa subsequência é estacionário.

■

4.2 Convergência Local

Agora vamos analisar as propriedades do métodos quando obtemos uma sequência convergente, com algumas condições adicionais. Mostraremos que o método tem convergência superlinear em dois passos. Para isso, vamos analisar o método em função das restrições ativas na solução.

Pediremos que as condições de segunda ordem tradicionais sejam satisfeitas e algumas condições extras, em geral específicas para o método.

Sejam $\{z^k\}$ e $\{z_c^k\}$ sequências geradas pelo algoritmo, convergentes a z^*. Seja $\{\lambda^k\}$ convergente a $\lambda^* = \lambda_{LS}(z^*, 0)$. Pelo algoritmo, temos

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

Defina $\mathcal{A}(x) = \{i \in E \cup I : c_i(x) = 0\}$, e $\mathcal{A}^* = \mathcal{A}(x^*)$. Defina λ_A^k e λ_A^* como as componentes de λ^k e λ^*, respectivamente, correspondentes às restrições ativas. Suponha que x^* é um "bom minimizador" para (2.1), no sentido de que $V = \{\nabla c_i(x^*) : i \in \mathcal{A}^*\}$ é L.I., e

$$y^T \Big[\nabla^2 f(x^*) + \sum_{i \in \mathcal{A}^*} \nabla^2 c_i(x^*) \lambda_i^*\Big] y = y^T \Big[\nabla^2 f(x^*) + \sum_{i=1}^{m} \nabla^2 c_i(x^*) \lambda_i^*\Big] y \geq \theta_1 \|y\|^2, \tag{4.61}$$

com $\theta_1 > 0$ e $y \in T = \{w : w^T \nabla c_i(x^*) = 0 : i \in E \cup J\}$, onde $J = \{i \in I : \lambda_i^* < 0\}$. Defina a matriz $\nabla c_A(x^*)$, cujas linhas são os vetores de V, e a matriz $\nabla c_B(x^*)$, cujas linhas são os vetores que faltam para completar $\nabla c(x^*)$. Como $\nabla c_A(x)$ é contínua, numa vizinhança de x^*, $\nabla c_A(x)$ tem posto linha completo (ver A.1). Assim, podemos definir

$$\lambda_A(x) = -[\nabla c_A(x) \nabla c_A(x)^T]^{-1} \nabla c_A(x) \nabla f(x),$$

$$g_A(x) = \nabla f(x) + \nabla c_A(x)^T \lambda_A(x),$$

e

$$H_A(x, \lambda) = \nabla^2 f(x) + \sum_{i \in \mathcal{A}^*} \nabla^2 c_i(x) \lambda_i.$$

Note que $\lambda_A(x_c^k) \longrightarrow \lambda_A^*$. Numa vizinhança V^* de x^*, podemos definir o projetor ortogonal no núcleo de $\nabla c_A(x)$ como

$$P(x) = I - \nabla c_A(x)^T [\nabla c_A(x) \nabla c_A(x)^T]^{-1} \nabla c_A(x),$$

e afirmamos que ele é Lipschitz contínuo, pois $c \in C^2$. Também definimos o passo completo $\delta_c^k = z_c^{k+1} - z_c^k = \delta_T^k + \delta_N^{k+1}$.

Além de considerar as hipóteses H.1-H.4, nossa análise da convergência local de x_c^k e x^k será baseada em cinco hipóteses locais. Quatro delas estão descritas a seguir. A quinta será descrita mais adiante.

Hipótese H.5.

$$\begin{aligned} \|\lambda^k - \lambda_{LS}(z_c^k, \mu^k)\| &= \mathcal{O}(\|g_p(z_c^k, \mu^k)\|), \\ (\lambda_I^{k+1})^T(s_c^{k+1} - s^k) &= \mathcal{O}(\|g_p(z_c^k, \mu^k)\|\rho^k) \end{aligned}$$

Hipótese H.6. *B^k é assintoticamente uniformemente definida positiva em $\mathcal{N}(A(z_c^k))$, isto é, em alguma vizinhança de z^*, podemos redefinir θ_1 de modo que*

$$\theta_1\|y\|^2 \leq y^T B^k y \leq \theta_2\|y\|^2, \tag{4.62}$$

para $y \in \mathcal{N}(A(z_c^k))$, onde $\theta_2 = \xi_0$.

Hipótese H.7. *Seja Z_A^k uma matriz cujas colunas formam uma base ortonormal para o núcleo de $\nabla c_A(x_c^k)$. Defina*

$$\delta_x^k = -Z_A^k[(Z_A^k)^T B_x^k Z_A^k]^{-1}(Z_A^k)^T g_A(x_c^k), \tag{4.63}$$

$$\delta_{s_i}^k = \frac{1}{s_{c_i}^k}\nabla c_{I_i}(x_c^k)^T \delta_x^k, \tag{4.64}$$

e

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

Assumimos que δ_A^k é o primeiro passo tentado pelo algoritmo se $\|\delta_A^k\| \leq \Delta$ e $s_c^k + S_c^k \delta_s^k \geq \varepsilon_\mu s_c^k$. Além disso, supomos que

$$P(x_c^k)[B_x^k - H_A(x^*, \lambda^*)]\delta_x^k = o(\|\delta_x^k\|). \tag{4.65}$$

Note que se $s_{c_i}^k \longrightarrow 0$, então $i \in \mathcal{A}^*$. Daí, $\nabla c_{I_i}(x_c^k)^T$ é uma das linhas de $\nabla c_A(x_c^k)$, de modo que $\nabla c_{I_i}(x_c^k)^T Z_A^k = 0$. Logo, $\delta_{s_i}^k$ é zero. Caso contrário, $s_{c_i}^k$ permanece afastado de zero. Portanto δ_s^k é limitado. Assim, definimos $s_{\min} > 0$ tal que se $i \notin \mathcal{A}^*$, então $s_{c_i}^k \geq s_{\min}$.

Hipótese H.8. *Existem constantes positivas θ_A e θ_p, tais que para k suficientemente grande,*

$$\|g_A(x_c^k)\| \leq \theta_A \|g_p^k\|,$$

$$\|g_p^k\| \leq \theta_p \|g_A(x_c^k)\|,$$

$$\|c_A(x_c^k)\| = \Theta(\|h(z_c^k)\|),$$

$$\|c_A(x^k)\| = \Theta(\|h(z^k)\|),$$

$$\|x_c^{k+1} - x^k\| = \mathcal{O}(\|c_A(x^k)\|).$$

Como $\nabla c_A(x)$ e $H_A(x,\lambda)$ são contínuas e $\nabla c_A(x^*)$ tem posto completo, nossas hipóteses implicam que existe $\theta_3 > 0$ e uma vizinhança V^* de x^*, tal que, para x, $x_c^k \in V^*$,

$$\|\nabla h(z)^T \lambda\| \;\geq\; \theta_3 \|\lambda\|, \qquad \lambda \in \mathbb{R}^{n+m_I},\ s \in \mathbb{R}^{m_I}_+. \tag{4.66}$$

Usando (4.65) e a continuidade de H_A, obtemos

$$\begin{aligned} P(x_c^k)[B_x^k - H_A(x_c^k, \lambda_A(x_c^k))]\delta_x^k &= P(x_c^k)[B_x^k - H_A(x_c^k, \lambda_A(x_c^k)) + H_A(x^*,\lambda^*) - H_A(x^*,\lambda^*)]\delta_x^k \\ &= o(\|\delta_x^k\|) + P(x_c^k)[H_A(x^*,\lambda^*) - H_A(x_c^k, \lambda_A(x_c^k))]\delta_x^k \\ &= o(\|\delta_x^k\|) + P(x_c^k)[o(1)]\delta_x^k \\ &= o(\|\delta_x^k\|). \end{aligned} \tag{4.67}$$

e

$$\begin{aligned} P(x_c^k)[B_x^k - \nabla^2_{xx}\mathcal{L}(x_c^k, \lambda^k)]\delta_x^k &= P(x_c^k)[B_x^k - \nabla^2_{xx}\mathcal{L}(x_c^k, \lambda^k) + H_A(x^*,\lambda^*) - H_A(x^*,\lambda^*)]\delta_x^k \\ &= o(\|\delta_x^k\|) + P(x_c^k)[H_A(x^*,\lambda^*) - \nabla^2_{xx}\mathcal{L}(x_c^k, \lambda^k)]\delta_x^k \\ &= o(\|\delta_x^k\|) + P(x_c^k)[o(1)]\delta_x^k \\ &= o(\|\delta_x^k\|). \end{aligned} \tag{4.68}$$

Podemos escrever $\delta_x^k = Z_A^k \nu^k \in \mathcal{N}(\nabla c_A(x_c^k))$. Note que ν^k é minimizador do problema

$$\min \overline{q}_x^k(\nu) = q_x^k(Z_A^k \nu) = \frac{1}{2}\nu^T (Z_A^k)^T B_x^k Z_A^k \nu + \nu^T (Z_A^k)^T \nabla f(x_c^k),$$

onde $q_x(\delta)$ é definido como

$$q_x^k(\delta) = \frac{1}{2}\delta^T B_x^k \delta + \delta^T \nabla f(x_c^k),$$

isto é, a parte de $q^k(\delta)$ que não envolve s ou μ. Então, como $(Z_A^k)^T \nabla f(x_c^k) = (Z_A^k)^T g_A(x_c^k)$, temos

$$(Z_A^k)^T (B_x^k Z_A^k \nu^k + g_A(x_c^k)) = 0. \tag{4.69}$$

Por (4.62) e o fato de que $(Z_A^k)^T Z_A^k = I$, a matriz $[(Z_A^k)^T B_x^k Z_A^k]^{-1}$ satisfaz, na vizinhança V^*, para todo $u \in \mathbb{R}^{\overline{n}-m_A}$,

$$\frac{1}{\mu_2}\|u\|^2 \leq u^T[(Z_A^k)^T B_x^k Z_A^k]^{-1} u \leq \frac{1}{\mu_1}\|u\|^2. \tag{4.70}$$

No próximo lema, mostraremos que o passo δ_A^k é aceito pelo algoritmo.

Lema 4.7. *O passo δ_A^k é aceito pelo algoritmo 3.1, para k suficientemente grande.*

Demonstração: Como $g_A(x_c^k) \in \mathcal{N}(\nabla c_A(x_c^k))$, existe ν_p^k tal que $g_A(x_c^k) = Z_A^k \nu_p^k$ com $\|g_A(x_c^k)\| = \|\nu_p^k\|$. Combinando (4.63) e (4.70) temos

$$\begin{aligned}
\|\delta_x^k\| &= \|Z_A^k[(Z_A^k)^T B_x^k Z_A^k]^{-1}(Z_A^k)^T g_A(x_c^k)\| = \|[(Z_A^k)^T B_x^k Z_A^k]^{-1}(Z_A^k)^T g_A(x_c^k)\| \\
&\leq \frac{1}{\mu_1}\|(Z_A^k)^T g_A(x_c^k)\| = \frac{1}{\mu_1}\|\nu_p^k\| \\
&= \frac{1}{\mu_1}\|g_A(x_c^k)\|,
\end{aligned} \tag{4.71}$$

Além disso, para $i \notin \mathcal{A}^*$,

$$\left|\delta_{A_i}^k\right| = \left|\frac{\nabla c_i(x_c^k)^T \delta_x^k}{s_i}\right| \leq \frac{\xi_0}{s_{\min}}\|\delta_x^k\| \leq \frac{\xi_0}{\mu_1 s_{\min}}\|g_A(x_c^k)\|,$$

de modo que

$$\|\delta_s^k\| \leq \frac{m_A \xi_0}{\mu_1 s_{\min}}\|g_A(x_c^k)\|. \tag{4.72}$$

Portanto,

$$\|\delta_A^k\| \leq \theta_4 \|g_A(x_c^k)\|, \tag{4.73}$$

onde $\theta_4 = \sqrt{1 + (m_A \xi_0 / s_{\min})^2}/\mu_1$, para k suficientemente grande. Como $g_A(x_c^k) \longrightarrow 0$, para k suficientemente grande, temos $\|\delta_A^k\| < \Delta_{\min}$ e $\delta_s^k \geq (\varepsilon_\mu - 1)e$, de modo que, pela Hipótese H.7, δ_A^k será o primeiro passo tentado pelo algoritmo, em alguma vizinhança V^*.

Para k suficientemente grande, $\overline{q}_x(\nu)$ será uma quadrática definida positiva. Portanto, seu mínimo, $\overline{q}_x(\nu^k)$, satisfaz

$$\begin{aligned} q_x(\delta_x^k) &= \overline{q}_x(\nu^k) \\ &= -\frac{1}{2}(g_A(x_c^k))^T Z_A^k [(Z_A^k)^T B_x^k Z_A^k]^{-1} (Z_A^k)^T g_A(x_c^k) \\ &\leq -\frac{1}{2\mu_2}\|g_A(x_c^k)\|^2, \end{aligned} \tag{4.74}$$

onde a última desigualdade vem de (4.70). Além disso, temos

$$\begin{aligned} q_s(\delta_s^k) &= \frac{1}{2}\delta_s^T B_s \delta_s + \delta_s^T(-\mu e) \\ &= \frac{1}{2}\sum_{i \notin \mathcal{A}^*} B_{s_i}\delta_{s_i}^2 - \mu \sum_{i \notin \mathcal{A}^*} \delta_{s_i} \\ &= \frac{1}{2}\sum_{i \notin \mathcal{A}^*} \frac{B_{s_i}}{s_i^2}\nabla c_i(x_c^k)^T \delta_x - \mu \sum_{i \notin \mathcal{A}^*} \frac{\nabla c_i(x_c^k)^T \delta_x}{s_i} \\ &\leq \frac{1}{2} m \frac{\xi_0^2}{s_{\min}^2}\|\delta_x\|^2 + \alpha_\mu \rho^k m \frac{\xi_0}{s_{\min}}\|\delta_x\| \\ &\leq \frac{m\xi_0^2}{2s_{\min}}\|\delta_x\|^2 + \frac{\alpha_\mu \rho_{\max}^{k-1} m}{s_{\min}}\|g_p^k\|\|\delta_x\| \\ &\leq \frac{m\xi_0^2}{2s_{\min}\mu_1^2}\|g_A(x_c^k)\|^2 + \frac{\alpha_\mu \rho_{\max}^0 m\theta_p}{s_{\min}\mu_1}\|g_A(x_c^k)\|^2. \end{aligned} \tag{4.75}$$

Assim, juntando (4.74) com (4.75), temos

$$\begin{aligned} q(\delta_A) &= q_x(\delta_x) + q_s(\delta_s) \\ &\leq \Big[-\frac{1}{2\mu_2} + \frac{m\xi_0^2}{2s_{\min}\mu_1^2} + \frac{\alpha_\mu \rho_{\max}^0 m\theta_p}{s_{\min}\mu_1} \Big] \|g_A(x_c^k)\|^2 \\ &= \mu_3 \|g_A(x_c^k)\|^2, \end{aligned} \tag{4.76}$$

onde μ_3 é o termo entre colchetes na segunda linha.

Agora, usando uma expansão de Taylor, temos

$$\begin{aligned}\Delta L_T^+ &= L(z_c^k + \Lambda_c^k \delta_A^k, \lambda^k, \mu^k) - L(z_c^k, \lambda^k, \mu^k) \\ &= \nabla_z L(z_c^k, \lambda^k, \mu^k)^T \Lambda_c^k \delta_A^k + \frac{1}{2}(\delta_A^k)^T \Lambda_c^k \nabla_{zz}^2 L(z_c^k, \lambda^k, \mu^k) \Lambda_c^k \delta_A^k + o(\|\Lambda_c^k \delta_A^k\|^2) \\ &= (g_p^k)^T \delta_A^k + \frac{1}{2}(\delta_A^k)^T W(z_c^k, \lambda^k, \mu^k) \delta_A^k + o(\|\delta_A^k\|^2). \end{aligned} \tag{4.77}$$

Lembrando que $A(z_c^k)\delta_A^k = 0$, o primeiro termo de (4.77) se expande em

$$\begin{aligned}(g_p^k)^T \delta_A^k &= [g(z_c^k, \mu^k) + A(z_c^k)^T \lambda^k]^T \delta_A^k \\ &= g(z_c^k, \mu^k)^T \delta_A^k + (\lambda^k)^T A(z_c^k) \delta_A^k \\ &= q(\delta_A^k) - \frac{1}{2}(\delta_A^k)^T B^k \delta_A^k. \end{aligned}$$

Substituindo isso em (4.77), temos

$$\Delta L_T^+ = q(\delta_A^k) + \frac{1}{2}(\delta_A^k)^T [W(z_c^k, \lambda^k, \mu^k) - B^k] \delta_A^k + o(\|\delta_A^k\|^2). \tag{4.78}$$

Usando o fato de que $\delta_x^k = P(x_c^k)\delta_x^k$, (4.68), (4.71) e (4.72), o segundo termo de (4.78) vira

$$\begin{aligned}(\delta_A^k)^T [W(z_c^k, \lambda^k, \mu^k) - B^k] \delta_A^k &= (\delta_x^k)^T [\nabla_{xx}^2 \mathcal{L}(x_c^k, \lambda^k) - B_x^k] \delta_x^k + (\delta_s^k)^T (\mu I - B_s^k) \delta_s^k \\ &= (\delta_x^k)^T P(x_c^k) [\nabla_{xx}^2 \mathcal{L}(x_c^k, \lambda^k) - B_x^k] \delta_x^k + (\delta_s^k)^T (\mu I - B_s^k) \delta_s^k \\ &= o(\|\delta_x^k\|^2) + o(\|\delta_s^k\|^2) \\ &= o(\|\delta_A^k\|^2). \end{aligned} \tag{4.79}$$

Agora, substituindo (4.79) em (4.78), e usando (4.73) obtemos

$$\begin{aligned}\Delta L_T^+ &= q(\delta_A^k) + o(\|\delta_A^k\|^2) \\ &= q(\delta_A^k) + o(\|g_A(x_c^k)\|^2). \end{aligned} \tag{4.80}$$

Segue de (4.76) e (4.80) que

$$\begin{aligned} r &= \frac{\Delta L_T^+}{q(\delta_A^k)} = 1 + \frac{o(\|g_A(x_c^k)\|^2)}{q(\delta_A^k)} \\ &\geq 1 + \frac{1}{\mu_3} \frac{o(\|g_A(x_c^k)\|^2)}{\|g_A(x_c^k)\|^2}. \end{aligned}$$

Lembrando que $\eta_1 \in (0,1)$, para k suficientemente grande, temos

$$\left|\frac{o(\|g_A(x_c^k)\|^2)}{\|g_A(x_c^k)\|^2}\right| \leq \mu_3(1-\eta_1),$$

logo,

$$r \;\geq\; 1 - \frac{1}{\mu_3}\mu_3(1-\eta_1) = \eta_1,$$

de modo que uma das condições da linha 3 do Algoritmo 3.1 é satisfeita para k suficientemente grande.

Por (4.17), (4.73), e a Hipótese H.8, temos

$$\begin{aligned}
\|\Lambda_c^k \delta_A^k\| &\leq \|\Lambda_c^k\|\|\delta_A^k\| \\
&\leq \xi_0\theta_4\|g_A(x_c^k)\| \\
&\leq \xi_0\theta_4\theta_A\|g_p^k\|. \qquad (4.81)
\end{aligned}$$

Então, de (4.81) e a Hipótese H.5, concluímos que as hipóteses da terceira parte do Lema 4.5 são satisfeitas, de modo que existe k_0 suficientemente grande tal que $\rho_{\max}^k = \rho_{\max}^{k_0}$, $k \geq k_0$. Assim, usando (4.3) e (4.12), para $k \geq k_0$, obtemos

$$\|g_p^k\| \leq 10^4\rho^k\frac{\|g(z_c^k,\mu^k)\|+1}{\rho_{\max}^k} \leq 10^4\rho^k\frac{\xi_0+1}{\rho_{\max}^{k_0}} = \beta\rho^k,$$

onde $\beta = 10^4(1+\xi_0)/\rho_{\max}^{k_0}$. Junto com (4.20), (4.73), a Hipótese H.8, (4.4), isto implica que, para k suficientemente grande,

$$\begin{aligned}
\|h(z_c^k + \Lambda_c^k\delta_A^k)\| &\leq \|h(z_c^k)\| + \|h(z_c^k + \Lambda_c^k\delta_A^k) - h(z_c^k)\| \\
&\leq \|h(z_c^k)\| + \overline{\xi}_0\|\delta_A^k\|^2 \\
&\leq \|h(z_c^k)\| + \overline{\xi}_0\theta_4^2\|g_A(x_c^k)\|^2 \\
&\leq \|h(z_c^k)\| + \overline{\xi}_0\theta_4^2\theta_A^2\|g_p^k\|^2 \\
&\leq \rho^k + \beta\overline{\xi}_0\theta_4^2\theta_A^2\|g_p^k\|\rho^k \\
&= \rho^k(1 + \beta\overline{\xi}_0\theta_4^2\theta_A^2\|g_p^k\|).
\end{aligned}$$

Assim para k suficientemente grande, $\beta\overline{\xi}_0\theta_4^2\theta_A^2\|g_p^k\| < 1$, de modo que o passo δ_A^k será aceito.

■

Até agora, pudemos trabalhar com um passo normal genérico que satisfizesse poucas condições. Doravante, vamos considerar um formato para o passo normal que nos ajudará na demonstração da convergência local.

Hipótese H.9. *Para k suficientemente grande, cada passo normal $\delta_N^{k+1} = z_c^{k+1} - z^k$ é calculado tomando um ou mais passos da forma*

$$\delta_N^+ = -J^+ h(z_c) = -J^T(JJ^T)^{-1}h(z_c), \tag{4.82}$$

onde J satisfaz

$$\|J - \nabla h(z_c)\| = \mathcal{O}(\|g_p^k\|). \tag{4.83}$$

Note que, para k suficientemente grande, usando (4.66), podemos redefinir θ_3 de modo que $\|J^T\lambda\| \geq \theta_3\|\lambda\|$.

Usando uma expansão de Taylor, (4.66), (4.82), (4.83), (4.19) e a continuidade de $A(z)$, é fácil mostrar que, se $z_c^{k+1} \neq z^k$, para k suficientemente grande, então o primeiro passo normal da iteração $k+1$, digamos δ_N^+, satisfaz

$$\|\delta_N^+\| \;=\; \|J^+h(z_c)\| \leq \frac{1}{\theta_3}\|h(z_c)\| = \mathcal{O}(\|h(z_c)\|) \tag{4.84}$$

e

$$\begin{aligned}
\|h(z_c^{k+1})\| &\leq \|h(z^k + \delta_N^+)\| \\
&= \|h(z^k) + \nabla h(z^k)\delta_N^+ + o(\|\delta_N^+\|)\| \\
&= \|h(z^k) + J\delta_N^+ + [\nabla h(z^k) - J]\delta_N^+ + o(\|\delta_N^+\|)\| \\
&= \|h(z^k) - JJ^+h(z^k) + [\nabla h(z^k) - J]\delta_N^+ + o(\|\delta_N^+\|)\| \\
&\leq \|\nabla h(z^k) - J\|\|\delta_N^+\| + o(\|\delta_N^+\|) \\
&= \mathcal{O}(\|g_p^k\|\|\delta_N^+\|) + o(\|\delta_N^+\|) \\
&= o(\|\delta_N^+\|) \\
&= o(\|h(z^k)\|).
\end{aligned} \tag{4.85}$$

No próximo lema, iremos mostrar que os valores de $\|g_A(x)\|$ e $\|c_A(x)\|$ definem uma medida de distância ao ponto x^*.

Lema 4.8. *Numa vizinhança V^* de x^*, temos*

$$\|x - x^*\| = \Theta(\|c_A(x)\| + \|g_A(x)\|). \tag{4.86}$$

Demonstração: Como $g_A(x)$ e $c_A(x)$ são Lipschitz contínuas, temos

$$\|g_A(x)\| = \|g_A(x) - g_A(x^*)\| = \mathcal{O}(\|x - x^*\|),$$

e

$$\|c_A(x)\| = \|c_A(x) - c_A(x^*)\| = \mathcal{O}(\|x - x^*\|).$$

Então

$$\|c_A(x)\| + \|g_A(x)\| = \mathcal{O}(\|x - x^*\|).$$

Agora consideremos o Lagrangeano do problema de igualdade associado às restrições ativas,

$$L_A(x, \lambda) = f(x) + c_A(x)^T \lambda,$$

cujas derivadas são

$$\nabla L_A(x, \lambda) = \begin{bmatrix} \nabla f(x) + \nabla c_A(x)^T \lambda \\ c_A(x) \end{bmatrix}$$

e

$$\nabla^2 L_A(x, \lambda) = \begin{bmatrix} H_A(x, \lambda) & \nabla c_A(x)^T \\ \nabla c_A(x) & 0 \end{bmatrix}.$$

Nesse caso, $\nabla L_A(x^*, \lambda^*) = 0$ e $\nabla^2 L_A(x^*, \lambda^*)$ é não-singular. Definamos $e_x = x^* - x, e_\lambda = \lambda^* - \lambda$ e $e = \begin{bmatrix} e_x \\ e_\lambda \end{bmatrix}$. Pelo Corolário A.3, temos

$$\|e\| = \mathcal{O}(\|\nabla L_A(x, \lambda)\|).$$

Daí,

$$\begin{aligned} \|x - x^*\| &\leq \left\| \begin{bmatrix} x - x^* \\ \lambda_A(x) - \lambda^* \end{bmatrix} \right\| \\ &= \mathcal{O}(\|\nabla L_A(x, \lambda_A(x))\|) \\ &= \mathcal{O}(\|g_A(x)\| + \|c_A(x)\|). \end{aligned}$$

■

Apresentamos agora o teorema de convergência local de nosso algoritmo. Obtivemos os mesmos resultados que o artigo original.

