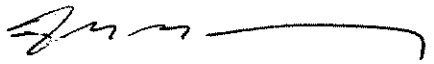
MÉTODOS NUMÉRICOS EM MINIMIZAÇÃO COM RESTRIÇÕES

ELVIO ANGEL PILOTTA

MÉTODOS NUMÉRICOS EM MINIMIZAÇÃO COM RESTRIÇÕES

Este exemplar corresponde a redação final da tese de doutorado devidamente corrigida e defendida por Elvio Angel Pilotta e aprovada pela Comissão Julgadora.

Campinas, 22 de Maio de 2000



Prof. Dr. José Mario Martínez
Orientador

Banca Examinadora:
Prof. Dr. José Mario Martínez (orientador) (IMECC–UNICAMP)
Profa. Dra. Márcia Aparecida Gomes Ruggiero (IMECC–UNICAMP)
Prof. Dr. Antonio Carlos Moretti (IMECC–UNICAMP)
Profa. Dra. Regina Sandra Burachik (COPPE–UFRJ)
Prof. Dr. Roberto Andreani (UNESP–SJRP)
Profa. Dra. Nataša Krejić (Univ. of Novi Sad, Yugoslávia)
Profa. Dra. Margarida Pinheiro Mello (IMECC–UNICAMP)

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas, como requisito parcial à obtenção do título de Doutor em Matemática Aplicada.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Pilotta, Elvio Angel

P647m Métodos numéricos em minimização com restrições / Elvio Angel Pilotta -- Campinas, [S.P. :s.n.], 2000.

Orientador : José Mario Martínez

Tese (doutorado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Otimização. 2. Algoritmos. 3. Matemática aplicada. I. Martínez, José Mario. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Tese de Doutorado defendida e em 22 maio de 2000 e aprovada pela Banca Examinadora composta pelos Profs. Drs.

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

Prof (a). Dr (a). MÁRCIA APARECIDA GOMES RUGGIERO

Prof (a). Dr (a). ANTONIO CARLOS MORETTI

Prof (a). Dr (a). REGINA SANDRA BURACHIK

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

Prof (a). Dr (a). NATAŠA KREJIĆ

Prof (a). Dr (a). MARGARIDA PINHEIRO MELLO

Agradecimentos

Ao Professor **José Mario Martínez**, pela orientação, dedicação e acompanhamento na elaboração deste trabalho.

Aos professores e colegas do IMECC, em particular aos professores do Grupo de Otimização, pelo apoio e incentivo.

À FAPESP pelo apoio financiero.

Aos meus pais

Resumo

Nesta tese apresentamos métodos numéricos para problemas de minimização com restrições. O Capítulo 1 está baseado no artigo "Validation of an Augmented Lagrangian algorithm with a Gauss-Newton Hessian approximation using a set of Hard-Spheres Problems", de Krejić, Martínez, Mello e Pilotta. O Capítulo 2 está baseado no artigo "Inexact–Restoration algorithm for constrained optimization", de Martínez e Pilotta, onde é considerado um novo método de tipo restauração inexata para um problema de minimização com restrições gerais. O Capítulo 3 está baseado no artigo "Spectral Gradient method for linearly constrained optimization", de Martínez, Pilotta e Raydan, onde é considerado um novo método para um problema de minimização com restrições lineares e canalizações usando gradiente espectral precondicionado e penalização exponencial. O Capítulo 4 está baseado no artigo "A limited-memory multipoint secant method for bound constrained optimization", de Burdakov, Martínez e Pilotta, onde é considerado um novo método para um problema de minimização com canalizações usando uma estratégia de restrições ativas e um método secante simétrico *multipoint* com memória limitada para resolver um subproblema quadrático em cada face.

Abstract

We present numerical methods for constrained minimization problems. Chapter 1 is based on the paper "Validation of an Augmented Lagrangian algorithm with a Gauss-Newton Hessian approximation using a set of Hard-Spheres Problems", by Krejić, Martínez, Mello and Pilotta. Chapter 2 is based on the paper "Inexact–Restoration algorithm for constrained optimization", by Martínez and Pilotta, where we introduce an inexact-restoration method for solving a general constrained minimization problem. Chapter 3 is based on the paper "Spectral Gradient method for linearly constrained optimization", by Martínez, Pilotta, and Raydan, where we introduce a new method for this problem which uses exponential penalization. Chapter 4 is based on the paper " A limited-memory multipoint secant method for bound constrained optimization", by Burdakov, Martínez and Pilotta, where we introduce a new method for bound constrained optimization that uses active set methods for solving a quadratic subproblem in each face.

Índice

Introdução Geral

O problema de minimizar uma função sujeita a restrições é um problema que aparece em diferentes áreas das ciências e tecnologia e que têm muitas aplicações. Embora, muitas vezes tais problemas podem ser formulados com grande simplicidade a resolução numérica deles pode ser um problema difícil. Portanto, é preciso dispor de métodos competitivos e algoritmos eficientes destinados a resolver estes problemas, e particularmente, quando esses problemas são de grande porte. Uma vez proposto um novo método, e além de provar os resultados teóricos tais como convergência, é de fundamental importância fazer numerosos experimentos numéricos para a validação do algoritmo. A validação de novos algoritmos requer a comparação deles com outros métodos computacionais já bem estabelecidos (e não necessariamente do mesmo tipo) para um mesmo conjunto de problemas.

O mais consolidado método do tipo Lagrangeano Aumentado atualmente disponível é o implementado no pacote LANCELOT [24]. Apresentamos um novo algoritmo, chamado ALBOX, baseado na idéia do Lagrangeano Aumentado. Ambos os códigos minimizam a função Lagrangeano Aumentado com canalizações nas variáveis resolvendo um suproblema quadrático. A principal diferença entre os códigos é que enquanto LANCELOT minimiza a quadrática na face determinada por uma aproximação do ponto de Cauchy generalizado, ALBOX procura o minimizador da quadrática em toda a caixa da região de confiança do subproblema. Uma outra característica de ALBOX é que usamos uma surpreendentemente efetiva simplificação da Hessiana verdadeira do Lagrangeano.

Para fazer os experimentos numéricos e comparações dos códigos usamos uma família paramétrica de problemas conhecida como *Hard-Spheres Problem* (HSP). Estes problemas pertencem á família de problemas de empacotamento, datam do século dezessete e vários deles continuam abertos. Além disso, estes problemas estão relacionados a problemas práticos em química, biologia e física. Para utilizar LANCELOT com este problema foi necessário formular o HSP no formato SIF (Stan-

dard Input Format), o formato utilizado no CUTE (Constrained and Unconstrained Testing Environment) [10].

Ao formular um novo método de minimização é conveniente explorar a estrutura do problema, e particularmente, das restrições. Assim, e dependendo das restrições propomos outros três novos métodos para resolver o problema de minimização de uma função geral e continuamente diferenciável, sujeita a três classes de restrições: gerais, lineares e canalizações.

Para o primeiro método de minimização proposto consideramos um problema com restrições não lineares e variáveis em Ω, um conjunto fechado e convexo do espaço n-dimensional. Embora, na prática estamos interessados no caso que Ω seja um politopo. Muitos algoritmos para resolver este problema estão baseados em métodos factíveis, os quais tem a vantagem que, às vezes, soluções factíves não ótimas são úteis nas aplicações, o que não acontece com as aproximações não factíveis. A dificuldade desta estratégia é que às vezes pode ser muito difícil obter essas aproximações factíveis. Por outro lado, existem outros métodos opostos a esta idéia baseados em Programação Quadrática Seqüêncial (SQP). Portanto, é interessante considerar métodos do tipo restauração inexata onde a factibilidade é controlada em cada iteração com um mecanismo interno que determina o grau de precisão requerida nas restrições. O novo algoritmo modelo gera aproximações factíveis com relação a Ω. Cada iteração do algoritmo tem duas fases: Restauração e Minimização. Dado um ponto x^k, na fase de Restauração é achado um ponto intermediário y^k em Ω tal que a infactibilidade seja menor que a infactibilidade em x^k, e usando esse ponto intermediário é construída uma região factível aproximada. Na fase de Minimização é encontrada uma nova aproximação na região factível aproximada usando uma estratégia de região de confiança. O novo ponto é aceito se o valor da função de mérito nesse ponto é menor que em x^k. Se Ω é um politopo as duas fases podem ser formuladas como um problema de minimização com restrições lineares e algoritmos disponíveis para estes problemas podem ser usados. Para testar o novo método fizemos uma implementação que comparamos com LANCELOT usando o problema HSP, obtendo ótimos resultados.

Para o segundo método proposto consideramos um problema de minimização com restrições lineares de igualdade. Apresentamos um método Quase-Newton onde a aproximação secante satisfaz uma equação secante fraca. O método pode ser visto como um método gradiente espectral precondicionado, onde a atualização secante é a matriz do precondicionamento e que pode ser reiniciada se algum indicador da performance indique que seja conveniente. A escolha espectral do passo é um quociente de Rayleigh que usamos, como fator na atualização secante, combinada com uma busca linear não monótona. O próximo passo foi acrescentar canalizações nas variáveis no problema inicial. Estas canalizações são incluidas na função objetivo com uma estratégia de penalização exponencial. Para obter a direção de descida temos que resolver um problema de programação quadrática convexa em

cada iteração. Uma conseqüência da escolha espectral do passo é que a matriz KKT associada é mantida constante a menos que o processo seja reiniciado e somente o lado direito do sistema KKT muda nas iterações internas, portanto foi possível explorar técnicas de fatorações esparsas. Fizemos uma implementação e apresentamos alguns resultados computacionais.

Finalmente, no terceiro método consideramos um problema de minimização sujeito a restrições de canalizações, o qual é um problema muito importante em algoritmos de otimização prática. Por um lado muitos problemas da física, engenharia e problemas industriais têm esta formulação. Por outro lado, muitos algoritmos de otimização, baseados em Lagrangeano Aumentado (LANCELOT, ALBOX), resolvem iterativamente subproblemas de minimização com canalizações. Do mesmo modo que esses algoritmos, no novo método é necessário resolver um subproblema quadrático em cada iteração mas as restrições são tratadas de um modo diferente. A região factível é subdividida em faces e um algoritmo para minimização irrestrita é aplicado em cada face até que algum parâmetro indique que não seja conveniente continuar explorando essa face. Nesse caso a face é abandonada na direção do gradiente chopado [42]. Existem diferentes formas de calcular a matriz Hessiana do modelo quadrático. A melhor alternativa é usar a Hessiana verdadeira embora, às vezes, é muito caro computacionalmente de se calcular. Outras alternativas possíveis, e que mostraram ser eficientes, são as aproximações Quase-Newton BFGS ou SR1 com memória limitada. No nosso algoritmo calculamos aproximações da matriz Hessiana usando um método *multipoint* secante simétrico [11]. Esta estratégia é uma extensão do clássico esquema secante multipoint que explora a simetria da matriz Hessiana.A idéia é que a aproximação Hessiana deveria a ser tal que o gradiente do modelo quadrático coincida como o gradiente da função objetivo em alguns pontos anteriores. Como esta estratégia entra em conflito com a simetria, a informação das últimas iterações é privilegiada. Para testar a performance do novo algoritmo fizemos uma implementação e mostramos alguns resultados numéricos com comparações com LANCELOT.

Este trabalho está organizado da seguinte forma: cada um dos quatro capítulos seguintes contém um artigo em inglês de co-autoria do dissertante, o qual já foi aceito para publicação ou está sendo submetido em alguma revista internacional, precedido pelo correspondente resumo em língua portuguesa. O Capítulo 1 está baseado no artigo "Validation of an Augmented Lagrangian Algorithm with a Gauss-Newton Hessian Approximation Using a Set of Hard-Spheres Problems", escrito em colaboração com Nataša Krejić (Institute of Mathematics,University of Novi Sad, Yugoslavia), José Mario Martínez e Margarida Mello (IMECC– Universidad Estadual de Campinas), e que já foi aceito e será publicado em *Computational Optimization and Applications* neste ano. O Capítulo 2 está baseado no artigo "Inexact–Restoration Algorithm for Constrained Optimization", escrito em colaboração com José Mario Martínez e que já foi aceito e será publicado em *Journal of Optimization*

and Applications (JOTA) no Vol. 104 Nro. 2 deste ano. O Capítulo 3 está baseado no artigo "Spectral Gradient Method for Linearly Constrained Optimization", escrito em colaboração com José Mario Martínez e Marcos Raydan (Departamento de Computación, Facultad de Ciencias, Universidad Central de Venezuela), e que foi submetido para publicação em *Journal of Optimization and Applications* em 1999. O Capítulo 4 está baseado no trabalho "A limited-memory multipoint secant method for bound constrained optimization", escrito em colaboração com Oleg Burdakov (Computing Center, Russian Academy of Sciences, Moscow, Russia) e José Mario Martínez, e que será submetido para publicação proximamente. Por último, apresentamos algumas Conclusões gerais deste trabalho, as Referências Bibliográficas e o Apêndice, onde foi incluida a formulação do Hard-Sphere Problems em formato SIF (Standard Input Format), escrita em colaboração com José Mario Martínez, e que foi aceita em Outubro de 1998 no conjunto de problemas testes CUTE (Constrained and Unconstrained Testing Environment) [10].

Capítulo 1

Validação de um algoritmo Lagrangeano Aumentado com uma aproximação Gauss–Newton da Hessiana usando o Hard–Spheres Problem (HSP)

Resumo

É apresentado um algoritmo Lagrangeano Aumentado que usa aproximações Gauss–Newton da Hessiana em cada iteração interna e é testado usando uma família de *Hard–Spheres Problems* (HSP). O modelo Gauss–Newton *convexifica* a aproximação quadrática da função Lagrangeano Aumentado aumentando a eficiência do *solver* quadrático iterativo. O método resultante é consideravelmente mais eficiente que o correspondente algoritmo que usa Hessianas verdadeiras. É apresentado um estudo comparativo com o bem conhecido pacote LANCELOT.

Validation of an Augmented Lagrangian Algorithm with a Gauss-Newton Hessian Approximation Using a Set of Hard-Spheres Problems

Nataša Krejić * José Mario Martínez † Margarida Mello †
Elvio A. Pilotta †

May 7, 1998

Abstract

An Augmented Lagrangian algorithm that uses Gauss-Newton approximations of the Hessian at each inner iteration is introduced and tested using a family of Hard-Spheres problems. The Gauss-Newton model convexifies the quadratic approximations of the Augmented Lagrangian function thus increasing the efficiency of the iterative quadratic solver. The resulting method is considerably more efficient that the corresponding algorithm that uses true Hessians. A comparative study using the well-known package LANCELOT is presented.

*Institute of Mathematics, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Yugoslavia, E-mail: natasa@unsim.im.ns.ac.yu. This author was supported by FAPESP (Grant 96/8163-9).

†Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil. These authors were supported by PRONEX, FAPESP (Grant 90-3724-6), CNPq and FAEP-UNICAMP. E-mail: martinez@ime.unicamp.br, margarid@ime.unicamp.br, pilotta@ime.unicamp.br

1 Introduction

In recent years we have been involved with the development of algorithms based on sequential quadratic programming [11] and inexact restoration [17, 16] for minimization problems with nonlinear equality constraints and bounded variables.

The validation of these algorithms require their comparison with well established computer methods for the same type of problems, which include methods of the same family (as other SQP methods in the first case and GRG like methods in the second) as well as methods that adopt a completely different point of view, as is the case of Penalty and Augmented Lagrangian algorithms. The most consolidated practical Augmented Lagrangian method currently available is the one implemented in the package LANCELOT, described in [4]. This was the method used, for example, in [11], to test the reliability of a new large-scale sequential quadratic programming algorithm.

In the course of the above mentioned experimental studies we felt the necessity of intervening in the Augmented Lagrangian code in a more active way than the one permitted to users of LANCELOT. As a result of this practical necessity, we became involved with the development of a different Augmented Lagrangian code, which preserves most of the principles of the LANCELOT philosophy, but also has some important differences.

Following the lines of [4], a modern Augmented Lagrangian method is essentially composed by three nested algorithms:

- The external algorithm updates the Lagrange multipliers and the penalty parameters, decides stopping criteria for the internal algorithm and the rules for declaring convergence or failure of the overall procedure.

- An internal algorithm minimizes the augmented Lagrangian function with bounds on the variables. Trust region methods, where the subproblem consists on the minimization of a quadratic model on the intersection of two boxes, the one that defines the problem and the trust-region box, are used both in [4] and in our implementation.

- A third algorithm deals with the resolution of the quadratic subproblem. While LANCELOT restricts its search to the face determined by an approximate generalized Cauchy point, our code explores the domain of the subproblem as a whole.

The second item, specifically where it deals with the formulation of the quadratic subproblem, is the one in which we felt more strongly the desire to intervene. On one hand, we tried many alternative sparse quasi-Newton schemes (without success, up to now). On the other hand, we used a surprisingly effective simplification of the true Hessian of the Lagrangian, called, in this paper, "the Gauss-Newton Hessian approximation" by analogy with the Gauss-Newton method for nonlinear least-squares, which can be interpreted as the result of excluding from the Hessian of a sum of squares those terms involving Hessian of individual components.

In order to validate our augmented Lagrangian implementation we selected a family of problems in which we have particular interest, known as the family of Hard-Spheres problems.

The Hard-Spheres Problem belongs to a family of sphere packing problems, a class of challenging problems dating from the beginning of the seventeenth century. In the tradition of famous problems in mathematics, the statements of these problems are elusively simple, and have withstood the attacks of many worthy mathematicians (e.g. Newton, Hilbert, Gregory), while most of its instances remain open problems. Furthermore, it is related to practical problems in chemistry, biology and physics, see, for instance, the list of examples in [19], concerning mainly three-dimensional problems, or peruse the 1550-item-long bibliography in [5]. The *Hard-Spheres Problem* is to maximize the minimum pairwise distance between p points on a sphere in $\mathbb{R}^n$. This problem may be reduced to a nonlinear optimization problem that turns out, as might be expected from the mentioned history, to be a particularly hard, nonconvex problem, with a potentially large number of (nonoptimal) points satisfying KKT conditions. We have thus a class of problems indexed by the parameters n and p, that provides a suitable set of test problems for evaluating Nonlinear Programming codes.

Very convenient is the fact that the Hard-Spheres Problem may be regarded as the feasibility problem associated with another famous problem in the area, the *Kissing Number Problem*, which seeks to determine the maximum number $\mathcal{K}_n$ of nonoverlapping spheres of given radius in $\mathbb{R}^n$ that can simultaneously touch (kiss) a central sphere of same radius. Thus, if the distance obtained in the solution of the Hard-Spheres Problem, for given n and p, is greater than or equal to the radius of the sphere on which the points lie, one may conclude that $\mathcal{K}_n \geq p$. We use the known solution of the three-dimensional Kissing Number Problem to calibrate our code, described below, and choose for testing the code values of n, p that might bring forth new knowledge about the problem, or strengthen existing conjectures about the true (but, alas, not rigorously established) value of $\mathcal{K}_n$, from the following table of known values/bounds of $\mathcal{K}_n$ given in [5]:

n	K_n
1	2
2	6
3	12
4	24–25
5	40–46
6	72–82
7	126–140
8	240
9	306–380
10	500–595
11	582–915
12	840–1416

Table 1: Known values/bounds of K_n.

This paper is organized as follows. In Section 2 we formulate the Hard-Spheres Problem as a nonlinear programming problem and we relate the main characteristics of ALBOX, our Augmented Lagrangian Algorithm. In Section 3 we explain how the main algorithmic parameters of ALBOX were chosen. (Here we follow a previous study in [15].) In Section 4 we introduce the Gauss-Newton Hessian approximation and discuss the effect of its use in comparison with the use of true Hessians of the Lagrangians. In Section 5 we describe the parameters used with LANCELOT. The numerical experiments, obtained by running ALBOX and LANCELOT for a large number of Hard-Spheres problems, are presented in Section 6. Finally, some conclusions are drawn in Section 7.

2 ALBOX

The straightforward formulation of the Hard-Spheres Problem leads to the following maxmin problem, where r is the radius of the sphere, centered on the origin, on which the points lie:

$$\begin{array}{lll} \max & \min_{i \neq j} & \|y^i - y^j\| \\ & \text{s.t.} & \|y^k\| = 2r, \quad k = 1, \ldots, p. \end{array} \tag{1}$$

The vectors y^k belong to $\mathbb{R}^n$ and $\|\cdot\|$ is the Euclidean norm. Since the answer to the problem is invariant under the choice of positive r, we let $r = 1/2$. Furthermore, using the definition of $\langle \cdot, \cdot \rangle$, the standard inner product in $\mathbb{R}^n$, and the constraints,

it is easy to see that (1) is equivalent to

$$\begin{array}{lll} \min & \max_{i \neq j} & \langle y^i, y^j \rangle \\ & \text{s.t.} & \|y^k\| = 2r, \quad k = 1, \ldots, p. \end{array} \tag{2}$$

Applying the classical trick for transforming minimax problems into constrained minimization problems, we reduce (2) to the nonlinear program

$$\begin{array}{lrcl} \min & z & & \\ \text{s.t.} & z & \geq & \langle y^i, y^j \rangle, \quad \forall\, i \neq j, \\ & \|y^k\| & = & 1, \quad k = 1, \ldots, p. \end{array} \tag{3}$$

Adding slack variables to the first set of constraints and squaring the second set of equations in order to avoid nonsmoothness in the first derivatives, we obtain

$$\begin{array}{llllll} \min & z & & & & \\ \text{s.t.} & z & - \langle y^i, y^j \rangle & - & w_{ij} & = 0, \quad \forall\, i \neq j, \\ & & \|y^k\|^2 & & & = 1, \quad k = 1, \ldots, p, \\ & & & & w & \geq 0. \end{array} \tag{4}$$

which is of the general form

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & h(x) & = 0 \\ & \ell \leq x \leq u. & \end{array} \tag{5}$$

ALBOX, the augmented Lagrangian code developed, approximately solves

$$\begin{array}{ll} \min & L(x, \lambda, \rho) \\ \text{s.t.} & \ell \leq x \leq u, \end{array} \tag{6}$$

at each Outer Iteration, where

$$L(x, \lambda, \rho) = f(x) + \sum_i \lambda_i h_i(x) + \sum_i \rho_i h_i^2(x) \tag{7}$$

is the augmented Lagrangian function associated with (5), λ is the current approximation to the Lagrange multipliers and ρ (≥ 0) is the current vector of penalty parameters. These are updated at the end of the Outer Iteration.

Subproblem (6) is solved using BOX, the box-constrained solver described in [10]. This iterative method minimizes a quadratic approximation to the objective function on the intersection of the original feasible set, the box $\ell \leq x \leq u$, and the trust region (also a box), at each iteration. If the original objective function is sufficiently reduced at the approximate minimizer of the quadratic, the corresponding trial point is accepted as the new iterate. Otherwise, the trust region is reduced. The main

algorithmic difference between BOX and the method used in [2] is that in BOX the quadratic is explored on the whole intersection of the original box and the trust region whereas in [2] only the face determined by an "approximate Cauchy point" is examined.

ALBOX is a double precision FORTRAN 77 code that aims to cope with large-scale problems. For this reason, factorization of matrices is not used at all. The quadratic solver used to solve the subproblems of the box-constraint algorithm, QUACAN, visits the different faces of its domain using conjugate gradients on the interior of each face and "chopped gradients" as search directions to leave the faces. We refer the reader to [1], [9] and [10], for details on the actual implementation of QUACAN. In most iterations of this quadratic solver, a matrix-vector product of the Hessian approximation and a vector is computed. Occasionally, an additional matrix-vector product may be neccessary.

The performance of ALBOX, and, in fact, of most sophisticated algorithms, depends on the choice of many parameters. The most sensitive parameters were adjusted using the Kissing Problem with $n = 3$ and $p = 12$ (*Icosahedron Problem*). We discuss these choices in the next section. A similar analysis was carried out for LANCELOT, and is described in section 5.

3 Choice of parameters for ALBOX

3.1 Penalty parameters and Lagrange multipliers

The vector ρ of penalty parameters associated with the equality constraints $h(x) = 0$ are updated after each Outer Iteration. We considered two possibilities: to update each component according to the decrease of the corresponding component of $h(x)$ or using a global criterion based on $h(x)$. The specific alternatives contemplated were, assuming x to be the initial point at some outer iteration and $\bar{x}$ the final one:

1. increase ρ_i only if $|h(\bar{x})_i|$ is not sufficiently smaller than $|h(x)_i|$;

2. increase ρ_i only if $\|h(\bar{x})\|_\infty$ is not sufficiently smaller than $\|h(x)\|_\infty$.