Teorema 4.9. *Com as hipóteses H.1-H.9, x^k e x_c^k são superlinearmente convergentes em dois passos para x^*. Se uma restauração é calculada a cada iteração, então x^k converge superlinearmente para x^*.*

Demonstração: As expressões (4.71) e (4.86) implicam que

$$\begin{aligned} \|x^{k+1} - x^*\| &\leq \|x^{k+1} - x_c^{k+1}\| + \|x_c^{k+1} - x^*\| \\ &= \|\delta_x^{k+1}\| + \|x_c^{k+1} - x^*\| \\ &= \mathcal{O}(\|g_A(x_c^{k+1})\|) + \|x_c^{k+1} - x^*\| \\ &= \mathcal{O}(\|x_c^{k+1} - x^*\|). \end{aligned} \tag{4.87}$$

Além disso, pela Hipótese H.8 e (4.86), temos

$$\begin{aligned} \|x_c^{k+1} - x^*\| &\leq \|x^k - x_c^{k+1}\| + \|x^k - x^*\| \\ &= \mathcal{O}(\|c_A(x^k)\|) + \|x^k - x^*\| \\ &= \mathcal{O}(\|x^k - x^*\|). \end{aligned} \tag{4.88}$$

Para mostrar a convergência local, precisamos das relações a seguir:

$$g_A(x^k) = o(\|x_c^k - x^*\|), \tag{4.89}$$

$$g_A(x_c^{k+1}) = o(\|x_c^{k-1} - x^*\|), \tag{4.90}$$

$$c_A(x^k) = o(\|x_c^{k-1} - x^*\|), \tag{4.91}$$

$$c_A(x_c^{k+1}) = o(\|x_c^k - x^*\|). \tag{4.92}$$

Vamos prová-las, começando por (4.89). Para uma vizinhança V^* de x^*, usando a expansão de

Taylor, temos

$$\begin{aligned}\|g_A(x^k)\| &= \|P(x^k)g_A(x^k)\| \\ &= \|P(x^k)[g_A(x_c^k) + \nabla g_A(x_c^k)\delta_x^k]\| + o(\|\delta_x^k\|) \\ &= \|P(x^k)\zeta_k\| + o(\|\delta_x^k\|) \\ &\leq \|[P(x^k) - P(x_c^k)]\zeta_k\| + \|P(x_c^k)\zeta_k\| + o(\|\delta_x\|), \end{aligned} \tag{4.93}$$

onde $\zeta_k = g_A(x_c^k) + \nabla g_A(x_c^k)\delta_x^k$. Pela continuidade de $P(x)$ em V^*, (4.8) e (4.71), temos

$$\begin{aligned}\|[P(x^k) - P(x_c^k)]\zeta_k\| &= \mathcal{O}(\|x_c^k - x^k\|\|\zeta_k\|) = o(\|\zeta_k\|) \\ &= o(\|g_A(x_c^k)\|) + o(\|\nabla g_A(x_c^k)\|\|\delta_x^k\|) \\ &= o(\|g_A(x_c^k)\|) + o(\|\delta_x^k\|) \\ &= o(\|g_A(x_c^k)\|). \end{aligned} \tag{4.94}$$

Além disso, usando o fato de que $P(x_c^k)\nabla c_A(x_c^k)^T = 0$, (4.69), (4.67) e (4.71), temos

$$\begin{aligned}\|P(x_c^k)\zeta_k\| &\leq \|P(x_c^k)[g_A(x_c^k) + B_x^k\delta_x^k]\| + \\ &\quad \|P(x_c^k)[H_A(x_c^k, \lambda_A(x_c^k)) - B_x^k]\delta_x^k\| + \\ &\quad \|P(x_c^k)[H_A(x_c^k, \lambda_A(x_c^k)) - \nabla g_A(x_c^k)]\delta_x^k\| \\ &= o(\|\delta_x^k\|) + \|P(x_c^k)\nabla c_A(x_c^k)^T\nabla\lambda_A(x_c^k)\delta_x^k\| \\ &= o(\|\delta_x^k\|) = o(\|g_A(x_c^k)\|). \end{aligned} \tag{4.95}$$

Assim, substituindo (4.94) e (4.95) em (4.93), e usando (4.86), obtemos

$$\|g_A(x^k)\| = o(\|g_A(x_c^k)\|) = o(\|x_c^k - x^*\|).$$

Para provar (4.92), precisamos considerar duas situações separadamente. Primeiro, suporemos que $\{k_i\}$ seja uma subsequência infinita em que nenhum passo normal é feito, i.e. $z_c^{k_i+1} = z^{k_i}$. Nesse caso, a dinâmica do controle da factibilidade, (4.2), a Hipótese H.8 e (4.89) implicam que

$$\begin{aligned}\|c_A(x_c^{k_i+1})\| &= \mathcal{O}(\rho^{k_i+1}) = \mathcal{O}(\|g_p^{k_i+1}\|) \\ &= \mathcal{O}(\|g_A(x_c^{k_i+1})\|) = \mathcal{O}(\|g_A(x^{k_i})\|) = o(\|x_c^{k_i} - x^*\|). \end{aligned} \tag{4.96}$$

Agora, vamos considerar uma subsequência infinita $\{k_j\}$, onde pelo menos um passo normal δ_N^+ satisfaz a Hipótese H.9. Nesse caso, as hipóteses H.8 e H.9, (4.86) e (4.87) implicam que

$$\begin{aligned}
\|c_A(x_c^{k_j+1})\| &= \mathcal{O}(h(z_c^{k_j+1})) = \mathcal{O}(\|h(z^{k_j}+\delta_N^+)\|) \\
&= o(\|h(z^{k_j})\|) = o(\|c_A(x^{k_j})\|) \\
&= o(\|x^{k_j}-x^*\|) = o(\|x_c^{k_j}-x^*\|). \qquad (4.97)
\end{aligned}$$

A expressão (4.92) segue de (4.96) e (4.97).

Combinando a Hipótese H.8, o Lema 4.7, (4.20), (4.85), (4.87), (4.86), (4.73) e (4.53), podemos escrever

$$\begin{aligned}
\|c_A(x^k)\| &= \mathcal{O}(\|h(z^k)\|) = \mathcal{O}(\|h(z_c^k)\|) + \mathcal{O}(\|h(z^k)-h(z_c^k)\|) \\
&= o(\|h(z^{k-1})\|) + \mathcal{O}(\|\delta_A^k\|^2) \\
&= o(\|c_A(x^{k-1})\|) + \mathcal{O}(\|\delta_A^k\|^2) \\
&= o(\|x^{k-1}-x^*\|) + \mathcal{O}(\|\delta_A^k\|^2) \\
&= o(\|x_c^{k-1}-x^*\|) + \mathcal{O}(\|\delta_A^k\|^2) \\
&= o(\|x_c^{k-1}-x^*\|) + \mathcal{O}(\|g_A(x_c^k)\|^2) \\
&= o(\|x_c^{k-1}-x^*\|) + \mathcal{O}(\|g_A(x_c^{k-1})\|^2) \\
&= o(\|x_c^{k-1}-x^*\|) + \mathcal{O}(\|x_c^{k-1}-x^*\|^2) \\
&= o(\|x_c^{k-1}-x^*\|).
\end{aligned}$$

Finalmente, para obter (4.90), usamos uma expansão de Taylor, a Hipótese H.8, (4.86), (4.89), (4.91), (4.88) e (4.87), de modo que

$$\begin{aligned}
\|g_A(x_c^{k+1})\| &= \|g_A(x^k)\| + \mathcal{O}(\|x_c^{k+1}-x^k\|) \\
&= \|g_A(x^k)\| + \mathcal{O}(\|c_A(x^k)\|) \\
&= o(\|x_c^{k-1}-x^*\|).
\end{aligned}$$

A convergência em dois passos segue utilizando (4.86), (4.89), (4.91), (4.88), (4.87), (4.90) e (4.92),

pois

$$\begin{aligned}\|x^{k+1} - x^*\| &= \mathcal{O}(\|g_A(x^{k+1})\| + \|c_A(x^{k+1})\|) \\ &= o(\|x_c^{k+1} - x^*\|) + o(\|x_c^k - x^*\|) \\ &= o(\|x^{k-1} - x^*\|), \end{aligned} \tag{4.98}$$

e

$$\begin{aligned}\|x_c^{k+1} - x^*\| &= \mathcal{O}(\|g_A(x_c^{k+1})\| + \|c_A(x_c^{k+1})\|) \\ &= o(\|x_c^{k-1} - x^*\|) + o(\|x_c^k - x^*\|) \\ &= o(\|x_c^{k-1} - x^*\|). \end{aligned} \tag{4.99}$$

Para concluir a prova, vamos supor que uma restauração não-nula é feita em cada iteração. Nesse caso, a Hipótese H.8, (4.20), (4.85), (4.86), (4.73) e (4.88) nos permitem melhorar (4.91), obtendo

$$\begin{aligned}\|c_A(x^k)\| &\leq \|c_A(x_c^k)\| + \|c_A(x^k) - c_A(x_c^k)\| \\ &= o(\|c_A(x^{k-1})\| + \mathcal{O}(\|\delta_A^k\|^2) \\ &= o(\|x^{k-1} - x^*\|) + \mathcal{O}(\|g_A(x_c^k)\|^2) \\ &= o(\|x^{k-1} - x^*\|) + \mathcal{O}(\|x_c^k - x^*\|^2) \\ &= o(\|x^{k-1} - x^*\|). \end{aligned} \tag{4.100}$$

Substituindo (4.89) e (4.100) em (4.98), temos

$$\begin{aligned}\|x^{k+1} - x^*\| &= \mathcal{O}(\|g_A(x^{k+1})\| + \|c_A(x^{k+1})\|) \\ &= o(\|x_c^{k+1} - x^*\|) + o(\|x^k - x^*\|) = o(\|x^k - x^*\|). \end{aligned}$$

■

4.3 Convergência para Pontos Estacionários da Infactibilidade

Algumas vezes, em nosso algoritmo, não é possível encontrar um ponto dentro do cilindro de confiança. Quando isso acontece, supomos que o problema é infactível, e desejamos que nosso método não faça iterações desnecessárias. Os teoremas de convergência anteriores consideram que a restauração não falha, isto é, que o passo normal consegue encontrar um ponto z_c^k tal que $\|h(z_c^k)\| \leq \rho^k$. Nesta seção, vamos mostrar que, se isso não é possível, então pelo menos o algoritmo deve convergir para um ponto estacionário da infactibilidade do problema (2.1).

Como apresentamos anteriormente, o problema normal é dado por

$$\begin{aligned} \min \quad & \tfrac{1}{2}\|h(z)\|^2 \\ \text{suj. a} \quad & s \geq 0. \end{aligned} \tag{3.18}$$

Já o problema da infactibilidade é

$$\min \qquad \|c_E(x)\|^2 + \|c_I^-(x)\|^2, \tag{4.101}$$

onde $v^- = (\min\{0, v_1\}, \min\{0, v_2\}, \ldots, \min\{0, v_n\})^T$.

Teorema 4.10. *Seja $\{z_N^j\}$ uma sequência gerada pela algoritmo normal com pontos de acumulação estacionários para o problema* (3.18)*, com*

$$z_N^j = \begin{bmatrix} x_N^j \\ s_N^j \end{bmatrix}.$$

Nesse caso, as componentes x_N^j de cada elemento dessa sequência formam uma sequência com pontos de acumulação estacionários para o problema da infactibilidade (4.101)*.*

Demonstração: Seja z^* um ponto de acumulação de $\{z_N^j\}$ estacionário para o problema (3.18). Então, pelas condições KKT, existe $w^* \in \mathbb{R}^{m_I}$ tal que

$$\nabla h(z^*)^T h(z^*) - \begin{bmatrix} 0 \\ w^* \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$w_i^* s_i^* = 0,$$

$$w^*, s^* \geq 0.$$

Como

$$\begin{aligned}\nabla h(z)^T h(z) &= \begin{bmatrix} \nabla c_E(x)^T & \nabla c_I(x)^T \\ 0 & -I \end{bmatrix} \begin{bmatrix} c_E(x) \\ c_I(x) - s \end{bmatrix} \\ &= \begin{bmatrix} \nabla c_E(x)^T c_E(x) + \nabla c_I(x)^T [c_I(x) - s] \\ s - c_I(x) \end{bmatrix},\end{aligned}$$

temos

$$\begin{aligned}\nabla c_E(x^*) c_E(x^*) + \nabla c_I(x^*)[c_I(x^*) - s^*] = 0, \\ s_i^* - c_{I_i}(x^*) = w_i^*.\end{aligned} \tag{4.102}$$

Para mostrar que x^* é um ponto estacionário do problema de infactibilidade (4.101), basta mostrar que

$$c_{I_i}(x^*) - s_i^* = c_{I_i}^-(x^*), \qquad i = 1, \ldots, m_I.$$

Para cada s_i^*, temos duas opções:

- Se $s_i^* > 0$, então $w_i^* = 0$. Daí, $c_{I_i}(x^*) = s_i^* > 0$, e portanto

$$c_{I_i}(x^*) - s_i^* = 0 = \min\{c_{I_i}(x^*), 0\}.$$

- Se $s_i^* = 0$, então $w_i^* \geq 0$. Daí, $c_{I_i}^* = -w_i^* \leq 0$, e portanto

$$c_{I_i}(x^*) - s_i^* = c_{I_i}(x^*) = \min\{c_{I_i}(x^*), 0\}.$$

De uma maneira ou de outra, temos

$$\nabla c_I(x^*)^T [c_I(x^*) - s^*] = \nabla c_I(x^*)^T c_I^-(x^*).$$

Substituindo essa expressão em (4.102), obtemos

$$\nabla c_E(x^*)^T c_E(x^*) + \nabla c_I(x^*)^T [c_I(x^*) - s_i^*] = \nabla c_E(x^*)^T c_E(x^*) + \nabla c_I(x^*)^T c_I^-(x^*) = 0,$$

de modo que x^* é ponto estacionário do problema (4.101).

■

Capítulo 5

Implementação

Neste capítulo, veremos os aspectos práticos do algoritmo, sua implementação, e detalhes computacionais. Vamos expandir a definição dos passos normal e tangente, detalhando os métodos internos para sua obtenção. Vamos considerar, ainda, o problema no formato geral utilizado no repositório de testes CUTEr[29]. Nossa implementação pode ser encontrada em [25].

5.1 Estruturas de Dados

Baseado no desempenho do algoritmo CDI para igualdades, decidimos utilizar a fatoração de Cholesky para resolver os sistemas lineares de nosso método. Sendo assim, procuramos uma implementação eficiente para usar como base de nosso algoritmo. Encontramos a biblioteca CHOLMOD [8, 13–16], escrita em C, que atendia aos nossos requisitos. Como trabalhamos em C++, decidimos criar uma biblioteca chamada de `base_matrices` [38] para servir de embrulho para as estruturas do CHOLMOD. As estruturas do CHOLMOD e seus respectivos embrulhos são

- `cholmod_common` - `base_common` - Estrutura que guarda as parâmetros, estatísticas e o espaço de trabalho das variáveis.

- `cholmod_sparse` - `base_sparse` - Estrutura que guarda matrizes esparsas no formato de coluna comprimida.

- `cholmod_factor` - `base_factor` - Estrutura que guarda a fatoração de Cholesky de uma matriz esparsa.

- `cholmod_dense` - `base_dense` - Estrutura que guarda matrizes ou vetores densos.

- `cholmod_triplet` - `base_triplet` - Estrutura que guarda matrizes esparsas no formato de triplas. É usada para receber facilmente os dados, e passá-los para o formato de coluna comprimida.

Também definimos tipos básicos de variáveis, considerando a possibilidade de expansão do algoritmo além dos tipos básicos de `C/C++`, e considerando a compatibilidade com Fortran e CUTEr. Atualmente, os tipos que temos são

- `Real` - Corresponde ao tipo `double`;

- `Int` - Corresponde ao tipo `int`;

- `Bool` - Corresponde ao tipo `int`.

Os tipos foram implementados de modo a permitir mudanças nas definições. Dessa maneira, o usuário que quiser utilizar alguma precisão diferente precisará mudar pouca coisa antes da compilação.

5.2 Interface CUTEr

Nosso algoritmo, intitulado `DCICPP` (do inglês, *Dynamic Control of Infeasibility - C Plus Plus*), trabalha com uma classe de interface, que recebe todas as informações necessárias para resolver o problema, como a função objetivo, as derivadas, e as restrições. Como decidimos utilizar as funções definidas pelo CUTEr como modelo para nossas funções, é esperado que o desempenho de um problema do CUTEr e de um problema feito "à mão" sejam equivalentes, em relação às chamadas de função. Obviamente, o custo de se calcular a função varia pois, em uma maneira, o usuário define-a explicitamente e, em outra, o CUTEr é responsável por retornar o valor.

O problema é recebido no formato utilizado pelo CUTEr, que é dado por

$$\begin{array}{lrcccl} \min & f(x) & & & & \\ \text{s.a} & & & c_E(x) & = & 0, \\ & c_L & \leq & c_I(x) & \leq & c_U, \\ & b_L & \leq & x & \leq & b_U. \end{array} \tag{5.1}$$

Para resolver um problema, é preciso fornecer uma quantidade mínima de informações, tais como um ponto inicial x^0 e os limitantes para as variáveis, b_L e b_U. Para indicar que o limitante superior não existe, definimos $b_{U_i} = 10^{20}$, e para indicar que o limitante inferior não existe, definimos $b_{L_i} = -10^{20}$. Além disso, o usuário precisa definir as funções computacionais que calculam os valores das funções matemáticas e suas derivadas, e, se o problema for irrestrito, alguns outros vetores com informações pertinentes.

Vamos apresentar agora as funções que o usuário precisa definir para executar o programa. Como alternativa, ele pode criar um arquivo `SIF` para o CUTEr, contendo as informações do problema, e resolver o problema usando o DCICPP para o CUTEr, apesar dessa estratégia ser, em geral, mais trabalhosa. É importante notar que, por compatibilidade com o CUTEr, que está em Fortran, todos os argumentos de função são ponteiros. Então, mesmo valores triviais, como o número de variáveis, são passados como ponteiros.

Para problemas irrestritos, é necessário definir as seguintes funções:

- `void UFN(Int *n, Real *x, Real *f)`
  Calcula o valor da função objetivo `*f` no ponto `x`, que tem dimensão `*n`.

- `void UOFG(Int *n, Real *x, Real *f, Real *g, Bool *grad)`
  Calcula o valor da função objetivo `*f`, no ponto `x`, que tem dimensão `*n`. Se `*grad` for 1, então também retorna o gradiente da função objetivo `g`.

- `void UPROD(Int *n, Bool *getder, Real *x, Real *p, Real *q)`
  Calcula o produto da Hessiana no ponto `x` por `p` e retorna o valor em `q`, isto é, retorna $q = \nabla^2 f(x)p$. Os vetores `x`, `p` e `q` têm dimensão `*n`. A variável `*getder` é sempre igual a 0 em nossa implementação.

Para problemas restritos, o usuário precisa definir as seguintes funções:

- `void CFN(Int *n, Int *m, Real *x, Real *f, Int *lc, Real *c)`
  Calcula o valor da função objetivo `*f` e o vetor de restrições `c` no ponto `x`. A dimensão de `x` é `*n` e a dimensão de `c` é `*m`. O Valor `*lc` é a dimensão real de `c`, que em nosso caso é sempre igual a `*m`.

- `void COFG(Int *n, Real *x, Real *f, Real *g, Bool *grad)`
  Calcula o valor da função objetivo `*f` no ponto `x`, que tem dimensão `*n`. Se `*grad` for 1, então também retorna o gradiente da função objetivo `g`.

- `void CPROD(Int *n, Int *m, Bool *getder, Real *x, Int *ly,`
  `Real *y, Real *p, Real *q)`
  Calcula o produto da Hessiana do Lagrangeano no ponto `x`, com multiplicadores `y`, pelo vetor `p`, retornando o vetor `q`, isto é, retornando $q = \nabla^2_{xx}\mathcal{L}(x, y)p$. `x`, `p` e `q` têm dimensão `*n`, e `y` tem dimensão `*m`. o valor `*getder` é sempre 0. O valor `*ly` é a dimensão real de `y`, que em nosso caso é sempre igual a `*m`.

- `void CCFSG(Int *n, Int *m, Real *x, Int *lc, Real *c,`
  `Int *nnzJ, Int *amax, Real *J, Int *indvar,`
  `Int *indfun, Bool *grad)`
  Calcula o valor das restrições `c` e, se `*grad` for igual a 1, também a Jacobiana das restrições no ponto `x`, guardando-a na forma de triplas, de modo que os valores J_{ij} são guardados em `J`, os índices das linhas são guardados em `indfun` e os índices das colunas são guardados em `indvar`. Apenas os valores não nulos são retornados. A dimensão de `x` é `*n`. A dimensão de `c` é `*m`. O vetor `*lc` é a dimensão real de `c`, que em nosso caso é sempre igual a `*m`. O valor `*nnzJ` é o número de elementos não nulos da Jacobiana, e `*amax` é a dimensão declarada de `J`, `indvar` e `indfun`, que tem que ser maior ou igual a `*nnzJ`.

Se o problema é restrito, além dessas funções o usuário também precisa fornecer um vetor binário `equatn`, de dimensão m, indicando quais restrições são de igualdade e quais são de desigualdade. Também é necessário fornecer os limitantes c_L e c_U das restrições, de modo que, se a restrição

i for de igualdade, então $c_{L_i} = c_{U_i} = 0$, e se for de desigualdade, então $c_{L_i} < c_{U_i}$. Finalmente, também é necessário fornecer `amax`, a quantidade máxima de elementos não nulos na Jacobiana. Se a restrição for ilimitada superiormente, defina $c_{U_i} = 10^{20}$, e se for ilimitada inferiormente, defina $c_{L_i} = -10^{20}$. Opcionalmente, o usuário pode passar o vetor `linear` indicando quais restrições são lineares. No entanto, ainda não estamos usando essa propriedade em todo o seu potencial.

O Apêndice B.1 mostra o arquivo de interface do CUTEr, e dois arquivos de exemplo para criação de problemas.

5.3 Generalização

Os teoremas apresentados nas seções anteriores foram desenvolvidos em torno do problema (2.1), que é o problema geral em um formato mais simples. No entanto, a implementação do método segue o formato utilizado pela biblioteca de testes CUTEr [29], mostrado em (5.1). Naturalmente, é possível converter o problema acima para o formato geral (2.1), definindo

$$\tilde{c}_I(x) = \begin{bmatrix} c_I(x) - c_L \\ c_U - c_I(x) \\ x - b_L \\ b_U - x \end{bmatrix},$$

de modo a obter o problema

$$\begin{aligned} \min \quad & f(x) \\ \text{s.a} \quad & c_E(x) = 0, \\ & \tilde{c}_I(x) \geq 0. \end{aligned}$$

No entanto, para trabalhar com esse problema, teríamos que lidar com a Jacobiana de $\tilde{c}_I$, cuja fatoração e armazenamento são caros. Decidimos, então, trabalhar com o formato do CUTEr e expandir o método para esse caso.

Nossa estratégia começa transformando o problema (5.1) num problema de igualdade com

limitantes nas variáveis. Para isso, introduzimos uma variável $s \in \mathbb{R}^{m_I}$, obtendo

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

Agora, definimos a variável

$$z = \begin{bmatrix} x \\ s \end{bmatrix},$$

a função

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

e os vetores

$$l = \begin{bmatrix} b_L \\ c_L \end{bmatrix} \qquad u = \begin{bmatrix} b_U \\ c_U \end{bmatrix},$$

para obter o problema

$$\begin{array}{rl} \min & f(x) \\ \text{s.a} & h(z) = 0, \\ & l \leq z \leq u. \end{array}$$

Lembramos que a variável ℓ_i pode valer $-\infty$, e u_i pode valer ∞, indicando que a componente i da variável z não tem limitante inferior ou superior, respectivamente. Note que podemos ter uma variável sem limitante algum.

Finalmente, vamos transformar esse problema no problema de pontos interiores,

$$\begin{array}{rl} \min & \varphi(z, \mu) = f(x) + \mu\beta(z) \\ \text{s.a} & h(z) = 0, \end{array} \tag{5.2}$$

onde β é uma função que se aproxima do infinito quando alguma componente de z se aproxima de seu limitante. Nossa ideia inicial foi fazer

$$\beta(z) = -\sum_{i:\ell_i > -\infty} \log(z_i - \ell_i) - \sum_{i:u_i < \infty} \log(u_i - z_i).$$

Nesse caso, nossa matriz de escalamento seria definida como $\Lambda(z) = (\Lambda_1(z), \ldots, \Lambda_N(z))$, onde

$$\Lambda_i(z) = \begin{cases} 1, & \text{se } \ell_i = -\infty \text{ e } u_i = \infty, \\ (z_i - \ell_i), & \text{se } \ell_i > -\infty \text{ e } u_i = \infty, \\ (u_i - z_i), & \text{se } \ell_i = -\infty \text{ e } u_i < \infty, \\ (z_i - \ell_i)(u_i - z_i), & \text{se } \ell_i > -\infty \text{ e } u_i < \infty. \end{cases} \tag{5.3}$$

Lembramos que esse escalamento provém da necessidade de melhorar a estabilidade das derivadas do problema. Assim, dada uma componente i de z que tenha limitante inferior e superior, a componente i de $\Lambda(z)\nabla\beta(z)$ será

$$\begin{aligned} \Lambda_i(z)\frac{\partial\beta(z)}{\partial z_i}(z) &= -(z_i - \ell_i)(u_i - z_i)\Big[\frac{1}{z_i - \ell_i} - \frac{1}{u_i - \ell_i}\Big], \\ &= -[u_i - z_i - (z_i - \ell_i)], \\ &= 2z_i - \ell_i - u_i. \end{aligned}$$

Podemos ver que essa componente pode gerar dificuldades computacionais caso z_i fique muito grande. Desse modo, nossa estratégia inicial não é satisfatória.

Decidimos, então, testar uma barreira β alternativa. Nossa escolha foi

$$\beta(z) = -\sum_{i=1}^{n} \beta_i(z),$$

onde

$$\beta_i(z) = \begin{cases} 0, & \text{se } \ell_i = -\infty \text{ e } u_i = \infty, \\ \log(z_i - \ell_i), & \text{se } \ell_i > -\infty \text{ e } u_i = \infty, \\ \log(u_i - z_i), & \text{se } \ell_i = -\infty \text{ e } u_i < \infty, \\ \log(z_i - \ell_i), & \text{se } \ell_i > -\infty \text{ e } u_i < \infty \text{ e } z_i < (u_i + \ell_i)/2, \\ \log(u_i - z_i), & \text{se } \ell_i > -\infty \text{ e } u_i < \infty \text{ e } z_i \geq (u_i + \ell_i)/2. \end{cases} \tag{5.4}$$

Essa escolha difere quando uma variável é limitada superior e inferiormente. Nesse caso, escolhemos penalizar de acordo com a proximidade ao limitante. A matriz de escalamento correspondente a

essa barreira é

$$\Lambda_i(z) = \begin{cases} 0, & \text{se } \ell_i = -\infty \text{ e } u_i = \infty, \\ z_i - \ell_i, & \text{se } \ell_i > -\infty \text{ e } u_i = \infty, \\ u_i - z_i, & \text{se } \ell_i = -\infty \text{ e } u_i < \infty, \\ z_i - \ell_i, & \text{se } \ell_i > -\infty \text{ e } u_i < \infty \text{ e } z_i < (u_i + \ell_i)/2, \\ u_i - z_i, & \text{se } \ell_i > -\infty \text{ e } u_i < \infty \text{ e } z_i \geq (u_i + \ell_i)/2. \end{cases} \tag{5.5}$$

Com essa nova barreira, agora temos a componente i do produto $\Lambda(z)\nabla\beta(z)$ constante:

$$\Lambda_i \frac{\partial \beta(z)}{\partial z_i} = \begin{cases} -1, & z_i < (u_i + \ell_i)/2, \\ 1, & z_i > (u_i + \ell_i)/2. \end{cases}$$

Note que as funções β_i e Λ_i são contínuas, porém não são diferenciáveis para $z_i = (u_i + \ell_i)/2$ se os limitantes forem finitos. Uma alternativa para contornar esse problema é suavizar Λ_i. Nesse caso, quando $\ell_i > -\infty$ e $u_i < \infty$, definimos Λ_i como

$$\Lambda_i(z_i) = \begin{cases} \dfrac{u_i - \ell_i}{2} - \left| z_i - \dfrac{\ell_i + u_i}{2} \right|, & \text{se } \left| z_i - \dfrac{\ell_i + u_i}{2} \right| > \sigma_i \\ \dfrac{u_i - \ell_i}{2} - \dfrac{\sigma_i}{2} - \dfrac{1}{2\sigma_i}\left(z - \dfrac{l + u}{2} \right)^2, & \text{caso contrário,} \end{cases}$$

onde $\sigma_i > 0, i = 1, \ldots, n + m_I$ são constantes, preferencialmente pequenas, que satisfazem $0 < \sigma_i < (u_i - \ell_i)/2$. A Figura 5.1 mostra o formato de função $\Lambda_i(z_i)$.

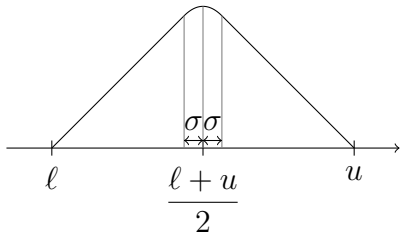


Figura 5.1: Ilustração da escolha de $\Lambda_i(z_i)$.

Na nossa implementação, escolhemos $\sigma_i = 0.01(u_i - \ell_i)/2$ e definimos β como

$$\beta(z) = \sum_{i=1}^{n+m_I} \beta_i(z_i) = \sum_{i=1}^{n+m_I} \ln(\Lambda_i(z)).$$

Com essa definição, temos

$$\beta_i'(z_i) = \frac{\Lambda_i'(z_i)}{\Lambda_i(z_i)},$$

e, para $\ell_i > -\infty$ e $u_i < \infty$, temos

$$\beta_i(z_i)\Lambda_i(z_i) = \Lambda_i'(z_i) = \begin{cases} 1, & z_i < \dfrac{\ell_i + u_i}{2} - \sigma_i, \\ -\dfrac{1}{\sigma_i}\Big(z - \dfrac{\ell_i + u_i}{2}\Big), & \Big|z_i - \dfrac{\ell_i + u_i}{2}\Big| \leq \sigma_i, \\ -1, & z_i > \dfrac{\ell_i + u_i}{2} + \sigma_i. \end{cases}$$

5.4 Passo Normal e Atualização de ρ

A iteração k do algoritmo começa com o passo normal, que consiste em encontrar um ponto z_c^k e atualizar o cilindro ρ^k de modo que $\|z_c^k\| \leq \rho^k$. Durante o passo normal, também atualizamos μ^k e λ^k. Esse passo é obtido por uma sequência de passos internos que resolvem aproximadamente o problema

$$\begin{aligned} \min \quad & \frac{1}{2}\|h(z)\|^2 \\ & l \leq z \leq u. \end{aligned} \tag{5.6}$$

Inicialmente, o cilindro tem raio ρ^{k-1}. Quando conseguirmos um ponto dentro desse cilindro, atualizamos seu raio, e atualizamos os multiplicadores. Se o ponto continuar dentro do cilindro, o definimos como o ponto z_c^k. Caso contrário, repetimos o procedimento. O algoritmo 5.1 explica o procedimento.

Algoritmo 5.1 Iteração k do Passo Normal

1: Dados $\phi_1 \in [10^{-4}, 1]$, $\rho = \rho^{k-1}$, $z_c = z^{k-1}$, e $\lambda = \lambda^{k-1}$

2: Atualize $\overline{A}$ e $\overline{J}$.

3: Faça $\Delta_N = \Delta_0$

4: **Enquanto** $\|h(z_c)\| > \rho$ **Faça**

5: Calcule z_c tal que $h(z_c) \leq \rho$ pelo **passo normal interno**

Continuação do Algoritmo 5.1

6: Atualize λ

7: Calcule $g_p = g(z_c, \mu) + \overline{A}^T \lambda$

8: Calcule $n_p = \|g_p\|/(\|g(z_c, \mu)\|+1)$

9: **Se** $\rho > 2\rho_{\max} n_p$ **Então**

10: $\rho = \min\{\phi_1 n_p, \max\{10^{-4} n_p, 0.75\}\}\rho_{\max}$

11: **Senão**

12: $\rho = \max\{\rho, \min\{\phi_1 \rho_{\max} n_p, \max\{10^{-4}\rho_{\max} n_p, 0.75\rho_{\max}\}\}\}$

13: **Fim do Se**

14: $\rho = \max\{\rho, \varepsilon_h\}$

15: gap $= \|\mu\nabla\beta_s(z) - \Lambda_s\lambda_I\|+\|g_p\|+\|h(z_c)\|$

16: $\mu = \min\{\mu, 100\rho, 100\rho^2, \text{gap}, 100\|h(z^{k-1})\|\}$

17: **Fim do Enquanto**

18: Faça $z_c^k = z_c$, $\rho^k = \rho$ e $\lambda^k = \lambda$.

No passo 2, atualizamos as matrizes $\overline{A}$ e $\overline{J}$. Como sugerido na Seção 4.2, essas matrizes são tais que

$$\|\overline{A} - A(z_c)\| = \mathcal{O}(\|g_p^k\|) \qquad \text{e} \qquad \|\overline{J} - \nabla h(z_c)\| = \mathcal{O}(\|g_p^k\|).$$

Nossa estratégia atual consiste em tomar simplesmente $\overline{A} = A(z_c)$ e $\overline{J} = \nabla h(z_c)$.

5.4.1 O Passo Normal Interno

No passo 5 do algoritmo 5.1, calculamos o passo normal interno do método. O objetivo do passo interno é, dado $\rho > 0$, obter um ponto dentro do cilindro de raio ρ. Segundo a Hipótese H.9, esperamos que, assintoticamente, os passos tenham a forma

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

chamada de forma de Gauss-Newton. Vamos utilizar uma modificação do Método de Dogleg, na linha de [23, 35], para obter o ponto acima, e vamos utilizar a teoria de convergência destes

métodos para mostrar que o método deve encontrar o ponto dentro do cilindro, e também que, assintoticamente, o passo de Gauss-Newton é utilizado.

A estratégia consiste em escolher um passo que seja combinação convexa da projeção dos passos de Cauchy $d_C = -\alpha_C \overline{J}^T h(z_c)$ e de Gauss-Newton $d_N = -\overline{J}(\overline{J}\overline{J}^T)^{-1}h(z_c)$ no interior da caixa definida pelos limitantes

$$\ell_{N_i} = \begin{cases} -\Delta_N, & \text{se } \ell_i = -\infty, \\ \max\{-\Delta_N, (\ell_i - z_{c_i})(1-\varepsilon_\mu)\} & \text{caso contrário,} \end{cases}$$

$$u_{N_i} = \begin{cases} \Delta_N, & \text{se } u_i = \infty \\ \min\{\Delta_N, (u_i - z_{c_i})(1-\varepsilon_\mu)\} & \text{caso contrário,} \end{cases}$$

Note que esses limitantes provêm de

$$l + \varepsilon_\mu(z_c - l) \leq z_c + d \leq u - \varepsilon_\mu(u - z_c), \tag{5.7}$$

que é a generalização da condição de fração-para-a-fronteira (3.17). O Algoritmo 5.2 exibe os procedimentos.

Algoritmo 5.2 Passo Normal Interno

1: Dados $\beta_1, \beta_2, \beta_3, \theta \in (0,1)$, $z_N^0 = z_c, \rho$

2: **Enquanto** $\|h(z_N^j)\| > \rho$ **Faça**

3: Defina $m(\delta) = \frac{1}{2}\|\overline{J}\delta + h(z_N^j)\|^2$

4: Calcule $v = \nabla m(0) = \overline{J}^T h(z_N^j)$.

5: Defina a matriz $D = \text{diag}(w_1, \ldots, w_N)$, onde

$$w_i = \begin{cases} (u_{N_i} - z_i)^{-1/2}, & \text{se } v_i < 0 \text{ e } u_{N_i} < \infty, \\ (z_i - \ell_{N_i})^{-1/2}, & \text{se } v_i > 0 \text{ e } \ell_{N_i} > -\infty, \\ 1, & \text{caso contrário.} \end{cases} \tag{5.8}$$

6: $d = -D^{-2}v$

7: $\gamma(\delta) = \arg\max\{t \geq 0 : \ell_{N_i} \leq t\delta \leq u_{N_i}\}$

8: Defina

$$P(\delta) = \begin{cases} \delta, & \text{se } \gamma(\delta) > 1 \\ \max\{\theta, 1 - \|\delta\|\}\gamma(\delta)\delta, & \text{caso contrário} \end{cases} \tag{5.9}$$

Continuação do Algoritmo 5.2

Calcule

$$\alpha_{cp} = \arg\min_{\alpha} \left\{ \frac{1}{2}\|\alpha\overline{J} + h(z_N^j)\|^2 \colon \alpha\|Dd\| \leq \Delta_N \right\} \tag{5.10}$$

Defina $d_{cp} = P(\alpha_{cp} d)$.

Defina

$$\rho_C(\delta) = \frac{m(0) - m(\delta)}{m(0) - m(d_{cp})}$$

Calcule $\overline{d}_N$, solução aproximada de $\min\{m(\delta) : \|D\delta\| \leq \Delta_N\}$.

Defina $d_N = P(\overline{d}_N)$.

Faça $t = 1$

Enquanto $\rho_C(td_N + (1-t)d_{cp}) \geq \beta_1$ **Faça**

    $t = 0.9t$

Fim do Enquanto

Defina $\delta_N^+ = td_N + (1-t)d_{cp}$.

Defina

$$\rho_h = \frac{\frac{1}{2}\|h(z_N^j)\|^2 - \frac{1}{2}\|h(z_N^j + \delta_N^+)\|^2}{m(0) - m(\delta_N^+)}.$$

Se $\rho_h \geq \beta_2$ **Então**

    $z_N^{j+1} = z_N^j + \delta_N^+$

    $\Delta_N \leftarrow 2\Delta_N$

Senão

    $z_N^{j+1} = z_N^j$

    $\Delta_N \leftarrow \Delta_N/4$

Fim do Se

Se $\|h(z_N^{j+1})\| \geq \beta_3\|h(z_N^j)\|$ 3 vezes consecutivas **Então**

    Então Atualize $\overline{J}$.

Fim do Se

$j \leftarrow j + 1$

Fim do Enquanto

Como dissemos, esperamos que esse método funcione e que, assintoticamente, tome direções de Gauss-Newton. Além disso, sabemos que o algoritmo converge para pontos estacionários do problema (5.6), de modo que, se não houver um ponto factível, ao menos teremos convergência para pontos estacionários da infactibilidade, como foi mostrado no Teorema 4.10. Vamos parafrasear as hipóteses de [23] para mostrar essas propriedades.

Hipótese H.10. (H1 de [23]) *A sequência* $\{z_N^j\}$ *gerada pelo algoritmo é limitada.*

Hipótese H.11. (H2 de [23]) *Para todo* z, w *num conjunto convexo, aberto e limitado* L *que contém toda a sequência gerada pelo algoritmo e todos os pontos da forma* $z_N^j + P(\delta_N^j)$*, temos*

$$\|\nabla h(z) - \nabla h(w)\| \leq 2\gamma_0 \|z - w\|.$$

Hipótese H.12. (H3 de [23]) $\nabla h(z)$ *tem posto linha completo para todo* $z \in L$*.*

Agora parafraseamos o teorema de convergência de [23], cuja demonstração pode ser encontrada no próprio artigo.

Teorema 5.1. *Suponha que as hipóteses H.10 e H.11 são satisfeitas e que o algoritmo gera um sequência infinita* $\{z_N^j\}$*. Então todo ponto limite é estacionário do problema* (5.6)*.*

O primeiro resultado do Teorema 5.1 já mostra que o ponto converge para pontos estacionários do problema normal, e podemos esperar que ele encontre um ponto dentro do cilindro. Agora, parafraseamos um lema do mesmo artigo, sobre o passo de Gauss-Newton

Lema 5.2 (Lemma 3.8 de [23])**.** *Suponha que H.10-H.12 são satisfeitas, e seja* $z^* \in \mathring{\Omega}$*, o interior de* Ω*, tal que* $h(z^*) = 0$*. Então, assintoticamente, o passo de Gauss-Newton é escolhido pelo algoritmo.*

5.4.2 Atualização dos Multiplicadores

No passo 6 do Algoritmo 5.1, é preciso calcular, ou atualizar, os multiplicadores de Lagrange. Tanto na hipótese 4.40 do Lema 4.5, quanto na Hipótese H.5, pedimos que

$$\|\lambda^k - \lambda_{LS}(z_c^k, \mu^k)\| = \mathcal{O}(\|g_p(z_c^k, \mu^k)\|).$$

Para obter essa propriedade, definimos $\psi : [0, \infty) \to [0, \infty)$, $\psi(\mu) = A\mu^n$, para $A, n > 0$, e fazemos

$$
\begin{aligned}
\tilde{\lambda} &= \lambda_{LS}(z_c^k, \mu^k) \\
\lambda_E^k &= \tilde{\lambda}_E \\
\lambda_{I_i}^k &= \begin{cases} \max\{\tilde{\lambda}_{I_i}, -\psi(\mu^k)\}, & \text{se } c_{L_i} = -\infty, \\ \min\{\tilde{\lambda}_{I_i}, \psi(\mu^k)\}, & \text{se } c_{U_i} = \infty, \\ \max\{\tilde{\lambda}_{I_i}, -\psi(\mu^k)\}, & \text{se } -\infty < c_{L_i} < c_{U_i} < \infty, \text{ e } s_i > (c_{L_i} + c_{U_i})/2, \\ \min\{\tilde{\lambda}_{I_i}, \psi(\mu^k)\}, & \text{se } -\infty < c_{L_i} < c_{U_i} < \infty, \text{ e } s_i \leq (c_{L_i} + c_{U_i})/2. \end{cases}
\end{aligned}
$$

5.5 Passo tangente

Após termos feito o passo normal e atualizado $\rho_{\max}$, obtemos o ponto z^k através do passo tangente, que é a solução aproximada do problema

$$
\begin{aligned}
\min_\delta \quad & q_k(\delta) = \tfrac{1}{2}\delta^T B^k \delta + \delta^T g_p^k \\
\text{s.a} \quad & A(z_c^k)\delta = 0 \\
& \ell_T \leq \delta \leq u_T.
\end{aligned}
\tag{5.11}
$$

onde B^k é uma aproximação para a Hessiana escalada $W(z_c^k, \lambda^k, \mu^k)$, e

$$
\begin{aligned}
\ell_{T_i} &= \begin{cases} -\Delta_T, & \text{Se } \ell_i = -\infty \\ \max\{-\Delta_T, -(z_{c_i}^k - \ell_i)(1 - \varepsilon_\mu)\}, & \text{caso contrário} \end{cases} \\
u_{T_i} &= \begin{cases} \Delta_T, & \text{Se } u_i = \infty \\ \min\{\Delta_T, (u_i - z_{c_i}^k)(1 - \varepsilon_\mu)\}, & \text{caso contrário} \end{cases}
\end{aligned}
$$

Resolvemos aproximadamente esse problema com o método de Steihaug [40] para obter o Passo Tangente Interno. Aceitamos o passo se tivermos decréscimo suficiente e se o ponto permanecer no cilindro de raio $2\rho^k$. Caso contrário, reduzimos a região de confiança e repetimos o processo. Além disso, pode ser necessário calcular uma Correção de Segunda Ordem. O Algoritmo 3.3 já mostra o algoritmo generalizado para o problema no formato CUTEr (5.1).

5.5.1 Passo Tangente Interno

Para encontrar um ponto que minimiza aproximadamente (5.11), utilizamos uma modificação do método de Steihaug. A grosso modo, esse método é uma generalização do método de gradientes conjugados para lidar com hessianas não definidas positivas, com restrições lineares e com uma região de confiança. Modificamos o método para lidar com limitantes no lugar da região de confiança, que é praticamente a mesma coisa que adotar uma região de confiança na norma infinito.

Algoritmo 5.3 Passo Tangente Interno

Dados: $\varepsilon_1, \varepsilon_2, \varepsilon_3 > 0$, $r^0 = g_p^k$, $p^0 = r^0$, $k = 0$, $d^0 = 0$, $\theta^0 = \langle r^0, r^0 \rangle$
Enquanto $\theta^k > \varepsilon_2$ E $\theta^k > \varepsilon_1 \theta^0$ **Faça**
    $\gamma^k = \langle d^k, B^k d^k \rangle$
    Se $\gamma^k \leq \varepsilon_3 \theta^k$ **Então**
        Defina $\delta_t = d^k + \nu p^k$ tal que $\ell_T \leq \Lambda(z_c^k)\delta_t \leq u_T$ e ν minimiza $q(d^k + \nu p^k)$.
    Fim do Se
    $\alpha^k = \theta^k / \gamma^k$
    Se $d^k + \alpha^k p^k < \ell_T$ **ou** $d^k + \alpha^k p^k > u_T$ **Então**
        Defina $\delta_t = d^k + \overline{\nu} p^k$, onde $\overline{\nu} = \arg\max\{\nu : \ell_T \leq \Lambda(z_c^k)\delta_t \leq u_T\}$.
    Fim do Se
    $d^{k+1} = d^k + \alpha^k p^k$
    $r^{k+1} = \text{proj}_{\mathcal{N}(A(z_c^k))}(r^k - \alpha^k B^k p^k)$
    $\theta^{k+1} = \langle r^{k+1}, r^{k+1} \rangle$
    $\beta^k = \theta^{k+1} / \theta^k$
    $p^{k+1} = r^{k+1} - \beta^k p^k$
    $k = k + 1$
Fim do Enquanto

5.5.2 Correção de Segunda Ordem

A correção de segunda ordem é adotada caso o primeiro passo interno tangente piore consideravelmente a factibilidade obtida no passo normal. Nos baseamos na ideia sugerida em [34] para

escolher a direção

$$\begin{aligned} d^+ &= \arg\min_{\delta} \|A(z_c^k)\delta + h(z_c^k + \delta_t) - h(z_c^k)\| \\ &= -\alpha A(z_c^k)^T [A(z_c^k)A(z_c^k)^T]^{-1}[h(z_c + \delta_t) - h(z_c^k)]. \end{aligned}$$

Esse passo é obtido da tentativa de trazer o passo tangente para o valor de infactibilidade de z_c^k. Note que a restrição do passo tangente é uma linearização de $h(z_c^k + \delta) = h(z_c^k)$. O que temos para esse passo é uma linearização de

$$h(z_c^k + \delta_t + \delta) = h(z_c^k).$$

Obtido esse passo, definimos a correção como

$$\delta_{soc} = \alpha d^+,$$

onde $\alpha \in (0, 1)$ é tal que

$$\tilde{\ell} \leq \Lambda(z_c^k)(\delta_t + \delta_{soc}) \leq \tilde{u}.$$

Escolhemos fazer essa correção se

$$\|h(z_c + \delta_t)\| > \min\{2\rho, 2\|h(z_c)\| + 0.5\rho\}$$

ou

$$\|h(z_c)\| \leq 10^{-5} \text{ e } \|h(z_c + \delta_t)\| > \max\{10^{-5}, 2\|h(z_c)\|\}$$

5.6 Sistemas Lineares

Precisamos resolver vários sistemas lineares no nosso algoritmo, todos na forma

$$AA^T x = b,$$

onde A é a Jacobiana escalada ou sem escalamento. Como vimos na Seção 5.1, os sistemas são resolvidos utilizando a fatoração de Cholesky. Note que a matriz AA^T é definida positiva se A tem posto linha completo. Caso seja detectado que A não tem posto linha completo, repetimos a fatoração de Cholesky com a matriz $AA^T + \varepsilon I$, onde ε é uma constante positiva. O único caso

em que a matriz A é a Jacobiana não escalada é no passo normal interno. Logo antes desse passo, precisamos atualizar a Jacobiana retirando o escalamento, e depois do método precisamos escalar novamente. Como a Jacobiana escalada é $A(z) = J(z)\Lambda(z)$, devemos calcular a fatoração de Cholesky da matriz $J(z)\Lambda(z)^2 J(z)^T$. Infelizmente, não podemos aproveitar a fatoração de $J(z)J(z)^T$ no passo tangente. Uma alternativa possível para nosso método, seria utilizar gradientes conjugados para resolver os sistemas. No entanto, isso iria requerer completa reestruturação do algoritmo, e o ganho no problemas de pequeno e médio porte não seria suficiente para justificar essa mudança. Também consideramos transformar o sistema definido positivo acima no sistema indefinido

$$\begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} 0 \\ -b \end{bmatrix},$$

e resolver esse sistema com a biblioteca MUMPS [1, 2], mas essa ideia não aumentou a robustez do algoritmo e o deixou um pouco mais lento.

5.7 Classificador de Problemas CUTEr

Criamos um programa extra que serve de classificador para o CUTEr [39]. Ao rodar um dos problemas do repositório com esse programa, o problema é adicionado a um arquivo (que é criado, caso não exista) com nome `classification.<rest>.<lib><fix><lin>`.

`<rest>` pode ser

- `unc`: Indica que não existem restrições no problema;
- `equ`: Indica que o problema tem apenas restrições de igualdade;
- `ineq`: Indica que o problema tem apenas restrições de desigualdade;
- `gencon`: Indica que o problema tem restrições de igualdade e desigualdade;

`<lim>` pode ser

- `free`: Indica que as variáveis não têm limitantes;
- `lower`: Indica que algumas variáveis têm limitante inferior;

- `upper`: Indica que algumas variáveis têm limitante superior;
- `box`: Indica que algumas variáveis têm limitante inferior, e algumas têm limitante superior;

`<lin>` pode ser

- (nulo): Se não existem restrições;
- `.linear`: Se todas as restrições são lineares;
- `.nonlin`: Se existem restrições não lineares;

e `<fix>` é nulo se não existem variáveis fixas ou `.fixed` caso contrário.

A partir dessa lista de arquivos, utilizamos comandos do shell do linux para montar listas com todos os problemas de um tipo específico. Isso serve para rodar apenas um tipo de problema, e também para separar os resultados por tipo.

Capítulo 6

Resultados Numéricos

Testamos nossa implementação com a interface de testes CUTEr, utilizando 767 problemas com variáveis ou restrições da ordem de até 10^6. Esses problemas são aqueles cujos arquivos SIF, específicos do CUTEr, utilizados para descrever o problema, são considerados pequenos. Não utilizamos problemas com variáveis fixas, pois ainda não temos uma versão estável com essa implementação.

O nosso algoritmo foi aplicado sequencialmente a cada problema, utilizando um máximo de 2×10^5 iterações e restaurações, e 2 horas de tempo de execução. Os possíveis resultados para a execução do algoritmo são

Convergiu: Indica que o algoritmo encontrou um ponto estacionário para o problema de minimização.

Estacionário da Infactibilidade: O algoritmo encontrou um ponto infactível estacionário para o problema de infactibilidade.

Máximo de Iterações: Indica que o algoritmo efetuou o número máximo de iterações permitidas, mas não convergiu.

Máximo de Restaurações: Indica que o algoritmo efetuou o número máximo de restaurações permitidas mas não conseguiu encontrar um ponto z_c^k dentro do cilindro de raio ρ.

$\rho_{\max}$ **pequeno:** Indica que $\rho_{\max}$ ficou muito pequeno. Isso sugere que os passos tangentes não fizeram progresso suficiente.

Limite de Tempo: Indica que o algoritmo atingiu o máximo de tempo permitido, mas não encontrou um ponto estacionário.

Ilimitado: Indica que o algoritmo encontrou algum ponto com norma maior que 10^{10}.

Falha: Indica que o algoritmo teve algum problema imprevisto, como falta de memória.

Um resumo dos resultados está na Tabela 6.1. A lista completa dos resultados do algoritmo pode ser encontrada no Apêndice C.

ExitFlag	**Total**	
	N°	**%**
Convergiu	689	89.60
Máximo de Iterações	3	0.39
Máximo de Restaurações	16	2.08
$\rho_{\max}$ **pequeno**	23	2.99
Limite de Tempo	14	1.82
Estacionário da Infactibilidade	9	1.17
Ilimitado	3	0.39
Falha	12	1.56
Total	767	100.00

Tabela 6.1: Resumo dos resultados

Separamos os testes em quatro conjuntos em função do tipo de restrições: os irrestritos; os que têm apenas restrições de igualdade; os que tem apenas restrições de desigualdade; e os que tem restrições dos dois tipos. As tabelas 6.2, 6.3, 6.4 e 6.5 mostram os resultados para cada um desses conjuntos, respectivamente.