Preliminary experiments revealed, perhaps surprisingly, that the "global strategy" 2 is better than the first. In fact, when ρ_i is not updated, but the other components of ρ are, the feasibility level $|h(\bar{x})_i|$ tends to deteriorate at the next iteration and, consequentely, a large number of Outer Iterations becomes necessary. In other words, it seems that a strategy based on 1 encourages a zigzagging behavior, with successive iterates alternatingly satisfying one constraint or another. Thus, although the original formulation allows for one penalty parameter for each

equality constraint, in practice it is as if we worked with one parameter for all of them, since they are all initialized at the same value (tests indicate that 10 is an adequate initial value) and are all updated according to the same rule (once again based on tests, they are increased by a factor of 10 when sufficient improvement of feasibility is not detected). Here we considered that "a sufficiently smaller than b" means that $a \leq 0.01b$.

It must be pointed out that the behavior of penalty parameters is not independent of the strategy for updating the Lagrange multipliers. With algorithmic simplicity in mind, we adopted a "first order formula". Letting $\bar{\lambda}$ be the Lagrange multiplier at the start of a new Outer Iteration and λ, ρ be the Lagrange multipliers and penalty parameters at the previous iteration, we set

$$\bar{\lambda}_i = \lambda_i + \rho_i h(\bar{x})_i$$

for all $i = 1, \ldots, m$. Initially, $\lambda = 0$.

3.2 Stopping criteria for box-constraint solver

Each outer iteration ends when one of the several stopping criteria for the algorithm that solves the augmented Lagrangian box-constrained minimization problem (6) is reached. There is the usual maximum number of iterations safeguard, which is set at 100 for QUACAN calls.

Other than that, we consider that the box-constraint algorithm BOX converges when

$$\|g_P(x)\|_2 \leq \varepsilon,$$

where $g_P(x)$ is the "continuous projected gradient" of the objective function of (6) at the point x. This vector is defined as the difference between the projection of $x - \nabla L(x, \lambda, \rho)$ on the box and the point x. The tolerance ε may change at each Outer Iteration. We tested two strategies for ε: one that defines ε dynamically depending on the degree of feasibility of the current iterate and another that fixes ε at 10^{-5}. Althought not conclusive, results for the Icosahedron Problem were better when the constant ε strategy was used. This was, therefore, the strategy adopted for further tests. Incidentally, the opposite was adopted in [8], where a similar Augmented Lagrangian Algorithm was used to solve linearly constrained problems derived from physical applications. Theoretical justifications for the inexact minimization of subproblems in the augmented Lagrangian context can also be found in [12, 13].

The box-constraint code admits other stopping criteria. For instance, execution may stop if the progress during some number of consecutive iterations is not good enough or if the the radius of the trust region becomes too small. Nevertheless, best results were obtained inhibiting these alternative stopping criteria.

3.3 Parameters for the quadratic solver

QUACAN is the code called to minimize quadratic functions (augmented Lagrangians in this case) subject to box constraints. Its efficiency, or lack thereof, plays a crucial role in the overall behavior of the Augmented Lagrangian Algorithm. Its parameters must therefore be carefully chosen.

Firstly we examine the convergence criterion. If the projected gradient of the quadratic is null, the corresponding point is stationary. Accordingly, convergence is considered achieved when the norm of this projected gradient is less than a fraction of the corresponding norm at the initial point. In this case, we use "non-continuous projected gradients," in which the projections are not computed on the feasible box but on the active constraints. Fractions 1/10, 1/100 and 1/100000 were tested on the Icosahedron Problem, and the first choice provided the best behavior, being the one employed subsequently.

The maximum number of iterations allowed is also an important parameter, since otherwise we may invest too much effort solving problems only distantly related to the original one. We found that the number of variables of the problem, $np + \binom{p}{2} + 1$, is a suitable delimiter in this case. Other non-convergence stopping criteria were inhibited.

The radius of the trust region determines the size of the auxiliary box used in QUACAN. The nonlinear programming algorithm is sensitive to the choice of δ, the first trust region radius. After testing different values, we selected $\delta = 10$ as an appropriate choice.

Another important parameter is $\eta \in (0, 1)$, the parameter that determines whether the next iterate must belong to the same face as the current one, or not. Roughly speaking, if η is small, the algorithm tends to leave the current face as soon as a mild decrease of the quadratic is detected. On the other hand, if $\eta \approx 1$, the algorithm only abandons the current face when the current point is close to a stationary point of the quadratic on that face. A rather surprising result was that, for the Icosahedron Problem, the conservative value $\eta = .95$ was better than smaller values.

Finally, when the quadratic solver hits the boundary of its feasible region, an extrapolation step may be tried, depending on the value of the extrapolation parameter $\gamma \geq 1$. If γ is large, new points will be tried at which the number of active bounds may be considerably increased. No extrapolation is tried when $\gamma = 1$. Tests indicated that $\gamma = 10$ is a convenient choice for the Hard-Spheres Problem.

4 Approximate Hessian

The nonlinear optimization problem (4) obtained in section 2 is the version of the Hard-Spheres Problem that was chosen for our tests. It was pointed out that (4) is of the general form

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & h(x) & = \ 0 \\ & \ell \leq x \leq u. & \end{array}$$

whose associated augmented Lagrangian is

$$L(x, \lambda, \rho) = f(x) + \langle \lambda, h(x) \rangle + \frac{\rho}{2} \|h(x)\|_2^2.$$

Thus

$$\nabla L(x, \lambda, \rho) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla h_i(x) + \rho h'(x)^T h(x)$$

and

$$\nabla^2 L(x, \lambda, \rho) = \nabla^2 f(x) + + \rho h'(x)^T h'(x) + \sum_{i=1}^{m} [\lambda_i + \rho h_i(x)] \nabla^2 h_i(x).$$

Although $\nabla^2 L(x, \lambda, \rho)$ tends to be positive definite when ρ is large, λ is close to the correct Lagrange multipliers and x is close to a solution, this is not the case at the early stages of augmented Lagrangian calculations. On the other hand, the simplified matrix obtained by neglecting the term involving second order derivatives of the constraint functions

$$B(x, \rho) = \nabla^2 f(x) + + \rho h'(x)^T h'(x)$$

is always positive semidefinite in our case, independently of ρ and x. Of course, this is always the case when f is a convex function.

Another insight into $B(x, \rho)$ is provided by examining the problem

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & h'(z)(x - z) + h(z) = 0 & \\ & \ell \leq x \leq u, & \end{array} \tag{8}$$

where z is the current point being used in a BOX iteration. Problem (8) is obtained by replacing the original $h(x) = 0$ constraints with its first order (linear) approximation. But $B(z, \rho)$ happens to be the Hessian of the augmented Lagrangian associated with (8) at z! Furthermore, both the augmented Lagrangian associated with (8) and its gradient evaluated at z coincide with their counterparts associated with the original problem (4), evaluated at z.

The matrix vector products $\nabla^2 L(x, \lambda, \rho)v$ and $B(x, \rho)v$ seem cumbersome to compute at a first glance. But taking advantadge of their structure enables the computation to be done in $O(np)$ time.

In principle, using the true Hessian of the Lagrangian should the best possible choice, since it represents better the structure of the true problem. However, available algorithms for minimizing quadratics in convex sets are much more efficient when the quadratic is convex than otherwise. QUACAN is not an exception to this rule. Therefore, in the interest of improving the overall performance of the augmented Lagrangian algorithm, we decided to use $B(x, \rho)$ as Hessian Lagrangian approximation.

The results were indeed impressive. Table 2 lists the average statistics obtained for four of the eighteen test sets, where each (n, p) pair was run for fifty random starting points. The average number of Outer iterations, BOX iterations, Function evaluations, Matrix Vector Products, CPU time in seconds and minimum distance are given for the runs using the exact Hessian (first row of each set) and the ones using the approximate Hessian (second row). The minimum distances obtained were very close and on some instances the minimum distance obtained using the approximate Hessian was smaller than the one obtained using the exact Hessian. While the number of Outer iterations does not differ very much from one choice to the other, the number of BOX iterations and, consequently, the number of Matrix Vector Products sensibly decreases. The overall result is a marked decrease in CPU time. In Figure 1 we plot the average CPU times, for all eighteen tests, using the exact Hessian versus the CPU times using the approximate Hessian. Also shown is the line that gives the best fit of the data by a linear (not affine) function, namely $y = 0.374138\,x$, that is, the approximate Hessian option implies in a decrease of almost two thirds in CPU times.

Problem size			Outer	Box	Funct.	MVP	CPU	Min
$\begin{bmatrix} n \\ p \end{bmatrix}$	var.	constr.	it.	it.	eval.		time	dist.
$\begin{bmatrix} 3 \\ 10 \end{bmatrix}$	76	55	4.86	37.06	52.14	1564.36	0.765	1.086487225412
			4.64	34.74	45.52	1194.70	0.476	1.083236334520
$\begin{bmatrix} 4 \\ 22 \end{bmatrix}$	320	253	4.60	91.10	123.90	16079.36	33.440	0.997314349536
			4.34	78.02	97.40	11222.14	20.032	0.997809583865
$\begin{bmatrix} 5 \\ 37 \end{bmatrix}$	852	703	5.00	274.10	358.54	142683.34	963.537	0.998632681285
			4.56	160.02	193.14	67020.22	373.141	0.998675348042

Table 2: Running ALBOX with exact (first row) and approximate Hessian (second row).

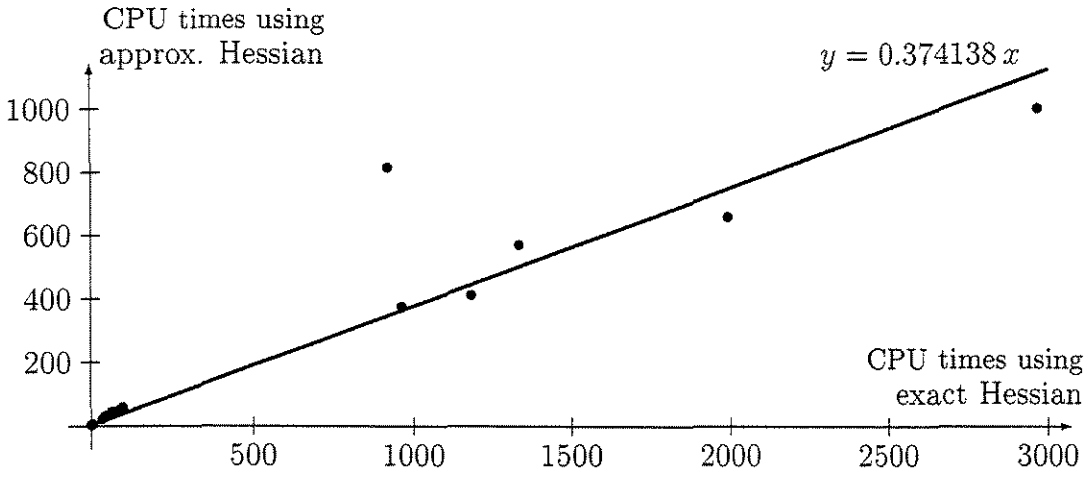


Figure 1: CPU times using exact Hessian (x-axis) versus using approximate Hessian (y-axis).

5 Choice of parameters for LANCELOT

LANCELOT allows for the choice of exact or approximate first and second order derivatives. However, LANCELOT's manual [3] (p.111) "strongly recommends the use of exact second derivatives whenever they are available", and, on the other hand, there is no provision for an user supplied Hessian approximation. In fact we ran a few tests with the default approximation (SR1) but the results were worse than those obtained using exact second derivatives, and thus this was the option adopted for all further tests. In the light of the experiments described in the previous section, this provides corroborating evidence to the effect that general purpose, consolidated packages, designed to provide a good performance with little interference from the user, may be more convenient to use than open ended, low-level interface codes, such as ALBOX; but, for the user willing to "get his hands dirty" the latter rawer code might not only prove competitive, it may actually outperform the former code, with its more polished though restrictive finish.

We also experimented with several different options for solving the linear equation solver, namely, without preconditioner, with diagonal preconditioner and with a band matrix preconditioner. The best results were obtained with the first option (no preconditioner). Another choice that slowed the algorithm, without noticeable improve the quality of solution, was requiring that the exact Cauchy point be computed. We settled to use the inexact Cauchy point option. The maximum number of iterations allowed is 1000. Finally, the gradient and constraints tolerances were the same chosen for ALBOX, namely 10^{-8}. The FORTRAN compiler option adopted for LANCELOT and ALBOX was "-O".

6 Numerical experiments

Tests were run on a Sun SparcStation 20, with the following main characteristics: 128Mbytes of RAM, 70MHz, 204.7 mips, 44.4 Mflops. Results for the fifty runs for each $(n,\ p)$ pair are summarized in the following tables. Table 3 summarizes the statistics that are "machine independent," typically involving number of iterations, number of function evaluations, with the exception of the optimal distances found. Quotes are needed because this is not completely accurate, since these numbers will in fact depend on machine precision, compiler manufacturer, and the like. Nevertheless, they certainly provide more independent grounds for comparison than CPU times, presented in Table 4, along with optimal distances.

Table 3 presents the mininum, maximum and average amounts of outer and BOX iterations, function evaluations, Quacan iterations and matrix-vector-products/conjugate-gradient iterations (for Box and LANCELOT, respectively). First row of each set corresponds to ALBOX and second to LANCELOT. Unfortunately the only statistics available for both is the number of function evaluations. We paired the number of matrix-vector-products (MVP) output by ALBOX with the number of conjugate-gradient iterations (CGI) produced by LANCELOT, since each conjugate-gradient iteration involves a matrix-vector-product.

Although the algorithms behave very differently timewise, as we will shortly see, this is not a direct consequence of the number of function evaluations each performs. The best least-squares fit by a first degree polynomial gives $y = 5.74631 + 0.855356\,x$, where y is the number of function evaluations of ALBOX and x is the corresponding amount for LANCELOT, whereas a similar fit involving CPU times will give a coefficient of less than a third. On Figure 2 we plot the function evaluation pairs for all eighteen instances along with the best fit obtained.

Problem size			Outer iter.			BOX iter.			Function eval.			Quacan iter.			MVP/CGI		
$\begin{bmatrix} n \\ p \end{bmatrix}$	var.	constr.	min.	max.	aver.	min.	max.	aver.	min.	max.	average	min.	max.	average	min.	max.	average
$\begin{bmatrix} 3 \\ 10 \end{bmatrix}$	76	55	4	5	4.64	21	55	34.74	25	77	45.52	309	2343	1063.70	340	2702	1194.70
						15	61	38.06	16	71	43.28				377	1949	992.02
$\begin{bmatrix} 3 \\ 11 \end{bmatrix}$	89	66	3	5	4.70	18	61	34.64	22	83	46.72	247	2940	1237.06	292	3292	1398.22
						20	62	38.02	21	80	43.44				511	2709	1031.52
$\begin{bmatrix} 3 \\ 12 \end{bmatrix}$	103	78	2	5	3.7	18	63	35.7	23	89	48.06	432	3302	1547.86	497	3677	1741.58
						22	58	39.66	24	66	45.34				553	1776	1069.04
$\begin{bmatrix} 3 \\ 13 \end{bmatrix}$	118	91	4	6	4.8	29	71	46.04	35	100	60.46	912	4279	2352.04	978	4719	2565.88
						24	85	51.58	25	94	59.18				991	3940	2044.16
$\begin{bmatrix} 3 \\ 14 \end{bmatrix}$	134	105	4	5	4.78	27	75	47.86	34	104	64.2	933	4923	2707.96	1017	5382	2962.84
						27	84	52.28	29	96	60.14				967	4248	2313.42
$\begin{bmatrix} 3 \\ 15 \end{bmatrix}$	151	120	4	5	4.56	32	110	60.16	41	140	77.76	1625	8385	4129.74	1751	8742	4443.5
						30	91	56.84	33	112	65.14				1107	5652	3014.98
$\begin{bmatrix} 4 \\ 22 \end{bmatrix}$	320	253	4	5	4.34	52	115	78.02	62	148	97.4	5688	16767	10502	6097	17871	11222.14
						45	225	104.12	49	262	120				5122	37546	12381.08
$\begin{bmatrix} 4 \\ 23 \end{bmatrix}$	346	276	3	6	4.7	50	168	89.86	59	199	111.04	5807	25830	13741.68	6240	26912	14598.14
						37	176	108.90	39	208	124.58				4799	29367	14607.08
$\begin{bmatrix} 4 \\ 24 \end{bmatrix}$	373	300	2	5	4.14	45	141	86.36	58	183	107.94	6282	28049	14076.54	6769	29825	15008.52
						49	210	118.02	53	251	136.04				6551	37259	17127.56

Table 3: ALBOX $\times$ LANCELOT test results

Problem size			Outer iter.			BOX iter.			Function eval.			Quacan iter.			MVP/CGI		
$\begin{bmatrix} n \\ p \end{bmatrix}$	var.	constr.	min.	max.	aver.	min.	max.	aver.	min.	max.	average	min.	max.	average	min.	max.	average
$\begin{bmatrix} 4 \\ 25 \end{bmatrix}$	401	325	4	5	4.18	63	180	97.8	75	226	120.6	10492	35660	17105.3	11034	37639	18143.06
						54	225	119.70	60	262	137.48				6870	38419	18736
$\begin{bmatrix} 4 \\ 26 \end{bmatrix}$	430	351	4	6	4.24	51	176	95.36	63	216	117.22	6765	38932	17185.16	7317	40932	18237.14
						53	266	131.40	59	311	150.5				5094	77233	21796.3
$\begin{bmatrix} 4 \\ 27 \end{bmatrix}$	460	378	4	5	4.3	62	206	99.48	76	254	122.12	11480	45129	19490.14	12169	47121	20616.06
						62	215	128.86	68	253	147.6				9420	41799	21533.96
$\begin{bmatrix} 5 \\ 37 \end{bmatrix}$	852	703	4	8	4.56	80	800	160.02	102	984	193.14	27836	471751	63777.98	29476	497038	67020.22
						85	334	190.42	95	381	218.6				9119	96036	56898.88
$\begin{bmatrix} 5 \\ 38 \end{bmatrix}$	894	741	4	6	4.56	89	600	166.2	107	717	200	29224	326333	67968.66	30804	340424	71260.98
						110	380	218.90	123	439	250.96				28894	161810	80371.82
$\begin{bmatrix} 5 \\ 39 \end{bmatrix}$	937	780	4	7	4.88	78	700	195.8	89	815	231.32	26692	448509	88565.94	27892	472730	92421.82
						91	385	231.24	99	453	263.44				24178	160611	85971.64
$\begin{bmatrix} 5 \\ 40 \end{bmatrix}$	981	820	4	7	4.94	90	700	202.86	106	880	242.94	34936	463883	98265.46	36614	485784	102815.68
						139	429	226.56	154	491	257.66				41872	180296	89191.74
$\begin{bmatrix} 5 \\ 41 \end{bmatrix}$	1026	861	4	8	4.86	93	800	225.08	117	954	271.64	36194	547421	117417.34	38311	577662	122923.74
						126	404	259.94	148	466	296.78				46484	202879	106628.42
$\begin{bmatrix} 5 \\ 42 \end{bmatrix}$	1072	903	4	7	4.64	109	700	212.34	132	887	256.08	47402	502810	109629.66	49993	529036	114743.76
						102	440	246.44	115	499	281.32				34730	200558	103288.96

Table 3: ALBOX $\times$ LANCELOT test results, cont.

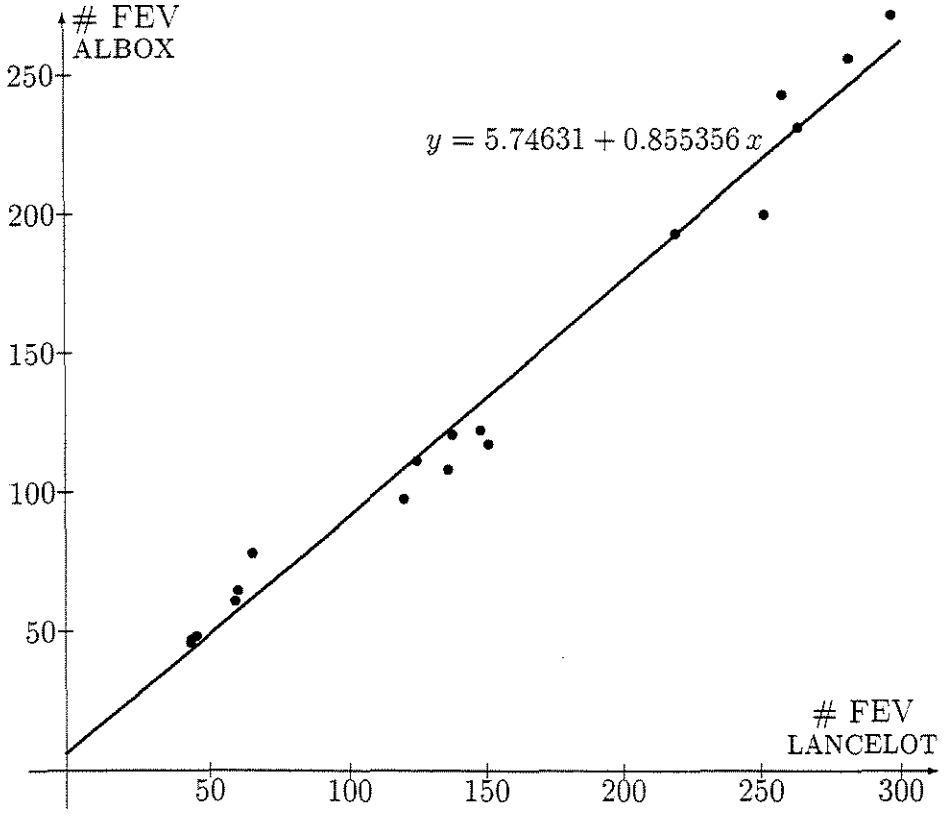


Figure 2: Number of function evaluations of LANCELOT versus ALBOX.

Further still from providing an explanation for the higher efficiency of ALBOX is the comparison of MVP versus CGI. In this case the best fit gives $y = -1320.36 + 1.10655x$, where y is the number of MVP and x is the number of CGI. This suggests that, although both iterations involve a matrix-vector-product, a CGI is substantially costlier, timewise, than the MVP performed in ALBOX. A main factor for this is that the matrix-vector-product in LANCELOT's conjugate gradient iteration deals with the true Hessian, whereas the one in ALBOX involves the approximate (and simpler) Hessian. Figure 3 contains the line corresponding to the best linear fit and the position of the (CGI, MVP) pairs.

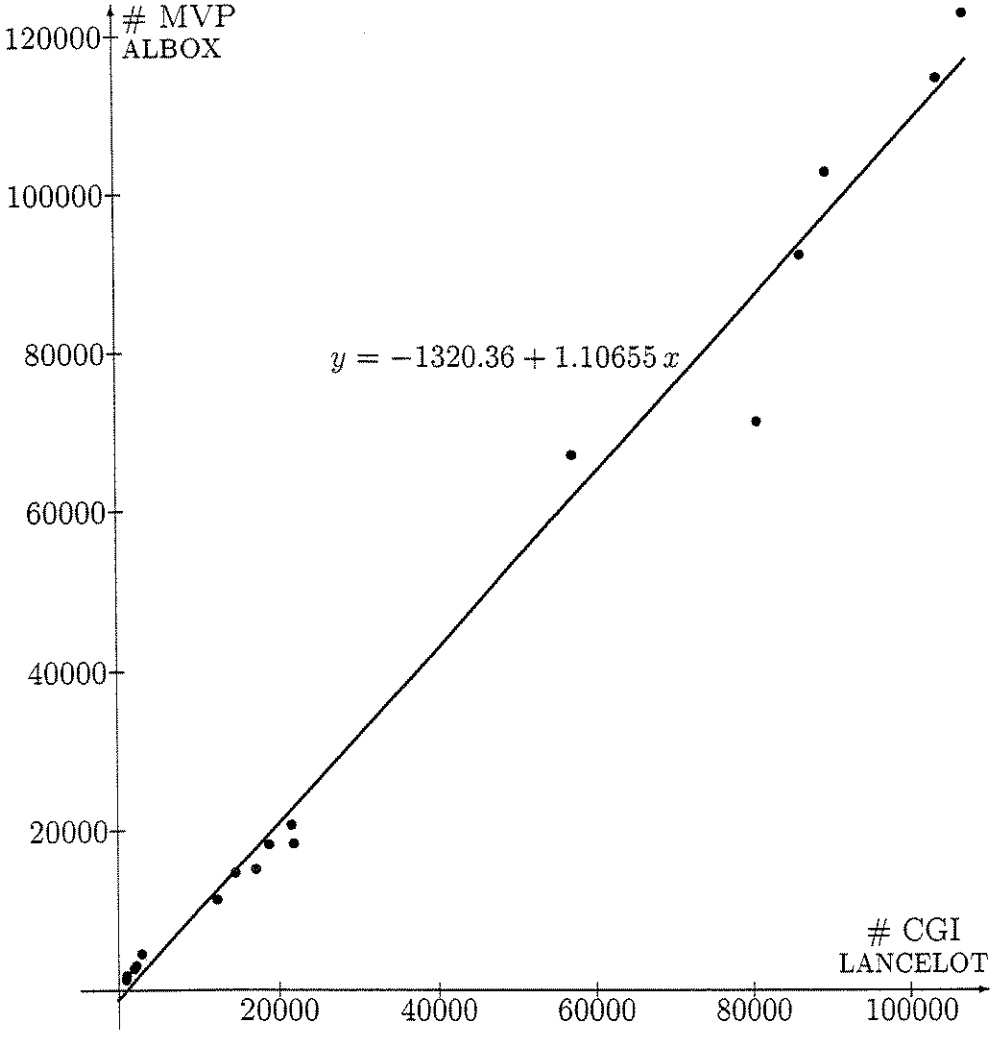


Figure 3: Number of CGIs of LANCELOT versus number of MVPs of ALBOX.