Escolhemos comparar nosso algoritmo com IPOPT [42], e o ALGENCAN [3, 4], dois métodos bastante competitivos da área. A comparação foi feita utilizando o conceito de perfil de desempenho

Saída do Algoritmo	Irrestritos	
	N°	%
Convergiu	230	94.26
Máximo de Iterações	2	0.82
Máximo de Restaurações	0	0.00
$\rho_{\max}$ **pequeno**	2	0.82
Limite de Tempo	5	2.05
Estacionário da Infactibilidade	0	0.00
Ilimitado	1	0.41
Falha	4	1.64
Total	244	100.00

Tabela 6.2: Resultados do algoritmo para os problemas irrestritos

Saída do Algoritmo	Igualdade	
	N°	%
Convergiu	221	85.33
Máximo de Iterações	1	0.39
Máximo de Restaurações	9	3.47
$\rho_{\max}$ **pequeno**	14	5.41
Limite de Tempo	5	1.93
Estacionário da Infactibilidade	5	1.93
Ilimitado	2	0.77
Falha	2	0.77
Total	259	100.00

Tabela 6.3: Resultados do algoritmo para os problemas de igualdade

definido por Dolan and Moré [19]. Considerando um conjunto S de algoritmos aplicados à um

Saída do Algoritmo	Desigualdade	
	N°	%
Convergiu	186	92.08
Máximo de Iterações	0	0.00
Máximo de Restaurações	5	2.48
$\rho_{\max}$ **pequeno**	3	1.49
Limite de Tempo	2	0.99
Estacionário da Infactibilidade	4	1.98
Ilimitado	0	0.00
Falha	2	0.99
Total	202	100.00

Tabela 6.4: Resultados do algoritmo para os problemas de desigualdade

Saída do Algoritmo	Rest. Gerais	
	N°	%
Convergiu	52	81.25
Máximo de Iterações	0	0.00
Máximo de Restaurações	2	3.12
$\rho_{\max}$ **pequeno**	4	6.25
Limite de Tempo	2	3.12
Estacionário da Infactibilidade	0	0.00
Ilimitado	0	0.00
Falha	4	6.25
Total	64	100.00

Tabela 6.5: Resultados do algoritmo para os problemas igualdade e desigualdade

conjunto P de problemas, definimos a razão de desempenho como

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}},$$

para cada $p \in P$ e $s \in S$, onde $t_{p,s}$ é o tempo gasto pelo algoritmo s para resolver o problema p. O desempenho do algoritmo, em comparação ao conjunto de algoritmos considerado, é dado pela função

$$\mathcal{P}_s(t) = \frac{1}{N_p}|\{p \in P : r_{p,s} \leq t\}|,$$

onde N_p é o número de problemas, isto é, a cardinalidade de P. Se um algoritmo s não consegue resolver o problema p, definimos $r_{p,s} = \infty$. Também definimos o maior valor finito para o qual os algoritmos convergem como

$$r_f = \max_{p,s}\{r_{p,s} : r_{p,s} < \infty\}.$$

O valor para ∞ computacionalmente depende da implementação. Nós utilizamos a linguagem Python e o valor `float('inf')`, que se comporta como o valor simbólico ∞.

A função $\mathcal{P}_s(t)$ mede a fração dos algoritmos resolvidos pelo algoritmo s dentro do tempo t escalado pelo tempo do algoritmo mais rápido. Note que, por definição, sempre temos $r_{p,s} \geq 1$, de modo que o gráfico deve ser desenhado no intervalo $[1, r_f]$. O valor $\mathcal{P}_s(1)$ indica a eficiência do algoritmo s, enquanto que o valor $\mathcal{P}_s(r_f)$ indica a robustez.

Os testes foram feitos num notebook Dell XPS 15 L502X, com processador i7-2820QM 2.3 GHz com 8Gb de RAM. Consideramos como critério de convergência as infactibilidades dual, primal e a complementaridade serem menores que 10^{-6}, isto é, $\varepsilon_g = \varepsilon_h = \varepsilon_a = 10^{-6}$ no Algoritmo 3.1; e um máximo de tempo de execução do algoritmo de 2 horas. Para o IPOPT, utilizamos a versão 3.10.4 com as opções

```
max_cpu_time 7200
tol 1e-6
constr_viol_tol 1e-6
```

Para o ALGENCAN, usamos a versão 2.4.0 com as opções

```
FEASIBILIT 1.0D-6
OPTIMALITY 1.0D-6
```

e a execução foi interrompida externamente. Os parâmetros utilizados na nossa implementação para a realização destes testes foram $\varepsilon_\mu = 10^{-6}$, $\nu = 1$, $\alpha_\rho = 100$, $\alpha_h = 100$, $\eta_1 = 10^{-3}$, $\eta_2 = 0.2$,

$\alpha_R = 0.75$, $\alpha_I = 2.5$, $\phi_1 = 1$, $\beta_1 = 0.1$, $\beta_2 = 0.25$, $\beta_3 = 0.95$, $\theta = 0.99995$, $\varepsilon_1 = 10^{-6}$, $\varepsilon_2 = 10^{-14}$, e $\varepsilon_3 = 10^{-8}$.

A Figura 6.1 mostra o gráfico de perfil de desempenho do DCICPP, IPOPT e ALGENCAN, para todos os problemas do nosso conjunto. Essa figura indica que o DCICPP é um método bastante competitivo. Nosso algoritmo resolve alguns problemas mais rapidamente, mas o ALGENCAN logo o ultrapassa. O IPOPT demora um pouco mais, mas recupera a diferença. Com o decorrer de 2 horas, nosso algoritmo consegue resolver uma quantidade maior de problemas que o IPOPT, mas menor que o ALGENCAN.

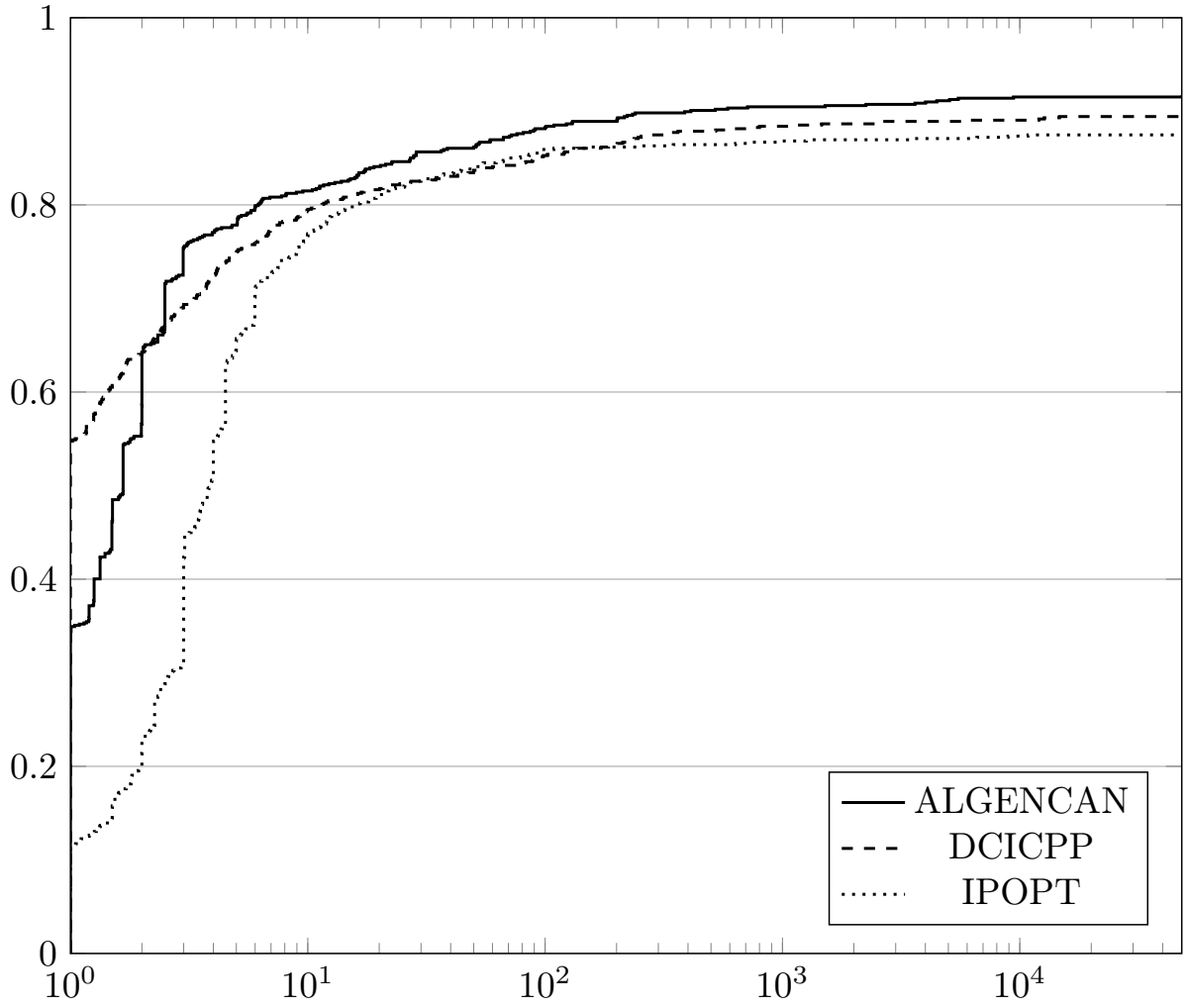


Figura 6.1: Comparação do DCICPP com o IPOPT e o ALGENCAN nos 767 problemas do CUTEr.

Numa tentativa de entender melhor as falhas de nosso algoritmo, fizemos uma separação do conjunto de problemas, em relação ao tipo de restrições e dos limitantes da variáveis. Como mostramos na Seção 5.7, em função das restrições, os problemas são divididos em

- `unc` - Problemas irrestritos;
- `equ` - Problemas apenas com restrições de igualdade;
- `ineq` - Problemas apenas com restrições de desigualdade;
- `gencon` - Problemas com restrições de igualdade e desigualdade.

Para os problemas restritos, podemos ter ainda

- `linear` - Problemas que contém apenas restrições lineares;
- `nonlin` - Problemas com restrições não lineares.

Em relação às variáveis, podemos classificar os problemas em

- `free` - Problemas em que nenhuma variável tem limitante.
- `lower` - Problemas em que as variáveis têm apenas limitante inferior, ou nenhum limitante.
- `upper` - Problemas em que as variáveis têm apenas limitante superior, ou nenhum limitante.
- `box` - Problemas em que existem variáveis com limitante inferior e variáveis com limitante superior, podendo haver variáveis com os dois limitantes.

Separamos os resultados para todas as possíveis combinações de classificação acima. Infelizmente, algumas destas combinações não são informativas, por conter uma quantidade muito pequena de testes. No entanto, achamos válido mostrar e comentar alguns dos resultados.

A Figura 6.2 mostra o gráfico de perfil de desempenho do DCICPP com o IPOPT e o ALGENCAN para o subconjunto dos problemas irrestritos. Podemos ver que o DCICPP é superior ao IPOPT nesse conjunto de problema, tanto em questão de eficiência, quanto de robustez. O ALGENCAN e o DCICPP são equivalentes. Note que para esse tipo de problema o método DCICPP

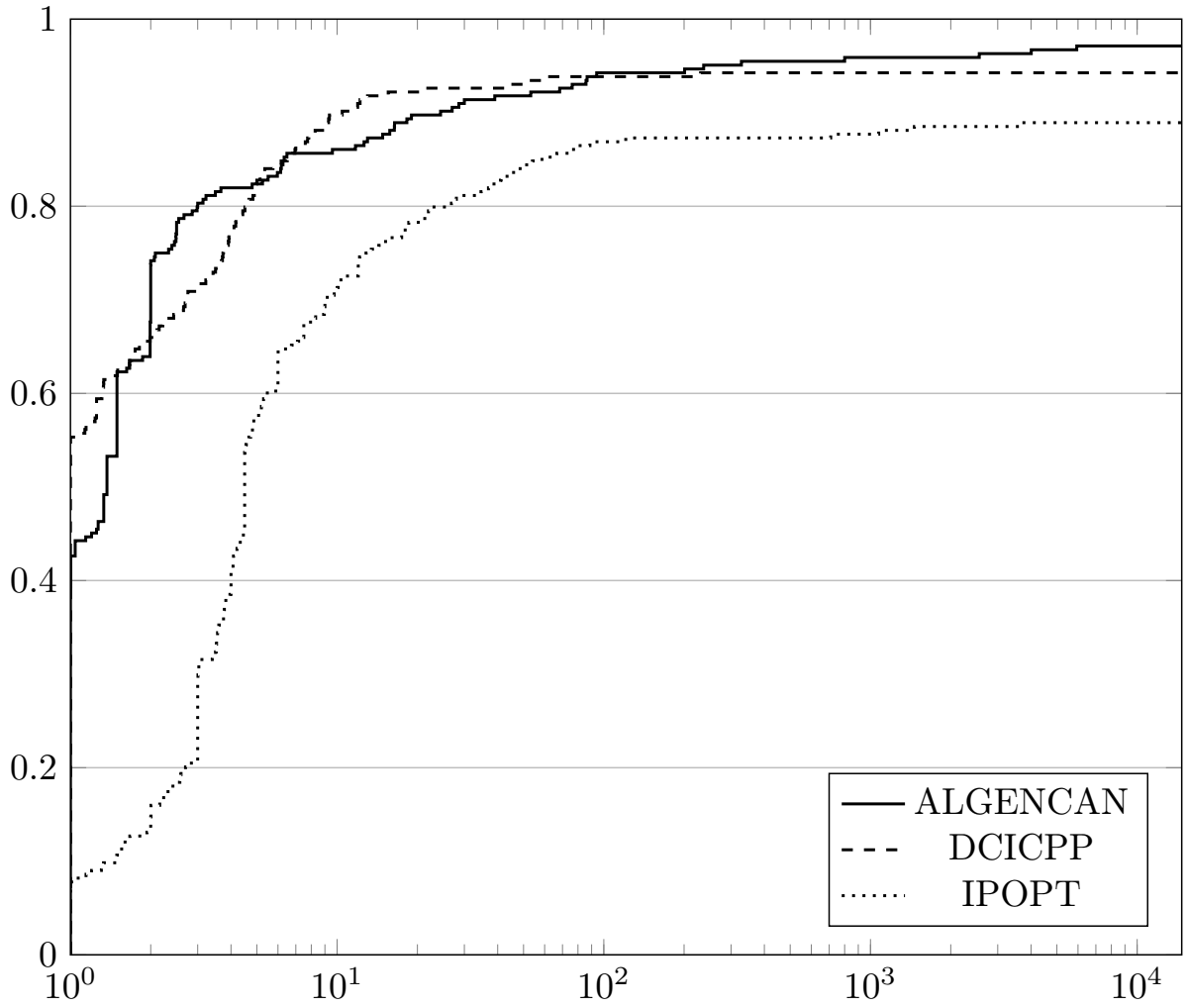


Figura 6.2: Comparação do DCICPP com o IPOPT e o ALGENCAN para problemas irrestritos.

é composto apenas pelo passo tangente, no entanto sem a restrição do espaço nulo. Nesse caso, o método é basicamente o método de Steihaug com caixas para aproximações quadráticas sucessivas.

A Figura 6.3 mostra o gráfico de perfil de desempenho do DCICPP com o IPOPT e o ALGENCAN para o subconjunto dos problemas com restrições apenas de igualdade. O desempenho do DCICPP para esses problemas é consideravelmente melhor. Podemos ver um alto nível de eficiência, e bastante competitividade.

A Figura 6.4 mostra o gráfico de perfil de desempenho do DCICPP com o IPOPT e o ALGENCAN para o subconjunto dos problemas com restrições apenas de desigualdade. Neste subconjunto, apesar de resolvermos mais problemas em menos tempo, o nosso método fica inferior para uma

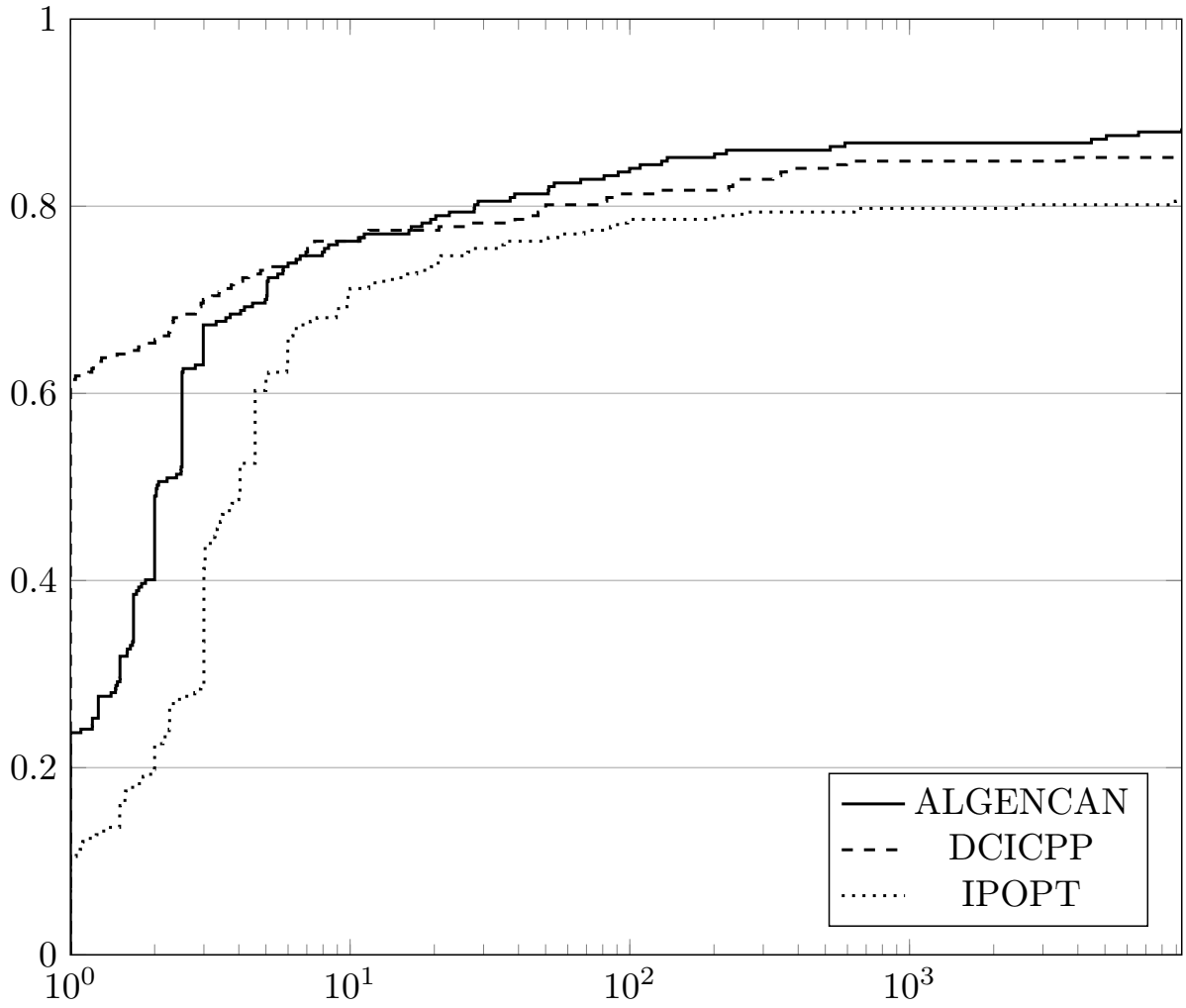


Figura 6.3: Comparação do DCICPP com o IPOPT e o ALGENCAN nos problemas com restrições apenas de igualdade.

certa parcela dos problemas.

A Figura 6.5 mostra o gráfico de perfil de desempenho do DCICPP com o IPOPT e o ALGENCAN para o subconjunto dos problemas com restrições de igualdade e de desigualdade. A quantidade de problemas neste subconjunto é muito pequena para que se possa tirar alguma conclusão definitiva. Tendo isso em mente, podemos afirmar que os métodos são equivalentes.

Os resultados mostrados aqui sugerem que nosso método pode render um software profissional. Alguns ajustes são necessários para expandir sua área de atuação. É importante lembrar que esta implementação, assim como o próprio método, foi iniciada a pouco tempo, e que não é esperado que

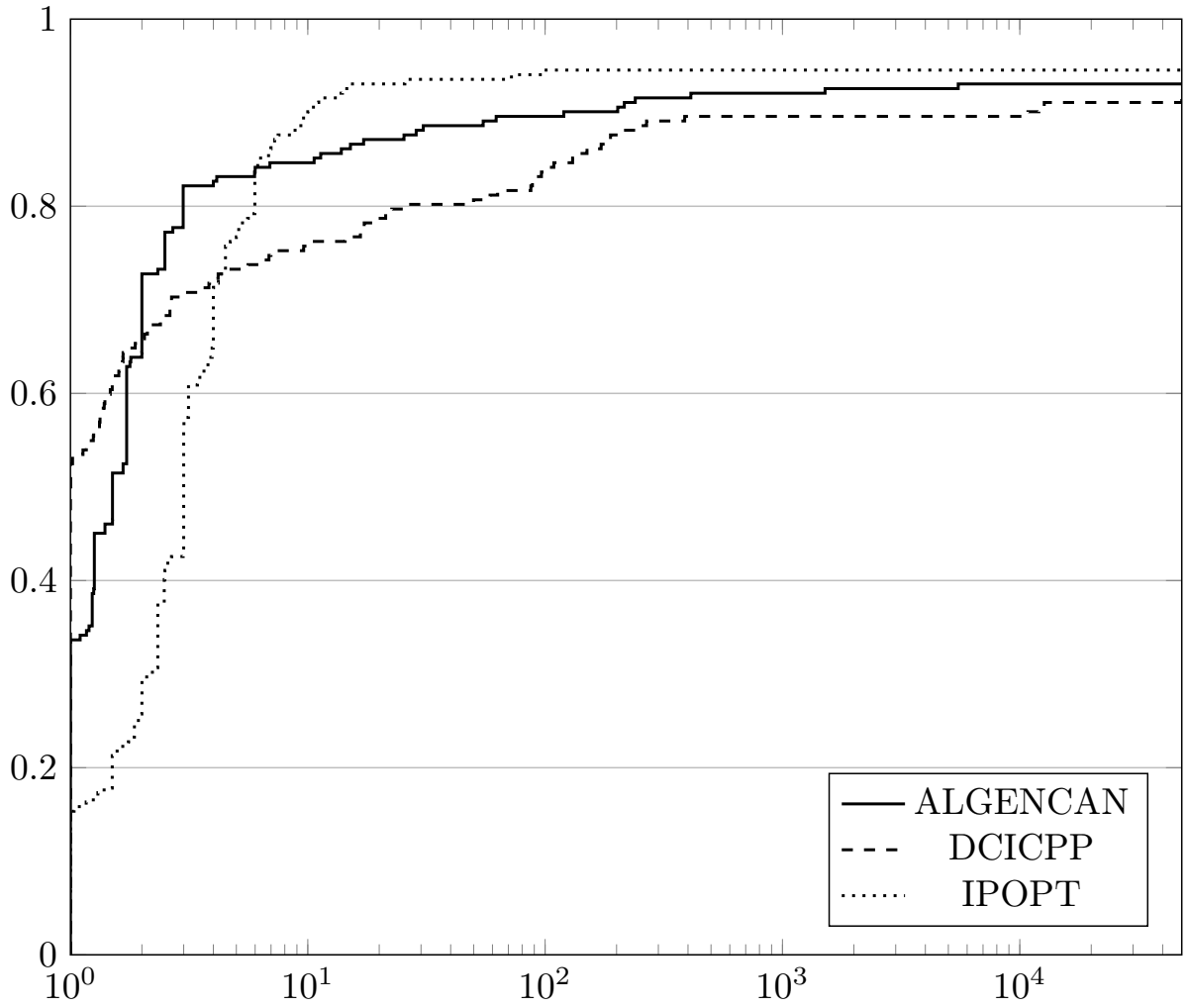


Figura 6.4: Comparação do DCICPP com o IPOPT e o ALGENCAN nos problemas com restrições apenas de desigualdade.

ela já supere os outros algoritmos bem estabelecidos. Com isso em mente, os resultados sugerem que as estratégias descritas aqui são viáveis e que devem gerar mais projetos de pesquisa.

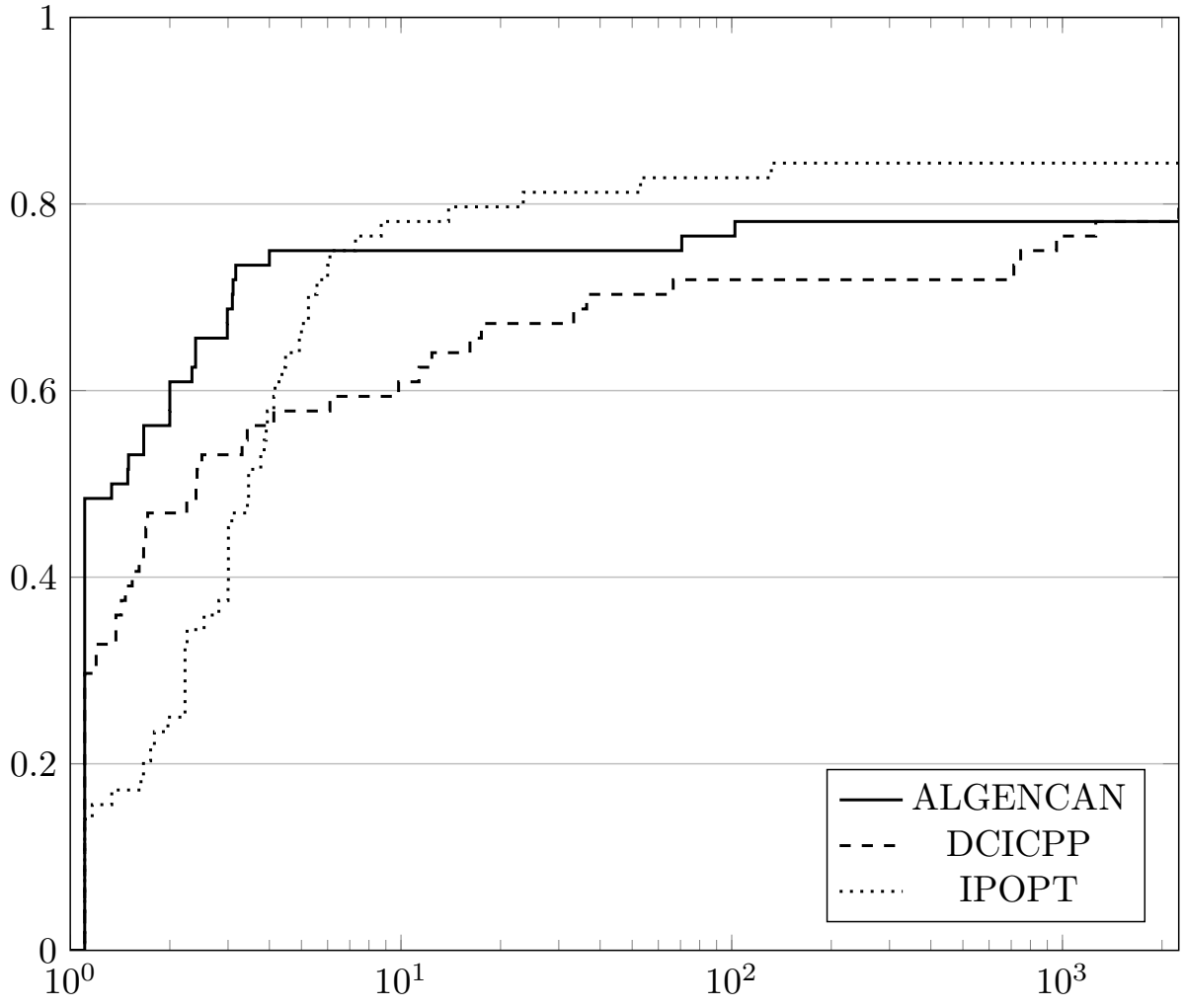


Figura 6.5: Comparação do DCICPP com o IPOPT e o ALGENCAN nos problemas com restrições de igualdade e de desigualdade.

Capítulo 7

Conclusões

Apresentamos nesta tese o método DCICPP, um método para o problema de otimização não linear contínua. Mostramos uma teoria satisfatória de convergência global e local e dentro das expectativas dos métodos usuais. Além disso, o algoritmo se mostrou competitivo em comparação ao IPOPT e ao ALGENCAN, dois dos melhores algoritmos gratuitos para programação não linear.

Ainda existe bastante espaço para trabalho tanto na teoria, quanto no algoritmo. Consideramos a extensão da teoria para condições mais gerais de passos tangentes e normais, possibilitando a implementação de algoritmos diferentes e mais robustos para os subproblemas. Uma possibilidade é utilizar algum pacote pronto para o subproblema. Também consideramos a possibilidade de implementar a resolução dos sistemas lineares utilizando métodos iterativos e/ou aproximados. Esperamos estudar os problemas com Jacobianas singulares e quase singulares. Também vale mencionar que o tratamento de variáveis fixas está sendo implementado, e esperamos obter uma versão estável para próximos trabalhos.

Para finalizar, vale mencionar que a escolha dos parâmetros também pode fazer muita diferença nos resultados, e que escolhemos apenas parâmetros canônicos ou sugeridos na literatura, de modo que é possível obter alguma melhoria no desempenho através da seleção criteriosa desses parâmetros.

Em trabalhos futuros, esperamos fazer novas comparações com outros algoritmos notáveis e implementar as mudanças sugeridas acima. Também gostaríamos de obter hipóteses mais fracas para a teoria de convergência do método, e estudar a estratégia de cilindros de confiança com

outras aplicações.

Bibliografia

[1] P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, and J. Koster. *A fully asynchronous multifrontal solver using distributed dynamic scheduling.* SIAM Journal on Matrix Analysis and Applications, 23(1):15–41, 2001.

[2] P. R. Amestoy, A. Guermouche, and S. Pralet. *Hybrid scheduling for the parallel solution of linear systems.* Parallel Computing, 32:136–156, 2006.

[3] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt. *On augmented lagrangian methods with general lower-level constraints.* SIAM Journal on Optimization, 18: 1286–1309, 2007.

[4] ——. *Augmented lagrangian methods under the constant positive linear dependence constraint qualification.* Mathematical Programming, 111:5–32, 2008.

[5] R. H. Bielschowsky and F. A. M. Gomes. *Dynamic control of infeasibility in equality constrained optimization.* SIAM Journal on Optimization, 19(3):1299–1325, 2008.

[6] H. Byrd, J. C. Gilbert, and J. Nocedal. *A trust region method based on interior point techniques for nonlinear programming.* Mathematical Programming, 89(1):149–185, 2000.

[7] H. Byrd, M. E. Hribar, and J. Nocedal. *An interior point algorithm for large scale nonlinear programming.* SIAM Journal on Optimization, 9(4):877–900, 2000.

[8] 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), 2009.

[9] A. R. Conn, N. I. M. Gould, and P. L. Toint. *LANCELOT: a Fortran package for large-scale nonlinear optimization*, volume 17 of *Springer Series in Computational Mathematics*. Springer Verlag (Heidelberg, New York), 1992.

[10] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. Springer, 2000.

[11] F. E. Curtis, J. Nocedal, and A. Wächter. *A matrix-free algorithm for equality constrained optimization problems with rank-deficient Jacobians.* SIAM Journal on Optimization, 20(3):1224–1249, 2009.

[12] F. E. Curtis, O. Schenk, and A. Wächter. *An interior-point algorithm for large-scale nonlinear optimization with inexact step computations.* SIAM Journal on Scientific Computing, 32(6):3447–3475, 2010.

[13] T. A. Davis and W. W. Hager. *Modifying a sparse Cholesky factorization.* SIAM Journal on Matrix Analysis and Applications, 20(3):606–627, 1999.

[14] ——. *Multiple-rank modifications of a sparse Cholesky factorization.* SIAM Journal on Matrix Analysis and Applications, 22(4):997–1013, 2001.

[15] ——. *Row modifications of a sparse Cholesky factorization.* SIAM Journal on Matrix Analysis and Applications, 26(3):621–639, 2005.

[16] ——. *Dynamic supernodes in sparse Cholesky update/downdate and triangular solves.* ACM Transactions on Mathematical Software, 35(4), 2009.

[17] J. E. Dennis and L. N. Vicente. *On the convergence theory of trust-region-based algorithms for equality-constrained optimization.* SIAM Journal on Optimization, 7(4):927–950, 1997.

[18] J. E. Dennis, Jr., M. El-Alem, and M. C. Maciel. *A global convergence theory for general trust-region-based algorithms for equality constrained optimization.* SIAM Journal on Optimization, 7(1):177–207, 1997.

[19] E. D. Dolan and J. J. Moré. *Benchmarking optimization software with performance profiles.* Mathematical Programming, 91(2):201–213, 2002.

[20] M. El-Alem. *A global convergence theory for Dennis, El-Alem, and Maciel's class of trust-region algorithms for constrained optimization without assuming regularity.* SIAM Journal on Optimization, 9(4):965–990, 1999.

[21] R. Fletcher and S. Leyffer. *Nonlinear programming without a penalty function.* Mathematical Programming, 91(2):239–269, 2002.

[22] A. Forsgren, P. E. Gill, and M. Wright. *Interior methods for nonlinear optimization.* SIAM review, 44(4):525–597, 2002.

[23] 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.

[24] P. E. Gill, W. Murray, and M. A. Saunders. *SNOPT: An SQP algorithm for large-scale constrained optimization.* SIAM Journal on Optimization, 12:979–1006, 2002.

[25] F. A. M. Gomes and A. S. Siqueira. *Dcicpp.* http://www.github.com/abelsiqueira/dcicpp, 2009.

[26] F. A. M. Gomes, M. C. Maciel, and J. M. Martínez. *Nonlinear programming algorithms using trust regions and augmented Lagrangians with nonmonotone penalty parameters.* Mathematical Programming, 84(1):161–200, 1999.

[27] F. A. M. Gomes. *A sequential quadratic programming algorithm that combines merit function and filter ideas.* Computational and Applied Mathematics, 26(3):337–379, 2007.

[28] C. C. Gonzaga, E. Karas, and M. Vanti. *A globally convergent filter method for nonlinear programming*. SIAM Journal on Optimization, 14(3):646–669, 2003.

[29] N. Gould, D. Orban, and P. L. Toint. *CUTEr and SifDec: A constrained and unconstrained testing environment, revisited*. ACM Transactions on Mathematical Software, 29(4): 373–394, 2003.

[30] W. Hock and K. Schittkowski. *Test examples for nonlinear programming codes*. Journal on Optimization Theory and Applications, 30(1):127–129, 1980.

[31] M. Lalee, J. Nocedal, and T. Plantenga. *On the implementation of an algorithm for large-scale equality constrained optimization*. SIAM Journal on Optimization, 8(3):682–706, 1998.

[32] D. G. Luenberger and Y. Ye. *Linear and nonlinear programming*. Springer, 3 edition, 2008.

[33] J. M. Martínez. *Otimização prática usando o lagrangiano aumentado*, 2009.

[34] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2 edition, 2006.

[35] M. Porcelli. *On the convergence of an inexact Gauss-Newton trust-region method for nonlinear least-squares problems with simple bounds*. Optimization Letters, 7(3):447–465, 2013.

[36] D. F. Shanno and R. J. Vanderbei. *An interior-point algorithm for nonconvex nonlinear programming*. Computational Optimization and Applications, 13:231–252, 1999.

[37] ——. *Interior-point methods for nonconvex nonlinear programming: Orderings and higher-order methods*. Mathematical Programming, 87(2):303–316, 2000.

[38] A. S. Siqueira. *Base matrices*. http://www.github.com/abelsiqueira/base_matrices, 2009.

[39] ——. *Classify cuter*. http://www.github.com/abelsiqueira/classify_cuter, 2010.

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

[41] A. Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering.* PhD thesis, Carnegie Mellon University, 2002.

[42] A. Wächter and L. T. Biegler. *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming.* Mathematical Programming, 106(1): 25–57, 2006.

Apêndice A

Demonstrações extras

A.1 Teoremas envolvendo o posto

Teorema A.1. *Dados a função matricial $A : \mathbb{R}^n \to \mathbb{R}^{m\times r}$ contínua, com $m < r$ e o ponto $\overline{x} \in \mathbb{R}^n$, tais que o posto de $(A(\overline{x}))$ é completo, então existe uma vizinhança de $\overline{x}$ onde $A(x)$ tem posto completo para todo ponto nessa vizinhança.*

Demonstração: Suponha que $\forall \varepsilon > 0$, $\exists x$ tal que $\|x - \overline{x}\| < \varepsilon$ e posto$(A(x)) < m$. Para $k \in \mathbb{N}^*$, defina x^k como o ponto que satisfaz $\|x^k - \overline{x}\| < 1/k$, posto$(A(x^k)) < m$. Daí, existe $y^k \in \mathbb{R}^m$ tal que

$$A(x^k)^T y^k = 0,$$

e $\|y^k\| = 1$. Como o conjunto $\{y \in \mathbb{R}^m : \|y\| = 1\}$ é compacto, existe um subconjunto infinito $\mathcal{K} \subset \mathbb{N}$ tal que $\{y^k\}_{k\in\mathcal{K}}$ é convergente. Defina $\overline{y}$ como o limite dessa subsequência. Trivialmente, $\|\overline{y}\| = 1$, e $x^k \to \overline{x}$. Como A é contínua, temos $A(x^k) \to A(\overline{x})$. Portanto

$$\lim_{k\in\mathcal{K}} A(x^k)^T y^k = A(\overline{x})^T \overline{y}.$$

Mas $A(x^k)^T y^k = 0$ para todo $k \in \mathcal{K}$, de modo que o limite é 0. Como o limite é único, temos $A(\overline{x})^T \overline{y} = 0$, e $\overline{y} \neq 0$, logo o posto de $A(\overline{x})$ é menor que m. Absurdo. Portanto existe uma vizinhança de $\overline{x}$, onde o posto de A é completo.

■

Teorema A.2. *Seja $U \subset \mathbb{R}^n$ um aberto, $F : U \to \mathbb{R}^m$ diferenciável em $\overline{x} \in U$. Se $A = F'(\overline{x}) \in \mathbb{R}^{m\times n}$ tem posto coluna completo, então existem $\delta > 0$ e $c > 0$ tais que se $\|h\| < \delta$, então $\overline{x} + h \in U$ e*

$$\|F(\overline{x} + h) - F(\overline{x})\| \geq c\|h\|.$$

Demonstração: Considere a decomposição em valores singulares $A = U\Sigma V^T$, onde $U \in \mathbb{R}^{m\times m}$ e $V \in \mathbb{R}^{n\times n}$ são matrizes ortogonais e $\Sigma \in \mathbb{R}^{m\times n}$ é dada por

$$\Sigma = \begin{bmatrix} \text{diag}(\sigma_1, \ldots, \sigma_n) \\ 0 \end{bmatrix},$$

com $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$. Como A tem posto coluna completo, então $\sigma_i > 0$. Para qualquer $x \in \mathbb{R}^n$, existe $\alpha \in \mathbb{R}^n$ tal que $x = V\alpha$. Então

$$\begin{aligned} \|Ax\| = \|U\Sigma V^T V\alpha\| &= \|\Sigma\alpha\| = \Big[\sum_{i=1}^{n}(\alpha_i\sigma_i)^2\Big]^{1/2} \\ &\geq \Big[\sum_{i=1}^{n}(\alpha_i\sigma_n)^2\Big]^{1/2} = \sigma_n\Big[\sum_{i=1}^{n}\alpha_i^2\Big]^{1/2} = \sigma_n\|\alpha\| \\ &= \sigma_n\|V^T x\| = \sigma_n\|x\|. \end{aligned}$$

Como F é diferenciável em $\overline{x} \in U$, então existe $\delta > 0$ tal que se $\|h\| \leq \delta$, então

$$F(\overline{x} + h) = F(\overline{x}) + F'(\overline{x})h + r(h),$$

com $\|r(h)\| \leq \sigma_n\|h\|/2$. Daí, usando a desigualdade $\|a + b\| \geq \|a\| - \|b\|$,

$$\begin{aligned} \|F(\overline{x} + h) - F(\overline{x})\| &= \|F'(\overline{x})h + r(h)\| \\ &\geq \|F'(\overline{x})h\| - \|r(h)\| \\ &\geq \sigma_n\|h\| - \frac{1}{2}\sigma_n\|h\| \\ &= \frac{\sigma_n}{2}\|h\|. \end{aligned}$$

Definindo $c = \sigma_n/2$, obtemos o resultado desejado.

■

Corolário A.3. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ continuamente diferenciável até segunda ordem, e $\overline{x} \in \mathbb{R}^n$ um ponto estacionário estrito para f, isto é, um ponto onde $\nabla f(\overline{x}) = 0$, e $\nabla^2 f(\overline{x})$ é não-singular. Então, numa vizinhança de $\overline{x}$,*

$$\|x - \overline{x}\| = \Theta(\|\nabla f(x)\|),$$

isto é, $\|\nabla f(x)\|$ é uma medida para a distância do ponto à solução.

Demonstração: Como f é continuamente diferenciável até segunda ordem, então ∇f é Lipschiptz, de modo que

$$\|\nabla f(x) - \nabla f(\overline{x})\| = \mathcal{O}(\|x - \overline{x}\|).$$

Agora, aplicando o teorema anterior na função ∇f, existe uma vizinhança V de x e uma constante $c > 0$, tal que

$$\|\nabla f(x) - \nabla f(\overline{x})\| \geq c\|x - \overline{x}\|.$$

Como $\nabla f(\overline{x}) = 0$, temos

$$\|x - \overline{x}\| = \Theta(\|\nabla f(x)\|).$$

■

Apêndice B

Arquivos

Apresentamos aqui os arquivos de interface para o DCICPP. Estes arquivos também estão em C++. O primeiro arquivo mostra a implementação da interface CUTEr. O usuário não precisa modificar este arquivo para utizar o programa. Os outros dois arquivos mostram exemplos de implementação de algum problema específico. O primeiro exemplo é de um problema irrestrito, e o segundo exemplo de um problema com restrições. O usuário que tiver algum problema específico para utilizar um dos exemplos dados como base para implementação de seus testes.

B.1 Interface do CUTEr

sources/cuter_interface.cpp

```
1  #include <iostream>
2  #include "dci.h"
3  extern "C" {
4  #include "cuter.h"
5  }
6
7  using namespace DCI;
8
9  int MAINENTRY () {
10   DCI::Interface dci;
```

```
  char fname[10] = "OUTSDIF.d";
  Int nvar = 0, ncon = 0, amax = 0;
  Int nmax, mmax;
  Int funit = 42, ierr = 0, fout = 6;

  FORTRAN_OPEN ((&funit), fname, (&ierr));
  CDIMEN ((&funit), (&nvar), (&ncon));

  dci.Cuter ();

  Real x[nvar], bl[nvar], bu[nvar];
  nmax = nvar;

  if (ncon == 0) {
    USETUP ((&funit), (&fout), (&nvar), x, bl, bu, (&nmax));

    dci.set_uofg (UOFG);
    dci.set_uprod (UPROD);
    dci.set_ufn (UFN);
    dci.set_unames (UNAMES);

  } else {
    mmax = ncon;
    Real y[ncon], cl[ncon], cu[ncon];
    Bool equatn[ncon], linear[ncon];
    Bool efirst = 0, lfirst = 0, nvfirst = 0;

    CSETUP ((&funit), (&fout), (&nvar), (&ncon), x, bl, bu, (&nmax), equatn,
     linear, y, cl, cu, (&mmax), (&efirst), (&lfirst), (&nvfirst));
    CDIMSJ ((&amax));

    dci.set_cofg (COFG);
    dci.set_cprod (CPROD);
    dci.set_cfn (CFN);
```

```
44      dci.set_ccfsg (CCFSG);
45      dci.set_ccifg (CCIFG);
46      dci.set_cnames (CNAMES);
47
48      dci.set_lambda (ncon, y);
49      dci.set_cl (ncon, cl);
50      dci.set_cu (ncon, cu);
51      dci.set_equatn (ncon, equatn);
52      dci.set_linear (ncon, linear);
53      dci.set_amax (amax);
54    }
55
56    dci.set_x (nvar, x);
57    dci.set_bl (nvar, bl);
58    dci.set_bu (nvar, bu);
59
60    dci.start ();
61    try {
62      dci.solve ();
63      dci.show ();
64      dci.printLatex ();
65    } catch (const char * ex) {
66      std::cout << ex << std::endl;
67    } catch (...) {
68      std::cout << "Unhandled␣exception␣caught" << std::endl;
69    }
70
71    real calls[7], time[2];
72    CREPRT(calls, time);
73    std::cout << "Setup␣time:␣" << time[0] << std::endl
74              << "Execution␣time:␣" << time[1] << std::endl;
75    FORTRAN_CLOSE ((&funit), (&ierr));
76
77    return 0;
```

```
78
79 }
```

B.2 Exemplos

O exemplo a seguir é para o problema irrestrito

$$\min f(x) = \sum_{i=1}^{n} (x_i - 1)^4,$$

com $n = 5$.

sources/unc_example.cpp

```
1  #include "dci.h"
2
3  using namespace DCI;
4
5  void UFN (Int * n, Real * x, Real * f) {
6    Real xi = 0, xi2 = 0;
7    *f = 0;
8    for (Int i = 0; i < *n; i++) {
9      xi = x[i] - 1;
10     xi2 = xi*xi;
11     *f += xi2*xi2;
12   }
13 }
14
15 void UOFG (Int * n, Real * x, Real * f, Real * g, Bool * grad) {
16   Real xi = 0, xi2 = 0;
17   *f = 0;
18   for (Int i = 0; i < *n; i++) {
19     xi = x[i] - 1;
20     xi2 = xi*xi;
21     *f += xi2*xi2;
22     if (*grad == dciTrue)
```

```
23        g[i] = 4*xi*xi2;
24    }
25  }
26
27  void UPROD (Int * n, Bool *, Real * x, Real * p, Real * q) {
28    Real xi = 0;
29    for (Int i = 0; i < *n; i++) {
30      xi = x[i] - 1;
31      q[i] = 12*xi*xi*p[i];
32    }
33  }
34
35  int main () {
36    Int n = 5;
37    DCI::Interface dci;
38    Real x[n], bl[n], bu[n];
39
40    dci.set_uofg (UOFG);
41    dci.set_uprod (UPROD);
42    dci.set_ufn (UFN);
43
44    for (Int i = 0; i < n; i++) {
45      x[i] = -1;
46      bl[i] = -dciInf;
47      bu[i] = dciInf;
48    }
49
50    dci.set_x (n, x);
51    dci.set_bl (n, bl);
52    dci.set_bu (n, bu);
53
54    dci.start ();
55    dci.solve ();
56    dci.show();
```

```
57
58 }
```

O exemplo a seguir é para o problema restrito

$$\begin{aligned} \min \quad & f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ \text{suj. a} \quad & x_1 x_2 \geq 1, \\ & x_1 + x_2^2 \geq 0. \\ & x_1 \leq 0.5, \end{aligned}$$

Esse é o problema de número 15 da coleção de problemas de Hock-Schittkowski [30].

sources/con_example.cpp

```
1  #include <iostream>
2  #include <cmath>
3  #include "dci.h"
4
5  using namespace DCI;
6
7  void CFN (Int * n, Int * m, Real * x, Real * f, Int * mmax, Real * c) {
8    if ( (*n < *m) || (*m != 2) || (*mmax < *m) )
9      throw("Error");
10   Real x1 = x[0], x2 = x[1];
11   Real a1 = x2 - x1*x1, a2 = 1 - x1;
12   *f = 100 * a1 * a1 + a2 * a2;
13   c[0] = x1 * x2 - 1;
14   c[1] = x1 + x2*x2;
15 }
16
17 void COFG (Int * n, Real * x, Real * f, Real * g, Bool * grad) {
18   if (*n != 2)
19     return;
20   Real x1 = x[0], x2 = x[1];
21   Real a1 = x2 - x1*x1, a2 = 1 - x1;
22   *f = 100 * a1 * a1 + a2 * a2;
```

```
23    if (*grad == dciTrue) {
24      g[0] = -400*x1*a1 - 2*a2;
25      g[1] = 200 * a1;
26    }
27  }
28
29  void CPROD (Int * n, Int * m, Bool * , Real * x, Int * mmax, Real * y, Real * p
        , Real * q) {
30    if ( (*n != 2) || (*m != 2) || (*mmax < *m) )
31      return;
32    Real x1 = x[0], x2 = x[1];
33    Real y1 = y[0], y2 = y[1];
34    q[0] = (1200*x1*x1 - 400*x2 + 2) * p[0] + (-400*x1 + y1) * p[1];
35    q[1] = (-400*x1 + y1) * p[0] + (200 + 2*y2) * p[1];
36  }
37
38  void CCFSG (Int * , Int * , Real * x, Int * , Real * c, Int * nnzJ, Int * ,
        Real * J, Int * indvar, Int * indfun, Bool * Grad) {
39    Real x1 = x[0], x2 = x[1];
40    c[0] = x1 * x2 - 1;
41    c[1] = x1 + x2*x2;
42    if (*Grad == dciFalse)
43      return;
44    if (x1 == 0) {
45      if (x2 == 0) {
46        *nnzJ = 0;
47      } else {
48        *nnzJ = 2;
49        indvar[0] = 0;
50        indfun[0] = 0;
51        J[0] = x2;
52        indvar[1] = 1;
53        indfun[1] = 1;
54        J[1] = 2*x2;
```

```
55      }
56    } else {
57      if (x2 == 0) {
58        *nnzJ = 1;
59        indvar[0] = 1;
60        indfun[0] = 0;
61        J[0] = x1;
62      } else {
63        *nnzJ = 3;
64        indvar[0] = 0;
65        indfun[0] = 0;
66        J[0] = x2;
67        indvar[1] = 1;
68        indfun[1] = 0;
69        J[1] = x1;
70        indvar[2] = 1;
71        indfun[2] = 1;
72        J[2] = 2*x2;
73      }
74    }
75    Int k = *nnzJ;
76    indvar[k] = 0;
77    indfun[k] = 1;
78    J[k] = 1;
79    *nnzJ = k + 1;
80  }
81
82  int main () {
83    Int n = 2, m = 2;
84    DCI::Interface dci;
85    Real x[n], bl[n], bu[n];
86    Real cl[m], cu[m];
87    Bool equatn[m];
88
```

```
89    dci.set_cofg (COFG);
90    dci.set_cprod (CPROD);
91    dci.set_cfn (CFN);
92    dci.set_ccfsg (CCFSG);
93
94    x[0] = -2;
95    x[1] = 1;
96
97    bl[0] = -dciInf;
98    bl[1] = -dciInf;
99    bu[0] = 0.5;
100   bu[1] = dciInf;
101
102   cl[0] = 0;
103   cl[1] = 0;
104   cu[0] = dciInf;
105   cu[1] = dciInf;
106
107   equatn[0] = dciFalse;
108   equatn[1] = dciFalse;
109
110   dci.set_x (n, x);
111   dci.set_bl (n, bl);
112   dci.set_bu (n, bu);
113   dci.set_cl (m, cl);
114   dci.set_cu (m, cu);
115   dci.set_equatn (m, equatn);
116
117   dci.start ();
118   dci.solve ();
119   dci.show();
120
121 }
```

Apêndice C

Tabela de Resultados do CUTEr

A seguir os resultados do método DCICPP nos problemas do CUTEr. As colunas são

- **Nome:** Nome do problema;
- **N:** Número de Variáveis;
- **ME:** Número de Restrições de Igualdade;
- **MI:** Número de Restrições de Desigualdade;
- **ExitFlag:** Saída do Algoritmo, podendo ser
  - **Convergiu:** Algoritmo convergiu para um ponto estacionário;
  - **Infactível:** Algoritmo convergiu para um ponto estacionário da infactibilidade;
  - **MaxIter:** Algoritmo chegou ao máximo de iterações;
  - **MaxRest:** Algoritmo chegou ao máximo de restaurações;
  - **Ilimitado:** Algoritmo encontrou um ponto de norma muito grande;
  - **MaxTempo:** Algoritmo chegou ao máximo de tempo de execução;
  - **Falha:** Aconteceu alguma falha não reconhecida;
  - **Rhomax:** $\rho_{\max}$ ficou muito pequeno;

- $f(x)$**:** Valor de função encontrado;
- R_p**:** Resíduo primal;
- R_d**:** Resíduo dual;
- **Iter:** Número de iterações feitas;
- **Tempo:** Tempo gasto pelo algoritmo.