Next we have Table 4, that presents similar statistics involving the optimal distances encountered and the CPU times, in seconds. The first (resp., second) row for each (n, p) pair gives the numbers obtained by ALBOX (resp., LANCELOT).

The information contained in Table 4 is depicted graphically below. The intervals (min., max) of distances/CPU times are represented by vertical segments, the averages are indicated with a diamond symbol for ALBOX and a bullet for LANCELOT. Graphs on the left refer to distances whereas graphs on the right refer to CPU times.

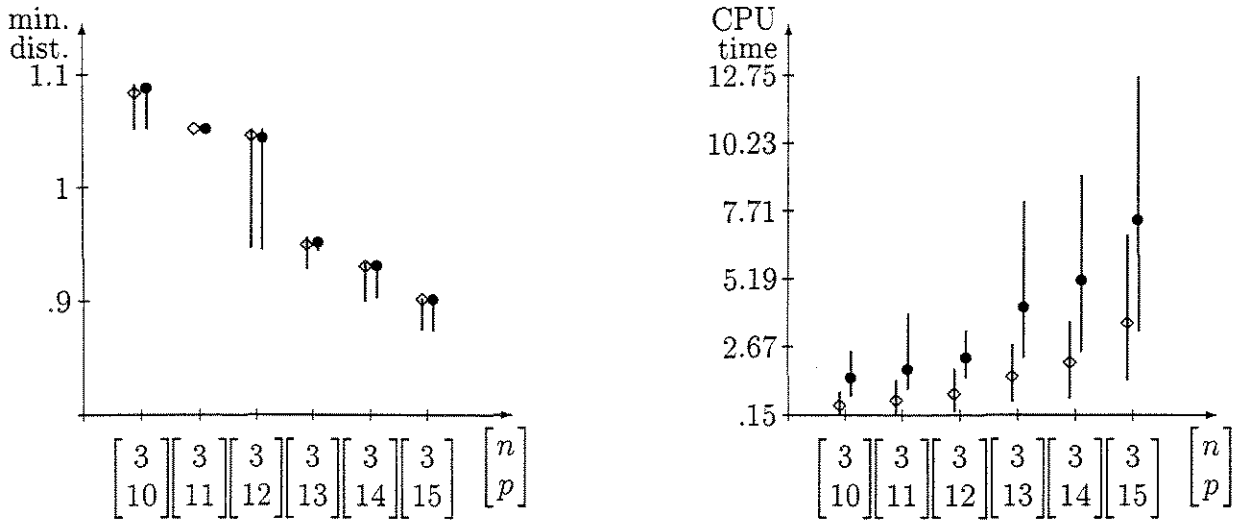


Figure 4: ALBOX ($\diamond$) and LANCELOT ($\bullet$) results for $n = 3$.

The graphs in Figures 4–6 evidence the qualitative relative behavior of both codes. Notice that the diamonds and bullets are always close together in the graphs on the left, indicating that the quality of the optimal solutions obtained by both codes is similar. On the other hand, the bullets rise faster than the diamonds on the graphs on the right, which means that the CPU times for LANCELOT tend to be higher than those for ALBOX. The linear fit of ALBOX CPU times versus LANCELOT CPU times, $y = 0.31054\,x$—the coefficient is less than one third—, ploted in Figure 7 confirms this.

Finally, it should be noted that CPU times increase sharply as a function of problem size (represented, for instance, by the number of constraints). We tried several fits (linear, quadratic, exponential) and, though none seemed to provide a very good model for the data, the quadratic fit was the best one.

Problem size			mininum distance between 2 points			CPU time (seconds)		
$\begin{bmatrix} n \\ p \end{bmatrix}$	var.	constr.	min.	max.	average	min.	max.	average
$\begin{bmatrix} 3 \\ 10 \end{bmatrix}$	76	55	1.0514622	1.0914262	1.08323633	0.170	1.010	0.476
			1.0514656	1.0914302	1.08740069	0.830	2.510	1.499
$\begin{bmatrix} 3 \\ 11 \end{bmatrix}$	89	66	1.0514622	1.0514622	1.05146223	0.170	1.420	0.636
			1.0514656	1.0514656	1.05146564	1.100	3.920	1.807
$\begin{bmatrix} 3 \\ 12 \end{bmatrix}$	103	78	0.9463817	1.0514622	1.04515739	0.290	1.870	0.906
			0.9447856	1.0514656	1.04306044	1.530	3.290	2.243
$\begin{bmatrix} 3 \\ 13 \end{bmatrix}$	118	91	0.9281797	0.9564136	0.94887941	0.640	2.770	1.546
			0.9443516	0.9564099	0.95127102	2.260	8.060	4.119
$\begin{bmatrix} 3 \\ 14 \end{bmatrix}$	134	105	0.8995904	0.9338626	0.92985570	0.750	3.640	2.060
			0.9025741	0.9338629	0.93055146	2.490	9.050	5.120
$\begin{bmatrix} 3 \\ 15 \end{bmatrix}$	151	120	0.8745438	0.9026561	0.90092361	1.420	6.830	3.536
			0.8734529	0.9026516	0.90092861	3.250	12.730	7.367
$\begin{bmatrix} 4 \\ 22 \end{bmatrix}$	320	253	0.9823127	1.0019894	0.99780958	11.560	31.310	20.032
			0.9840223	1.0019880	0.99676146	30.490	209.270	69.851
$\begin{bmatrix} 4 \\ 23 \end{bmatrix}$	346	276	0.9723134	0.9999999	0.98629231	12.080	52.350	27.931
			0.9740944	0.9918568	0.98476502	29.260	178.840	89.796
$\begin{bmatrix} 4 \\ 24 \end{bmatrix}$	373	300	0.9630174	0.9999999	0.97698487	14.830	61.260	30.918
			0.9580083	0.9828733	0.97519850	43.160	239.770	112.781
$\begin{bmatrix} 4 \\ 25 \end{bmatrix}$	401	325	0.9529038	0.9619429	0.95809771	25.150	86.570	41.290
			0.9465833	0.9619563	0.95749630	49.000	268.490	131.181
$\begin{bmatrix} 4 \\ 26 \end{bmatrix}$	430	351	0.9386238	0.9583427	0.95079467	18.830	106.930	46.326
			0.9367603	0.9583423	0.94916150	39.900	565.900	164.466
$\begin{bmatrix} 4 \\ 27 \end{bmatrix}$	460	378	0.9204666	0.9390862	0.93510260	32.770	126.840	55.681
			0.9273834	0.9389142	0.93457532	79.260	332.120	173.134
$\begin{bmatrix} 5 \\ 37 \end{bmatrix}$	852	703	0.9923706	1.0036282	0.99867534	163.670	2786.230	373.141
			0.9911508	1.0025367	0.99791248	444.810	2501.760	1154.082
$\begin{bmatrix} 5 \\ 38 \end{bmatrix}$	894	741	0.9864997	0.9977500	0.99318755	186.141	1988.719	409.580
			0.9864684	1.0019880	0.99307107	546.550	3105.860	1538.541
$\begin{bmatrix} 5 \\ 39 \end{bmatrix}$	937	780	0.9758879	0.9926700	0.98798399	166.400	2944.560	568.289
			0.9808159	0.9920786	0.98811785	502.380	3161.880	1782.302
$\begin{bmatrix} 5 \\ 40 \end{bmatrix}$	981	820	0.9750429	0.9883235	0.98179030	233.309	3114.030	657.233
			0.9701958	0.9920282	0.98108644	863.850	3820.430	1907.569
$\begin{bmatrix} 5 \\ 41 \end{bmatrix}$	1026	861	0.9685664	0.9835789	0.97594060	256.301	3834.699	812.938
			0.9644272	0.9819470	0.97574346	1148.770	4669.870	2521.840
$\begin{bmatrix} 5 \\ 42 \end{bmatrix}$	1072	903	0.9606935	0.9779378	0.96928704	420.170	4527.652	1000.608
			0.9599791	0.9798367	0.97025160	807.570	4664.630	2473.782

Table 4: Minimum distances and CPU times

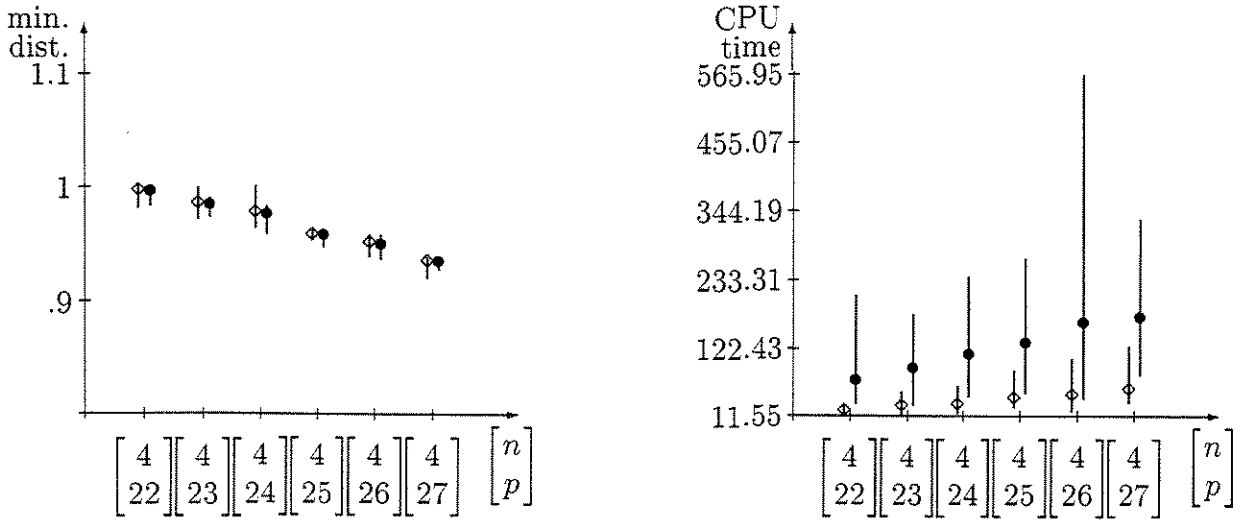


Figure 5: ALBOX ($\diamond$) and LANCELOT ($\bullet$) results for $n = 4$.

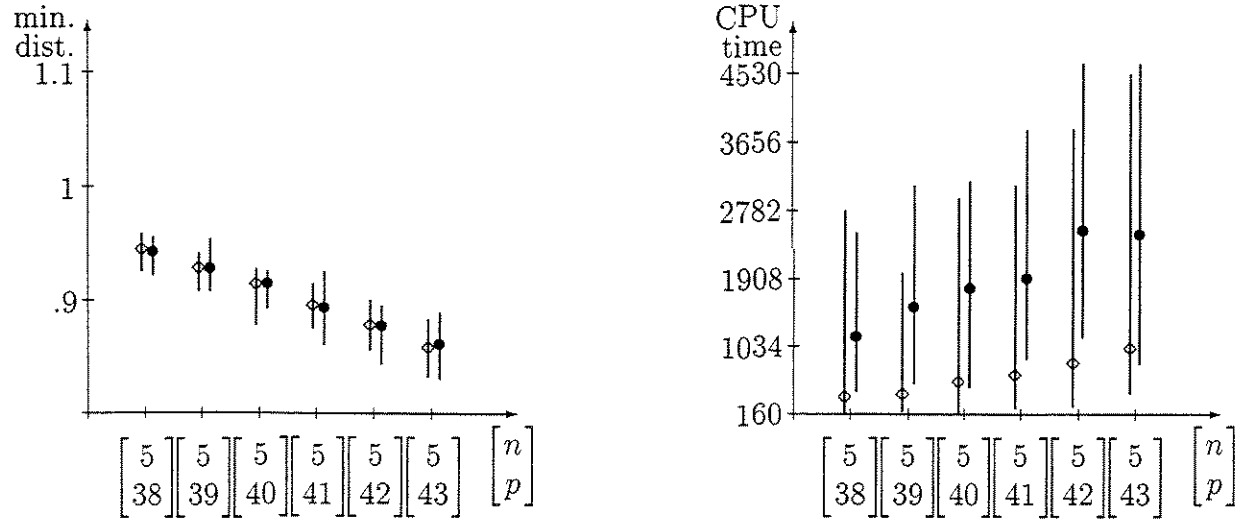


Figure 6: ALBOX ($\diamond$) and LANCELOT ($\bullet$) results for $n = 5$.

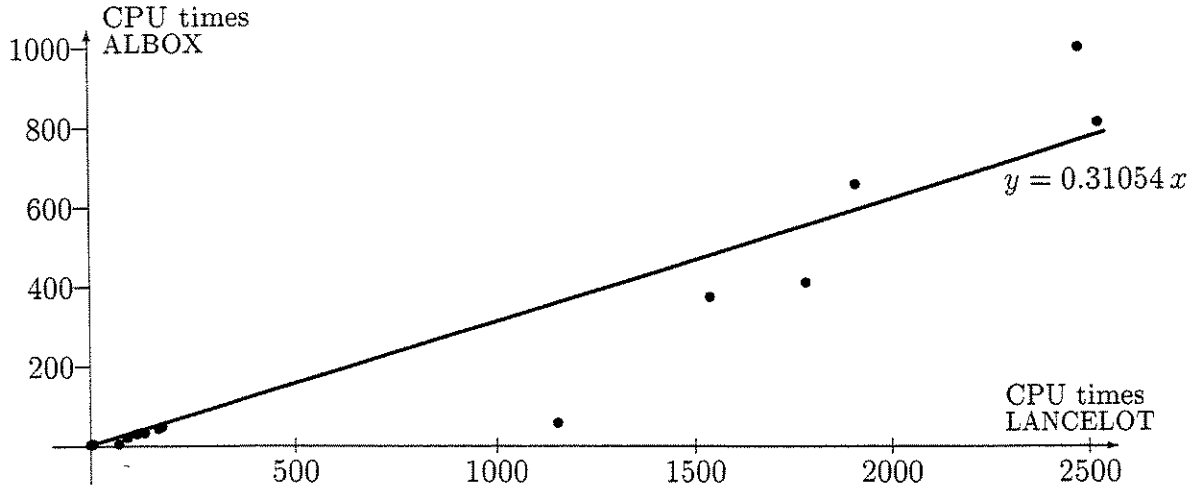


Figure 7: CPU times of LANCELOT versus those of ALBOX.

7 Conclusions

The main aspects of the Augmented Lagrangian methodology for solving large-scale nonlinear programming problems have been consolidated after the works of Conn, Gould and Toint which gave origin to the LANCELOT package. This algorithmic framework has been very useful in the last ten years for solving practical problems and for comparison purposes with innovative nonlinear programming methods. Very likely, this tendency will be maintained in the near future.

The present research was born as a result of our need to have more freedom in the formulation and resolution of the quadratic subproblems that arise in the LANCELOT-like approach to the Augmented Lagrangian philosophy. On one hand, we decided to exploit more deeply the whole trust region by means of the use of a box-constraint quadratic solver. On the other hand, perhaps more importantly, we tested a Gauss-Newton convex simplification of the quadratic model which turned out to be much more efficient than the straight Newton-like version of this model. Behind this gain of efficiency is the fact that the quadratic solver, though able to deal with nonconvex models, is far more efficient when the underlying quadratic has a positive semidefinite Hessian. It is usual, in Numerical Analysis, that a decision on the implementation of a high level algorithm depends on the current technology for solving low-level subproblems. It must only be warned that such a decision could change if new more efficient algorithms for solving the subproblems (nonconvex quadratic programming in our case) are found.

Our main objective is to use ALBOX, not only for solving real-life problems, but also for testing alternative nonlinear programming methods against it. We feel

that having a deep knowledge of the implementation details of the code will enable us to be much more exacting when testing new codes, since it will be possible to fine tune the standard against which the new code is tested. The present study, apart from calling the reader's attention to convex simplified Gauss-Newton like subproblems, had the objective of validating our code, by means of its comparison with LANCELOT, using a set of problems that have an independent interest. The result of this comparison seems to indicate that ALBOX can be used as a competitive tool for nonlinear programming calculations.

References

[1] R.H. Bielschowsky, A. Friedlander, F.M. Gomes, J.M. Martínez and M. Raydan. An adaptative algorithm for bound constrained quadratic minimization. Technical report, Institute of Mathematics, State University of Campinas, 1995.

[2] A.R. Conn, N.I.M. Gould and Ph.L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, 28:545–572, 1991.

[3] A.R. Conn, N.I.M. Gould and Ph.L. Toint. *LANCELOT, a Fortran Package for Large-Scale Nonlinear Optimization (Release A).* Springer-Verlag, Berlin, 1992.

[4] A. R. Conn, N. I. M. Gould and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis* 25:433–460, 1988. See also *SIAM Journal on Numerical Analysis* 26:764–767, 1989.

[5] J.H. Conway and N.J.C. Sloane. *Sphere Packings, Lattices and Groups.* Springer-Verlag, New York, 1988.

[6] K. Devlin. *Mathematics: The Science of Patterns.* Scientific American Library, New York, 1994.

[7] M.A. Diniz-Ehrhardt, M.A. Gomes-Ruggiero and S.A. Santos. Comparing the numerical performance of two trust-region algorithms for large-scale bound-constrained minimization. In *Revista Latinoamericana de Investigación Operativa*, Vol. 7, N. 1–2, pp. 23–54.

[8] Z. Dostál, A. Friedlander and S.A. Santos. Augmented Lagrangians with adaptative precision control for quadratic programming with equality constraints. To appear in *Computational Optimization and Applications.*

[9] A. Friedlander and J.M. Martínez. On the maximization of a concave quadratic function with box constraints. *SIAM Journal on Optimization*, 4:177–192, 1994.

[10] A. Friedlander, J.M. Martínez and S.A. Santos. A new trust-region algorithm for bound constrained minimization. *Applied Mathematics and Optimization*, 30:235–266, 1994.

[11] F. M. Gomes, M. C. Maciel and J.M. Martínez. Nonlinear programming algorithms using trust regions and augmented Lagrangians with nonmonotone penalty parameters. To appear in *Mathematical Programming.*

[12] W. W. Hager. Analysis and implementation of a dual algorithm for constraint optimization. *Journal of Optimization Theory and Applications* 79:427–462, 1993.

[13] W. W. Hager. Dual techniques for constraint optimization. *Journal of Optimization Theory and Applications* 55:37–71, 1987.

[14] N. Maculan, P. Michelon and J. MacGregor Smith. Bounds on the kissing numbers in $\mathbb{R}^n$: mathematical programming formulations. Technical report, COPPE, University of Rio de Janeiro, Brazil, 1996.

[15] J.M. Martínez. Augmented Lagrangians and the resolution of packing problems. Technical Report 08/97, Institute of Mathematics, University of Campinas, Brazil.

[16] J.M. Martínez and E. A. Pilotta. Inexact-Restoration algorithm for constrained optimization. To appear in *Journal of Optimization Theory and Applications.*

[17] J.M. Martínez. A Two-Phase Model Algorithm with global convergence for Nonlinear Programming. to appear in *Journal of Optimization Theory and Applications*, Vol. 96, Feb. 1998.

[18] S.G. Nash. Preconditioning of truncated-newton methods. *SIAM Journal on Scientific and Statistical Computing*, 6:599–616, 1985.

[19] E.B. Saff and A.B.J. Kuijlaars. Distributing many points on a sphere. *Mathematical Intelligencer*, 19:5–11, 1997.

Capítulo 2

Um algoritmo de Restauração–Inexata para minimização com restrições

Resumo

É apresentado um novo algoritmo modelo para resolver problemas de programação não linear. Não são acrescentadas variáveis de folga para tratar as restrições de desigualdade. Cada iteração do método procede em duas fases. Na primeira fase, é melhorada a factibilidade da aproximação atual e na segunda fase o valor da função objetivo é reduzido num conjunto factível aproximado. O ponto que resulta da segunda fase é comparado com o ponto atual usando uma função de mérito que combina factibilidade e otimalidade. Esta função de mérito inclui um parâmetro de penalização que muda em cada iteração. Além disso, para este parâmetro de penalização é implementado um adequado procedimento pelo qual esse parâmetro pode crescer ou decrescer ao longo das iterações. As condições para a factibilidade na primeira fase e para a otimalidade na segunda são simples e o método resultante admite implementações para problemas de grande porte. Provamos com adequadas hipóteses, e sem usar regularidade ou existência de derivadas segundas, que todos os pontos limites de uma seqüência gerada pelo algoritmo são factíveis, e que uma adequada medida da otimalidade pode ser feita tão pequena quando se desejar. O algoritmo é implementado e comparado com LANCELOT usando um conjunto de *Hard-Spheres problems.*

Inexact–Restoration Algorithm for Constrained Optimization

José Mario Martínez * Elvio A. Pilotta †

September 29, 1998

Abstract

We introduce a new model algorithm for solving nonlinear programming problems. No slack variables are introduced for dealing with inequality constraints. Each iteration of the method proceeds in two phases. In the first phase, feasibility of the current iterate is improved and in second phase the objective function value is reduced in an approximate feasible set. The point that results from the second phase is compared with the current point using a nonsmooth merit function that combines feasibility and optimality. This merit function includes a penalty parameter that changes between different iterations. A suitable updating procedure for this penalty parameter is included by means of which it can be increased or decreased along different iterations. The conditions for feasibility improvement at the first phase and for optimality improvement at the second phase are mild, and large-scale implementations of the resulting method are possible. We prove that under suitable conditions, that do not include regularity or existence of second derivatives, all the limit points of an infinite sequence generated by the algorithm are feasible, and that a suitable optimality measure can be made as small as desired. The algorithm is implemented and tested against LANCELOT using a set of hard-spheres problems.

Key words: Nonlinear programming, trust regions, feasible methods, global convergence, numerical experiments.

*Departamento de Matemática Aplicada, IMECC-UNICAMP, CP 6065, 13081-970 Campinas SP, Brazil (martinez@ime.unicamp.br). Work sponsored by FAPESP (Grant 90-3724-6), CNPq and FAEP-UNICAMP.

†Departamento de Matemática Aplicada, IMECC-UNICAMP, CP 6065, 13081-970 Campinas SP, Brazil (pilotta@ime.unicamp.br). Work sponsored by FAPESP (Grant 96-0681-0), CNPq and FAEP-UNICAMP.

1 Introduction

Feasible methods for solving minimization problems with inequality and equality constraints [1, 2, 17, 19, 20, 21, 22, 27, 28, 29, 30, 31] have a strong reputation among practitioners of nonlinear programming and, for this reason, are incorporated to well known user-oriented libraries. The reason is that, very frequently, feasible nonoptimal solutions are useful in engineering applications, whereas nonfeasible approximations are not, even when they are "quasi-optimal". In the 80's very few papers in the mainstream of the optimization literature were dedicated to feasible methods. That decade was dominated by SQP (sequential quadratic programming) models and the usual criticism against feasible methods was that it is very difficult and, frequently, not worthwhile, to follow very curved feasible regions, especially when the current approximation is far from the solution. In the last few years (we write in 1998) many researchers realized that at least a subfamily of feasible methods (those based on the barrier approach) was perhaps unfairly despised. See [33]. Obviously, the barrier approach is not applicable to equality constraints and must be combined with SQP-like schemes in order to deal with equalities.

The preference for feasibility cannot be ignored in practical applications but, on the other hand, the SQP criticism based on high-curvature domains must also be taken into account. These two facts motivated us to develop (see [18]) theoretically justified algorithms for constraints of the form $h(x) = 0, \quad \ell \leq x \leq u$ where feasibility is controlled at every iteration, with an internal mechanism that automatically determines the degree of precision required in the constraints. An interesting related method that does not use merit functions was introduced in [2]. We notice that some practical SGRA algorithms [20, 21, 22] successfully used "Inexact-Restoration" procedures in applications.

In [18] we need to introduce slack variables for dealing with inequality constraints, so that the feasible region takes the the canonical form above. This transformation can increase the number of variables in an undesirable way, leading to expensive subproblems. Therefore, it is interesting to introduce Inexact–Restoration algorithms that deal with inequality constraints without the slack–variable transformation.

Let us state the nonlinear programming problem in the form

$$\text{Minimize } f(x)$$

$$\text{subject to } C(x) \leq 0, \quad x \in \Omega, \tag{1}$$

where $f : I\!R^n \to I\!R$ and $C : I\!R^n \to I\!R^m$ are continuously differentiable and $\Omega \subset I\!R^n$ is closed and convex. In practice, we are mostly interested in the case in which Ω is a polytope. Each equality constraint appearing in the original formulation of the nonlinear programming problem can be transformed into two inequality constraints. It will be seen that this does not increase the complexity of the method introduced here.

The new model algorithm generates feasible iterates with respect to Ω ($x^k \in \Omega$ for all $k = 0, 1, 2, \ldots$) Each iteration includes two different procedures: Restoration and Minimization. In the Restoration Step (which is executed once per iteration) an intermediate point $y^k \in \Omega$ is found such that the infeasibility at y^k is a fraction of the infeasibility at x^k. Immediately after Restoration we construct an approximation π_k of the feasible region using available information at y^k. In the Minimization Step we compute a trial point $z^{k,i} \in \pi_k$ such that $f(z^{k,i}) << f(y^k)$ ($<<$ means "sufficiently smaller than" here) and $\|z^{k,i} - y^k\| \leq \delta_{k,i}$, where $\delta_{k,i}$ is a trust-region radius. The trial point $z^{k,i}$ is accepted as new iterate if the value of a nonsmooth (exact penalty) merit function at $z^{k,i}$ is sufficiently smaller than its value at x^k. If $z^{k,i}$ is not acceptable, the trust-region radius is reduced.

When Ω is a polytope, the approximate feasible region π_k is a polytope too. So, if $\|\cdot\|$ is the sup-norm, the Minimization Step consists of an inexact (approximate) minimization of f with linear constraints. In that case, the Restoration Step also represents an inexact minimization of infeasibility with linear constraints. Therefore, available algorithms for (large-scale) linearly constrained minimization (see [13, 14, 23]) can be fully exploited.

As mentioned above, the new algorithm is related to classical *feasible* methods for nonlinear programming, such as the Generalized Reduced Gradient (GRG) method and the family of Sequential Gradient Restoration algorithms (SGRA). See [1, 2, 17, 20, 21, 22, 27, 28, 29, 30, 31]. However, in our approach the successive approximations to the solution of (1) are not necessarily feasible (or nearly feasible) with respect to $C(x) \leq 0$. In spite of that, the necessity of considering and probably improving feasibility is taken actively into account at all the iterations. This strategy is quite different than the one adopted in Sequential Quadratic Programming (SQP) algorithms, where the trial point at each iteration is obtained after considering only a linear model of the constraints.

The convergence theory developed in this paper has several points in common with global convergence theories for different SQP-like algorithms with trust-regions (see [5, 10, 12, 25, 26]), in particular the one developed in [15]. The new model algorithm is also related to the method introduced in [18] for problems where the constraints are given in the form $C(x) = 0$, $x \in \Omega$. In [18] the merit function is an augmented Lagrangian, while here we consider the exact penalty-like merit function used, for example, in [3, 4, 16, 25] for forcing convergence of SQP and other nonlinear programming algorithms. Another remarkable difference is that the algorithm introduced in this paper use trust-regions centered on the intermediate point y^k instead of the more usual trust-regions centered on the current point x^k. Consequently, only the Minimization Step is repeated after a reduction of the trust-region radius.

A rigorous description of the new model algorithm is given in Section 2, together with further motivation. In Section 3 we prove that the algorithm is well defined,

that is, given a current point $x^k \in \Omega$ that does not satisfy the stopping criteria, a new iterate x^{k+1} is found after a finite number of reductions of the trust-region radius. In the same section we prove that, when an infinite sequence is generated, we obtain points arbitrarily close to feasibility. In Section 4 we prove that a quantity that measures first-order optimality can be made as small as desired. In Section 5 we give an application and we describe the practical implementation oriented to it. In Section 6 we compare our implementation against the well-known augmented Lagrangian code LANCELOT. Conclusions are given in Section 7.

Notation.

In this work we use two (perhaps different) norms. We denote $|\cdot|$ a monotone norm on $\mathbb{R}^m$ ($|v| \leq |w|$ whenever $0 \leq v \leq w$) and $\|\cdot\|$ an arbitrary norm on $\mathbb{R}^n$.

We denote $C'(x) \in \mathbb{R}^{m\times n}$ the Jacobian matrix of $C(x)$ and $C_j'(x) = \nabla C_j(x)^T$ for all $j = 1, \ldots, m$.

We also denote $C_j^+(x) = \max\{C_j(x), 0\}$ and $C^+(x) = (C_1^+(x), \ldots, C_m^+(x))^T$.

2 Description of the Model Algorithm

Before giving a rigorous description of the algorithm, we will comment some of its main features.

2.1 Restoration Step

As we mentioned in the Introduction, given the current iterate $x^k \in \Omega$, the model algorithm computes an intermediate "more feasible" point $y^k \in \Omega$. The conditions that must be satisfied by y^k are

$$|C^+(y^k)| \leq r|C^+(x^k)| \tag{2}$$

$$\|y^k - x^k\| \leq \beta|C^+(x^k)|. \tag{3}$$

where $r \in [0, 1)$ and $\beta > 0$ are parameters given independently of k. Condition (2) states the necessity of having an intermediate point at least as feasible as x^k. Condition (3) imposes that y^k must be equal to x^k if the current point is feasible.

2.2 Approximate Linearized Feasible Region

After the computation of y^k with the conditions (2) and (3) we define a linear approximation of the feasible region of (1), containing the intermediate point y^k. This auxiliary region is given by

$$\pi_k = \{x \in \Omega \mid C_j(y^k) + C_j'(y^k)(x - y^k) \leq C_j^+(y^k) \text{ whenever } C_j(y^k) \geq -p\}, \tag{4}$$

where $p > 0$ is a parameter given independently of the iteration index k. So, π_k is the intersection of Ω with the linear approximations of the sets $C_j(x) \leq C_j^+(y^k)$, excluding the indices j that correspond to constraints that, according to the tolerance p, are strongly satisfied at y^k. If p is large the approximate feasible region takes into account all the constraints $C_j(x) \leq 0$, independently of $C_j(y^k)$. On the other hand, if p is small, only the constraints violated at y^k tend to be considered in the definition of π_k. In other words, if $C_j(y^k) < -p$, it is considered that the approximation of the set $C_j(x) \leq 0$ that uses information at y^k is the whole space $\mathbb{R}^n$. In principle, it should be better to use a large p, for this gives a more faithful representation of the true feasible region. However, the subproblem involved in the Minimization Step is simpler when p is small.

2.3 Minimization Step

The objective of the Minimization Step is to obtain $z^{k,i} \in \pi_k \cap \mathbb{B}_{k,i}$ such that $f(z^{k,i}) << f(y^k)$, where

$$\mathbb{B}_{k,i} = \{x \in \mathbb{R}^n \mid \|x - y^k\| \leq \delta_{k,i}\}, \tag{5}$$

and $\delta_{k,i} > 0$ is a trust-region radius. The first trial point at each iteration is obtained using a trust-region radius $\delta_{k,0}$. Successive trust-region radius are tried until a point $z^{k,i}$ is found such that the merit function at this point is sufficiently smaller than the merit function at x^k.

The minimization step is preceded by the computation of the Cauchy-like direction (independent of i)

$$d^{k,tan} = P_k(y^k - \eta \nabla f(y^k)) - y^k, \tag{6}$$

where $P_k(z)$ denotes the orthogonal projection of z on π_k and $\eta > 0$ is an arbitrary scaling parameter independent of k. It turns out that $d^{k,tan}$ is a feasible descent direction for f on π_k. Its norm will be used to define a convergence criterion for the algorithm. The trial point $y^k + d^{k,tan}$ belongs to π_k but it does not necessarily belong to $\mathbb{B}_{k,i}$. So, we define the breakpoint $y^k + t_{(k,i,break)}d^{k,tan}$ by

$$t_{(k,i,break)} = \sup \{t \in [0,1] \mid [y^k, y^k + td^{k,tan}] \subset \mathbb{B}_{k,i}\}. \tag{7}$$

Moreover, the value of the objective function f at $y^k + t_{(k,i,break)}d^{k,tan}$ is not necessarily smaller than $f(y^k)$, therefore a sufficiently smaller functional value $f(y^k + t_{(k,i,dec)}d^{k,tan})$ must be obtained using a classical backtracking procedure. Finally, $z^{k,i} \in \pi_k \cap \mathbb{B}_{k,i}$ will be any point such that $f(z^{k,i}) \leq f(y^k + t_{(k,i,dec)}d^{k,tan})$. Alternatively, $z^{k,i}$ can be any point of $\pi_k \cap \mathbb{B}_{k,i}$ such that $f(z^{k,i}) \leq f(y^k) - \tau_1\delta_{k,i}$ or $f(z^{k,i}) \leq f(y^k) - \tau_2$, where τ_1 and τ_2 are nonnegative parameters of the algorithm.

This means that, for computing the trial point $z^{k,i}$ in an efficient way, we can apply any reasonable algorithm (with a mild convergence criterion) to the resolution of the minimization problem

$$\text{Minimize } f(x) \quad \text{subject to} \quad x \in \pi_k \cap I\!B_{k,i}. \tag{8}$$

Clearly, (8) is a linearly constrained optimization problem if $\|\cdot\|$ is the sup-norm.

2.4 Merit Function and Penalty Parameter

The comparison of $z^{k,i}$ and x^k involves the evaluation of a merit function at both points. We decided to use the exact penalty-like nonsmooth merit function, given by

$$\psi(x, \theta) = \theta f(x) + (1 - \theta)|C^+(x)| \tag{9}$$

where $\theta \in (0, 1]$ is a penalty parameter used to give different weights to the objective function and to the feasibility objective. The choice of the parameter θ at each iteration depends of practical and theoretical considerations. For example, if $|C^+(x^k)|$ is large, the weight assigned to $f(x)$ must be small, for it does not make sense to worry about the functional values if the current point is far from the feasible region. Our choice of the penalty parameter automatically takes into account this practical necessity.

Roughly speaking, we wish that the merit function at the new point should be less than the merit function at the current point x^k. That is, we want $\mathbf{Ared}_{k,i} > 0$, where $\mathbf{Ared}_{k,i}$, the "actual reduction of the merit function", is defined by

$$\mathbf{Ared}_{k,i} = \psi(x^k, \theta_{k,i}) - \psi(z^{k,i}, \theta_{k,i}). \tag{10}$$

So,

$$\mathbf{Ared}_{k,i} = \theta_{k,i}[f(x^k) - f(z^{k,i})] + (1 - \theta_{k,i})[|C^+(x^k)| - |C^+(z^{k,i})|].$$

However, as in unconstrained optimization, merely a reduction of the merit function is not sufficient to guarantee convergence. In fact, we need a "sufficient reduction" of the merit function, that will be defined by the satisfaction of the following test:

$$\mathbf{Ared}_{k,i} \geq 0.1\mathbf{Pred}_{k,i}, \tag{11}$$

where $\mathbf{Pred}_{k,i}$ is a positive "predicted reduction" of the merit function between x^k and $z^{k,i}$. In our case, we define

$$\mathbf{Pred}_{k,i} = \theta_{k,i}[f(x^k) - f(z^{k,i})] + (1 - \theta_{k,i})[|C^+(x^k)| - |C^+(y^k)|]. \tag{12}$$

The quantity $\mathbf{Pred}_{k,i}$ defined above can be nonpositive depending on the value of the penalty parameter. Fortunately, if $\theta_{k,i}$ is small enough, $\mathbf{Pred}_{k,i}$ is arbitrarily close

to $|C(x^k)| - |C(y^k)|$ which is necessarily nonnegative. Therefore, we will always be able to choose $\theta_{k,i} \in (0,1]$ such that

$$\mathbf{Pred}_{k,i} \geq \frac{1}{2}[|C^+(x^k)| - |C^+(y^k)|]. \tag{13}$$

When the criterion (11) is satisfied, we accept $x^{k+1} = z^{k,i}$. Otherwise, we reduce the trust-region radius.

2.5 Description of the Model Algorithm

Assume that $p > 0$, $\eta > 0$, $\beta > 0$, $r \in [0,1)$, $\delta_{min} > 0$, $\tau_1 > 0, \tau_2 > 0$ are algorithmic parameters given independently of k and $\sum_{k=0}^{\infty} \omega_k$ is a convergent series of nonnegative terms. Suppose that $x^0 \in \Omega$ is an initial approximation to the solution and that $\theta_{-1} \in (0,1)$ is an initialization of the penalty parameter. Given $x^k \in \Omega$, $\theta_{k-1} \in (0,1]$, $\delta_{k,0} \geq \delta_{min}$, the steps for computing x^{k+1} or for stopping the process are given by the following algorithm.

Algorithm 2.1

Step 1. *Compute y^k, $d^{k,tan}$ and decide termination*

Compute $y^k \in \Omega$ such that (2) and (3) hold. If this is not possible, stop the execution of the algorithm declaring "failure in improving feasibility". Otherwise, set $i \leftarrow 0$, define

$$\theta_{k,-1} = \min\{1, \min\{\theta_{-1}, \ldots, \theta_{k-1}\} + \omega_k\}$$

and compute $d^{k,tan}$ using (6). If $C^+(x^k) = 0$ and $d^{k,tan} = 0$ terminate the execution of the algorithm declaring "finite convergence".

Step 2. *Minimization Step*

Compute $t_{(k,i,break)}$ using (7). Define $t_{(k,i,dec)}$ as the first term t of the sequence $\{t_{k,1}, t_{k,2}, \ldots\}$ such that

$$f(y^k + td^{k,tan}) \leq f(y^k) + 0.1t\langle \nabla f(y^k), d^{k,tan}\rangle, \tag{14}$$

where $\{t_{k,j}\}$ is defined by $t_{k,1} = t_{(k,i,break)}$ and $t_{k,j+1} \in [0.1t_{k,j}, 0.9t_{k,j}]$ for all $j = 1, 2, \ldots$

Compute $z^{k,i} \in \pi_k \cap I\!B_{k,i}$ such that

$$f(z^{k,i}) \leq \max\{f(y^k + t_{(k,i,dec)}d^{k,tan}),\ f(y^k) - \tau_1\delta_{k,i},\ f(y^k) - \tau_2\}. \tag{15}$$

Step 3. *Choice of the penalty parameter*

Define, for all $\theta \in [0,1]$,

$$Pred_{k,i}(\theta) = \theta[f(x^k) - f(z^{k,i})] + (1-\theta)[|C^+(x^k)| - |C^+(y^k)|].$$

Choose $\theta_{k,i}$ the supremum of the values of θ in the interval $[0, \theta_{k,i-1}]$ such that

$$Pred_{k,i}(\theta) \geq \frac{1}{2}[|C^+(x^k)| - |C^+(y^k)|]. \tag{16}$$

Step 4. *Acceptance or rejection of the trial point*

Define $\mathbf{Ared}_{k,i}$ and $\mathbf{Pred}_{k,i}$ as in (10) and (12) respectively. If the test (11) is satisfied, define $x^{k+1} = z^{k,i}$, $\theta_k = \theta_{k,i}$, $iacc(k) = i$ ("*iacc*" means "accepted i") and finish the iteration. If (11) does not hold, choose $\delta_{k,i+1} \in [0.1\delta_{k,i}, 0.9\delta_{k,i}]$, set $i \leftarrow i+1$ and go to Step 2.

2.6 Some Remarks and Elementary Properties

By means of the introduction of the nonnegative parameters ω_k a "moderate" increase of the penalty parameter between different iterations is permitted. This prevents the possibility of inheriting artificially small penalty parameters from the very beginning of the iterative process. It is easy to see that the sequence of penalty parameters finally used at each iteration $\{\theta_k\}$ is convergent. In fact, defining $\theta_{k,small} = \min\{\theta_{-1}, \ldots, \theta_k\}$ and $\theta_{k,large} = \theta_{k,small} + \omega_k$, we see that $\theta_{k+1} \leq \theta_{k,large}$ and $\theta_k \geq \theta_{k,small}$ for all k. Clearly, $\{\theta_{k,large}\}$ and $\{\theta_{k,small}\}$ are convergent to the same limit, so $\{\theta_k\}$ is also convergent. We can also prove, by induction, that $\theta_{k,i} > 0$ for all k, i.

It is easy to verify that $d^{k,tan}$ is a descent direction. In fact, since $y^k \in \pi_k$, we have that

$$\|(y^k - \eta\nabla f(y^k)) - P_k(y^k - \eta\nabla f(y^k))\|_2 \leq \|(y^k - \eta\nabla f(y^k)) - y^k\|_2.$$

Therefore,

$$\|y^k - P_k(y^k - \eta\nabla f(y^k))\|_2^2 + \|\eta\nabla f(y^k)\|_2^2 + 2\eta\langle P_k(y^k - \eta\nabla f(y^k)) - y^k, \nabla f(y^k)\rangle \leq \|\eta\nabla f(y^k)\|_2^2,$$

so,

$$\langle d^{k,tan}, \nabla f(y^k)\rangle \leq -\frac{1}{2\eta}\|d^{k,tan}\|_2^2 \leq -\frac{c}{2\eta}\|d^{k,tan}\|^2, \tag{17}$$

where $c > 0$ is a norm-dependent constant. We can use classical arguments for justifying backtracking with Armijo-like conditions (see [11], Chapter 6), to show that $t_{(k,i,dec)}$ is well defined at Step 2 of Algorithm 2.1. In other words, given the current point x^k and the trust-region radius $\delta_{k,i}$ it is possible to compute, in finite time, the trial point $z^{k,i}$.

3 General Assumptions and Consequences

From now on, we will suppose that the nonlinear programming problem (1) satisfies the assumptions A1, A2 and A3 stated below. These will be the only assumptions on the problem that are needed for proving convergence. In particular, no regularity assumptions are used in the proofs and second derivatives of f and C are not assumed to exist.

A1. Ω is convex and compact.
A2. The Jacobian matrix of $C(x)$ exists and satisfies the Lipschitz condition

$$\|C'(y) - C'(x)\| \leq L_1\|y - x\| \text{ for all } x, y \in \Omega. \tag{18}$$

A3. The gradient of f exists and satisfies the Lipschitz condition

$$\|\nabla f(y) - \nabla f(x)\| \leq L_2\|y - x\| \text{ for all } x, y \in \Omega. \tag{19}$$

Due to the equivalence of norms on $\mathbb{R}^n$, similar conditions to (18) and (19) hold if we consider different norms than $\|\cdot\|$. So, in order to simplify the notation, we can assume that (18) and (19) hold with the same constants L_1 and L_2 for all the norms considered in this work. From these Lipschitz conditions it follows that

$$\|C(y) - C(x) - C'(x)(y - x)\| \leq \frac{L_1}{2}\|y - x\|^2 \tag{20}$$

and

$$|f(y) - f(x) - \langle \nabla f(x), y - x\rangle| \leq \frac{L_2}{2}\|y - x\|^2 \tag{21}$$

for all $x, y \in \Omega$. Again, we can assume, without loss of generality, that (20) and (21) hold for different norms with the same constants and that

$$|C_j(y) - C_j(x) - C_j'(x)(y - x)| \leq \frac{L_1}{2}\|y - x\|^2 \tag{22}$$

for all $j = 1, \ldots, m$.

The assumption on the boundedness of Ω can be replaced by hypotheses that state boundedness of a set of quantities depending on the iterates. This is frequently done in global convergence theories for SQP algorithms. We prefer to state directly Assumption A1 since it seems to be the only reasonable assumption *on the problem* that guarantees boundedness of the required quantities.

The following theorem is directly deduced from the general assumptions. It states a bounded deterioration result for the feasibility of $z^{k,i}$ in relation to the feasibility

of y^k. Briefly speaking, we prove that only a second order deterioration of feasibility can be expected for a trial point $x \in \pi_k$.

Theorem 3.1. *There exists $c_1 > 0$ (independent of k) such that, whenever $y^k \in \Omega$ is defined and $x \in \pi_k$, we have*

$$|C^+(x)| \leq |C^+(y^k)| + c_1\|x - y^k\|^2 \tag{23}$$

Proof. Let $j \in \{1, \ldots, m\}$. By the compactness of Ω and the continuity of C_j there exists $\rho > 0$ such that whenever $C_j(y) < -p$ and $C_j(x) \geq 0$ it holds that $\|x-y\| \geq \rho$.

If $C_j^+(x) = 0$, the inequality

$$C_j^+(x) \leq C_j^+(y^k) \tag{24}$$

holds trivially. If $C_j^+(x) > 0$ we analyze three different cases.
Case 1: If $C_j(y^k) \geq 0$ (so $C_j^+(y^k) = C_j(y^k)$) we have, by (22) that

$$C_j(x) \leq C_j(y^k) + C_j'(y^k)(x - y^k) + \frac{L_1}{2}\|x - y^k\|^2.$$

So, if $x \in \pi_k$,

$$C_j(x) \leq C_j^+(y^k) + \frac{L_1}{2}\|x - y^k\|^2.$$

Therefore,

$$C_j^+(x) \leq C_j^+(y^k) + \frac{L_1}{2}\|x - y^k\|^2. \tag{25}$$

Case 2: If $0 > C_j(y^k) \geq -p$ (so $C_j^+(y^k) = 0$) and $x \in \pi_k$ we have that $C_j(y^k) + C_j'(y^k)(x - y^k) \leq 0$. But, by (22) we have that

$$C_j(x) \leq C_j(y^k) + C_j'(y^k)(x - y^k) + \frac{L_1}{2}\|x - y^k\|^2.$$

So,

$$C_j(x) \leq \frac{L_1}{2}\|x - y^k\|^2 = C_j^+(y^k) + \frac{L_1}{2}\|x - y^k\|^2.$$

This implies that (25) also holds in this case.
Case 3: Now consider the case $C_j(y^k) < -p$ (so $C_j^+(y^k) = 0$). Let us define $\rho_1 = \max\{C_j^+(x), \ x \in \Omega\}$. Clearly, we have that

$$C_j^+(x) \leq C_j^+(y^k) + \frac{\rho_1}{\rho^2}\|x - y^k\|^2 \tag{26}$$

for all $x \in \Omega$.

The desired results follows from the monotonicity of the norm $|\cdot|$ using (24), (25) and (26). $\square$

In the next theorem we compute the decrease of the objective function that can be expected when we move from y^k to $z^{k,i}$.

Theorem 3.2. *There exist $c_2 > 0$, $c_3 > 0$ (independent of k) such that, whenever $y^k \in \Omega$ is defined and $z^{k,i}$ is computed at Step 2 of Algorithm 2.1, we have that*

$$f(z^{k,i}) \leq f(y^k) - \min\{\tau_2,\ c_2\|d^{k,tan}\|^2,\ \tau_1\delta_{k,i},\ c_3\|d^{k,tan}\|\delta_{k,i}\}.$$

Proof. By (21) we have that

$$f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L_2}{2}\|y - x\|^2$$

for all $x, y \in \Omega$. So, since $y^k + d^{k,tan} \in \Omega$ we have, for all $t \in [0,1]$, that

$$f(y^k + td^{k,tan}) \leq f(y^k) + t\langle \nabla f(y^k), d^{k,tan}\rangle + \frac{t^2 L_2}{2}\|d^{k,tan}\|^2$$

$$= f(y^k) + 0.1t\langle \nabla f(y^k), d^{k,tan}\rangle + 0.9t\langle \nabla f(y^k), d^{k,tan}\rangle + \frac{t^2 L_2}{2}\|d^{k,tan}\|^2.$$

So, (17) implies that

$$f(y^k + td^{k,tan}) \leq f(y^k) + 0.1t\langle \nabla f(y^k), d^{k,tan}\rangle - \frac{0.9ct\|d^{k,tan}\|^2}{2\eta} + \frac{t^2 L_2}{2}\|d^{k,tan}\|^2$$

$$= f(y^k) + 0.1t\langle \nabla f(y^k), d^{k,tan}\rangle + \frac{t\|d^{k,tan}\|^2}{2}(tL_2 - \frac{0.9c}{\eta}).$$

Therefore, if $t \leq \frac{0.9c}{\eta L_2}$, we have that

$$f(y^k + td^{k,tan}) \leq f(y^k) + 0.1t\langle \nabla f(y^k), d^{k,tan}\rangle.$$

This implies that $t_{(k,i,dec)} \geq \min\{t_{(k,i,break)}, \frac{0.09c}{\eta L_2}\}$.