Nome	**N**	**ME**	**MI**	**ExitFlag**	$f(x)$	R_p	R_d	**Iter**	**Tempo**
3PK	30	0	0	Convergiu	1.72012	0	7.47015e-08	26	0.00
AGG	163	36	452	Rhomax	-3.29627e+07	9.64169e-08	8.53428e-06	2524	10.93
AIRPORT	84	0	42	Convergiu	47952.7	2.42671e-07	7.67234e-08	56	0.09
AKIVA	2	0	0	Convergiu	6.16604	0	9.42582e-07	6	0.00
ALJAZZAF	1000	1	0	Convergiu	37438.8	1.23769e-11	2.64274e-13	4	4.01
ALLINITU	4	0	0	Convergiu	5.74438	0	3.54415e-07	7	0.00
ALSOTAME	2	1	0	Convergiu	0.082085	1.27764e-12	1.77032e-12	14	0.00
ANTWERP	27	8	2	Convergiu	28170.7	1.80476e-08	9.70558e-07	9843	0.71
ARGAUSS	3	15	0	Infactível	0	0.00197183	0	1	1.79
ARGLALE	200	400	0	MaxTempo	0	14.1422	0	1	7200.11
ARGLBLE	200	400	0	Ilimitado	0	2.17035e+12	0	1	7200.20
ARGLCLE	200	399	0	Ilimitado	-1	3.62138e+13	0	1	7200.12
ARGLINA	200	0	0	Convergiu	200	0	5.06185e-16	2	0.00
ARGLINB	200	0	0	Convergiu	99.6255	0	7.4591e-07	2	0.00
ARGLINC	200	0	0	Convergiu	101.125	0	3.64665e-07	2	0.00
ARGTRIG	200	200	0	Convergiu	0	8.15175e-10	0	1	0.12
ARWHDNE	500	998	0	MaxTempo	0	11.808	0	1	7200.01
ARWHEAD	5000	0	0	Convergiu	3.33e-12	0	7.90532e-09	6	0.02
AUG2D	20200	10000	0	Convergiu	1.68741e+06	1.22434e-10	9.47406e-08	3	0.14
AUG2DC	20200	10000	0	Convergiu	1.81837e+06	5.82405e-11	8.10897e-15	2	0.09
AUG2DCQP	20200	10000	0	Convergiu	6.50218e+06	9.45931e-07	8.60115e-07	275	99.75
AUG2DQP	20200	10000	0	Convergiu	6.23889e+06	3.55031e-07	1.79443e-07	626	536.43
AUG3D	27543	8000	0	Convergiu	24561.5	1.1489e-11	4.38324e-08	3	1.08
AUG3DC	27543	8000	0	Convergiu	27654.1	6.56442e-12	4.841e-16	2	0.83
AUG3DCQP	27543	8000	0	Convergiu	61608.6	8.51406e-10	8.52752e-07	19	13.24
AUG3DQP	27543	8000	0	Convergiu	54273	5.87076e-10	1.07354e-07	18	13.14
AVGASA	8	0	10	Convergiu	-4.63193	1.52915e-14	9.99058e-08	24	0.00
AVGASB	8	0	10	Convergiu	-4.48322	2.59849e-14	3.09419e-09	12	0.00
AVION2	49	15	0	Convergiu	0	1.18293e-12	0	2	0.02
BARD	3	0	0	Convergiu	0.00821488	0	6.3605e-07	10	0.00

BATCH	48	12	61	Convergiu	259180	3.89541e-09	5.6016e-07	17	0.01
BDEXP	5000	0	0	Convergiu	0	0	0	2	0.01
BDQRTIC	5000	0	0	Convergiu	20006.3	0	4.28262e-07	8	0.04
BEALE	2	0	0	Convergiu	2.59247e-14	0	2.90499e-08	8	0.00
BIGGS6	6	0	0	Convergiu	0.00565565	0	2.81316e-08	19	0.00
BIGGSB1	5000	0	0	Convergiu	0.0678635	0	4.18146e-07	5	0.13
BIGGSC4	4	0	7	Convergiu	-3.25	1.79171e-13	2.48228e-09	7	0.00
BLOCKQP1	10010	5000	1	Convergiu	4.9998	8.13442e-07	8.04389e-09	3	111.61
BLOCKQP2	10010	5000	1	Convergiu	-4993.79	4.44306e-10	9.69997e-07	11	336.94
BLOCKQP3	10010	5000	1	Convergiu	4.9997	3.08062e-07	1.13731e-08	3	111.32
BLOCKQP4	10010	5000	1	Convergiu	-2495.54	6.84957e-07	5.7999e-07	16	512.76
BLOCKQP5	10010	5000	1	Convergiu	4.99982	2.36273e-11	1.18307e-10	3	114.33
BLOWEYA	4002	2002	0	Convergiu	-4.46248e-05	5.57239e-09	4.44569e-09	2	0.03
BLOWEYB	4002	2002	0	Convergiu	-2.22029e-05	4.36871e-09	5.36171e-09	2	0.03
BLOWEYC	4002	2002	0	Convergiu	-9.5326e-05	6.63104e-09	4.58556e-09	2	0.01
BOOTH	2	2	0	Convergiu	0	9.93014e-16	0	1	0.00
BOX	10000	0	0	Convergiu	-1864.54	0	3.70035e-09	7	0.07
BOX3	3	0	0	Convergiu	7.30022e-15	0	7.59708e-09	9	0.00
BQP1VAR	1	0	0	Convergiu	2.5e-07	0	1.56776e-07	2	0.00
BQPGASIM	50	0	0	Convergiu	-2.35317e-05	0	5.27892e-08	5	0.00
BRITGAS	450	360	0	Convergiu	0.000214303	3.30636e-07	4.50111e-07	13	0.05
BRKMCC	2	0	0	Convergiu	0.169043	0	2.52555e-07	3	0.00
BROWNAL	200	0	0	Convergiu	2.03747e-08	0	2.77564e-08	2	0.00
BROWNALE	200	200	0	Convergiu	0	2.5776e-08	0	1	0.78
BROWNBS	2	0	0	Convergiu	0	0	0	5	0.00
BROWNDEN	4	0	0	Convergiu	85822.2	0	3.92735e-09	8	0.00
BROYDN3D	5000	5000	0	Convergiu	0	1.07411e-09	0	1	0.02
BROYDN7D	5000	0	0	Convergiu	1865.52	0	9.17946e-07	733	8.97
BROYDNBD	5000	5000	0	Convergiu	0	2.54219e-11	0	1	0.05
BRYBND	5000	0	0	Convergiu	0.000218773	0	6.88962e-07	18	0.52
BT10	2	2	0	Convergiu	-1	4.4039e-09	7.4476e-16	1	0.00
BT11	5	3	0	Convergiu	0.824892	7.90019e-08	1.24242e-08	13	0.00
BT1	2	1	0	Convergiu	-0.999998	2.22344e-08	2.58906e-08	5	0.00
BT12	5	3	0	Convergiu	6.18812	9.16573e-13	5.50411e-07	5	0.00
BT13	5	1	0	Convergiu	3.05714e-08	9.61006e-07	1.34085e-10	3	0.00
BT2	3	1	0	Convergiu	0.0325682	1.30145e-11	2.73113e-07	25	0.00
BT3	5	3	0	Convergiu	4.09302	1.86425e-14	1.35749e-18	2	0.00
BT4	3	2	0	Convergiu	-45.5106	3.0731e-13	7.26911e-09	5	0.00
BT5	3	2	0	Convergiu	961.715	9.83436e-10	1.36569e-08	3	0.00
BT6	5	2	0	Convergiu	0.277045	6.78825e-07	3.01513e-10	12	0.00
BT7	5	3	0	Convergiu	306.5	3.05572e-11	2.18911e-15	9	0.00

BT8	5	2	0	Convergiu	1	3.37175e-07	3.64327e-14	2	0.00
BT9	4	2	0	Convergiu	-1	1.02804e-10	1.62574e-13	6	0.00
BURKEHAN	1	0	1	Infactível	-5.22839e-08	1.0108	1.41421	4	0.00
BYRDSPHR	3	2	0	Convergiu	-4.6833	7.41398e-07	2.54384e-16	3	0.00
CAMEL6	2	0	0	Convergiu	-0.215464	0	8.52108e-08	7	0.00
CAMSHAPE	800	0	1603	Convergiu	-4.18365	7.39148e-07	4.04416e-07	50	0.67
CANTILVR	5	0	1	Convergiu	1.33996	1.13365e-08	5.81822e-08	10	0.00
CB2	3	0	3	Convergiu	1.95222	5.38438e-09	2.58908e-11	255	0.01
CB3	3	0	3	Convergiu	2	2.87239e-09	1.39058e-10	39	0.00
CHACONN1	3	0	3	Convergiu	1.95222	7.93682e-09	9.69263e-07	165	0.00
CHACONN2	3	0	3	Convergiu	2	4.09756e-09	1.44774e-10	29	0.00
CHAINWOO	4000	0	0	Convergiu	15747.2	0	7.96126e-07	9	0.28
CHANDHEQ	100	100	0	Convergiu	0	7.97667e-07	0	1	0.04
CHANDHEU	500	500	0	Convergiu	0	4.46812e-07	0	1	3.61
CHARDIS0	-	-	-	Falha	-	-	-	-	-
CHEBYQAD	100	0	0	Convergiu	0.00947023	0	7.57772e-07	38	7.83
CHEMRCTA	5000	5000	0	Convergiu	0	7.39325e-11	0	2	0.14
CHEMRCTB	5000	5000	0	Convergiu	0	1.53873e-09	0	2	0.13
CHENHARK	5000	0	0	Convergiu	-0.861093	0	9.3214e-07	98	2.70
CHNROSNB	50	0	0	Convergiu	1.35768e-08	0	7.79588e-08	64	0.01
CHNRSBNE	50	98	0	Convergiu	0	3.50382e-10	0	1	0.00
CLIFF	2	0	0	Convergiu	0.208512	0	9.1839e-07	24	0.00
CLUSTER	2	2	0	Convergiu	0	7.74787e-08	0	1	0.00
CONCON	15	11	0	Convergiu	-6230.8	3.27104e-08	7.807e-08	2	0.00
CONGIGMZ	3	0	5	Convergiu	28	7.30173e-09	1.62452e-09	8	0.00
CONT6-QQ	20002	10197	0	MaxTempo	-4.29323	9.72718e-05	1.49656e-07	2	7200.14
COOLHANS	9	9	0	Convergiu	0	8.24414e-07	0	1	0.00
CORE1	65	41	18	Convergiu	91.0564	1.59896e-07	6.56849e-07	15	0.02
CORE2	157	108	26	Convergiu	99.0305	4.42547e-08	2.23971e-07	11	1.65
COSHFUN	6001	0	2000	Convergiu	-0.748945	3.95537e-09	9.32375e-07	92	398.84
COSINE	10000	0	0	Rhomax	-9999	0	0.70056	65	1.43
CRAGGLVY	5000	0	0	Convergiu	1688.22	0	7.32934e-08	12	0.37
C-RELOAD	342	200	84	Convergiu	-1.00835	8.37865e-08	9.79576e-07	157	0.77
CRESC100	6	0	200	MaxRest	0.000737793	19171.1	0.835911	2	745.21
CRESC4	6	0	8	Convergiu	1.95248	2.94112e-07	9.57998e-07	2114	0.27
CRESC50	6	0	100	Convergiu	0.898068	6.19618e-08	8.17944e-07	7	1.29
CSFI1	5	2	2	Convergiu	-49.0752	1.49067e-07	7.01527e-08	52	0.00
CSFI2	5	2	2	Convergiu	55.0176	1.0204e-16	2.68015e-07	4	0.00
CUBE	2	0	0	Convergiu	4.3085e-09	0	4.60956e-07	29	0.00
CUBENE	2	2	0	Convergiu	0	1.99951e-14	0	1	0.00
CURLY10	10000	0	0	Convergiu	-1.00316e+06	0	9.57488e-07	11	76.66

CURLY20	10000	0	0	Convergiu	-1.00316e+06	0	9.85965e-08	12	143.99
CURLY30	10000	0	0	Convergiu	-1.00316e+06	0	1.30077e-07	14	223.68
CVXBQP1	10000	0	0	Convergiu	2.25027e+06	0	5.79522e-07	20	0.24
CVXQP1	10000	5000	0	Convergiu	1.08705e+08	6.25783e-07	1.63064e-07	38	49.64
CVXQP2	10000	2500	0	Convergiu	8.18429e+07	2.57441e-07	2.51514e-07	51	15.80
CVXQP3	10000	7500	0	Rhomax	1.15711e+08	9.63131e-07	2.94122e-06	423	44.29
DALLASM	196	151	0	Convergiu	86384.3	2.31423e-08	4.32915e-07	40	0.16
DALLASS	46	31	0	Convergiu	-18636.7	2.38363e-11	5.31335e-07	13	0.00
DECONVB	63	0	0	Convergiu	6.9413	0	2.61823e-07	23	0.01
DECONVNE	63	40	0	Convergiu	0	4.39703e-10	0	1	0.00
DECONVU	63	0	0	Convergiu	5.37634e-07	0	2.67869e-07	18	0.01
DEGDIAG	100001	0	0	Convergiu	16683.7	0	3.20593e-07	18	2.45
DEGENLPA	20	15	0	Convergiu	15.4177	2.05379e-07	2.95175e-07	3	0.00
DEGENLPB	20	15	0	Convergiu	-15.4177	5.14207e-07	3.50071e-07	9	0.01
DEGENQP	-	-	-	Falha	-	-	-	-	-
DEGTRID	100001	0	0	Convergiu	-99999.5	0	9.12763e-08	14	3.27
DEGTRID2	100001	0	0	Convergiu	-99999.5	0	7.17255e-08	17	2.61
DEGTRIDL	100001	1	0	Convergiu	0.50035	1.05133e-08	7.39698e-08	2	0.31
DEMBO7	16	0	20	Convergiu	174.911	8.90608e-07	2.05621e-07	13	0.00
DEMYMALO	3	0	3	Convergiu	-3	6.68809e-10	5.07627e-07	30	0.00
DENSCHNA	2	0	0	Convergiu	2.21391e-12	0	1.90769e-07	6	0.00
DENSCHNB	2	0	0	Convergiu	1.69695e-17	0	1.61392e-09	6	0.00
DENSCHNC	2	0	0	Convergiu	5.33959e-11	0	2.55961e-08	10	0.00
DENSCHND	3	0	0	Convergiu	0.0380005	0	5.37204e-07	16	0.00
DENSCHNE	3	0	0	Convergiu	2.78336e-09	0	6.27921e-07	16	0.00
DENSCHNF	2	0	0	Convergiu	2.31489e-10	0	4.06836e-07	6	0.00
DIPIGRI	7	0	4	Convergiu	680.63	3.07506e-10	1.17343e-08	63	0.00
DISC2	29	17	6	Convergiu	1.56251	1.63914e-07	1.6942e-07	23	0.00
DITTERT	1133	1034	0	Convergiu	-1.99939	2.14571e-10	3.51962e-07	2	0.33
DIXCHLNG	10	5	0	Convergiu	2471.9	9.6767e-13	9.09042e-07	8	0.00
DIXCHLNV	1000	500	0	Convergiu	3.3036e-06	1.80813e-10	9.57353e-08	45	19.31
DIXMAANA	3000	0	0	Convergiu	1.00001	0	2.55892e-07	6	0.03
DIXMAANB	3000	0	0	Convergiu	1	0	6.29257e-09	7	0.04
DIXMAANC	3000	0	0	Convergiu	1.00001	0	5.7961e-08	8	0.05
DIXMAAND	3000	0	0	Convergiu	1.00003	0	6.53423e-08	14	0.06
DIXMAANE	3000	0	0	Convergiu	1.00011	0	9.60072e-08	9	0.12
DIXMAANF	3000	0	0	Convergiu	1.00058	0	1.99383e-07	18	0.15
DIXMAANG	3000	0	0	Convergiu	1.00912	0	7.18725e-07	20	0.13
DIXMAANH	3000	0	0	Convergiu	1.07279	0	5.62317e-07	21	0.12
DIXMAANI	3000	0	0	Convergiu	1.00154	0	4.94732e-07	10	0.27
DIXMAANJ	3000	0	0	Convergiu	1.0019	0	3.18754e-07	17	0.12

DIXMAANK	15	0	0	Convergiu	1	0	3.01402e-07	14	0.00
DIXMAANL	3000	0	0	Convergiu	1.00246	0	1.23297e-07	18	0.12
DIXON3DQ	10000	0	0	Convergiu	1.0959e-23	0	1.65085e-12	3	11.02
DJTL	2	0	0	Convergiu	-8951.54	0	1.89061e-08	215	0.00
DQDRTIC	5000	0	0	Convergiu	8.13283e-06	0	7.98881e-08	3	0.01
DQRTIC	5000	0	0	Convergiu	1.55456	0	8.30609e-07	26	0.21
DUALC1	9	1	214	Convergiu	138572	4.09234e-09	4.79998e-07	25	0.20
DUALC2	7	1	228	Convergiu	4713.61	9.06246e-09	8.00111e-09	7	0.05
DUALC5	8	1	277	Convergiu	545.732	3.58037e-09	4.6539e-08	5	0.08
EDENSCH	2000	0	0	Convergiu	12003.3	0	4.43123e-08	11	0.03
EG1	3	0	0	Convergiu	-1.1328	0	2.91704e-10	9	0.00
EG2	1000	0	0	Convergiu	-998.947	0	7.08856e-12	4	0.01
EG3	10001	1	19999	MaxTempo	0.13898	51.0774	0.000775257	4	7585.55
EIGENA2	2550	1275	0	Convergiu	1.93468e-28	0	6.88152e-19	2	0.47
EIGENA	2550	2550	0	MaxTempo	0	12.8285	0	1	7201.59
EIGENACO	2550	1275	0	Convergiu	1.93468e-28	0	6.88152e-19	2	0.51
EIGENALS	2550	0	0	Convergiu	0.000246933	0	5.34279e-07	45	16.38
EIGENAU	2550	2550	0	Convergiu	0	7.51688e-09	0	1	10.20
EIGENB2	2550	1275	0	Convergiu	98	0	4.24349e-17	2	0.47
EIGENB	2550	2550	0	Convergiu	0	9.40057e-09	0	1	2393.77
EIGENBCO	2550	1275	0	Convergiu	49	0	2.53752e-16	2	0.51
EIGENBLS	2550	0	0	Convergiu	2.9471e-06	0	6.73327e-07	708	627.13
EIGENC2	2652	1326	0	Convergiu	1.84538e-08	2.54323e-11	1.09805e-07	22	10.47
EIGENC	2652	2652	0	Convergiu	0	1.18561e-08	0	1	45.25
EIGENCCO	2652	1326	0	Convergiu	3.226e-08	1.57914e-12	9.24406e-08	27	19.23
EIGENCLS	2652	0	0	Convergiu	0.000169345	0	9.73472e-07	650	490.00
EIGMAXA	101	101	0	Convergiu	-1	4.6008e-14	5.32907e-15	1	0.06
EIGMAXB	101	101	0	Convergiu	-0.000967435	8.40409e-13	6.48761e-14	1	0.01
EIGMINA	101	101	0	Convergiu	1	4.6008e-14	5.32907e-15	1	0.06
EIGMINB	101	101	0	Convergiu	0.000967435	8.40409e-13	6.48761e-14	1	0.01
ELATTAR	7	0	102	MaxRest	23.5772	7.35117e+20	2.70266e+28	6	299.45
ELEC	600	200	0	Convergiu	18439	1.98034e-12	1.23196e-07	73	2.26
ENGVAL1	5000	0	0	Convergiu	5548.67	0	9.25934e-08	7	0.03
ENGVAL2	3	0	0	Convergiu	2.44766e-10	0	2.78161e-08	17	0.00
ERRINBAR	-	-	-	Falha	-	-	-	-	-
ERRINROS	50	0	0	Convergiu	39.9172	0	2.59588e-07	48	0.01
EXPFIT	2	0	0	Convergiu	0.240511	0	5.19063e-07	9	0.00
EXPFITA	5	0	22	Convergiu	3.42369	1.60804e-07	5.11335e-08	13	0.00
EXPFITB	5	0	102	Convergiu	6.65391	2.89754e-09	4.99106e-07	14	0.01
EXPFITC	5	0	502	Convergiu	0.18191	5.74089e-09	8.75649e-08	6	0.32
EXPLIN	1200	0	0	Convergiu	-7.19255e+07	0	7.59461e-08	35	0.07

EXPLIN2	1200	0	0	Convergiu	-7.19988e+07	0	4.27183e-07	24	0.05
EXPQUAD	1200	0	0	Convergiu	-3.68494e+09	0	1.03545e-07	67	0.16
EXTRASIM	2	1	0	Convergiu	1	2.22045e-16	2.74307e-08	12	0.00
EXTROSNB	1000	0	0	Convergiu	0.0341727	0	5.47772e-07	17	0.02
FCCU	19	8	0	Convergiu	11.1491	1.02346e-12	2.16899e-07	4	0.00
FLETCBV2	5000	0	0	Convergiu	-0.500286	0	1.00047e-07	2	12.36
FLETCBV3	5000	0	0	Ilimitado	-2.40715e+13	0	0.000101962	481	4.02
FLETCHBV	5000	0	0	MaxTempo	-1e+20	0	3756.95	80745	7200.00
FLETCHCR	1000	0	0	Convergiu	2.46758e-11	0	8.15005e-08	1790	4.90
FLETCHER	4	1	3	Convergiu	11.6569	1.47127e-08	8.55298e-07	226	0.01
FLT	2	2	0	Convergiu	0	5.92902e-07	0	2	0.00
FMINSRF2	5625	0	0	Convergiu	1	0	9.71486e-08	240	9.76
FMINSURF	5625	0	0	Convergiu	1	0	9.73079e-07	245	9.79
FREUROTH	5000	0	0	Convergiu	608159	0	1.65219e-07	9	0.06
GENHS28	10	8	0	Convergiu	0.927174	2.67895e-15	7.12346e-17	2	0.00
GENHUMPS	5000	0	0	Convergiu	1.49076	0	6.94188e-07	6126	157.34
GENROSE	500	0	0	Convergiu	1	0	7.23641e-08	680	0.53
GIGOMEZ1	3	0	3	Convergiu	-3	3.17968e-13	2.74221e-09	8	0.00
GIGOMEZ2	3	0	3	Convergiu	2	4.32117e-07	1.94208e-08	5	0.00
GIGOMEZ3	3	0	3	Convergiu	2	3.63637e-09	1.39273e-10	31	0.00
GILBERT	5000	1	0	Convergiu	2459.47	6.29585e-12	2.21761e-08	14	0.09
GOFFIN	51	0	50	Convergiu	0.000160864	2.95864e-09	7.84438e-07	9	0.03
GOTTFR	2	2	0	Convergiu	0	1.13922e-09	0	1	0.00
GOULDQP1	32	17	0	Convergiu	-2896.43	3.05713e-08	2.55159e-07	16	0.00
GOULDQP2	19999	9999	0	Convergiu	1.59629e-12	4.04535e-07	4.77942e-19	2	0.54
GOULDQP3	19999	9999	0	Convergiu	4.71247e-05	4.04535e-07	7.86191e-14	2	0.48
GPP	1000	0	1998	Convergiu	231925	6.77779e-14	8.67987e-07	1529	894.16
GRIDNETB	7564	3844	0	Convergiu	127.615	1.40172e-09	1.64077e-07	4	0.16
GRIDNETC	7564	3844	0	Convergiu	164.49	1.262e-07	9.19613e-07	44	1.41
GRIDNETE	7564	3844	0	Convergiu	206.481	2.16051e-09	9.62923e-08	4	0.16
GRIDNETF	7564	3844	0	Convergiu	246.22	1.60644e-07	5.72643e-07	45	2.37
GRIDNETH	7564	3844	0	Convergiu	206.481	3.29249e-09	9.26979e-08	4	0.18
GRIDNETI	7564	3844	0	Convergiu	246.22	1.48247e-07	5.09299e-07	45	2.35
GROUPING	100	125	0	Convergiu	13.8504	5.62944e-08	1.09884e-09	1	0.00
GROWTH	3	12	0	Infactível	0	293.194	0	1	0.00
GROWTHLS	3	0	0	Convergiu	12.4524	0	9.35396e-08	9	0.00
GULF	3	0	0	Convergiu	2.51439e-09	0	2.41146e-08	20	0.01
HADAMARD	401	210	800	Convergiu	1.20154	3.10007e-09	9.47253e-07	50	20.97
HAIFAS	13	0	9	Convergiu	-0.45	1.1403e-07	3.95627e-08	13	0.00
HAIRY	2	0	0	Convergiu	20	0	6.56523e-17	31	0.00
HALDMADS	6	0	42	Convergiu	2.5065	0	4.48333e-07	39	0.03

HART6	6	0	0	Convergiu	-3.11018	0	5.91952e-07	10	0.00
HATFLDA	4	0	0	Convergiu	2.07455e-12	0	2.55734e-07	6	0.00
HATFLDB	4	0	0	Convergiu	0.00557285	0	8.27961e-07	6	0.00
HATFLDC	25	0	0	Convergiu	1.27452e-14	0	6.01891e-08	5	0.00
HATFLDD	3	0	0	Convergiu	6.61827e-08	0	9.27485e-07	18	0.00
HATFLDE	3	0	0	Convergiu	5.12048e-07	0	3.34931e-07	17	0.00
HATFLDF	3	3	0	Convergiu	0	2.28154e-10	0	1	0.00
HATFLDFL	3	0	0	Convergiu	6.24417e-05	0	9.83927e-07	57	0.00
HATFLDG	25	25	0	Convergiu	0	5.57428e-08	0	1	0.00
HATFLDH	4	0	7	Convergiu	-24.5	3.2073e-14	5.18443e-07	8	0.00
HEART6	6	6	0	Convergiu	0	3.97541e-13	0	1	0.00
HEART6LS	6	0	0	Convergiu	3.5753e-05	0	7.03648e-07	1255	0.02
HEART8	8	8	0	Convergiu	0	3.10148e-07	0	1	0.00
HEART8LS	8	0	0	Convergiu	3.93455e-11	0	3.13574e-08	111	0.00
HELIX	3	0	0	Convergiu	1.38783e-09	0	1.30069e-07	12	0.00
HET-Z	2	0	1002	Convergiu	1.0001	1.05768e-11	1.85582e-08	8	1.72
HILBERTA	2	0	0	Convergiu	1.7092e-30	0	1.83275e-16	2	0.00
HILBERTB	10	0	0	Convergiu	1.11466e-10	0	9.86962e-08	3	0.00
HIMMELBA	2	2	0	Convergiu	0	0	0	1	0.00
HIMMELBB	2	0	0	Convergiu	9.7388e-09	0	6.34651e-08	6	0.00
HIMMELBC	2	2	0	Convergiu	0	6.47625e-08	0	1	0.00
HIMMELBD	2	2	0	MaxRest	0	4.91214e+13	0	1	1.75
HIMMELBE	3	3	0	Convergiu	0	2.22045e-16	0	1	0.00
HIMMELBF	4	0	0	Convergiu	318.58	0	3.16466e-08	21	0.00
HIMMELBG	2	0	0	Convergiu	3.54091e-24	0	6.02437e-12	7	0.00
HIMMELBH	2	0	0	Convergiu	-1	0	3.32554e-13	5	0.00
HIMMELBI	100	0	12	Convergiu	-1729.7	3.83133e-07	9.43326e-08	31	0.02
HIMMELBK	24	14	0	Convergiu	0.0518153	7.81048e-15	2.67679e-07	16	0.01
HIMMELP1	2	0	0	Convergiu	-62.0539	0	7.93138e-09	8	0.00
HIMMELP2	2	0	1	Convergiu	-59.6189	0	1.75223e-09	6	0.00
HIMMELP3	2	0	2	Convergiu	-59.0132	1.34712e-07	4.46792e-11	6	0.00
HIMMELP4	2	0	3	Convergiu	-59.0132	1.42153e-13	2.83094e-09	24	0.00
HIMMELP5	2	0	3	Convergiu	-59.0132	9.44808e-13	1.78837e-09	32	0.00
HIMMELP6	2	0	5	Convergiu	-59.0132	3.13345e-12	3.73227e-08	30	0.00
HONG	4	1	0	Convergiu	22.5711	1.66533e-16	7.92868e-08	6	0.00
HS100	7	0	4	Convergiu	680.63	3.07509e-10	1.17343e-08	63	0.00
HS100LNP	7	2	0	Convergiu	680.63	1.90614e-10	9.04126e-08	8	0.00
HS100MOD	7	0	4	Convergiu	678.68	5.78074e-08	5.02626e-07	41	0.00
HS101	7	0	5	Convergiu	1809.76	3.95783e-07	1.86682e-10	44	0.01
HS10	2	0	1	Convergiu	-1	1.50833e-10	9.72385e-07	13	0.00
HS102	7	0	5	Convergiu	911.881	2.56959e-07	8.62221e-08	34	0.01

HS103	7	0	5	Convergiu	543.668	0	4.26751e-07	34	0.01
HS104	8	0	5	Convergiu	3.95116	8.16427e-07	5.2906e-08	130	0.01
HS105	8	0	1	Convergiu	1062.05	2.22045e-16	2.25846e-08	10	0.01
HS106	8	0	6	Convergiu	7049.25	1.32619e-10	3.93518e-08	13	0.00
HS107	9	6	0	Convergiu	5055.01	3.09508e-07	7.9106e-07	7	0.00
HS108	9	0	13	Convergiu	-0.866013	2.76436e-07	5.44944e-07	41	0.00
HS109	9	6	4	Convergiu	5362.07	7.75417e-07	9.95822e-07	21222	3.17
HS110	10	0	0	Convergiu	-45.7785	0	6.18504e-07	14	0.00
HS111	10	3	0	Convergiu	-47.7611	2.8688e-09	2.69516e-08	16	0.00
HS111LNP	10	3	0	Convergiu	-47.7611	9.59317e-07	6.57811e-08	11	0.00
HS11	2	0	1	Convergiu	-8.49846	1.45319e-12	3.61689e-07	7	0.00
HS112	10	3	0	Convergiu	-47.759	1.10593e-12	1.04315e-07	17	0.00
HS113	10	0	8	Convergiu	24.3063	9.84009e-11	8.9986e-08	43	0.00
HS114	10	3	8	Convergiu	-1768.81	1.43206e-07	9.15358e-07	58	0.00
HS116	13	0	14	Convergiu	250	0	1.30411e-10	21	0.00
HS117	15	0	5	Convergiu	34.2509	7.13192e-10	5.09409e-07	23	0.00
HS118	15	0	17	Convergiu	664.822	8.5875e-10	8.90219e-07	17	0.00
HS119	16	8	0	Convergiu	2863.46	1.14068e-12	2.5524e-07	7	0.00
HS1	2	0	0	Convergiu	1.60361e-10	0	3.157e-08	44	0.00
HS12	2	0	1	Convergiu	-30	1.01252e-13	3.20523e-08	10	0.00
HS13	2	0	1	Convergiu	0.980653	9.18533e-07	1.14964e-10	24	0.00
HS14	2	1	1	Convergiu	1.39346	1.68611e-08	8.74464e-16	4	0.00
HS15	2	0	2	Convergiu	306.501	0	5.79749e-07	2	0.00
HS16	2	0	2	Convergiu	23.1447	1.75991e-10	1.37359e-10	6	0.00
HS17	2	0	2	Convergiu	1.00004	4.63128e-07	3.02066e-07	41	0.00
HS18	2	0	2	Convergiu	5	8.44355e-08	1.50423e-10	15	0.00
HS19	2	0	2	Convergiu	-6961.81	1.45945e-11	1.06645e-11	16	0.00
HS20	2	0	3	Convergiu	38.1987	1.21993e-08	1.84987e-09	57	0.00
HS21	2	0	1	Convergiu	-99.96	6.03961e-14	6.78522e-12	5	0.00
HS21MOD	7	0	1	Convergiu	-95.96	5.39835e-12	4.06667e-07	5	0.00
HS2	2	0	0	Convergiu	4.94123	0	8.55095e-10	21	0.00
HS22	2	0	2	Convergiu	1	2.5011e-08	2.92819e-11	20	0.00
HS23	2	0	5	Convergiu	2	3.46433e-07	5.42615e-08	34	0.00
HS24	2	0	3	Convergiu	-0.999999	2.39232e-14	2.49168e-07	4	0.00
HS25	3	0	0	Convergiu	0.00224932	0	1.03958e-08	18	0.00
HS26	3	1	0	Convergiu	4.23607e-08	6.31689e-07	7.46995e-07	13	0.00
HS268	5	0	5	Convergiu	6.14818e-10	3.28146e-14	1.99183e-07	11	0.00
HS27	3	1	0	Convergiu	0.04	6.94562e-11	2.27828e-09	17	0.00
HS28	3	1	0	Convergiu	1.93887e-29	4.44089e-15	5.38486e-16	2	0.00
HS29	3	0	1	Convergiu	-22.6274	6.68434e-07	3.81858e-09	13	0.00
HS30	3	0	1	Convergiu	1	9.88046e-07	2.15317e-09	12	0.00

HS31	3	0	1	Convergiu	6	9.88739e-10	1.40676e-10	9	0.00
HS3	2	0	0	Convergiu	7.30644e-13	0	3.85998e-09	16	0.00
HS32	3	1	1	Convergiu	1.24259	3.96818e-09	7.58387e-10	4	0.00
HS33	3	0	2	Convergiu	2	4.97982e-09	4.0532e-07	35	0.00
HS34	3	0	2	Convergiu	-0.834032	8.96297e-09	4.95357e-09	4	0.00
HS35	3	0	1	Convergiu	0.111111	1.17126e-16	2.74039e-08	4	0.00
HS35I	3	0	1	Convergiu	0.111111	1.17126e-16	2.74039e-08	4	0.00
HS36	3	0	1	Convergiu	-3300	1.60316e-13	6.56049e-07	5	0.00
HS37	3	0	2	Convergiu	-3456	5.5738e-10	1.14358e-07	7	0.00
HS38	4	0	0	Convergiu	605.964	0	1.37436e-07	11	0.00
HS39	4	2	0	Convergiu	-1	1.02804e-10	1.62574e-13	6	0.00
HS3MOD	2	0	0	Convergiu	9.99999e-07	0	1.21951e-08	5	0.00
HS40	4	3	0	Convergiu	-0.25	2.06992e-08	1.22036e-10	4	0.00
HS41	4	1	0	Convergiu	1.92593	8.88178e-16	2.17775e-08	12	0.00
HS4	2	0	0	Convergiu	2.66667	0	1.91881e-08	12	0.00
HS42	4	2	0	Convergiu	13.8579	5.63923e-09	8.4671e-17	3	0.00
HS43	4	0	3	Convergiu	-44	2.76642e-09	6.61708e-08	27	0.00
HS44	4	0	6	Convergiu	-15	1.0057e-07	4.23737e-08	9	0.00
HS44NEW	4	0	6	Convergiu	-15	2.42289e-13	5.5479e-07	9	0.00
HS45	5	0	0	Convergiu	1	0	3.24613e-08	18	0.00
HS46	5	2	0	Convergiu	9.00726e-09	1.91555e-07	4.39807e-07	14	0.00
HS47	5	3	0	Convergiu	1.78578e-08	9.35321e-09	2.93071e-07	13	0.00
HS48	5	2	0	Convergiu	2.5638e-30	2.70129e-15	5.33356e-17	2	0.00
HS49	5	2	0	Convergiu	4.57325e-06	8.0428e-15	5.87717e-07	12	0.00
HS50	5	3	0	Convergiu	6.38372e-13	8.77602e-14	2.20001e-10	9	0.00
HS51	5	3	0	Convergiu	4.93038e-32	9.15513e-16	5.79615e-17	2	0.00
HS5	2	0	0	Convergiu	-1.91322	0	5.67285e-11	5	0.00
HS52	5	3	0	Convergiu	5.32665	1.06962e-14	1.32567e-16	3	0.00
HS53	5	3	0	Convergiu	4.09302	2.27257e-15	2.67852e-15	4	0.00
HS54	6	1	0	Convergiu	-0.908075	2.00089e-11	9.97907e-07	38964	0.65
HS55	6	6	0	Convergiu	6.33333	1.09693e-11	1.24163e-08	3	0.00
HS56	7	4	0	Convergiu	-3.456	2.16202e-09	2.80973e-10	15	0.00
HS57	2	0	1	Convergiu	0.0284598	3.56131e-07	4.83835e-08	6	0.00
HS59	2	0	3	Convergiu	-6.7495	1.14511e-07	8.57884e-10	19	0.00
HS60	3	1	0	Convergiu	0.0325682	7.23536e-10	4.76216e-11	10	0.00
HS61	3	2	0	Convergiu	-143.646	3.31806e-10	1.92724e-07	5	0.00
HS6	2	1	0	Convergiu	3.87589e-20	2.77417e-09	3.63819e-11	8	0.00
HS62	3	1	0	Convergiu	-26272.5	1.11022e-15	1.3873e-09	13	0.00
HS63	3	2	0	Convergiu	961.715	1.06159e-07	1.40133e-07	4	0.00
HS64	3	0	1	Convergiu	6299.84	7.31742e-09	8.82829e-08	10	0.00
HS65	3	0	1	Convergiu	0.953529	1.40982e-07	8.20298e-07	10	0.00

HS66	3	0	2	Convergiu	1.33277	8.77967e-09	3.75862e-09	4	0.00
HS67	3	0	14	Convergiu	-1162.12	7.44048e-07	1.56773e-09	17	0.00
HS68	4	2	0	Convergiu	-0.920425	3.79407e-10	6.7363e-10	29	0.00
HS69	4	2	0	Convergiu	-956.713	6.28644e-07	1.03603e-08	27	0.00
HS70	4	0	1	Convergiu	0.269057	4.40536e-13	5.94737e-07	9	0.00
HS71	4	1	1	Convergiu	17.014	3.53026e-09	4.57295e-09	9	0.00
HS7	2	1	0	Convergiu	-1.73205	8.43769e-14	3.41242e-10	4	0.00
HS72	4	0	2	Convergiu	727.679	1.44158e-10	2.1058e-10	12	0.00
HS73	4	1	2	Convergiu	29.8944	6.93889e-18	2.77363e-07	15	0.00
HS74	4	3	2	Convergiu	5126.5	8.94002e-12	6.41766e-07	9	0.00
HS75	4	3	2	Convergiu	5174.41	4.25901e-08	5.00088e-11	13	0.00
HS76	4	0	3	Convergiu	-4.68182	2.40989e-14	3.44932e-07	5	0.00
HS76I	4	0	3	Convergiu	-4.68182	2.40989e-14	3.44932e-07	5	0.00
HS77	5	2	0	Convergiu	0.241505	6.7502e-13	1.15085e-08	13	0.00
HS78	5	3	0	Convergiu	-2.9197	4.90327e-12	8.23085e-08	3	0.00
HS79	5	3	0	Convergiu	0.0787768	1.02519e-08	1.86795e-10	8	0.00
HS80	5	3	0	Convergiu	0.0539498	3.25837e-07	4.38778e-10	7	0.00
HS81	5	3	0	Convergiu	0.0539498	3.14811e-07	3.84186e-11	7	0.00
HS8	2	2	0	Convergiu	-1	1.39447e-10	0	1	0.00
HS83	5	0	3	Convergiu	-30665.5	4.76376e-07	7.33506e-07	30	0.00
HS84	5	0	3	Convergiu	-5.28033e+06	1.81899e-09	6.51661e-07	62	0.00
HS85	5	0	21	Convergiu	-2.21548	3.7254e-09	7.61626e-07	405	0.10
HS86	5	0	10	Convergiu	-32.3487	2.45176e-12	2.74731e-08	8	0.00
HS87	6	4	0	MaxIter	8996.98	1.10702e-07	0.000158334	200001	7.98
HS88	2	0	1	Convergiu	1.36261	4.10832e-08	7.04305e-11	12	0.01
HS89	3	0	1	Infactível	4.53674e-05	0.873647	0.00145834	4	0.01
HS90	4	0	1	Convergiu	1.36262	3.31208e-08	3.18363e-09	17	0.03
HS91	5	0	1	Infactível	1.30087e-06	1.22188	0.000435633	4	0.01
HS9	2	1	0	Convergiu	-0.5	4.65661e-10	1.9449e-07	5	0.00
HS92	6	0	1	Infactível	5.88953e+06	0.133233	930.26	9	0.05
HS93	6	0	2	Convergiu	135.076	3.15278e-08	1.59154e-08	43	0.00
HS95	6	0	4	Convergiu	0.0157324	3.86552e-08	7.33458e-07	30	0.00
HS96	6	0	4	Convergiu	0.0156913	5.32528e-08	6.41398e-07	40	0.00
HS97	6	0	4	Convergiu	3.13588	1.92214e-07	5.1033e-07	13	0.00
HS98	6	0	4	Convergiu	4.0713	6.24647e-07	3.62847e-07	34	0.00
HS99	7	2	0	Convergiu	-8.3108e+08	2.95405e-09	8.35887e-09	4	0.00
HUBFIT	2	0	1	Convergiu	0.0168936	4.96003e-13	2.4992e-08	4	0.00
HUES-MOD	5000	2	0	Convergiu	3.48245e+07	1.18702e-07	9.68128e-08	37	85.92
HUESTIS	5000	2	0	Convergiu	1.74122e+11	9.93789e-08	5.05378e-05	36	67.63
HUMPS	2	0	0	Convergiu	0.00626638	0	9.05938e-07	771	0.01
HYDC20LS	99	0	0	Convergiu	0.976221	0	5.60176e-07	251	4.05

HYDCAR20	99	99	0	Convergiu	0	1.16558e-09	0	1	0.00
HYDCAR6	29	29	0	Convergiu	0	3.04199e-08	0	1	0.00
HYPCIR	2	2	0	Convergiu	0	1.19776e-11	0	1	0.00
INDEF	5000	0	0	MaxIter	-9.42056e+10	0	0.00592425	200001	1263.55
JANNSON3	20000	1	2	MaxRest	19998.9	0.000792889	9.99825e-05	1	2606.20
JANNSON4	10000	0	2	Convergiu	9801.97	1.43949e-12	1.50368e-08	14	0.19
JENSMP	2	0	0	Convergiu	124.362	0	3.60627e-07	9	0.00
JIMACK	-	-	-	Falha	-	-	-	-	-
KISSING	127	42	861	Convergiu	0.892127	2.60761e-07	9.88267e-07	1526	363.22
KIWCRESC	3	0	2	Convergiu	-1.28554e-09	3.65347e-09	1.67423e-12	34	0.00
KOEBHELB	3	0	0	Convergiu	77.5163	0	1.30487e-07	117	0.02
KOWOSB	4	0	0	Convergiu	0.000307801	0	2.87267e-09	13	0.00
KSIP	20	0	1001	Convergiu	0.712986	5.11245e-07	2.64551e-07	28	19.27
LAKES	90	78	0	Convergiu	350525	6.19156e-11	1.78854e-07	3	0.00
LAUNCH	25	9	19	Convergiu	9.00532	1.88191e-08	9.61995e-07	84	0.03
LCH	3000	1	0	Convergiu	-4.34179	1.21295e-11	2.14619e-07	958	29.76
LEAKNET	156	153	0	Convergiu	8.05194	4.5856e-07	9.26738e-07	8	0.01
LEWISPOL	6	9	0	Infactível	2.92611	26.3249	2.44259e-06	1	0.01
LIARWHD	5000	0	0	Convergiu	0.000577787	0	4.8154e-08	12	0.04
LIN	4	2	0	Convergiu	-0.0175775	1.41406e-12	2.03653e-08	4	0.00
LINVERSE	1999	0	0	Convergiu	681.002	0	8.67372e-08	36	0.16
LIPPERT1	20201	10000	40000	MaxTempo	-0.01	33659	1.82301e-08	1	7553.71
LIPPERT2	-	-	-	Falha	-	-	-	-	-
LISWET10	2002	0	2000	Convergiu	10.1546	2.26212e-07	6.5305e-07	25	0.59
LISWET11	2002	0	2000	Convergiu	10.0043	6.44843e-08	6.50303e-07	15	0.24
LISWET1	2002	0	2000	Convergiu	7.26621	3.39561e-07	1.17188e-07	15	0.24
LISWET12	2002	0	2000	Convergiu	347.58	6.83371e-08	8.65246e-08	27	0.33
LISWET2	2002	0	2000	Convergiu	5.01484	1.42822e-07	4.72275e-07	13	0.26
LISWET3	2002	0	2000	Convergiu	5.00384	1.82946e-08	9.25351e-07	13	0.23
LISWET4	2002	0	2000	Convergiu	5.01287	1.49446e-08	5.99226e-07	35	0.62
LISWET5	2002	0	2000	Convergiu	5.00525	1.91424e-08	1.84924e-07	13	0.23
LISWET6	2002	0	2000	Convergiu	5.02359	1.85302e-08	2.53323e-07	12	0.22
LISWET7	2002	0	2000	Convergiu	100.482	9.84063e-07	2.58318e-07	15	0.23
LISWET8	2002	0	2000	Convergiu	143.325	4.16601e-07	3.33081e-07	20	0.28
LISWET9	2002	0	2000	Convergiu	392.996	3.70125e-07	1.32095e-07	24	0.31
LOADBAL	31	11	20	Convergiu	0.452854	1.54044e-11	6.55407e-07	161	0.01
LOGHAIRY	2	0	0	Convergiu	0.182322	0	3.20969e-12	2340	0.03
LOGROS	2	0	0	Convergiu	3.58602e-12	0	1.83978e-07	69	0.00
LOOTSMA	3	0	2	Convergiu	1.41422	6.56995e-13	5.00244e-07	8	0.00
LOTSCHD	12	7	0	Convergiu	2398.42	1.03838e-10	3.27168e-08	6	0.00
LSNNODOC	5	4	0	Convergiu	123.113	1.27505e-07	1.40095e-08	4	0.00

LSQFIT	2	0	1	Convergiu	0.0337872	1.80971e-12	4.7699e-08	4	0.00
LUKVLE10	10000	9998	0	Convergiu	3535.1	5.79865e-13	1.75436e-07	6	0.23
LUKVLE1	10000	9998	0	Convergiu	6.23246	8.90436e-10	2.67185e-08	4	0.34
LUKVLE11	9998	6664	0	MaxRest	1e+20	1.00003e+20	2.19468e+33	1	2675.99
LUKVLE12	9997	7497	0	Convergiu	171495	2.62796e-07	5.4274e-07	44	6023.11
LUKVLE13	9998	6664	0	Convergiu	91372	2.90929e-08	3.3851e-07	9	0.27
LUKVLE14	9998	6664	0	Convergiu	3.13811e+08	2.99016e-08	7.14781e-07	14	562.41
LUKVLE15	9997	7497	0	MaxRest	7.03543e+06	69.4984	0.573876	36	1164.34
LUKVLE16	9997	7497	0	Convergiu	16615.1	1.16256e-09	7.74771e-08	29	28.95
LUKVLE17	9997	7497	0	Rhomax	33562.6	8.70215e-07	0.00755517	46	10.71
LUKVLE18	9997	7497	0	Rhomax	11202.9	8.48168e-07	0.0031668	44	7.11
LUKVLE2	10000	4999	0	Rhomax	-1e+20	2.41308e-07	2.43431e+09	60	12.01
LUKVLE3	10000	2	0	Convergiu	27.5866	1.32528e-10	7.79348e-08	9	0.13
LUKVLE4	10000	4999	0	Rhomax	6.39713e+09	1.29647e-11	5.78738e+13	91	3.49
LUKVLE6	9999	4999	0	Convergiu	628644	6.63636e-08	9.52918e-08	13	0.40
LUKVLE7	10000	4	0	Convergiu	-2165.07	1.45792e-07	6.27449e-08	36	63.23
LUKVLE8	10000	9998	0	Rhomax	1.06096e+06	1.22916e-08	0.00129202	8731	147.68
LUKVLE9	10000	6	0	Convergiu	1000.26	3.38902e-09	9.23478e-07	834	18.68
LUKVLI10	10000	0	9998	Convergiu	3535.15	3.09982e-07	4.55739e-08	3542	194.38
LUKVLI1	10000	0	9998	Convergiu	9898.53	2.20046e-08	9.22583e-07	3400	463.51
LUKVLI11	9998	0	6664	Convergiu	3.4338	2.28057e-08	9.88254e-07	4889	244.29
LUKVLI12	9997	0	7497	MaxTempo	18548.7	7.68605e-07	0.00357201	423	7207.07
LUKVLI13	9998	0	6664	Convergiu	165.219	1.4146e-08	8.35399e-07	36	4.29
LUKVLI14	9998	0	6664	MaxTempo	1.1213e+08	9.52924e-08	0.766917	289	7208.25
LUKVLI15	9997	0	7497	Convergiu	12.5481	1.1139e-07	9.70278e-07	46	4.02
LUKVLI16	9997	0	7497	Convergiu	2972.14	1.04855e-07	9.97475e-07	4294	126.95
LUKVLI17	9997	0	7497	Convergiu	788.306	1.21103e-09	9.64002e-07	5332	157.94
LUKVLI18	9997	0	7497	Convergiu	0.495892	2.9425e-13	9.99322e-07	2716	80.16
LUKVLI2	10000	0	4999	Rhomax	-1e+20	6.96067e-13	2.08705e+09	56	11.07
LUKVLI3	10000	0	2	Convergiu	11.9398	1.32644e-08	6.17177e-07	298	4.77
LUKVLI4	10000	0	4999	Rhomax	1.02867e+10	2.04678e-10	1.49646e+14	114	4.86
LUKVLI6	9999	0	4999	Convergiu	628644	1.67644e-09	4.39378e-08	335	26.00
LUKVLI7	10000	0	4	Convergiu	-2167.88	1.40905e-10	7.33542e-07	137	95.15
LUKVLI8	10000	0	9998	Convergiu	827403	2.06039e-07	3.26992e-07	366	16.03
LUKVLI9	10000	0	6	Convergiu	998.941	2.01821e-10	9.67986e-07	3870	75.01
MADSEN	3	0	6	Convergiu	0.616432	1.10051e-09	2.80174e-08	67	0.00
MADSSCHJ	201	0	398	Convergiu	-4992.13	2.43907e-07	4.28328e-07	33	13.09
MAKELA1	3	0	2	Convergiu	-1.41421	6.03973e-09	1.37972e-10	22	0.00
MAKELA2	3	0	3	Convergiu	7.2	3.415e-09	9.38597e-11	8	0.00
MAKELA3	21	0	20	Convergiu	1.22432e-05	2.00237e-10	6.37439e-07	15	0.01
MAKELA4	21	0	40	Convergiu	0.000100084	5.69908e-11	3.94077e-07	10	0.00

MANCINO	100	0	0	Convergiu	5.66214e-10	0	6.52816e-08	7	0.32
MARATOS	2	1	0	Convergiu	-1	1.67684e-09	1.25025e-17	2	0.00
MARATOSB	2	0	0	Convergiu	0.995893	0	7.94809e-07	4	0.00
MARINE	11215	11192	0	Convergiu	1.97466e+07	6.09192e-08	1.07811e-07	29	393.90
MATRIX2	6	0	2	Convergiu	3.08331e-06	2.56626e-07	2.90647e-07	42	0.00
MAXLIKA	8	0	0	Convergiu	1149.35	0	5.16144e-08	21	0.02
MCCORMCK	5000	0	0	Convergiu	-4566.58	0	4.49597e-08	15	0.34
MCONCON	15	11	0	Convergiu	-6230.8	3.27104e-08	7.807e-08	2	0.00
MDHOLE	2	0	0	Convergiu	1.00001e-05	0	2.47678e-08	3	0.00
METHANB8	31	31	0	Convergiu	0	1.27662e-07	0	1	0.00
METHANL8	31	31	0	Convergiu	0	5.26527e-11	0	1	0.00
MEXHAT	2	0	0	Convergiu	-0.0400092	0	5.0837e-07	24	0.00
MEYER3	3	0	0	Convergiu	97.4967	0	8.06822e-07	566	0.01
MIFFLIN1	3	0	2	Convergiu	-1	0	1.16476e-10	23	0.00
MIFFLIN2	3	0	2	Convergiu	-1	1.59789e-13	4.08589e-07	4	0.00
MINMAXBD	5	0	20	Convergiu	115.706	5.31097e-08	9.16748e-09	619	0.07
MINMAXRB	3	0	4	Convergiu	4.56788e-06	7.84282e-07	4.36056e-07	39	0.00
MINPERM	1113	1033	0	Convergiu	0.00036288	2.54141e-13	3.31647e-09	1	0.24
MISTAKE	9	0	13	Convergiu	-0.999988	7.55139e-10	5.62411e-07	43	0.00
MODBEALE	20000	0	0	Convergiu	3.03361	0	1.81678e-07	21	2.51
MOREBV	5000	0	0	Convergiu	2.36802e-11	0	7.9672e-07	2	0.41
MOSARQP1	2500	0	700	Convergiu	-3821.06	3.23809e-11	9.48759e-08	62	2.16
MOSARQP2	2500	0	700	Convergiu	-5052.59	2.9278e-11	9.53452e-08	27	1.09
MSQRTA	1024	1024	0	Convergiu	0	1.99439e-09	0	1	3.36
MSQRTALS	1024	0	0	Convergiu	6.44866e-06	0	9.95005e-08	23	5.81
MSQRTB	1024	1024	0	Convergiu	0	3.38949e-09	0	1	3.34
MSQRTBLS	1024	0	0	Convergiu	9.88797e-07	0	1.53049e-07	19	3.90
MSS3	2070	1981	0	Convergiu	-335.999	3.07916e-10	4.57177e-07	66	6.89
MWRIGHT	5	3	0	Convergiu	24.9788	8.76692e-08	2.81476e-08	7	0.00
NCB20B	-	-	-	Falha	-	-	-	-	-
NCB20	-	-	-	Falha	-	-	-	-	-
NCVXBQP1	10000	0	0	Convergiu	-1.98554e+10	0	2.73743e-07	72	0.44
NCVXBQP2	10000	0	0	Convergiu	-1.33402e+10	0	5.51471e-07	151	1.81
NCVXBQP3	10000	0	0	Convergiu	-6.45927e+09	0	9.1205e-08	475	7.05
NCVXQP1	10000	5000	0	Convergiu	-7.51351e+09	9.47091e-07	9.68558e-07	37	2.98
NCVXQP2	10000	5000	0	Rhomax	-5.83741e+09	9.23632e-07	0.000241488	76	4.89
NCVXQP3	10000	5000	0	Rhomax	-3.07912e+09	9.91921e-07	3.7122e-05	560	40.96
NCVXQP4	10000	2500	0	Convergiu	-9.38499e+09	7.23317e-08	6.49683e-07	42	0.35
NCVXQP5	10000	2500	0	Convergiu	-6.63491e+09	9.00319e-07	4.46881e-08	88	0.93
NCVXQP6	-	-	-	Falha	-	-	-	-	-
NCVXQP7	10000	7500	0	Rhomax	-5.21974e+09	5.89052e-07	2.7024e-06	70	33.84