Now, $t_{(k,i,break)} = \min\{1, \frac{\delta_{k,i}}{\|d^{k,tan}\|}\}$. So,

$$t_{(k,i,dec)} \geq \min\{1, \frac{0.09c}{\eta L_2}, \frac{\delta_{k,i}}{\|d^{k,tan}\|}\}.$$

Thus, by the definition of $t_{(k,i,dec)}$, it follows that

$$f(y^k + t_{(k,i,dec)}d^{k,tan}) \leq f(y^k) + \min\{0.1, \frac{0.009c}{\eta L_2}, \frac{0.1\delta_{k,i}}{\|d^{k,tan}\|}\}\langle \nabla f(y^k), d^{k,tan}\rangle.$$

So, by (17), we obtain

$$f(y^k + t_{(k,i,dec)}d^{k,tan}) \leq f(y^k) - \min\{\frac{0.1c\|d^{k,tan}\|^2}{2\eta}, \frac{0.009c^2\|d^{k,tan}\|^2}{2\eta^2 L_2}, \frac{0.1c\delta_{k,i}\|d^{k,tan}\|}{2\eta}\}.$$

Therefore,

$$f(y^k + t_{(k,i,dec)}d^{k,tan}) \leq f(y^k) - \min\{c_2\|d^{k,tan}\|^2, c_3\delta_{k,i}\|d^{k,tan}\|\},$$

where $c_2 = \min\{\frac{0.1c}{2\eta}, \frac{0.009c^2}{2\eta^2 L_2}\}$ and $c_3 = \frac{0.1c}{2\eta}$.

The desired result follows from the last inequality and (15). □

In the last theorem of this section we prove that Algorithm 2.1 is well defined. This amounts to show that, for small enough $\delta_{k,i}$, the inequality (11) is satisfied and, so, the trial point $z^{k,i}$ is accepted as new iterate.

Theorem 3.3. *Algorithm 2.1 is well defined.*

Proof. Observe that

$$\begin{aligned}
&\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \\
&= 0.9\theta_{k,i}[f(x^k) - f(z^{k,i})] + (1-\theta_{k,i})[|C^+(x^k)| - |C^+(z^{k,i})|] - 0.1(1-\theta_{k,i})[|C^+(x^k)| - |C^+(y^k)|] \\
&= 0.9\theta_{k,i}[f(x^k) - f(z^{k,i})] + 0.9(1-\theta_{k,i})[|C^+(x^k)| - |C^+(y^k)|] \\
&\quad + (1-\theta_{k,i})[|C^+(x^k)| - |C^+(z^{k,i})|] - (1-\theta_{k,i})[|C^+(x^k)| - |C^+(y^k)|] \\
&= 0.9\mathbf{Pred}_{k,i} + (1-\theta_{k,i})[|C^+(y^k)| - |C^+(z^{k,i})|].
\end{aligned}$$

So, by (13) and (2),

$$\begin{aligned}
\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} &\geq 0.45[|C^+(x^k)| - |C^+(y^k)|] - |(|C^+(y^k)| - |C^+(z^{k,i})|)| \\
&\geq 0.45(1-r)|C^+(x^k)| - |(|C^+(y^k)| - |C^+(z^{k,i})|)|.
\end{aligned}$$

Therefore, if $|C^+(x^k)| > 0$, since $\|y^k - z^{k,i}\| \leq \delta_{k,i}$ and $|C^+(x)|$ is continuous, it follows that $\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \geq 0$ if $\delta_{k,i}$ is small enough. So, we proved that the algorithm is well defined if the current point x^k is infeasible.

If x^k is feasible, (3) implies that $y^k = x^k$ and $|C^+(y^k)| = 0$. If $d^{k,tan} \neq 0$ we have that $f(z^{k,i}) < f(y^k)$ for all $i = 0, 1, 2, \ldots$. So, the condition (13) is always satisfied and, consequently, $\theta_{k,i} = \theta_{k,-1}$ for all $i = 0, 1, 2, \ldots$ Therefore, in this case, we have

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} = 0.9\theta_{k,-1}[f(y^k) - f(z^{k,i})] - (1-\theta_{k,-1})|C^+(z^{k,i})|.$$

So, by Theorems 3.1 and 3.2, we obtain that

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \geq 0.9\theta_{k,-1} \min\{\tau_2, c_2\|d^{k,tan}\|^2, \tau_1\delta_{k,i}, c_3\|d^{k,tan}\|\delta_{k,i}\} - c_1\|z^{k,i} - y^k\|^2.$$

Therefore, (11) holds if

$$\delta_{k,i} \leq \min\{(\frac{0.9\theta_{k,-1}\tau_2}{c_1})^{1/2}, (\frac{0.9\theta_{k,-1}c_2}{c_1})^{1/2}\|d^{k,tan}\|, \frac{0.9\theta_{k,-1}\tau_1}{c_1}, (\frac{0.9\theta_{k,-1}c_3}{c_1})^{1/2}\|d^{k,tan}\|\}.$$

So, we proved that x^{k+1} is well defined when x^k is feasible and $d^{k,tan} \neq 0$. □

The next theorem is an important tool for proving convergence of the model algorithm. We are going to prove that the actual reduction $\mathbf{Ared}_{k,iacc(k)}$ effectively achieved at each iteration necessarily tends to 0. An immediate consequence will be the feasibility of the limit points generated by the algorithm.

Theorem 3.4. *Suppose that Algorithm 2.1 generates an infinite sequence. Then*

$$\lim_{k\to\infty} \psi(x^k, \theta_k) - \psi(x^{k+1}, \theta_k) = 0$$

Proof. Suppose, by contradiction, that there exists an infinite set of indices $K_1 \subset \{0, 1, 2, \ldots\}$ and a positive number $\gamma > 0$ such that

$$\psi(x^{k+1}, \theta_k) \leq \psi(x^k, \theta_k) - \gamma$$

for all $k \in K_1$. Let us write $\psi_k = \psi(x^k, \theta_k)$ for all $k \in \{0, 1, 2, \ldots\}$.

Then, for all $k \in \{0, 1, 2, \ldots\}$ we have that

$$\begin{aligned}
\psi_{k+1} &= \theta_{k+1}f(x^{k+1}) + (1-\theta_{k+1})|C^+(x^{k+1})| \\
&= \theta_{k+1}f(x^{k+1}) + (1-\theta_{k+1})|C^+(x^{k+1})| - [\theta_k f(x^{k+1}) + (1-\theta_k)|C^+(x^{k+1})|] \\
&\quad + [\theta_k f(x^{k+1}) + (1-\theta_k)|C^+(x^{k+1})|] \\
&= (\theta_{k+1} - \theta_k)f(x^{k+1}) + (\theta_k - \theta_{k+1})|C^+(x^{k+1})| + [\theta_k f(x^{k+1}) + (1-\theta_k)|C^+(x^{k+1})|] \\
&= (\theta_k - \theta_{k+1})(|C^+(x^{k+1})| - f(x^{k+1})) + [\theta_k f(x^k) + (1-\theta_k)|C^+(x^k)|] - \gamma_k \\
&= (\theta_k - \theta_{k+1})(|C^+(x^{k+1})| - f(x^{k+1})) + \psi_k - \gamma_k, \qquad (27)
\end{aligned}$$

where $\gamma_k \geq 0$ for all $k \in \{0, 1, 2, \ldots\}$ and $\gamma_k \geq \gamma > 0$ for all $k \in K_1$. Now, by the definition of $\theta_{k,-1}$ at Algorithm 2.1, we have that

$$\theta_k - \theta_{k+1} + \omega_k \geq 0. \qquad (28)$$

for all $k \in \{0, 1, 2, \ldots\}$. By the compactness of Ω, there exists an upper bound $c > 0$ such that

$$|C^+(x^k)| - f(x^k)| \leq c$$

for all $k \in \{0, 1, 2, \ldots\}$. Therefore, by (27) and (28), we have that

$$\psi_{j+1} = (\theta_j - \theta_{j+1} + \omega_j)(|C^+(x^{j+1})| - f(x^{j+1})) + \psi_j - \gamma_j - \omega_j(|C^+(x^{j+1})| - f(x^{j+1}))$$

$$\leq (\theta_j - \theta_{j+1} + \omega_j)c + \psi_j - \gamma_j + \omega_j c = (\theta_j - \theta_{j+1})c + \psi_j - \gamma_j + 2\omega_j c$$

for $j = 0, 1, \ldots, k-1$. Adding these k inequalities, we obtain

$$\psi_k \leq \psi_0 + (\theta_0 - \theta_k)c + \sum_{j=0}^{k-1} 2c\omega_j - \sum_{j=0}^{k-1} \gamma_j \leq \psi_0 + 2c + \sum_{j=0}^{k-1} 2c\omega_j - \sum_{j=0}^{k-1} \gamma_j \qquad (29)$$

for all $k \geq 1$. Since the series $\sum_{j=0}^{\infty} \omega_j$ is convergent, and γ_k is bounded away from 0 for $k \in K_1$, (29) implies that ψ_k is unbounded below. This contradicts the compactness of Ω. □

An easy consequence of Theorem 3.4 is that, when Algorithm 2.1 generates an infinite sequence (that is, it is not stopped at Step 1), we have that $\lim_{k\to\infty} |C^+(x^k)| = 0$. This means that points arbitrarily close to feasibility are eventually generated.

Theorem 3.5. *If Algorithm 2.1 does not stop at Step 1 for all $k = 0, 1, 2, \ldots$, then*

$$\lim_{k\to\infty} |C^+(x^k)| = 0.$$

(In particular, every limit point of $\{x^k\}$ is feasible.)

Proof. By (2), (11) and (13) we have that

$$|C^+(x^k)| \leq \frac{|C^+(x^k)| - |C^+(y^k)|}{1-r} \leq \frac{2}{1-r}\mathbf{Pred}_{k,iacc(k)} \leq \frac{20}{1-r}\mathbf{Ared}_{k,iacc(k)}$$

$$= \frac{20}{1-r}[\psi(x^k, \theta_k) - \psi(x^{k+1}, \theta_k)].$$

So, the desired result follows from Theorem 3.4. □

4 Convergence to Optimality

In the former section we proved that, if the algorithm does not break down at Step 1, it achieves approximate feasibility up to any desired precision. In this section we are

going to prove that, in that case, the optimality indicator $\|d^{k,tan}\|$ cannot be bounded away from zero. In practice, this implies that given arbitrarily small convergence tolerances $\varepsilon_{feas}, \varepsilon_{opt} > 0$, Algorithm 2.1 eventually finds an iterate x^k such that $\|C^+(x^k)\| \leq \varepsilon_{feas}$ and $\|d^{k,tan}\| \leq \varepsilon_{opt}$. For proving this result, we will proceed by contradiction, assuming that $\|d^{k,tan}\|$ is bounded away from zero for k large enough. From this hypothesis (stated as Hypothesis C below) we will deduce intermediate results that, finally, will lead us to a contradiction.

Hypothesis C. *Algorithm 2.1 generates an infinite sequence* $\{x^k\}$ *and there exists* $\varepsilon > 0$, $k_0 \in \{0, 1, 2, \ldots\}$ *such that*

$$\|d^{k,tan}\| \geq \varepsilon \quad \textit{for all} \quad k \geq k_0.$$

Lemma 4.1. *Suppose that Hypothesis C holds. Then, there exist* $c_4, c_5 > 0$ *(independent of* k*) such that*

$$f(y^k) - f(z^{k,i}) \geq \min\{c_4, c_5\delta_{k,i}\}$$

for all $k \geq k_0, i = 0, 1, \ldots, iacc(k)$

Proof. The result follows trivially from Theorem 3.2 and Hypothesis C. □

Lemma 4.2. *Suppose that Hypothesis C holds. Then, there exist* $\alpha, \varepsilon_1 > 0$*, independent of* k *and* i*, such that* $|C^+(x^k)| \leq \min\{\varepsilon_1, \alpha\delta_{k,i}\}$ *implies that* $\theta_{k,i} = \theta_{k,i-1}$.

Proof. Observe that

$$Pred_{k,i}(1) = f(x^k) - f(z^{k,i})$$
$$\geq f(y^k) - f(z^{k,i}) - |f(x^k) - f(y^k)| \geq f(y^k) - f(z^{k,i}) - c\|y^k - x^k\|$$

where c is a constant that only depends on the norms and on a bound of $\|\nabla f(x)\|$ on Ω. Therefore, by (3), and Lemma 4.1,

$$Pred_{k,i}(1) - \frac{1}{2}|C^+(x^k)| \geq \min\{c_4, c_5\delta_{k,i}\} - (c\beta + 0.5)|C^+(x^k)|.$$

Define

$$\varepsilon_1 = \frac{c_4}{c\beta + 0.5}, \quad \alpha = \frac{c_5}{c\beta + 0.5}.$$

If $|C^+(x^k)| \leq \min\{\varepsilon_1, \alpha\delta\}$ we have that

$$Pred_{k,i}(1) - \frac{1}{2}|C^+(x^k)| \geq 0.$$

This implies that any value of $\theta_{k,i}$ in the interval $[0,1]$ satisfies (13). In particular $\theta_{k,i-1}$ satisfies (13), as we wanted to prove. $\square$

In the next Lemma, we prove that, under Hypothesis C, the penalty parameters $\{\theta_k\}$ are bounded away from zero. It must be warned that this is a property of sequences that satisfy Hypothesis C (which, in turn, will be proved to be non-existent!) and not of *all* the sequences effectively generated by the model algorithm.

Lemma 4.3. *Suppose that Hypothesis C holds. Then, there exists $\bar{\theta} > 0$ such that $\theta_k \geq \bar{\theta}$ for all $k \in \{0, 1, 2, \ldots\}$.*

Proof. We are going to show first that, if $|C^+(x^k)|$ is sufficiently small, a step $\delta_{k,i}$ that satisfies

$$|C^+(x^k)| \geq \frac{\alpha}{10}\delta_{k,i} \tag{30}$$

is necessarily accepted, where α is defined in Lemma 4.2.

In fact, assume that (30) holds. Then, by (13) and (2),

$$\mathbf{Pred}_{k,i} \geq \frac{1}{2}[|C^+(x^k)| - |C^+(y^k)|] \geq \frac{1-r}{2}|C^+(x^k)| \geq \frac{(1-r)\alpha}{20}\delta_{k,i}.$$

So, (30) implies that

$$\delta_{k,i} \leq \frac{20}{(1-r)\alpha}\mathbf{Pred}_{k,i}. \tag{31}$$

Now, by Theorem 3.1,

$$\mathbf{Ared}_{k,i} = \mathbf{Pred}_{k,i} + (1-\theta_{k,i})[|C^+(y^k)| - |C^+(z^{k,i})|] \geq \mathbf{Pred}_{k,i} - c_1\delta_{k,i}^2.$$

Therefore, by (31), (30) implies that

$$\mathbf{Ared}_{k,i} \geq \mathbf{Pred}_{k,i} - \frac{20c_1}{(1-r)\alpha}\delta_{k,i}\mathbf{Pred}_{k,i} \geq (1 - \frac{200c_1}{(1-r)\alpha^2}|C^+(x^k)|)\mathbf{Pred}_{k,i}.$$

So, if (30) holds and $|C^+(x^k)| \leq \frac{0.9(1-r)\alpha^2}{200c_1}$, the trial point $z^{k,i}$ is necessarily accepted.

Let us define

$$\varepsilon_2 = \min\{\varepsilon_1, \frac{0.9(1-r)\alpha^2}{200c_1}, \alpha\delta_{min}\},$$

where ε_1 is defined in Lemma 4.2. Let $k_1 \geq k_0$ be such that $|C^+(x^k)| \leq \varepsilon_2$ for all $k \geq k_1$. Since $\delta_{min} \geq \frac{|C^+(x^k)|}{\alpha}$, this implies that $\delta_{k,0} \geq \frac{|C^+(x^k)|}{\alpha}$ for all $k \geq k_1$. Therefore, a possible trust region radius such that $\delta_{k,i} < \frac{|C^+(x^k)|}{\alpha}$ cannot correspond to $i = 0$, so it is preceded by $\delta_{k,i-1}$ which necessarily verifies

$$\delta_{k,i-1} \leq 10\frac{|C^+(x^k)|}{\alpha}.$$

By the reasoning displayed above, the trial point $z^{k,i-1}$ is accepted for all $k \geq k_1$. Therefore, $\delta_{k,i} \geq \frac{|C^+(x^k)|}{\alpha}$ for all $k \geq k_1$, $i = 0, 1, \ldots, iacc(k)$. So, by Lemma 4.2, the penalty parameter $\theta_{k,i}$ is never decreased for all $k \geq k_1$. This implies the desired result. □

Finally, we prove, in Theorem 4.4, that Hypothesis C cannot be true.

Theorem 4.4. *Let $\{x^k\}$ be an infinite sequence generated by Algorithm 2.1. Then, there exists K_2, an infinite subset of $\{0, 1, 2, \ldots\}$, such that*

$$\lim_{k \in K_2} \|d^{k,tan}\| = 0. \tag{32}$$

Proof. Suppose that the thesis of the theorem is not true. Then, there exists $k_0 \in \{0, 1, 2, \ldots\}$, $\varepsilon > 0$ such that Hypothesis C holds.

As in the beginning of the proof of Theorem 3.3, observe that, by Theorem 3.1,

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i}$$

$$= 0.9\{\theta_{k,i}[f(x^k) - f(z^{k,i})] + (1-\theta_{k,i})[|C^+(x^k)| - |C^+(y^k)|]\} + (1-\theta_{k,i})[|C^+(y^k)| - |C^+(z^{k,i})|]$$

$$\geq 0.9\theta_{k,i}[f(y^k) - f(z^{k,i})] + 0.9\theta_{k,i}[f(x^k) - f(y^k)] - (1-r)|C^+(x^k)| - c_1\delta_{k,i}^2.$$

So, by Lemma 4.1, Lemma 4.3, and (3),

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \geq 0.9\bar{\theta} \min\{c_4, c_5\delta_{k,i}\} - c|C^+(x^k)| - c_1\delta_{k,i}^2$$

for all $k \geq k_0$, $i = 0, 1, iacc(k)$, where c is a norm-dependent constant that also depends on a bound of $\|\nabla f(x)\|$ on Ω.

Let us define

$$\bar{\delta} = \min\{(0.45\bar{\theta}c_4/c_1)^{1/2}, 0.45\bar{\theta}c_5/c_1\}.$$

If $\delta_{k,i} \leq \bar{\delta}$ we have that

$$c_1\delta_{k,i}^2 \leq 0.45\bar{\theta} \min\{c_4, c_5\delta_{k,i}\},$$

so, when $\delta_{k,i} \leq \bar{\delta}$, we have that

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \geq 0.45\bar{\theta} \min\{c_4, c_5\delta_{k,i}\} - c|C^+(x^k)| \tag{33}$$

for all $k \geq k_0$, $i = 0, 1, iacc(k)$. Let $k_2 \geq k_0$ be such that

$$c|C^+(x^k)| \leq 0.45\bar{\theta} \min\{c_4, c_5\frac{\bar{\delta}}{10}\} \tag{34}$$

for all $k \geq k_2$. By (33) and (34) we have that, for all $k \geq k_2$, if $i \in \{0, 1, 2, \ldots\}$ corresponds to the first trust-region radius $\delta_{k,i}$ less than or equal to $\bar{\delta}$ (so, $\bar{\delta} \geq \delta_{k,i} \geq \frac{\bar{\delta}}{10}$),

$$\mathbf{Ared}_{k,i} - 0.1\mathbf{Pred}_{k,i} \geq 0.$$

This means that $\delta_{k,i} \geq \frac{\bar{\delta}}{10}$ must be accepted. Therefore,

$$\delta_{k,iacc(k)} \geq \frac{\bar{\delta}}{10}$$

for all $k \geq k_2$. So, if $k \geq k_2$ we have, by Lemma 4.1, Lemma 4.3 and (3), that

$$\mathbf{Pred}_{k,iacc(k)} = \theta_{k,iacc(k)}[f(x^k) - f(z^{k,i})] + (1 - \theta_{k,iacc(k)})[|C^+(x^k)| - |C^+(y^k)|]$$

$$= \theta_{k,iacc(k)}[f(y^k) - f(z^{k,i})] + \theta_{k,iacc(k)}[f(x^k) - f(y^k)] + (1 - \theta_{k,iacc(k)})[|C^+(x^k)| - |C^+(y^k)|]$$

$$\geq \bar{\theta}[f(y^k) - f(z^{k,i})] - |f(x^k) - f(y^k)| - |C^+(x^k)| \geq \bar{\theta} \min\{c_4, \frac{c_5\bar{\delta}}{10}\} - c'|C^+(x^k)| \quad (35)$$

for all $k \geq k_2$, where c' is a constant that depends on the norm and the bound of $\|\nabla f(x)\|$ on Ω. Now, let $k_3 \geq k_2$ be such that

$$c'|C^+(x^k)| \leq \frac{\bar{\theta}}{2} \min\{c_4, \frac{c_5\bar{\delta}}{10}\}$$

for all $k \geq k_3$. By (35), $\mathbf{Pred}_{k,iacc(k)}$ is bounded away from zero for all $k \geq k_3$. This implies, by (11), that $\mathbf{Ared}_{k,iacc(k)}$ is bounded away from zero for all $k \geq k_3$. Clearly, this contradicts Theorem 3.4. This means that Hypothesis C cannot be true. Therefore, the desired result is proved. □

5 Application: Hard-Spheres Problems

The Hard-Spheres problem belongs to the family of sphere packing problems, a class of challenging problems dating from the beginning of the seventeenth century which is related to practical problems in Chemistry, Biology and Physics (see [7, 32]). It consists on maximizing the minimum pairwise distance between q points on a sphere in $\mathbb{R}^{dim}$. This problem may be reduced to a nonconvex nonlinear optimization problem with a potentially large number of (nonoptimal) points satisfying optimality conditions. We have, thus, a class of problems indexed by the parameters dim and q, that provides a suitable set of test problems for evaluating nonlinear programming codes.

The straightforward formulation of the Hard-Spheres problem is:

$$\begin{array}{lll} \text{Maximize} & \min_{i \neq j} & \|w^i - w^j\| \\ & \text{subject to} & \|w^k\| = 1, k = 1, \ldots, q, \end{array} \quad (36)$$

where the vectors w^k belong to $\mathbb{R}^{dim}$ and $\|\cdot\|$ is the Euclidean norm.

This is equivalent to

$$\begin{array}{lll} \text{Minimize} & \max_{i \neq j} & \langle w^i, w^j \rangle \\ & \text{subject to} & \|w^k\|^2 - 1 = 0, k = 1, \ldots, q. \end{array} \tag{37}$$

Applying the classical trick for transforming minimax problems into constrained minimization problems, we reduce (37) to the nonlinear program

$$\begin{array}{lrcl} \text{Minimize} & z & & \\ \text{subject to} & \langle w^i, w^j \rangle - z & \leq & 0, \text{ for all } i \neq j, \\ & \|w^k\|^2 - 1 & = & 0, k = 1, \ldots, q. \end{array} \tag{38}$$

The structure of the Hard-Spheres problems suggests a natural Restoration Step, which does not rely on sophisticated algorithms for solving (2)–(3). Assume that $x^k = (w^1, \ldots, w^q, z)$ is the current point at the $k-$th iteration. Replacing

$$w^j \leftarrow \frac{w^j}{\|w^j\|}, \; j = 1, \ldots, q$$

and

$$z \leftarrow \max\{\langle w^i, w^j \rangle, \; i \neq j\}$$

we obtain a point $x = (w^1, \ldots, w^q, z)$ that satisfies exactly the constraints. If (3) is violated by x (so $\|x - x^k\| > \beta \|C^+(x^k)\|$), we replace x by $x^k + \frac{\beta \|C^+(x^k)\|}{\|x - x^k\|}(x - x^k)$. If this point violates (2) we declare "failure in improving feasibility" at the Restoration Phase. In our experiments we used $\beta = 4, r = 0.99$. Obviously, this restoration procedure relies on the specific structure of the constraints (38) and we take advantage of the freedom allowed by the Inexact-Restoration algorithm on the choice of the restored point.

For the Minimization Step we use the well-known linearly constrained minimization solver implemented in the MINOS system, Version 5.4 (see [24]). The problem to be solved by MINOS is to minimize the variable z on the intersection of polytope defined by the linearization of the inequality constraints of (38) and the trust region box around of y^k. We used the defaults of MINOS for optimality and feasibility and the "Warm Start" option at each Minimization Step. Since the subproblem solved by MINOS is a Linear Programming problem, we can assume that MINOS finds a global solution, so that the inequality $f(z^{k,i}) \leq f(y^k + t_{(k,i,dec)} d^{k,tan})$ (see (15)) necessarily holds. Therefore, in this case it is not necessary to specify the parameters τ_1, τ_2 and η. In practice, each execution of MINOS was stopped with the default convergence criterion relatively to the norm of the reduced gradient and signs of the multipliers.

The nonnegative sequence for the penalty parameter of the merit function at Step 1 of Algorithm 2.1, was $\omega_k = \frac{n}{(1+k)^2}$, where $n = q \times dim + 1$ and the initial penalty parameter was $\theta_{-1} = 0.5$. After some preliminary tests we used $p = 10$.

We used the following criterion to update the trust region radius $\delta_{k,i}$. If the sufficient reduction condition (11) does not hold at Step 4 in Algorithm 2.1, we set $\delta_{k,i+1} = \delta_{k,i}/8$. On the other hand, to restart at the beginning of an iteration, we set $\delta_{k,0} = \max\{\delta_{min}, 4\delta_{k-1,acc}\}$, with $\delta_{min} = \delta_{0,0} = 0.5$.

The theoretical properties of the Inexact–Restoration algorithm guarantee that, if break-down does not occur at the Restoration Step, then given any $\varepsilon > 0$ there exists k such that $\|C^+(x^k)\| \leq \varepsilon$ and $\|d^{k,tan}\| \leq \varepsilon$. In our practical implementation we declared "convergence" when $\|C^+(x^k)\|_\infty \leq 10^{-8}$. Since x^k comes from the Minimization Step performed by MINOS, when this occurs we necessarily have that $d^{k-1,tan} \approx 0$.

Let us comment now the choice of the parameters of LANCELOT. The manual [6] (p.111) "strongly recommends the use of exact second derivatives whenever they are available". In fact we ran a few tests with the default approximation SR1 but the results were worse than those obtained using exact second derivatives, and thus this was the option adopted for all further tests. We also experimented several different options for the linear equation solver: without preconditioner, with diagonal preconditioner and with a band matrix preconditioner. The best results were obtained with the first option (no preconditioner). Moreover, after some preliminary tests, we decided to use the "inexact Cauchy point" option. The maximum number of iterations allowed was 1000. Finally, the gradient and constraints tolerances were the same chosen for the Inexact–Restoration algorithm, namely 10^{-8}. Both codes are in FORTRAN and the compiler option adopted for both was "-O".

6 Numerical experiments

Tests were run on a Sun SparcStation 20, with the following main characteristics: 128Mbytes of RAM, 70MHz, 204.7 mips, 44.4 Mflops. We ran both codes using 50 initial random points for each problem. The results are summarized in Table 1. This table lists the eigtheen problems with the number of variables and constraints and the statistic information related to the minimum distance between two points (minimum, maximum, average) and CPU time (minimum,maximum, average) using the Inexact–Restoration algorithm (first row of each set) and the ones using LANCELOT (second row).

The information contained in Table 1 is depicted graphically below. The intervals (min, max) of distances/log(CPU times) are represented by vertical segments, the averages are indicated with a diamond symbol for the Inexact–Restoration algorithm and a bullet for LANCELOT. Graphs on the left refer to distances whereas graphs on the right refer to log(CPU times).

Problem size			minimum distance between 2 points			CPU time (seconds)		
$\begin{bmatrix} n \\ p \end{bmatrix}$	var.	constr.	min.	max.	average	min.	max.	average
$\begin{bmatrix} 3 \\ 10 \end{bmatrix}$	31	55	1.0514622	1.0914262	1.0822176	0.46	0.79	0.61
			1.0514656	1.0914302	1.0874007	0.83	2.51	1.50
$\begin{bmatrix} 3 \\ 11 \end{bmatrix}$	34	66	1.0514622	1.0514622	1.0514622	0.64	0.91	0.76
			1.0514656	1.0514656	1.0514656	1.10	3.92	1.81
$\begin{bmatrix} 3 \\ 12 \end{bmatrix}$	37	78	0.9447876	1.0514622	1.0493287	0.81	1.37	0.99
			0.9447856	1.0514656	1.0430604	1.53	3.29	2.24
$\begin{bmatrix} 3 \\ 13 \end{bmatrix}$	40	91	0.9427907	0.9564136	0.9499126	0.88	1.25	1.00
			0.9443516	0.9564099	0.9512710	2.26	8.06	4.12
$\begin{bmatrix} 3 \\ 14 \end{bmatrix}$	43	105	0.9161167	0.9338626	0.9293394	1.04	1.47	1.24
			0.9025741	0.9338629	0.9305515	2.49	9.05	5.12
$\begin{bmatrix} 3 \\ 15 \end{bmatrix}$	46	120	0.8745439	0.9026562	0.9008776	1.16	1.92	1.47
			0.8734529	0.9026516	0.9009286	3.25	12.73	7.37
$\begin{bmatrix} 4 \\ 22 \end{bmatrix}$	89	253	0.9824163	1.0019895	0.9951659	5.29	17.43	8.12
			0.9840223	1.0019880	0.9967615	30.49	209.27	69.85
$\begin{bmatrix} 4 \\ 23 \end{bmatrix}$	93	276	0.9693916	1.0000000	0.9827767	6.73	16.74	10.31
			0.9740944	0.9918568	0.9847650	29.26	178.84	89.80
$\begin{bmatrix} 4 \\ 24 \end{bmatrix}$	97	300	0.9573460	1.0000000	0.9734775	7.13	19.26	12.34
			0.9580083	0.9828733	0.9751985	43.16	239.77	112.78
$\begin{bmatrix} 4 \\ 25 \end{bmatrix}$	101	325	0.9477678	0.9616207	0.9569177	8.25	17.97	12.58
			0.9465833	0.9619563	0.9574963	49.00	268.49	131.18
$\begin{bmatrix} 4 \\ 26 \end{bmatrix}$	105	351	0.9327032	0.9583427	0.9474299	9.99	29.60	15.57
			0.9367603	0.9583423	0.9491615	39.90	565.90	164.47
$\begin{bmatrix} 4 \\ 27 \end{bmatrix}$	109	378	0.9276386	0.9394150	0.9344075	11.08	33.88	17.06
			0.9273834	0.9389142	0.9345753	79.26	332.12	173.13
$\begin{bmatrix} 5 \\ 37 \end{bmatrix}$	186	703	0.9905835	1.0045763	0.9993300	68.66	369.42	149.48
			0.9911508	1.0025367	0.9979124	444.81	2501.76	1154.08
$\begin{bmatrix} 5 \\ 38 \end{bmatrix}$	191	741	0.9842019	1.0019176	0.9917008	93.85	527.66	168.08
			0.9864684	1.0019880	0.9930711	546.55	3105.86	1538.54
$\begin{bmatrix} 5 \\ 39 \end{bmatrix}$	196	780	0.9772092	0.9929902	0.9871450	108.71	461.15	204.96
			0.9808159	0.9920786	0.9881178	502.38	3161.88	1782.30
$\begin{bmatrix} 5 \\ 40 \end{bmatrix}$	201	820	0.9734556	0.9886857	0.9818932	100.08	600.04	220.59
			0.9701958	0.9920282	0.9810864	863.85	3820.43	1907.57
$\begin{bmatrix} 5 \\ 41 \end{bmatrix}$	206	861	0.9686624	0.9818115	0.9746239	117.34	435.91	195.79
			0.9644272	0.9819470	0.9757435	1148.77	4669.87	2521.84
$\begin{bmatrix} 5 \\ 42 \end{bmatrix}$	211	903	0.9612090	0.9793985	0.9693361	105.37	641.68	213.74
			0.9599791	0.9798367	0.9702516	807.57	4664.63	2473.78

Table 1: Minimum distances and CPU times

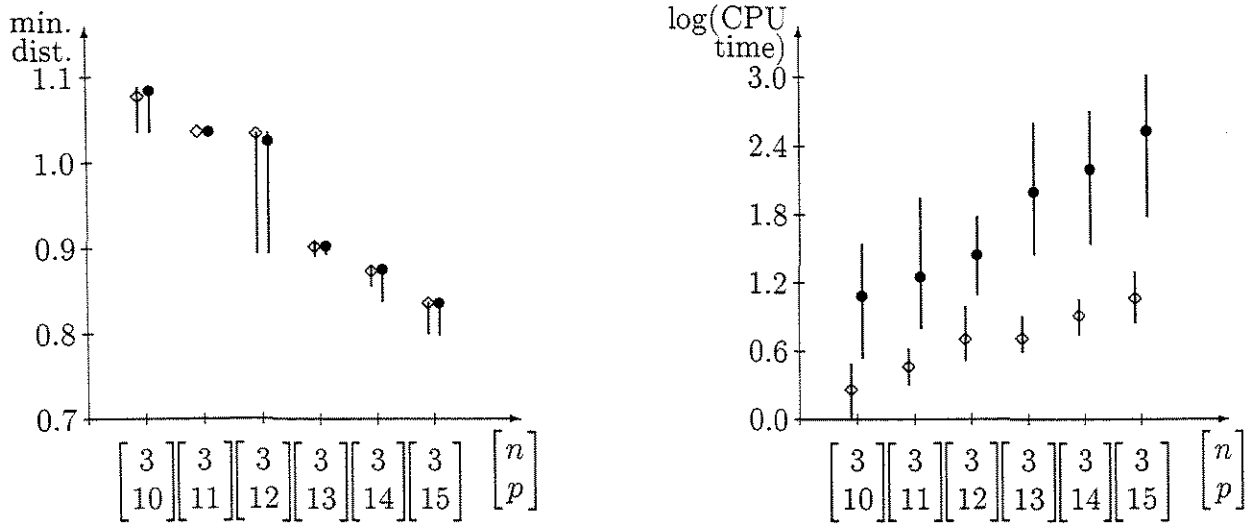


Figure 1: Inexact-Restoration ($\diamond$) and LANCELOT ($\bullet$) results for $n = 3$.

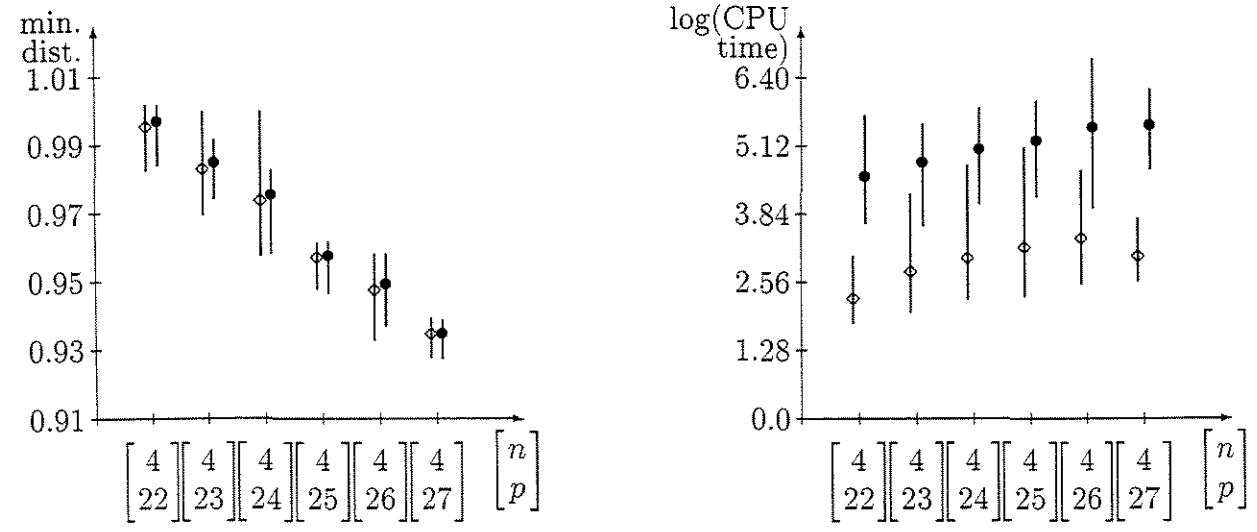


Figure 2: Inexact-Restoration ($\diamond$) and LANCELOT ($\bullet$) results for $n = 4$.

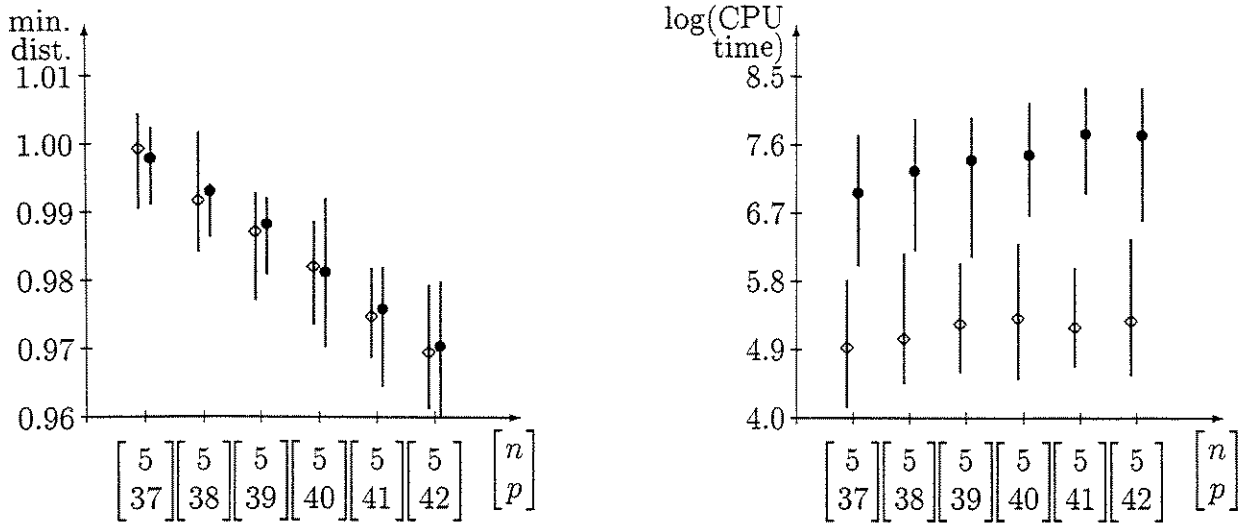


Figure 3: Inexact-Restoration ($\diamond$) and LANCELOT ($\bullet$) results for $n = 5$.

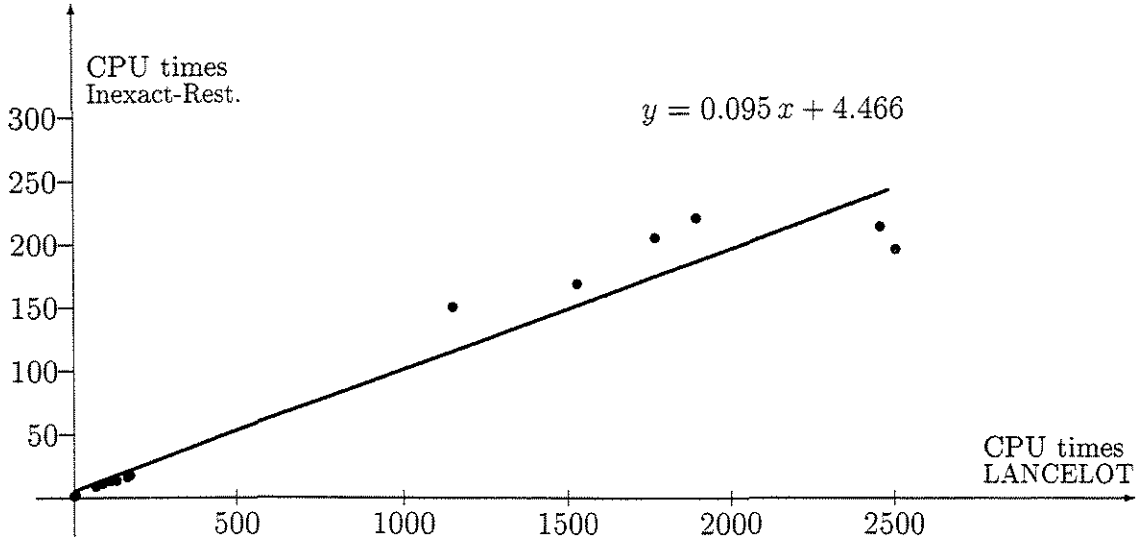


Figure 4: CPU times of LANCELOT versus those of Inexact-Restoration Alg.

The graphs in Figures 1–3 evidence the qualitative relative behavior of both codes. Notice that the diamonds and bullets are always close together in the graphs on the left, indicating that the quality of the optimal solutions obtained by both codes is similar. On the other hand, the bullets rise faster than the diamonds on the graphs on the right, which means that the CPU times for LANCELOT tend to be higher than those of the Inexact–Restoration code. The linear fit of Inexact–Restoration CPU times versus LANCELOT CPU times is $y = 0.095\,x + 4.466$ (see Figure 4). Observe that, in fact, the linear coefficient is less than 0.1 .

In Figure 5 we compare the CPU times of both algorithms for the eigtheen problems considered. This figure shows clearly the good performance of our Algorithm, specially when the size of the problem increases.

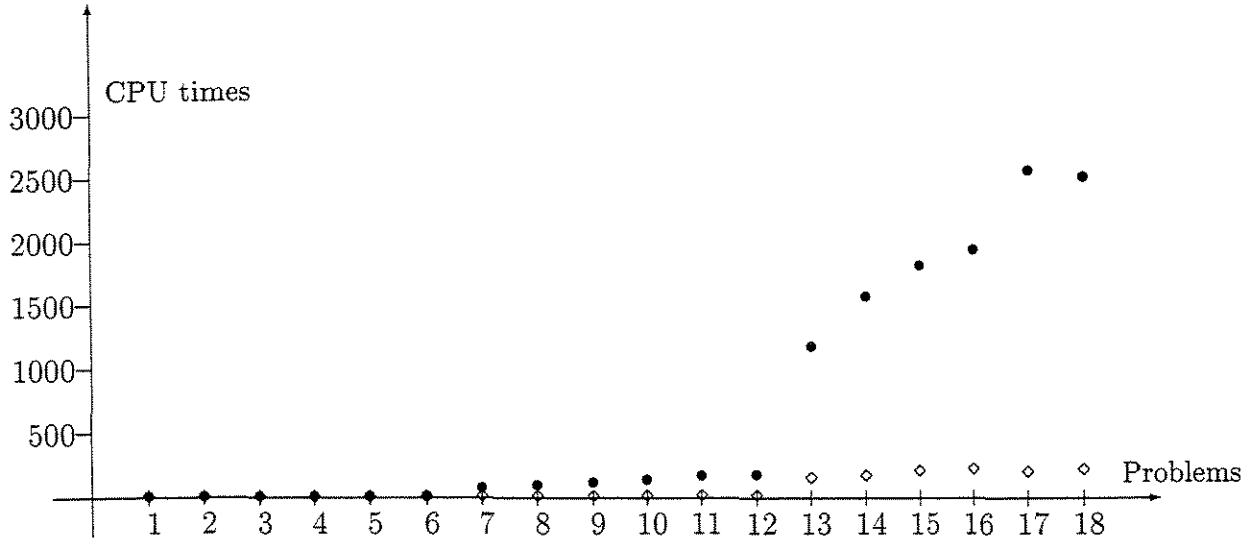


Figure 5: CPU times: Inexact-Restoration ($\diamond$) and LANCELOT ($\bullet$).

7 Final remarks

Since the method presented in this paper is a model algorithm, many possible implementations can be given. The efficiency of different implementations should be linked to the quality of the algorithms chosen for performing different steps. For the Restoration Step we need an algorithm that solves (2)–(3). Since, in most cases, $\|\cdot\|$ will be the sup-norm and Ω will be a box, we can choose any of the many available methods for large-scale box-constrained minimization for solving this problem.

In the Minimization Step we need an approximate solution of (8). Generally, this is a linearly constrained minimization problem. For its resolution active set methods are generally recommended (see, for example, [23]). However, last decade large-scale

optimization research suggests that efficient implementations can also result from the application of interior point methods to (8). See [33].

In this paper we did not use regularity assumptions to prove global convergence of infinite sequences generated by the algorithm. This does not mean that regularity is not playing any role in practical circumstances. Roughly speaking, lack of regularity can cause a failure in Restoration Phase, resulting in break-down at Step 1. In fact, our theoretical results show that, if the original problem is infeasible, break-down will necessary take place for some (finite) value of the iteration k, that is, an infinite sequence will not be generated. On the other hand, we proved that when infinitely many points are generated, all the limit points are feasible. Finally, the results on Section 4 show that at least one of these limit points is stationary in the sense that $\lim_{k\in K_2} \|d^{k,tan}\| = 0$ when $\{x_k\}_{k\in K_2}$ is the corresponding convergent subsequence. The relations between this type of stationarity and necessary or sufficient conditions for local minimization remain to be investigated.

References

[1] J. Abadie and J. Carpentier, *Generalization of the Wolfe reduced-gradient method to the case of nonlinear constraints*, Optimization, Edited by R. Fletcher, Academic Press, New York, pp. 37-47, 1968.

[2] R. H. Bielschowsky, *Nonlinear programming algorithms with dynamic definition of near-feasibility: theory and implementations*, Tese de Doutorado, IMECC-UNICAMP, Campinas, 1996.

[3] R. H. Byrd, *Robust trust region methods for constrained optimization*, Contributed presentation, SIAM Conference on Optimization, Houston, Texas, 1987.

[4] R. H. Byrd, J. Ch. Gilbert and J. Nocedal, *A trust region method based on interior point techniques for nonlinear programming*, Technical Report OTC 96/02, Optimization Technology Center, Argonne National Laboratory and Northwestern University, Illinois, 1996.

[5] M. R. Celis, J. E. Dennis and R. A. Tapia, *A trust-region strategy for nonlinear equality constrained optimization*, Numerical Optimization, Edited by P. Boggs, R. Byrd and R. Schnabel, SIAM Publications, Philadelphia, Pennsylvania, pp. 71-82, 1985.

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

[7] J. H. Conway and N. J. C. Sloane, *Sphere Packings, Lattices and Groups.*, Springer-Verlag, New York, 1988.

[8] R. S. Dembo, S. C. Eisenstat and T. Steihaug, *Inexact Newton methods*, SIAM Journal on Numerical Analysis, Vol. 19, pp. 400-408, 1982.

[9] J. E. Dennis and M. El-Alem, *A trust-region algorithm for solving feasibility problems*, Technical Report, Department of Applied and Computational Mathematics, Rice University, Houston, Texas, 1996.

[10] J. E. Dennis, 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 (1997), 177-207.

[11] J. E. Dennis and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice Hall, Englewood Cliffs, NJ, 1983.

[12] M. El-Alem, *A robust trust region algorithm with a nonmonotonic penalty parameter scheme for constrained optimization*, SIAM Journal on Optimization, Vol. 5, pp. 348-378, 1995.

[13] D. M. Gay, *A trust-region approach to linearly constrained optimization*, Poceedings of the Dundee Biennial Conference on Numerical Analysis, 1983.

[14] P. E. Gill, W. Murray and M. H. Wright, *Practical optimization*, Academic Press, London, England, 1981.

[15] F. M. Gomes, M. C. Maciel and J. M. Martínez, *Nonlinear programming algorithms using trust regions and augmented Lagrangians with nonmonotone penalty parameters*, Relatório Técnico, IMECC-UNICAMP, Campinas, 1995. To appear in Mathematical Programming.

[16] M. Lalee, J. Nocedal and T. Plantenga, *On the implementation of an algorithm for large-scale equality constrained optimization*, Technical Report N.A.M. 08, E.E.C.S. Dept., Northwestern University, 1993.

[17] L. S. Lasdon, *Reduced gradient methods*, in *Nonlinear Optimization 1981*, edited by M. J. D. Powell, Academic Press, New York, pp. 235-242, 1982.

[18] J. M. Martínez, *A two-phase trust-region model algorithm with global convergence for nonlinear programming, Journal of Optimization Theory and Applications*, Vol. 96, pp. 397-436, 1998.

[19] J. M. Martínez and S. A. Santos, *A trust region strategy for minimization on arbitrary domains*, Mathematical Programming, Vol. 68, pp. 267-302, 1995.

[20] A. Miele, H. Y. Huang and J. C. Heideman, *Sequential gradient-restoration algorithm for the minimization of constrained functions, ordinary and conjugate gradient version*, Journal of Optimization Theory and Applications, Vol. 4, pp. 213-246, 1969.

[21] A. Miele, A. V. Levy and E. E. Cragg, *Modifications and extensions of the conjugate-gradient restoration algorithm for mathematical programming problems*, Journal of Optimization Theory and Applications, Vol. 7, pp. 450-472, 1971.

[22] A. Miele, E. M. Sims and V. K. Basapur, *Sequential Gradient-Restoration algorithm for mathematical programming problem with inequality constraints, Part 1, Theory*, Rice University, Aero-Astronautics Report No. 168, 1983.

[23] R. B. Murtagh and M. A. Saunders, *Large-scale linearly constrained optimization*, Mathematical Programming, Vol. 14 , pp. 41-72, 1978.

[24] B. A. Murtagh and M. A. Saunders, *MINOS 5.4 User's guide*, Technical Report SOL 83-20 R, 1995, Stanford University.

[25] E. Omojokun, *Trust-region strategies for optimization with nonlinear equality and inequality constraints*, PhD Thesis, Department of Computer Science, University of Colorado, Boulder, Colorado, 1989.

[26] M. J. D. Powell and Y. Yuan, *A trust-region algorithm for equality constrained optimization*, Mathematical Programming, Vol. 49, pp. 190-211, 1991.

[27] M. Rom and M. Avriel, *Properties of the sequential gradient-restoration algorithm (SGRA), Part 1: Introduction and comparison with related methods*, Journal of Optimization Theory and Applications, Vol. 62, pp. 77-98, 1989.

[28] M. Rom and M. Avriel, *Properties of the sequential gradient-restoration algorithm (SGRA), Part 2: Convergence Analysis*, Journal of Optimization Theory and Applications, Vol. 62, pp. 99-126, 1989.