NCVXQP8	10000	7500	0	Rhomax	-3.57685e+09	9.95384e-07	24.2168	94	110.82
NCVXQP9	10000	7500	0	Rhomax	-2.12084e+09	9.76165e-07	2.39446e-05	323	313.77
NONCVXU2	5000	0	0	Convergiu	11586.1	0	2.51027e-07	1287	25.20
NONCVXUN	5000	0	0	Convergiu	11618	0	4.27412e-07	1376	29.91
NONDIA	5000	0	0	Convergiu	0.00025697	0	1.30842e-08	3	0.01
NONDQUAR	5000	0	0	Convergiu	0.000527068	0	4.5547e-07	13	0.09
NONMSQRT	4900	0	0	Convergiu	710.455	0	5.77262e-07	207	3101.45
NONSCOMP	5000	0	0	Convergiu	0.272073	0	4.64889e-08	18	0.13
ODFITS	10	6	0	Convergiu	-2380.03	6.03585e-13	3.74276e-08	20	0.00
OET1	3	0	1002	Convergiu	3.10302	4.32275e-08	2.98605e-09	11	2.33
OET2	3	0	1002	Convergiu	2.00339	3.56821e-07	9.95408e-07	2339	681.77
OET3	4	0	1002	Convergiu	0.0258559	1.43495e-09	2.80988e-08	10	2.56
OET4	4	0	1002	Convergiu	0.062836	3.97923e-07	9.72519e-07	9820	2407.56
OET5	5	0	1002	Convergiu	0.0476189	6.3613e-14	9.84379e-07	3615	972.26
OET6	5	0	1002	Convergiu	2.00401	3.93446e-07	9.88436e-07	2539	760.52
OET7	7	0	1002	Convergiu	2.005	2.01541e-14	9.99954e-07	2676	829.69
OPTPRLOC	30	0	30	Convergiu	-16.3444	4.74789e-07	4.30404e-07	58	0.01
ORTHRDM2	8003	4000	0	Convergiu	311.015	8.71134e-09	1.52733e-08	5	275.61
ORTHRDS2	5003	2500	0	Convergiu	39237.4	2.78667e-07	4.40887e-08	35	355.95
ORTHREGA	8197	4096	0	Convergiu	22647.8	3.7499e-10	1.13603e-07	40	1034.66
ORTHREGB	27	6	0	Convergiu	7.33039e-15	1.69357e-12	1.71219e-07	3	0.00
ORTHREGC	5005	2500	0	Convergiu	331.567	8.72331e-08	6.99715e-07	33	311.81
ORTHREGD	5003	2500	0	Rhomax	39238.1	2.96418e-07	1.18152	92	785.01
ORTHREGE	7506	5000	0	MaxTempo	1276.07	4.78794e-11	4.81734e-05	150	7208.91
ORTHREGF	4805	1600	0	Convergiu	62.7687	4.21955e-07	7.34177e-08	18	51.80
ORTHRGDM	10003	5000	0	Convergiu	77264.9	9.772e-07	5.97201e-08	39	2761.82
ORTHRGDS	5003	2500	0	Convergiu	762.064	4.95845e-10	9.83159e-07	6	77.69
OSBORNEA	5	0	0	Convergiu	6.09772e-05	0	3.54439e-07	19	0.00
OSBORNEB	11	0	0	Convergiu	0.0401377	0	5.98327e-08	20	0.01
OSCIGRAD	100000	0	0	Convergiu	4.4799e-11	0	9.44826e-08	13	3.03
OSCIGRNE	100000	100000	0	Convergiu	0	1.42804e-07	0	1	0.80
OSCIPANE	10	10	0	MaxRest	0	1.05896e+08	0	1	2.93
OSCIPATH	10	0	0	MaxIter	2.60841e-05	0	0.00225935	200001	4.95
OSLBQP	8	0	0	Convergiu	6.25	0	3.68588e-09	15	0.00
PALMER1	4	0	0	Convergiu	28191.6	0	2.96712e-07	28	0.00
PALMER1A	6	0	0	Convergiu	0.0935137	0	1.19189e-07	53	0.00
PALMER1B	4	0	0	Convergiu	3.44735	0	4.03468e-08	30	0.00
PALMER1C	8	0	0	Convergiu	5.1998	0	5.58963e-08	4	0.00
PALMER1D	7	0	0	Convergiu	11.6072	0	2.88972e-07	7	0.00
PALMER1E	8	0	0	Convergiu	1.1421	0	5.69172e-08	23	0.00
PALMER2	4	0	0	Convergiu	4581.17	0	5.57076e-07	4	0.00

PALMER2A	6	0	0	Convergiu	0.0250571	0	8.35225e-07	37	0.00
PALMER2B	4	0	0	Convergiu	0.623267	0	5.86758e-08	23	0.00
PALMER2C	8	0	0	Convergiu	15.8465	0	3.30177e-07	5	0.00
PALMER2E	8	0	0	Convergiu	0.0611336	0	7.79942e-12	5	0.00
PALMER3	4	0	0	Convergiu	2416.98	0	9.14134e-11	6	0.00
PALMER3A	6	0	0	Convergiu	0.0295676	0	6.04624e-07	69	0.00
PALMER3B	4	0	0	Convergiu	4.22765	0	4.35694e-09	30	0.00
PALMER3C	8	0	0	Convergiu	0.289058	0	2.73857e-08	6	0.00
PALMER3E	8	0	0	Convergiu	0.100888	0	5.44017e-08	26	0.00
PALMER4	4	0	0	Convergiu	2424.02	0	6.45282e-11	6	0.00
PALMER4A	6	0	0	Convergiu	0.0417998	0	7.50763e-07	60	0.00
PALMER4B	4	0	0	Convergiu	6.83514	0	8.81055e-09	20	0.00
PALMER4C	8	0	0	Convergiu	0.566018	0	3.35906e-08	6	0.00
PALMER4E	8	0	0	Convergiu	0.370867	0	1.92413e-10	6	0.00
PALMER5A	8	0	0	Convergiu	2.12809	0	9.79462e-08	12	0.00
PALMER5B	9	0	0	Convergiu	0.120004	0	8.17556e-08	23	0.00
PALMER5C	6	0	0	Convergiu	2.12809	0	4.68326e-08	3	0.00
PALMER5D	4	0	0	Convergiu	87.3394	0	2.24087e-10	2	0.00
PALMER5E	8	0	0	Convergiu	1.63051	0	3.03944e-17	3	0.00
PALMER6A	6	0	0	Convergiu	0.0602001	0	8.13705e-07	119	0.00
PALMER6C	8	0	0	Convergiu	0.973536	0	2.52168e-08	3	0.00
PALMER6E	8	0	0	Convergiu	0.112953	0	9.29456e-08	22	0.00
PALMER7A	6	0	0	Convergiu	11.5036	0	9.73491e-07	119	0.00
PALMER7C	8	0	0	Convergiu	5.38725	0	5.84476e-08	3	0.00
PALMER7E	8	0	0	Convergiu	10.1926	0	1.95831e-07	38	0.00
PALMER8A	6	0	0	Convergiu	6.96971	0	2.62299e-11	5	0.00
PALMER8C	8	0	0	Convergiu	3.12533	0	6.07074e-08	3	0.00
PALMER8E	8	0	0	Convergiu	0.0597575	0	5.59291e-07	12	0.00
PENALTY1	1000	0	0	Convergiu	0.0284478	0	3.4054e-07	27	0.01
PENALTY2	200	0	0	Convergiu	4.71163e+13	0	1.01484e-07	10	0.03
PENALTY3	200	0	0	Convergiu	0.00099909	0	5.21724e-08	17	0.60
PENTAGON	6	0	15	Convergiu	0.00013699	2.74576e-14	1.16032e-07	27	0.00
PENTDI	5000	0	0	Convergiu	-0.0348019	0	5.92341e-07	14	0.18
PFIT1	3	3	0	MaxRest	0	1.73205e+20	0	2	2.79
PFIT1LS	3	0	0	Convergiu	4.90942e-05	0	4.73348e-07	26	0.00
PFIT2	3	3	0	MaxRest	0	1.73205e+20	0	2	2.91
PFIT2LS	3	0	0	Convergiu	0.000287239	0	9.32442e-08	46	0.00
PFIT3	3	3	0	MaxRest	0	1.73205e+20	0	2	2.49
PFIT3LS	3	0	0	Convergiu	0.00723357	0	2.22559e-07	106	0.00
PFIT4	3	3	0	MaxRest	0	1.5064e+20	0	2	2.57
PFIT4LS	3	0	0	Convergiu	0.0294351	0	2.85093e-07	110	0.00

POLAK1	3	0	2	Convergiu	2.71829	2.44285e-08	5.00426e-07	3	0.00
POLAK2	11	0	2	MaxRest	-35.38	1.00012e+20	0.107407	2	3.44
POLAK3	12	0	10	MaxRest	-121.261	3.16228e+20	4.31795e+12	3	29.33
POLAK4	3	0	3	MaxRest	-0.999983	1.41454	0.0546864	20	2.22
POLAK5	3	0	2	Convergiu	50	1.41213e-12	4.25405e-09	2	0.00
POLAK6	5	0	4	Convergiu	-44	1.47263e-09	2.09146e-10	52	0.00
PORTFL1	12	1	0	Convergiu	0.0204931	3.99876e-10	7.37892e-07	5	0.00
PORTFL2	12	1	0	Convergiu	0.0297135	3.99998e-10	8.71238e-07	5	0.00
PORTFL3	12	1	0	Convergiu	0.0327506	3.99963e-10	3.02227e-07	6	0.00
PORTFL4	12	1	0	Convergiu	0.0263077	3.99859e-10	1.69608e-07	6	0.00
PORTFL6	12	1	0	Convergiu	0.0257924	3.99945e-10	1.89457e-08	7	0.00
PORTSNQP	100000	2	0	Convergiu	33332.3	8.21258e-16	5.50432e-13	1	0.36
PORTSQP	100000	1	0	Convergiu	33333.3	3.77559e-12	6.25979e-08	1	0.10
POWELL20	5000	0	5000	Convergiu	6.51198e+09	2.99945e-07	1.82463e-07	10	24.89
POWELLBC	1000	0	0	Convergiu	310243	0	7.87653e-07	360	100.28
POWELLBS	2	2	0	Convergiu	0	7.83462e-07	0	1	0.00
POWELLSG	5000	0	0	Convergiu	0.0207813	0	3.22704e-07	11	0.04
POWELLSQ	2	2	0	Convergiu	0	5.18332e-08	0	1	0.00
POWER	10000	0	0	Convergiu	0.000286233	0	6.5441e-07	28	1.43
PRIMALC1	230	0	9	Convergiu	-6155.25	7.11344e-10	4.39984e-07	203	0.10
PRIMALC2	231	0	7	Convergiu	-3519.85	5.87239e-11	3.60688e-07	31	0.01
PRIMALC5	287	0	8	Convergiu	-426.184	1.24055e-07	9.47536e-07	22	0.01
PROBPENL	500	0	0	Convergiu	3.99198e-07	0	6.79485e-12	3	0.00
PRODPL0	60	20	9	Convergiu	63.3643	5.76768e-08	3.97363e-07	20	0.06
PRODPL1	60	20	9	Convergiu	35.7398	4.43954e-08	6.56592e-07	26	0.01
PSPDOC	4	0	0	Convergiu	2.41421	0	9.49472e-09	8	0.00
PT	2	0	501	Convergiu	0.186912	5.9531e-11	9.45267e-08	4	0.15
QPBAND	50000	0	25000	Convergiu	-49996.5	0	8.57868e-09	2	0.21
QPCBLEND	83	43	31	Convergiu	-0.0054823	2.0349e-12	6.54674e-07	46	0.01
QPCBOEI2	143	4	162	Convergiu	8.26069e+06	7.16865e-07	1.79242e-06	107	0.32
QPNBAND	50000	0	25000	Convergiu	-249985	0	2.24275e-07	2	0.20
QPNBLEND	83	43	31	Convergiu	-0.00663249	1.76736e-12	7.84936e-07	47	0.02
QPNBOEI2	143	4	162	Convergiu	1.38425e+06	7.90696e-07	1.42748e-07	96	0.95
QR3D	610	610	0	Convergiu	0	6.27818e-08	0	1	0.31
QR3DBD	457	610	0	Convergiu	0	4.38813e-07	0	1	0.31
QR3DLS	610	0	0	Convergiu	2.77779e-10	0	5.1086e-07	444	22.68
QRTQUAD	5000	0	0	Convergiu	-2.64855e+11	0	6.71089e-07	2011	13.45
QUARTC	5000	0	0	Convergiu	1.55456	0	8.30609e-07	26	0.23
QUDLIN	5000	0	0	Convergiu	-1.25e+09	0	6.62216e-07	8	0.02
READING3	4002	2001	0	Convergiu	-0.00122336	3.5738e-08	2.95298e-09	2	0.11
RECIPE	3	3	0	Convergiu	0	3.72529e-07	0	1	0.00

RES	20	12	2	Convergiu	0	3.6558e-12	0	2	0.00
RK23	17	11	0	Convergiu	0.0833558	7.94618e-07	8.10872e-07	15	0.00
ROSENBR	2	0	0	Convergiu	7.1079e-12	0	2.65952e-07	23	0.00
ROSENMMX	5	0	4	Convergiu	-44	6.24481e-07	2.43282e-08	35	0.00
RSNBRNE	2	2	0	Convergiu	0	4.49057e-15	0	1	0.00
S268	5	0	5	Convergiu	6.14818e-10	3.28146e-14	1.99183e-07	11	0.00
S277-280	4	0	4	Convergiu	5.0762	2.6941e-13	1.62466e-07	7	0.00
S308	2	0	0	Convergiu	0.773199	0	2.03278e-07	9	0.00
S316-322	2	1	0	Infactível	800	1	0.0707107	1	0.00
S365	-	-	-	Falha	-	-	-	-	-
S365MOD	-	-	-	Falha	-	-	-	-	-
S368	8	0	0	Convergiu	-0.75	0	8.68398e-07	11	0.00
SBRYBND	5000	0	0	MaxTempo	14518.4	0	0.239306	34239	7200.00
SCHMVETT	5000	0	0	Convergiu	-14994	0	8.52433e-07	3	0.14
SCOSINE	5000	0	0	Rhomax	546.034	0	7.46866e+06	154	5.21
SCURLY10	10000	0	0	MaxTempo	-1.00316e+06	0	0.054233	414	7200.00
SCURLY20	10000	0	0	MaxTempo	-1.00315e+06	0	0.0888847	387	7200.00
SCURLY30	10000	0	0	MaxTempo	-1.00314e+06	0	0.285173	376	7200.00
SENSORS	100	0	0	Convergiu	-2095.88	0	5.42116e-09	23	1.65
SIMBQP	2	0	0	Convergiu	4.95001e-07	0	9.68909e-10	3	0.00
SIMPLLPA	2	0	2	Convergiu	1	1.71451e-12	1.76026e-08	12	0.00
SIMPLLPB	2	0	3	Convergiu	1.1	6.43061e-15	1.73163e-09	13	0.00
SINEALI	1000	0	0	Convergiu	-99899	0	3.41573e-08	17	0.09
SINEVAL	2	0	0	Convergiu	1.12684e-28	0	1.37974e-14	49	0.00
SINQUAD	5000	0	0	Convergiu	-6.75701e+06	0	4.94452e-09	14	0.17
SINROSNB	1000	0	999	Convergiu	201.994	8.9615e-08	4.54611e-08	24	0.24
SINVALNE	2	2	0	Convergiu	0	2.22045e-15	0	1	0.00
SIPOW1	2	0	2000	Convergiu	-0.173599	1.85095e-09	1.49443e-07	4	7.12
SIPOW1M	2	0	2000	Convergiu	-0.171628	2.20674e-11	4.96971e-07	3	5.31
SIPOW2	2	0	2000	Convergiu	-0.184336	4.88522e-10	1.5211e-07	4	7.06
SIPOW2M	2	0	2000	Convergiu	-0.181251	1.74199e-09	1.6479e-07	4	7.07
SIPOW3	4	0	2000	Convergiu	0.538348	1.65402e-10	3.61878e-07	5	10.66
SIPOW4	4	0	2000	Convergiu	0.278965	3.87822e-10	4.5273e-07	4	7.16
SISSER	2	0	0	Convergiu	1.07131e-08	0	4.58412e-07	13	0.00
SMBANK	117	64	0	Rhomax	-6.93226e+06	8.27643e-10	0.000193078	23879	4.40
SMMPSF	720	240	23	Convergiu	1.10128e+06	1.16615e-08	8.12054e-07	541	0.96
SNAIL	2	0	0	Convergiu	1.24766e-17	0	4.11865e-10	79	0.03
SNAKE	2	0	2	Convergiu	-0.000254797	9.21267e-07	1.86637e-08	125	0.01
SOSQP1	5000	2501	0	Convergiu	1.30158e-10	2.88267e-11	1.11214e-13	2	0.02
SOSQP2	5000	2501	0	Convergiu	-1248.7	6.81068e-08	3.29225e-07	15	0.11
SPARSINE	5000	0	0	Convergiu	0.163011	0	8.01222e-07	109	53.51

SPARSQUR	10000	0	0	Convergiu	0.0504766	0	5.71475e-07	13	0.48
SPECAN	9	0	0	Convergiu	1721.98	0	8.3504e-08	9	0.20
SPIN	1327	1325	0	Convergiu	0	8.48936e-07	0	1	2.15
SPIN2	102	100	0	Convergiu	0	3.83641e-14	0	1	0.02
SPIN2OP	102	100	0	Convergiu	1.46138e-18	1.99896e-09	1.5077e-10	6	0.11
SPINOP	1327	1325	0	Convergiu	8.6359	5.18101e-11	9.9397e-07	261	121.98
SPIRAL	3	0	2	Rhomax	0.0278004	8.53587e-07	0.00355405	125	0.00
SPMSQRT	4999	8329	0	Convergiu	0	4.31003e-08	0	1	0.07
SPMSRTLS	4999	0	0	Convergiu	6.19655e-08	0	9.17113e-08	17	0.51
SREADIN3	4002	2001	0	Convergiu	-0.0302369	6.25728e-08	1.17632e-07	2	0.11
SROSENBR	5000	0	0	Convergiu	8.81182e-05	0	9.69185e-07	7	0.02
STANCMIN	3	0	2	Convergiu	4.25	2.60725e-14	4.96964e-08	4	0.00
STCQP1	8193	4095	0	Convergiu	367557	4.34171e-08	8.64272e-08	23	41.68
STCQP2	8193	4095	0	Convergiu	37189.3	2.3979e-12	8.55761e-07	12	3.66
STEENBRA	432	108	0	Convergiu	16959.6	9.96246e-10	6.42498e-08	36	0.03
STEENBRB	468	108	0	Convergiu	9476.09	4.79488e-08	9.75259e-07	350	0.20
STEENBRC	540	126	0	Convergiu	32367.9	2.39294e-07	9.94142e-07	853	0.47
STEENBRD	468	108	0	Convergiu	11073.4	3.74637e-07	9.62409e-07	318	0.18
STEENBRE	540	126	0	Convergiu	31255.4	6.93913e-07	9.90116e-07	667	0.38
STEENBRF	468	108	0	Convergiu	9397.39	1.46796e-08	9.93648e-07	321	0.18
STEENBRG	540	126	0	Convergiu	31257.6	3.86002e-07	9.99358e-07	854	0.49
STNQP1	8193	4095	0	Convergiu	-311704	3.02614e-07	1.02396e-07	23	42.23
STNQP2	8193	4095	0	Convergiu	-564409	5.46507e-09	4.56999e-07	315	3.63
SUPERSIM	2	2	0	Convergiu	0.666667	0	1.57009e-16	1	0.00
SVANBERG	5000	0	5000	Convergiu	8362.16	2.07869e-07	9.43106e-07	451	19.13
SWOPF	83	78	14	Convergiu	0.0678615	3.58315e-14	8.54157e-07	179	0.06
SYNTHES1	6	0	6	Convergiu	0.75929	5.6135e-07	1.39725e-07	44	0.00
SYNTHES2	11	1	13	Convergiu	-0.554337	6.38234e-08	2.44706e-07	103	0.01
SYNTHES3	17	2	21	Convergiu	15.0822	1.16997e-07	1.58754e-07	52	0.01
TAME	2	1	0	Convergiu	0	0	0	1	0.00
TENBARS1	-	-	-	Falha	-	-	-	-	-
TENBARS2	-	-	-	Falha	-	-	-	-	-
TENBARS3	18	8	0	MaxRest	19.0302	3318.18	1.48893e-13	2	4.63
TENBARS4	18	8	1	Convergiu	368.493	6.34286e-07	6.22576e-08	83	0.02
TESTQUAD	5000	0	0	Convergiu	4.99982e-05	0	7.57265e-08	4	0.47
TFI1	3	0	101	Convergiu	5.33471	5.20456e-12	8.01136e-07	37	0.04
TFI2	3	0	101	Convergiu	0.921936	2.28917e-10	9.24903e-07	5	0.00
TFI3	3	0	101	Convergiu	4.32284	4.39852e-12	6.68138e-08	14	0.01
TOINTGOR	50	0	0	Convergiu	1373.91	0	6.48553e-07	6	0.00
TOINTGSS	5000	0	0	Convergiu	10	0	5.85612e-08	3	0.02
TOINTPSP	50	0	0	Convergiu	225.56	0	6.2736e-08	38	0.00

TOINTQOR	50	0	0	Convergiu	1175.47	0	5.09303e-08	3	0.00
TQUARTIC	5000	0	0	Convergiu	1.90889e-22	0	2.17105e-09	2	0.01
TRIDIA	5000	0	0	Convergiu	5.417e-05	0	4.05522e-07	3	0.29
TRIMLOSS	142	20	55	Convergiu	9.23745	6.83602e-12	7.84091e-07	26	0.03
TRO11X3	150	60	1	Rhomax	1273.05	3.64545e-07	0.000534448	14894	19.60
TRO21X5	540	200	1	Rhomax	1916.25	1.09347e-07	0.000101918	32382	177.21
TRO3X3	30	12	1	Rhomax	9.00005	7.18429e-07	5.46158e-05	689	0.17
TRO4X4	63	24	1	Convergiu	9.00034	7.03518e-07	8.74492e-07	126	0.09
TRO5X5	108	40	1	Convergiu	9.00087	1.14582e-10	8.16772e-07	361	0.44
TRO6X2	45	20	1	Convergiu	1225	1.57262e-07	8.35324e-08	139	0.06
TRUSPYR1	11	3	1	Convergiu	11.2287	9.16565e-08	3.05471e-08	19	0.00
TRUSPYR2	11	3	8	Convergiu	11.2288	5.77083e-07	3.07166e-07	56	0.00
TRY-B	2	1	0	Convergiu	2.31641e-15	2.90209e-11	1.18837e-09	4	0.00
TWIRIMD1	1247	521	191	Convergiu	-0.898845	3.06019e-08	4.36131e-07	35	1.13
TWIRISM1	343	224	89	Convergiu	99.0063	2.41251e-11	8.67033e-07	45	0.62
TWOBARS	2	0	2	Convergiu	1.50865	1.11778e-10	4.915e-07	13	0.00
VANDERM1	100	100	99	Convergiu	0	6.8735e-07	1.3028e-10	37	0.77
VANDERM2	100	100	99	Convergiu	0	6.8735e-07	1.3028e-10	37	0.78
VANDERM3	100	100	99	Convergiu	0	4.06109e-07	1.58171e-10	41	0.88
VANDERM4	100	100	99	MaxRest	0	8.55666e+20	0.063335	1	2108.94
VARDIM	200	0	0	Convergiu	8.46509e-17	0	3.01617e-11	29	0.00
VAREIGVL	50	0	0	Convergiu	1.12739e-12	0	9.40035e-08	14	0.01
VIBRBEAM	8	0	0	Convergiu	0.162871	0	7.28711e-08	42	0.02
WACHBIEG	3	2	0	Convergiu	1	5.40012e-13	1.84452e-07	3	0.00
WATER	31	10	0	Convergiu	10549.4	8.7964e-09	2.3238e-07	37	0.00
WATSON	12	0	0	Convergiu	8.51252e-06	0	4.55368e-07	19	0.00
WEEDS	3	0	0	Convergiu	2639.27	0	9.77325e-07	1287	0.01
WOMFLET	3	0	3	Convergiu	1.08278e-06	1.88657e-13	1.02625e-07	32	0.00
WOODS	4000	0	0	Convergiu	0.000150655	0	5.21022e-07	51	0.20
WOODSNE	4000	3001	0	Infactível	-8925.25	28.8625	7.1477e-06	2	0.02
YFIT	3	0	0	Convergiu	2.67087e-11	0	1.77232e-09	38	0.00
YFITNE	3	17	0	Convergiu	0	9.05869e-07	0	1	0.01
YFITU	3	0	0	Convergiu	0.0253216	0	6.5624e-07	42	0.00
YORKNET	312	256	0	Rhomax	14391.6	6.04548e-07	3.34929e-06	70	0.20
ZANGWIL2	2	0	0	Convergiu	-18.2	0	0	2	0.00
ZANGWIL3	3	3	0	Convergiu	0	0	0	1	0.00
ZECEVIC2	2	0	2	Convergiu	-4.125	7.6596e-14	7.33912e-07	4	0.00
ZECEVIC3	2	0	2	Convergiu	97.3095	2.94106e-09	1.23398e-10	33	0.00
ZECEVIC4	2	0	2	Convergiu	7.55751	1.34663e-08	2.10017e-07	12	0.00
ZY2	3	0	2	Convergiu	5.6151	1.47663e-07	2.33819e-07	181	0.00

Tabela C.1: Tabela de Resultados do DCICPP aplicado aos problemas do CUTEr

Anexo I

Licença

I.1 Sobre a licença dessa obra

A licença Creative Commons Atribuição-CompartilhaIgual 3.0 Não Adaptada utilizada nessa obra diz que:

1. Você tem a liberdade de:

- Compartilhar — copiar, distribuir e transmitir a obra;
- Remixar — criar obras derivadas;
- fazer uso comercial da obra.

2. Sob as seguintes condições:

- Atribuição — Você deve creditar a obra da forma especificada pelo autor ou licenciante (mas não de maneira que sugira que estes concedem qualquer aval a você ou ao seu uso da obra).
- Compartilhamento pela mesma licença — Se você alterar, transformar ou criar em cima desta obra, você poderá distribuir a obra resultante apenas sob a mesma licença, ou sob uma licença similar à presente.