[29] J. B. Rosen, *The gradient projection method for nonlinear programming, Part 1, Linear constraints*, SIAM Journal on Applied Mathematics, Vol. 8, pp. 181-217, 1960.

[30] J. B. Rosen, *The gradient projection method for nonlinear programming, Part 2, Nonlinear constraints*, SIAM Journal on Applied Mathematics, Vol. 9, pp. 514-532, 1961.

[31] J. B. Rosen, *Two-phase algorithm for nonlinear constraint problems*, Nonlinear Programming 3, Edited by O. L. Mangasarian, R. R. Meyer and S. M. Robinson, Academic Press, London and New York, pp. 97-124, 1978.

[32] E. B. Saff and A. B. J. Kuijlaars, *Distributing many points on a sphere*, Mathematical Intelligencer, Vol. 19, pp. 5–11, 1997.

[33] M. H. Wright, *Interior point methods for constrained optimization*, Acta Numerica 1992, pp. 341-407, 1992.

Capítulo 3

Métodos gradiente espectral para otimização com restrições lineares de igualdade

Resumo

No presente trabalho são considerados problemas de otimização com restrições lineares e canalizações. Primeiro, é definido um método gradiente espectral precondicionado para o caso sem canalizações.Este algoritmo pode ser visto como um método Quase–Newton onde as aproximações das Hessianas satisfazem uma equação secante fraca. A escolha espectral do passo está inserida na aproximação Hessiana e o algoritmo completo está combinado com uma estratégia de busca linear não monótona. As canalizações são incluídas na função objetivo usando um esquema de penalização exponencial. Este esquema de penalização define as iterações externas do algoritmo geral. Cada iteração externa inclui a aplicação do método gradiente espectral previamente definido para problemas com restrições lineares de igualdade. Conseqüentemente, um problema de programação quadrática convexa com restrições lineares de igualdade deve ser resolvido em cada iteração interna. A matriz KKT extendida associada a este problema fica constante a menos que o processo seja reiniciado. Nas iterações internas somente o lado direito do sistema KKT é modificado. Conseqüentemente, técnicas de fatorações esparsas podem ser efetivamente aplicadas e exploradas. São apresentados promissórios resultados numéricos.

Spectral gradient methods for linearly constrained optimization

J. M. Martínez * E. A. Pilotta † M. Raydan ‡

April 9, 1999

Abstract

Linearly constrained optimization problems with simple bounds are considered in the present work. First, a preconditioned spectral gradient method is defined for the case in which no simple bounds are present. This algorithm can be viewed as a quasi-Newton method in which the approximate Hessians satisfy a weak secant equation. The spectral choice of steplength is embedded into the Hessian approximation, and the whole process is combined with a nonmonotone line search strategy. The simple bounds are then taken into account by placing them in an exponential penalty term that modifies the objective function. The exponential penalty scheme defines the outer iterations of the process. Each outer iteration involves the application of the previously defined preconditioned spectral gradient method for linear equality constrained problems. Therefore, an equality constrained convex quadratic programming problem needs to be solved at every inner iteration. The associated extended KKT matrix remains constant unless the process is reinitiated. In ordinary inner iterations, only the right hand side of the KKT system changes. Therefore, suitable sparse factorization techniques can be effectively applied and exploited. Encouraging numerical experiments are presented.

Key words: Linearly constrained optimization, quasi-Newton methods, exponential penalty methods, spectral gradient method, nonmonotone line search.

*Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil (martinez@ime.unicamp.br). This author was supported by FAPESP (Grant 90-3724-6), FINEP and FAEP-UNICAMP.

†Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil (pilotta@ime.unicamp.br). This author was supported by FAPESP (Grant 90-3724-6), FINEP and FAEP-UNICAMP.

‡Departamento de Computación, Facultad de Ciencias, Universidad Central de Venezuela, Ap. 47002, Caracas 1041-A, Venezuela (mraydan@reacciun.ve). Work sponsored by FAPESP, FAEP-UNICAMP and the Numerical Analysis Laboratory at UCV.

1 Introduction

Spectral gradient methods have proved to be of great value in unconstrained optimization problems. They were introduced by Barzilai and Borwein [1], and later analyzed by Raydan [19]. They have been applied to find local minimizers of large scale problems (Raydan [5, 4, 20]), and also to explore faces of large dimensions in box-constrained optimization (see Bielschowsky et al. [3] and Friedlander et al. [11]). More recently, spectral gradient methods were extended by Birgin et al. [6] to minimize general smooth functions on convex sets. In this case, the spectral choice of steplength was combined with the projected gradient method to obtain a robust and effective low cost computational scheme.

In this work, we present a spectral gradient method for the linearly constrained optimization problem

$$\text{Minimize } f(x) \text{ subject to } Ax = b,\ x \geq 0,$$

where $f : I\!R^n \to I\!R$ is a smooth function. We consider, first, the case in which the bounds $x \geq 0$ are not present. For this case, we present a quasi-Newton method in which the secant approximation satisfies a weak secant equation. Our method can also be viewed as a preconditioned spectral gradient method (see Luengo et al. [16]), in which the secant update plays the role of a preconditioner matrix that can be reinitialized whenever some indicator of performance reveals that this is convenient. The spectral choice of steplength is embedded into the secant matrix via a simple Rayleigh quotient scalar multiplication, and the whole process is combined with a nonmonotone line search strategy. The simple bounds are then taken into account by means of an exponential penalty term that modifies the objective function. Each modification of the penalty term defines a different outer iteration. At each outer iteration we apply the preconditioned spectral gradient method for linear equality constrained problems defined before.

Concerning the implementation, a basis for the null space of A is not required at all. To obtain the search direction we need to solve a convex quadratic programming problem at every inner iteration. As a consequence of the spectral choice of steplength, the associated extended KKT matrix remains constant unless the process is reinitialized, and only the right hand side of the KKT system changes. Therefore, suitable sparse factorization techniques can be effectively exploited.

This work is organized as follows. In section 2 we present the spectral gradient method for linear equality constrained problems. We describe the calculation of the search direction, the nonmonotone line search and the convergence properties of the method. In section 3, we present an exponential penalty approach to solve the linearly constrained optimization problem with simple bounds. In section 4 we give a global convergence result for the proposed outer scheme. In section 5, we describe

implementation features. In section 6, we present computational results. Finally, in section 7, we state some conclusions.

2 Linearly constrained preconditioned spectral gradient without bounds

Let us consider the problem

$$\text{Minimize} \quad F(x) \quad \text{subject to} \quad Ax = b, \tag{1}$$

where $F : \mathbb{R}^n \to \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$, $F \in C^1(\mathbb{R}^n)$, and b lies in the range of A. We denote $g(x) = \nabla F(x)$ for all $x \in \mathbb{R}^n$.

In this section we define a quasi-Newton algorithm for solving (1). Assume that $0 < \alpha_{\min} < \alpha_{\max} < \infty$. Given $x_k \in \mathbb{R}^n$ such that $Ax_k = b$ and a positive definite matrix B_k (this can be relaxed to $Z^T B_k Z > 0$, where the columns of Z are a basis of the null space of A), the steps for obtaining x_{k+1} are given by the following algorithm.

Algorithm 1 *Preconditioned spectral gradient with linear equality constraints*
Step 1: Obtain $d_k \in \mathbb{R}^n$ the unique solution of

$$\text{Minimize } \frac{1}{2} d^T B_k d + g(x_k)^T d \quad \text{subject to} \quad Ad = 0. \tag{2}$$

If $d_k = 0$ terminate. (x_k is a stationary point of problem (1).)
Step 2: Compute, using a procedure that will be specified later (in Algorithm 2), $\lambda_k > 0$ (the steplength). Define

$$\begin{aligned} x_{k+1} &= x_k + \lambda_k d_k, \\ s_k &= x_{k+1} - x_k, \\ y_k &= g(x_{k+1}) - g(x_k), \\ B_{k+1} &= \alpha_k B_0, \end{aligned} \tag{3}$$

where

$$\alpha_k = \begin{cases} \max\left(\alpha_{\min}, \min\left(\alpha_{\max}, \frac{s_k^T y_k}{s_k^T B_0 s_k}\right)\right) & \text{if } s_k^T y_k > 0 \\ \alpha_{\min} & \text{else .} \end{cases}$$

Return to Step 1.

The solution of (2) is given by

$$\begin{bmatrix} B_k & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \psi_k \end{bmatrix} = \begin{bmatrix} -g_k \\ 0 \end{bmatrix}, \tag{4}$$

where $g_k = g(x_k)$. The matrix of the linear system is referred as the KKT-matrix. Note that in the particular case in which $B_k = I$, the solution d_k is the orthogonal projection of $-g(x_k)$ on the null-space of A.

Remarks

1. With the appropriate changes on the bounds for α_k we can define $B_{k+1} = \alpha_k B_k$ using $\frac{s_k^T y_k}{s_k^T B_k s_k}$ instead of $\frac{s_k^T y_k}{s_k^T B_0 s_k}$. In this way it is easier to visualize the algorithm as a quasi-Newton method in which, when the bounds for α_k are not violated, the matrix B_{k+1} satisfies the "weak secant equation"

$$s_k^T B s_k = s_k^T y_k. \tag{5}$$

   The geometrical meaning of (5) is that the directional derivative of the quadratic model of F coincides with the directional derivative of F at the previous point. Obviously, the gradient of the quadratic model coincides with the gradient of F at the current point. With our spectral choice, the directional derivatives of the quadratic model on directions that are B_k-orthogonal to the increment coincide with the directional derivative of F at the current point.

2. If no constraints are present at all and $B_0 = I$ this algorithm is, essentially, the one defined by Raydan in [20]. The present algorithm generalizes also in many senses the preconditioned spectral gradient method introduced in [16], where B_0 plays the role of the preconditioner matrix.

3. The quantity $\frac{s_k^T y_k}{s_k^T B_0 s_k}$ is frequently refered as a Rayleigh quotient. In fact, if $F \in C^2(\mathbb{R}^n)$, we have:

$$y_k = [\int_0^1 \nabla^2 F(x_k + t s_k) dt] s_k.$$

   So, defining $w_k = B_0^{\frac{1}{2}} s_k$ and

$$A_k = B_0^{-\frac{1}{2}} [\int_0^1 \nabla^2 F(x_k + t s_k) dt] B_0^{-\frac{1}{2}},$$

   we obtain:

$$\frac{s_k^T y_k}{s_k^T B_0 s_k} = \frac{w_k^T A_k w^k}{w_k^T w_k}.$$

   Therefore, $\frac{s_k^T y_k}{s_k^T B_0 s_k}$ is a Rayleigh quotient of A_k in the classical sense.

In Algorithm 2 we describe the line-search procedure, that is to say, the way in which we choose λ_k at each iteration. We adopted a nonmonotone line-search

strategy. This means that, following [6, 15, 20], we do not impose decrease of the objective function at every iteration. Instead, we choose a positive integer M at the beginning of the process and we accept a trial point when a sufficient decrease is obtained in relation to the maximum functional value among the M last iterations.

Algorithm 2 *Nonmonotone line-search procedure*

We assume that $\gamma \in (0,1)$ is given independently of the iteration number k and that d_k has been computed using (2).

Step 1 Set $\lambda \leftarrow 1$.

Step 2 Set $x_+ = x_k + \lambda d_k$.

Step 3 If

$$F(x_+) \leq \max_{0 \leq j \leq \min\{k,M\}} F(x_{k-j}) + \gamma \lambda \langle d_k, g(x_k) \rangle, \tag{6}$$

then define $\lambda_k = \lambda$ and finish the line-search.

If (6) does not hold, define

$$\lambda_{new} \in [0.1\lambda, 0.9\lambda], \tag{7}$$

set $\lambda \leftarrow \lambda_{new}$ and go to Step 2.

Algorithm 1 admits many possible implementations according to the choice of the matrix B_0. The "pure" spectral gradient method with linear constraints corresponds to $B_0 = I$. Our approach here, similarly to the one of [16], is to consider that Algorithm 1 could be re-initiated with a different initial matrix B_0 whenever some indicator of performance reveals that this is convenient. The whole process is described below.

Algorithm 3 *Spectral gradient for linear equality constraints with re-initialization*

Assume that the positive definite matrix B_0 is given, as well as the positive integer M and the sufficient decrease parameter $\gamma \in (0,1)$.

Step 1. Set $kount \leftarrow 0, k \leftarrow kount$

Step 2. Execute Algorithm 1 with the line-search procedure given by Algorithm 2.

Step 3. If Algorithm 1 terminates at Step 1, terminate the execution of Algorithm 3 too.

Step 4. Decide whether it is necessary to re-initiate Algorithm 1 (YES-NO).

Step 5. If NO, set $kount \leftarrow kount + 1, k \leftarrow k + 1$ and go to Step 2.

Step 6. If YES, define a new B_0, set $kount \leftarrow kount + 1$, $k \leftarrow 0$ and go to Step 2.

The following observation plays an important role in our convergence analysis and also in our implementation. Let (d_k, ψ_k) be a solution of the system (4). Then d_k belongs to the null space of A, that is, for some w_k, $d_k = Zw_k$ where the columns of Z form a basis of the null space of A. Therefore

$$B_k Z w_k + A^T \psi_k = -g_k.$$

So,

$$Z^T B_k Z w_k + Z^T A^T \psi_k = -Z^T g_k.$$

Since $AZ = 0$ and B_k is positive definite, we have

$$w_k = -(Z^T B_k Z)^{-1} Z^T g_k,$$

and hence

$$d_k = -Z(Z^T B_k Z)^{-1} Z^T g_k. \tag{8}$$

Now, using the update for the matrix B_k in Algorithm 1, we obtain

$$d_k = -Z(Z^T B_0 Z)^{-1} Z^T \left(\frac{g_k}{\alpha_{k-1}} \right). \tag{9}$$

Therefore, the search direction d_k can be obtained using the KKT-matrix of the last re-initialization the corresponding independent vector by means of

$$\begin{bmatrix} B_0 & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \psi_k \end{bmatrix} = \begin{bmatrix} -\frac{g_k}{\alpha_{k-1}} \\ 0 \end{bmatrix} \tag{10}$$

Our next result establishes the convergence properties of Algorithm 3.

Theorem 1 *Suppose that at all initializations of B_0 the eigenvalues of $Z^T B_0 Z$ are clustered in a strictly positive closed and bounded interval. Then Algorithm 3 is well defined and every limit point is stationary.*

Proof. Since the eigenvalues of $Z^T B_0 Z$ remain in the strictly positive interval $[\lambda_{\min}, \lambda_{\max}]$, for all initializations of B_0, then we can assume for the sake of clarity, and without loss of generality, that no re-initialization is performed, and that the smallest and largest eigenvalues of $Z^T B_0 Z$ are $\lambda_{\min}$ and $\lambda_{\max}$ respectively.

If x_k is not a constrained stationary point, then $Z^T g_k \neq 0$, and the search direction d_k in (9) satisfies

$$g_k^T d_k = -\frac{1}{\alpha_{k-1}} (Z^T g_k)^T (Z^T B_0 Z)^{-1} (Z^T g_k) \leq -\frac{\|Z^T g_k\|^2}{\alpha_{\max} \lambda_{\max}} < 0.$$

(Throughout this paper $\|\cdot\|$ means $\|\cdot\|_2$). Hence, a stepsize satisfying (6) will be found after a finite number of trials, and Algorithm 3 is well defined.

Let $\bar{x}$ be an accumulation point of $\{x_k\}$, and relabel $\{x_k\}$ a subsequence converging to $\bar{x}$. Suppose, by contradiction, that $\bar{x}$ is not stationary. Then $\|Z^T g(\bar{x})\| = \hat{\delta} > 0$. This implies by continuity that

$$\|Z^T g(x_k)\| > \hat{\delta}/2, \tag{11}$$

for k large enough on the subsequence that converges to $\bar{x}$. We consider two cases:

Case 1. Assume that $\inf \lambda_k = 0$. Hence, there exists a subsequence $\{x_k\}_K$ such that

$$\lim_{k\to\infty,\, k\in K} \lambda_k = 0.$$

In that case, from the way λ_k is chosen in (6), there exists an index $\bar{k}$ sufficiently large such that for all $k \geq \bar{k}$, $k \in K$, there exists ω_k, $(0 < 0.1 \leq \omega_k \leq 0.9)$, for which $\lambda_k/\omega_k > 0$ fails to satisfy condition (6), i.e.,

$$F(x_k + \frac{\lambda_k}{\omega_k} d_k) > \max_{0\leq j\leq M} F(x_{k-j}) + \gamma \frac{\lambda_k}{\omega_k} \langle g(x_k), d_k\rangle \geq F(x_k) + \gamma \frac{\lambda_k}{\omega_k} \langle g(x_k), d_k\rangle.$$

As a consequence,

$$\frac{F(x_k + \frac{\lambda_k}{\omega_k} d_k) - F(x_k)}{\lambda_k/\omega_k} > \gamma \langle g(x_k), d_k\rangle.$$

By the mean value theorem, this relation can be written as

$$\langle g(x_k + t_k d_k), d_k\rangle > \gamma \langle g(x_k), d_k\rangle, \quad \text{for all } k \in K,\ k \geq \bar{k}, \tag{12}$$

where t_k is a scalar in the interval $[0, \lambda_k/\omega_k]$ that goes to zero as $k \in K$ goes to infinity.

Taking a convenient subsequence such that $d_k/\|d_k\|$ is convergent to $\bar{d}$, and taking limits in (12) we deduce that $(1-\gamma)\langle g(\bar{x}), \bar{d}\rangle \geq 0$. (In fact, observe that $\{\|d_k\|\}_K$ is bounded and so $t_k\|d_k\| \to 0$.) Since $(1-\gamma) > 0$ and $\langle g(x_k), d_k\rangle < 0$ for all k, then $\langle g(\bar{x}), \bar{d}\rangle = 0$.

Using (9) this implies that $\{\alpha_{k-1}^{-1} (Z^T g_k)^T (Z^T B_0 Z)^{-1} (Z^T g_k)\}$ goes to zero on that subsequence. However,

$$\frac{1}{\alpha_{k-1}} (Z^T g_k)^T (Z^T B_0 Z)^{-1} (Z^T g_k) \geq \frac{\|Z^T g_k\|^2}{\lambda_{\max}\, \alpha_{\max}}.$$

Therefore, $\|Z^T g_k\|$ goes to zero when $k \in K$ goes to infinity. Thus, by continuity, for k large enough on that subsequence we have that

$$\|Z^T g_k\| < \hat{\delta}/2\,,$$

which contradicts (11).

Case 2. Assume that $\inf \lambda_k \geq \omega > 0$. Using the first part of the proof of the theorem in [15, p. 709], we obtain a monotonically nonincreasing sequence $\{F(x_{l(k)})\}$. Indeed, let $l(k)$ be an integer such that $k - \min\{k, M-1\} \leq l(k) \leq k$, and

$$F(x_{l(k)}) = \max_{0 \leq j \leq \min\{k,M\}} [F(x_{k-j})].$$

From (6) it follows that, for $k > M$ (see [15] for details)

$$F(x_{l(k)}) \leq F(x_{l(l(k)-1)}) + \gamma \lambda_{l(k)-1} \langle g(x_{l(k)-1}), d_{l(k)-1} \rangle.$$

Using (9) and (11), we obtain

$$\begin{aligned} F(x_{l(k)}) &= F(x_{l(l(k)-1)}) - \frac{\gamma\, \lambda_{l(k)-1}}{\alpha_{l(k)-1}} (Z^T g(x_{l(k)-1}))^T (Z^T B_0 Z)^{-1} (Z^T g(x_{l(k)-1})) \\ &\leq F(x_{l(l(k)-1)}) - \frac{\gamma\, \omega\, \|Z^T g(x_{l(k)-1})\|^2}{\alpha_{\max}\, \lambda_{\max}} \\ &\leq F(x_{l(l(k)-1)}) - \frac{\gamma\, \omega\, \delta^2}{4\, \alpha_{\max}\, \lambda_{\max}} \end{aligned}$$

When $k \to \infty$, clearly $F(x_{l(k)}) \to -\infty$ which is a contradiction. In fact, f is a continuous function and so $F(x_k)$ converges to $F(\bar{x})$. □

3 Adding bounds on the variables

Now, we consider the following linearly constrained optimization problem with simple bounds :

$$\text{Minimize } f(x) \text{ subject to } Ax = b,\ x \geq 0, \tag{13}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, $f \in C^1(\mathbb{R}^n)$. Our arguments below can be easily extended to a problem with simple upper and lower bounds:

$$\text{Minimize } f(x) \text{ subject to } Ax = b,\ \ell \leq x \leq u,$$

however we will restrict ourselves to the case (13) in order to simplify the notation.

We eliminate the positivity constraints in (13) using penalization. There are many possibilities to do this. The reputation of the logarithmic barrier function lead us to try an interior-point scheme at early stages of this work. This idea did not work well. Roughly speaking, the first trial step almost always violates the bounds $x > 0$ so that, at most inner iterations, the step must be reduced. As a consequence, if there is any benefit in the spectral choice of the step, this benefit

disappears when the method is associated to the logarithmic barrier function. Of course, this does not mean that the *log*-function is not effective when it is associated to the pure Newtonian direction, as many theoretical and practical studies show.

In order to preserve the smoothness of the objective function and to keep the variables unconstrained, we decided to consider the modified exponential penalty function [2, 17, 18] given by:

$$\text{axp}(z) = \begin{cases} e^z & \text{if } z \leq 0 \\ 1 + z + \frac{1}{2}z^2 & \text{if } z > 0. \end{cases}$$

As a consequence, we have to solve the following minimization subproblem for each penalty parameter ρ:

$$\text{Minimize } F(x) \equiv f(x) + \frac{1}{\rho}\sum_{i=1}^{n} \mu_i \text{axp}(-\rho x_i), \text{ subject to } Ax = b. \qquad (14)$$

The meaning of μ_i will be given later. We are only going to assume that $0 < \mu_{min} \leq \mu_i \leq \mu_{max} < \infty$ for all $i = 1, \ldots, n$. The outer algorithm consists in solving a sequence of penalty problems (14). Each penalty problem is solved using Algorithm 3. Clearly, the exponential penalty approach does not require positivity of the initial point.

4 Global convergence

In this section, we will show that, under certain conditions, the scheme based on successive approximate minimizations of (14) really works. Similar results, for related algorithms, were given, for example, in [2] and [17]. The outer scheme proposed in the previous section is an exponential Lagrangian method. At each outer iteration, the sequence of inner steps will be given by Algorithm 1. Accordingly, an inexact stopping convergence criterion will be defined, based on computable quantities.

We will need several assumptions. The first assumption guarantees that, at each outer iteration, Algorithm 1 will be able to stop.

Assumption 1. *There exists $\rho_0 > 0$ such that, if $\rho \geq \rho_0$ and $\mu_i \in [\mu_{min}, \mu_{max}]$, the level sets of $F(x)$ in (14) are bounded.*

Observe that Assumption 1 is stronger than saying that $f(x)$ has bounded level sets on the feasible region $Ax = b$, $x \geq 0$, since it involves the behavior of $f(x)$ on infeasible points too. If $f(x)$ goes to $-\infty$ very quickly for $x_i \to -\infty$, Assumption 1 might not hold. It can be argued that, in this case, it is inexpensive to modify the definition of $f(x)$ in the infeasible region, but even this operation is risky and can create undesirable local minimizers.

Now, we are able to define the outer algorithm. Each iterate of this algorithm will be $x^k \in I\!R^n$, whose existence is guaranteed by Assumption 1.

Algorithm 4 *Outer iterations*

Let $\varepsilon_k > 0$ for all $k = 0, 1, 2, \ldots$ be such that $\lim_{k\to\infty} \varepsilon_k = 0$. Let ρ_0 be given by Assumption 1, as well as μ_{min} and μ_{max}. Let $\tau \in (0, 1)$, $\eta > 1$. Initialize $\sigma_{-1} = 0$ and $k \leftarrow 0$.

Step 1. Choose $\mu_i^k \in [\mu_{min}, \mu_{max}]$ for all $i = 1, \ldots, n$.

Step 2. Define

$$F_k(x) = f(x) + \frac{1}{\rho_k} \sum_{i=1}^{n} \mu_i^k \ \mathrm{axp}(-\rho_k x_i).$$

Apply Algorithm 1 for

$$\text{Minimize } F_k(x) \quad \text{subject to} \quad Ax = b.$$

Use, as stopping criterion, the test

$$\|d\| \leq \varepsilon_k,$$

where d is the solution of (2). (From now on, with some abuse of notation, we will denote $d_k = d$ in this section.) The final iterate of Algorithm 1 so far obtained will be called x^k.

Step 3. Compute $\underline{\mu}^k \in I\!R^n$ by

$$\underline{\mu}_i^k = \mu_i^k \ \ \mathrm{axp}'(-\rho_k x_i^k), i = 1, \ldots, n. \tag{15}$$

Step 4. Define

$$\sigma_k = \ \max \ \{|\min \ \{\underline{\mu}_i^k, x_i^k\}|, i = 1, \ldots, n\}.$$

Step 5. If $\sigma_k \leq \tau\sigma_{k-1}$ define $\rho_{k+1} = \rho_k$. Else, define $\rho_{k+1} = \eta\rho_k$.

Step 6. Set $k \leftarrow k + 1$. Go to Step 1.

The vector $\underline{\mu}^k$ is intended to be an estimator of the Lagrange multipliers associated to the inequality constraints. For this reason, the choice $\mu^k = \underline{\mu}^{k-1}$ could be quite natural at Step 2, if the safeguards defined by μ_{min} and μ_{max} are not violated. However, this specific choice is not necessary for proving convergence. The motivation of Step 4 of the algorithm is that the conditions on positivity and complementarity that relate the solution and the Lagrange multipliers can be written as

$$\min \ \{\mu_i, x_i\} = 0, \ \ i = 1, \ldots, n.$$

Therefore, Step 4 measures the progress related to the positivity-complementarity requirement. If this progress is satisfactory, it is not necessary to increase the penalty

parameter. In our practical implementations we used $\mu_i^k \equiv 1$, so that the whole practical process can be interpreted as a penalty scheme where the tests of Step 4 determine if the penalty parameter must be increased or not.

In addition to Assumption 1, other assumptions for convergence of Algorithm 4 are merged in the hypothesis of the following theorem. The main one is that the sequence of inequality Lagrange multiplier estimates must be bounded. If this happens, we can guarantee that limit points of $\{x^k\}$ are stationary points of (13). Of course, in practical terms we also need that these limit points exist, which means that a bounded subsequence of $\{x^k\}$ must exist.

Theorem 2. *Assume that $\{\underline{\mu}^k\}$ is bounded. Then, every limit point of $\{x^k\}$ is a stationary point of (13).*

Proof. Let $x^* \in \mathbb{R}^n$ be a limit point of $\{x^k\}$ and let $K_1 \subset \mathbb{N}$ be such that $\lim_{k\in K_1} x^k = x^*$. Take $K_2 \subset K_1$ such that $\lim_{k\in K_2} \underline{\mu}^k = \underline{\mu}^*$. Clearly,

$$\underline{\mu}^* \geq 0.$$

Define, with some abuse of notation, $B_k \in \mathbb{R}^{n\times n}$ the matrix used at the last inner iteration of the outer iteration k (which gave $\|d_k\| \leq \varepsilon_k$). By the hypothesis of Theorem 1 and the safeguards on α_k, the eigenvalues of the B_k's are clustered in a strictly positive interval. Therefore, there exists $K_3 \subset K_2$ and a positive definite matrix B_* such that

$$\lim_{k\in K_3} B_k = B_*.$$

Taking limits for $k \in K_3$ in $\|d_k\| \leq \varepsilon_k$, we obtain, by (8), that

$$Z(Z^T B_* Z)^{-1} Z^T \nabla(\nabla f(x^*) - \underline{\mu}^*) = 0.$$

This implies that there exists $\lambda \in \mathbb{R}^m$ such that

$$\nabla f(x^*) + A^T\lambda - \underline{\mu}^* = 0, \tag{16}$$

with $\underline{\mu}^* \geq 0$.

Now, we consider two cases. In the first case, there exists $k_0 \in \mathbb{N}$ such that $\sigma_k \leq \tau\sigma_{k-1}$ for all $k \geq k_0$. By continuity, this implies that

$$\min\{\underline{\mu}_i^*, x_i^*\} = 0 \ \ \forall \ \ i = 1, \ldots, n. \tag{17}$$

Clearly, (16) and (17) imply that x^* is a stationary point of (13).

In the second case $\lim_{k\to\infty} \rho_k = \infty$. By hypotheses, $\{\underline{\mu}^k\}$ is bounded. Therefore, by (15) and the choice of μ^k, $\{\mathrm{axp}'(-\rho_k x_i^k)\}$ is bounded for all $i = 1, \ldots, n$. By the definition of $\mathrm{axp}(z)$, this implies that $\inf x_i^k \geq 0$ for all $i = 1, \ldots, n$. Therefore, $x^* \geq 0$.

Now, if $x_i^* > 0$, taking $k \in K_3$ large enough we have that

$$x_i^k \geq \frac{x_i^*}{2} > 0.$$

So,

$$\lim_{k \in K_3} \rho_k x_i^k = \infty.$$

By the boundedness of μ_k and (15), this implies that $\underline{\mu}_i^k \to 0$. So, $\underline{\mu}_i^* = 0$. This means that complementarity holds. Thus, the proof is complete. □

5 Implementation features

This section deals with some specific algorithmic choices. We present the results of computer experiments in the next section. Remember that, as we said in the previous section, we used $\mu_i^k \equiv 1$ in our implementation of outer iterations.

Computation of α_k and d_k in Algorithm 1. The expression for α_k can be simplified when $k \geq 1$ as follows. Notice first that

$$s_k^T B_0 s_k = \frac{s_k^T B_k s_k}{\alpha_{k-1}} = \frac{\lambda_k s_k^T B_k d_k}{\alpha_{k-1}}.$$

From (4) we obtain that $B_k d_k = -g_k - A^T \psi_k$. Since s_k is a feasible direction, then $s_k^T A^T = 0$, and so $s_k^T B_k d_k = -s_k^T g_k$. Hence,

$$s_k^T B_0 s_k = -\frac{\lambda_k s_k^T g_k}{\alpha_{k-1}} = -\frac{\lambda_k^2}{\alpha_{k-1}} d_k^T g_k.$$

So, the expression for α_k can be reduced to

$$\alpha_k = -\frac{d_k^T y_k}{\lambda_k d_k^T g_k} \alpha_{k-1}.$$

For this new expression the vector s_k is not required, and the calculation of $d_k^T g_k$ can be reused in (6) for each trial step during the nonmonotone line search.

On the other hand, the direction d_k could be obtained by solving the system (4). If the order of the system were small it could be solved using the LU factorization with partial pivoting or the QR factorization. Using these linear solvers, the cost of solving the system would be $O(n^3)$ floating point operations. For large-scale linear systems this cost becomes prohibitive. So, it is convenient to compute d_k by solving instead the system (10). In this case, we will solve several systems for consecutive

iterations, with different independent vectors, using only one factorization. Moreover, since we want to exploit the fact that the KKT matrix is symmetric we use the subroutine MA27 of Duff and Reid [10]. This routine is suitable for solving specifically sparse symmetric indefinite linear systems. Finally, we note that the block B_0 changes when a re-initialization is needed in Step 6 of Algorithm 3, so the KKT-matrix changes too. However the sparsity structure is constant during the whole process and we can also exploit this fact using suitable routines from MA27.

Initial point. To start Algorithm 3 we need an initial point x_0, which must satisfy the linear equality constraints $Ax = b$. That point can be easily obtained solving the linear system

$$\begin{bmatrix} -I & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

Observe that the matrix of this system has the same structure as the KKT matrix of (4). So, we used again the subroutine MA27.

Stopping criterion. The outer exponential algorithm was stopped when, at a solution of (14), we have that

$$x_i \geq -\varepsilon \ \ \forall \ \ i = 1, \ldots, n \tag{18}$$

and, for all $i = 1, \ldots, n$,

$$\exp'(-\rho x_i) \leq \varepsilon \ \text{ whenever } \ x_i \geq \varepsilon. \tag{19}$$

Condition (18) indicates that the current outer iterate is (almost) feasible. The quantities $\exp'(-\rho x_i)$ are estimates of the Lagrange multipliers corresponding to the constraints $-x_i \leq 0$. Therefore, (19) indicates that the current point satisfies (approximately) the complementarity conditions. The fact that the gradient of the objective function is (approximately) a linear combination of the gradients of the active constraints must be guaranteed by the convergence criterion used in the inner algorithm. By (8) this property will hold if $d_k = 0$. Therefore, as inner convergence criterion of Algorithm 1, we used:

$$\|d_k\| \leq \varepsilon'. \tag{20}$$

Angle criterion. Since we do not have direct access to positive-definiteness of the matrices $Z^T B_k Z$, we must control the fact that descent directions are effectively generated in an indirect way. Here we decided to use

$$\nabla F(x_k)^T d_k \leq -\theta \|d_k\|^2. \tag{21}$$

Inequality (21) says that the appropriate Rayleigh quotient (see the proof of Theorem 1) related to $Z^T B_k Z$ is positive.

When (21) is not satisfied we have to consider two cases:

1. If it happens immediately after a re-initialization (k = 0) we define $B_k = B_k + \rho I$, $\rho > 0$ and a new factorization of the KKT-matrix is performed. This procedure is repeated until the angle condition is satisfied.

2. Otherwise, we perform a re-initialization and continue with Step 2 of Algorithm 3.

Re-initialization criterion at Step 4 in Algorithm 3. Among the many possible re-starting procedures, we decided to use, after preliminary experiments, the following one, which depends of a positive integer parameter p:

(a) If we are at the first outer iteration or if at the previous outer iteration the inner algorithm converged within *limit* iterations, then at the current outer iteration the algorithm is restarted every p^k inner iterations.

(b) If, at the previous outer iteration, the inner algorithm did not converge after *limit* iterations, at the current outer iteration the algorithm is restarted every p iterations.

(c) The inner algorithm is restarted if condition (21) is not satisfied.

The rationale behind (a) is that, at the first iterations of the inner algorithm, large steps are expected so that the Hessian changes more abruptly than at the final iterations. Therefore, restarts must be more frequent at the first iterations than at the final ones. However, we turn to criterion (b), that restarts the algorithm after equally spaced iterations if criterion (a) did not work well.

Several strategies can be proposed for choosing B_0 when re-initialization is recommended. Since in all our experiments $f \in C^2(\mathbb{R}^n)$ we adopt the most obvious one, that is

$$B_0 = \nabla^2 F(x_{kount}). \tag{22}$$

Nonmonotone line-search. When (6) does not hold in Algorithm 2, we compute the new step size using quadratic interpolation with safeguards. See for instance [9, p. 127].

Updating the penalty parameters. After finishing each outer iteration we update the penalty parameter ρ in a typical way: $\rho_{k+1} = 10\rho_k$. At the first outer iteration we used $\rho_1 = 10$.

6 Numerical experiments

We tested our algorithm using the following family of problems. Given a positive integer κ, we define

$$\begin{gathered}
\text{Minimize } \tfrac{1}{2}\sum_{i=1}^{\kappa-2}(x_{\kappa+i+1}-x_{\kappa+i})^2 \\
\text{subject to } x_{\kappa+i}-x_{i+1}+x_i=0, \quad i=1,\ldots,\kappa-1, \\
\alpha_i \le x_i \le \alpha_{i+1}, \quad i=1,\ldots,\kappa, \\
0.4(\alpha_{i+2}-\alpha_i) \le x_{\kappa+i} \le 0.6((\alpha_{i+2}-\alpha_i)), \quad i=1,\ldots,\kappa-1,
\end{gathered}$$

where the constants α_i are defined by $\alpha_i = 1.0 + (1.01)^{i-1}$.

These problems arise in the optimal placement of nodes in a scheme for solving ordinary differential equations with given boundary values [13]. We solve these problems for different values of κ.

For the numerical experiments we set the following default parameters:

limit $= 300$, maximum number of inner iterations.

$\varepsilon = 10^{-8}$, tolerance for stopping criterion for outer iterations.

$\varepsilon' = 10^{-6}$, tolerance for stopping criterion for internal iterations.

$\gamma = 10^{-4}$, tolerance for nonmonotone line search.

$\theta = 10^{-9}$, constant in the angle criterion.

$\alpha_{\min} = 10^{-3}$, $\alpha_{\max} = 10^{3}$, the safeguards interval for the spectral parameter α.

We tried to solve all the problems using $M = 0, 5$ and 10 for the nonmonotone line search procedure (Algorithm 2). Note that for $M = 0$ we are forcing monotone decrease of the objective function at every iteration. We consider $p = 1, 2, 3$. The case $p = 1$ means that we re-start Algorithm 1 at every iteration. Therefore, this case corresponds to Newton's method.

All the numerical experiments were run on a SUN Ultra 1 Creator in double precision FORTRAN, with the optimization option -O.

The numerical results are shown in Tables 1, 2 and 3, corresponding to different values of the parameter p. For each table we report the parameter κ, the number of rows of the matrix A (m), the number of variables (columns of A) (n), the outer iterations (*Outer*), the inner iterations (*Inner*) and the CPU time in seconds (*Time*) for $M = 0, 5, 10$. Tables 4 and 5 show the performance of our code (with $p = 2, 3$) against an implementation of the Stationary-Newton Method, with $M = 10$. We report the number of outer iterations, inner iterations and the CPU time for the same set of problems.

Finally, in Table 6 we compare the CPU time obtained with our method ($p = 2, 3$, and $M = 0, 10$.) against the well-known package LANCELOT [7, 8]. We used the following defaults parameters for LANCELOT:

- `exact-second-derivatives-used`
- `cg-method-used`
- `inexact-Cauchy-point-required`
- `infinity-norm-trust-region-used`
- `gradient-tolerance 1.0D-06`
- `constraints-tolerance 1.0D-06`

In all the experiments we verify that the three methods that we used (spectral gradient, stationary Newton and LANCELOT) converge to the same solution.

Table 1: Spectral Method, $p = 1$ (NEWTON)

Problems			M = 0			M = 5			M = 10		
κ	m	n	Outer	Inner	Time	Outer	Inner	Time	Outer	Inner	Time
50	49	99	5	38	0.14	5	32	0.11	5	32	0.11
100	99	199	5	42	0.23	5	55	0.26	5	57	0.27
150	149	299	5	48	0.38	5	62	0.43	5	65	0.45
200	199	399	5	80	0.80	5	67	0.65	5	67	0.65
250	249	499	5	81	1.08	5	84	1.11	5	84	1.11
300	299	599	5	90	1.47	5	96	1.50	5	95	1.49
350	349	699	5	77	1.53	5	100	1.85	5	113	2.04
400	399	799	5	148	3.35	5	120	2.55	5	120	2.55
450	449	899	5	136	3.38	5	133	3.29	5	138	3.38
500	499	999	5	139	4.05	5	139	4.01	5	141	4.00
550	549	1099	5	320	12.56	5	711	31.39	5	688	30.60
600	599	1199	5	704	34.44	5	732	34.94	5	718	34.46
650	649	1299	5	623	33.16	5	703	38.10	5	549	28.79
700	699	1399	5	722	59.28	5	719	58.84	5	715	58.65
750	749	1499	5	739	45.88	5	739	45.46	5	732	45.28
800	799	1599	5	731	48.62	5	754	48.82	5	609	37.92
850	849	1699	4	449	42.52	4	441	41.67	4	444	41.80
900	899	1799	4	441	32.73	4	439	32.40	4	439	32.43

Table 2: Spectral Method, $p = 2$

Problems			M = 0			M = 5			M = 10		
κ	m	n	Outer	Inner	Time	Outer	Inner	Time	Outer	Inner	Time
50	49	99	5	204	0.20	5	65	0.09	5	62	0.09
100	99	199	5	148	0.32	5	127	0.24	5	136	0.25
150	149	299	5	147	0.45	5	266	0.67	5	225	0.57
200	199	399	5	342	1.30	5	368	1.46	5	366	1.44
250	249	499	5	350	1.77	5	361	1.77	5	353	1.75
300	299	599	5	380	2.44	5	396	2.49	5	425	2.90
350	349	699	5	527	4.34	5	487	4.17	5	563	4.93
400	399	799	5	404	3.92	5	415	4.03	5	493	5.69
450	449	899	4	605	7.10	4	570	6.82	4	630	7.67
500	499	999	5	447	5.83	5	458	5.97	5	486	6.55
550	549	1099	5	860	16.34	5	641	9.00	5	696	9.30
600	599	1199	5	764	11.58	5	715	11.40	5	696	10.96
650	649	1299	5	934	15.16	5	796	14.57	5	915	16.77
700	699	1399	5	974	25.96	5	956	22.46	5	1270	47.13
750	749	1499	6	888	20.15	5	901	18.14	5	880	18.84
800	799	1599	5	975	25.00	5	1303	44.40	5	909	20.98
850	849	1699	5	1346	54.73	5	916	25.06	4	899	23.18
900	899	1799	5	973	24.74	5	1392	54.48	5	989	26.68

Table 3: Spectral Method, $p = 3$

Problems			M = 0			M = 5			M = 10		
κ	m	n	Outer	Inner	Time	Outer	Inner	Time	Outer	Inner	Time
50	49	99	5	318	0.30	5	144	0.15	5	222	0.20
100	99	199	5	183	0.36	5	295	0.50	5	330	0.55
150	149	299	5	266	0.73	5	282	0.69	5	323	0.80
200	199	399	5	371	1.34	5	360	1.33	5	407	1.70
250	249	499	5	342	1.67	5	434	2.18	5	434	2.15
300	299	599	5	488	3.59	5	560	3.33	4	477	2.98
350	349	699	5	606	4.92	5	457	3.64	5	465	3.70
400	399	799	5	444	4.24	5	452	4.29	5	485	4.66
450	449	899	4	624	6.91	5	534	5.72	5	543	5.72
500	499	999	5	525	6.62	5	546	7.08	5	531	6.93
550	549	1099	5	954	14.97	5	753	10.23	5	716	9.73
600	599	1199	5	907	12.75	5	745	11.44	5	827	12.31
650	649	1299	5	965	15.45	5	888	15.53	5	907	14.94
700	699	1399	5	1030	24.89	6	914	14.90	6	908	17.98
750	749	1499	5	967	17.70	5	842	17.00	6	905	19.25
800	799	1599	5	1231	27.56	6	1046	22.22	6	1039	22.25
850	849	1699	5	1125	31.38	4	977	23.10	5	930	24.53
900	899	1799	5	858	23.54	5	956	21.92	5	1026	25.05

Table 4: Spectral Method vs. Stationary-Newton, $p = 2, M = 10$

Problems			Spectral Method			Stationary-Newton		
κ	m	n	Outer	Inner	Time	Outer	Inner	Time
50	49	99	5	62	0.09	5	264	0.19
100	99	199	5	136	0.25	5	310	0.47
150	149	299	5	225	0.57	5	327	0.78
200	199	399	5	366	1.44	5	410	1.52
250	249	499	5	353	1.75	5	568	2.38
300	299	599	5	425	2.90	5	648	3.93
350	349	699	5	563	4.93	5	585	4.50
400	399	799	5	493	5.69	5	877	10.26
450	449	899	4	630	7.67	5	520	4.85
500	499	999	5	486	6.55	5	697	7.72
550	549	1099	5	696	9.30	5	1067	23.24
600	599	1199	5	696	10.96	5	1115	19.51
650	649	1299	5	915	16.77	5	1275	31.66
700	699	1399	5	1270	47.13	5	1169	24.89
750	749	1499	5	880	18.84	5	1462	40.16
800	799	1599	5	909	20.98	5	1376	40.07
850	849	1699	4	899	23.18	5	1322	45.85
900	899	1799	5	989	26.68	5	1077	33.15

Table 5: Spectral Method vs. Stationary-Newton, $p = 3, M = 10$

Problems			Spectral Method			Stationary-Newton		
κ	m	n	Outer	Inner	Time	Outer	Inner	Time
50	49	99	5	222	0.20	5	309	0.19
100	99	199	5	330	0.55	5	322	0.40
150	149	299	5	323	0.80	5	328	0.63
200	199	399	5	407	1.34	4	463	1.48
250	249	499	5	434	2.15	5	386	1.57
300	299	599	5	477	2.98	5	412	2.16
350	349	699	5	465	3.70	5	411	2.57
400	399	799	5	485	4.66	5	479	4.16
450	449	899	5	543	5.72	5	686	5.92
500	499	999	5	531	6.93	5	721	8.17
550	549	1099	5	716	9.73	5	1074	14.37
600	599	1199	5	827	12.31	5	1163	17.32
650	649	1299	5	907	14.94	5	939	16.17
700	699	1399	6	908	17.98	5	1236	21.17
750	749	1499	6	905	19.25	5	1298	33.54
800	799	1599	6	1039	22.25	5	875	20.93
850	849	1699	5	930	24.53	5	1287	38.05
900	899	1799	5	1026	25.05	5	1288	40.76

Table 6: CPU time of Spectral Method vs. CPU time of LANCELOT

Problems			p = 2		p = 3		LANCELOT
κ	m	n	$M = 0$	$M = 10$	$M = 0$	$M = 10$	
50	49	99	0.20	0.09	0.30	0.20	1.07
100	99	199	0.32	0.25	0.36	0.55	5.98
150	149	299	0.45	0.57	0.73	0.80	14.63
200	199	399	1.30	1.44	1.34	1.70	33.64
250	249	499	1.77	1.75	1.67	2.15	36.69
300	299	599	2.44	2.90	3.59	2.98	52.81
350	349	699	4.34	4.93	4.92	3.70	68.03
400	399	799	3.92	5.69	4.24	4.66	115.01
450	449	899	7.10	7.67	6.91	5.72	93.12
500	499	999	5.83	6.55	6.62	6.93	167.43
550	549	1099	16.34	9.30	14.97	9.73	210.72
600	599	1199	11.58	10.96	12.75	12.31	242.94
650	649	1299	15.16	16.77	15.45	14.94	263.84
700	699	1399	25.96	47.13	24.89	17.98	331.65
750	749	1499	20.15	18.84	17.70	19.25	349.01
800	799	1599	25.00	20.98	27.56	22.25	351.11
850	849	1699	54.73	23.18	31.38	24.53	430.93
900	899	1799	24.74	26.68	23.54	25.05	475.93

7 Final remarks

As we mentioned in the Introduction, spectral gradient methods for unconstrained and bound constrained problems have shown to be very effective when compared to conjugate gradient methods and with other methods especially designed for large scale problems. See [6, 20]. In fact, as we can see in Tables 1–6 the performance of our method was satisfactory for small scale problems and very attractive and encouraging for large scale problems. Now, we proceed to the analysis of these tables.

First of all, we consider Tables 1–3 separately. Table 1 ($p = 1$) shows that there is no difference between monotone ($M = 0$) and nonmonotone versions ($M = 5, 10$), except for problems with $\kappa = 650$ and $\kappa = 800$. Clearly, $p = 1, M = 0$ corresponds, essentially, to a projected Newton's method with backtracking (and modified exponential penalization). On the other hand, Tables 2 and 3 ($p = 2$ and 3 respectively) show that the nonmonotone version was better than the monotone one, specially for large scale problems.

Now, if we consider Tables 1–3 jointly we note that in most of the experiments the number of inner iterations for $p = 2$ and 3 was greater than for $p = 1$. However, for large scale problems, the CPU time for $p = 2$ and 3 and nonmonotone versions were less than those corresponding to the case $p = 1$.

Tables 4 and 5 show that performing the spectral correction of the Hessian at

inner iterations is better than not correcting at all, as the Stationary Newton method does.

Finally, Table 6 shows that using any implementation of the spectral method ($p = 2$ or 3) in the monotone or nonmonotone line search is more effective than using LANCELOT (with the mentioned default parameters) on this set of problems.

Let us try to explain the reasons for this state of facts.

(1) The performance of our method was better than Newton's method except for small problems. (Observe that the sparsity pattern of the matrix (4) is suitable for a rather quick factorization using MA27.) The fact that Newton's method, in any of its variations, is very effective when the linear algebra associated to it is not prohibitive is well known in numerical optimization. Moreover, in that case, the association of Newton with the logarithmic barrier function (see, for example [21, 22]) is usually profitable. It could be easy to construct examples where the situation is completely different, just placing an objective function with a dense or badly structured Hessian. On the other hand, the fact that in our method we do not factorize that matrix in each iteration is a very important advantage, as can be clearly observed comparing the CPU time for large scale problems.

(2) More important is the fact that the spectral preconditioned methods outperformed stationary Newton variations. In fact, stationary Newton methods (with restarts) are good alternatives to Newton when this is not affordable. So, it is interesting to observe that with minimal corrections, which do not increase the computational cost of Newton's stationary methods, we can obtain significant improvements of the overall performance.

(3) The fact that explains the superiority of the methods introduced in this paper against LANCELOT is, essentially, that the matrix A has a structure that makes the factorization of (4) affordable. LANCELOT does not use factorizations at all and, so, this advantage is lost.

(4) As in [6, 15, 20] the numerical results for large scale problems using a nonmonotone line search were better than those using monotone ones.

Given the above observations and possible explanations, let us forecast the practical future of algorithms like the ones presented in this paper. First of all, it is clear for us that they are not going to replace Newton's method, when the linear algebra cost associated to the latter is moderate. However, in situations where a Newton iteration can be performed but is rather expensive, a spectral Hessian scaling performed at most iterations must be more effective than merely repeating the previous Hessian.

If a factorization of (4) is so expensive that it cannot be performed even in a single iteration, two different situations can be distinguished. In the first situation, the matrix responsible by the linear algebra cost associated to the factorization of (4) is the constraint matrix A. In this case, Algorithm 1 cannot be implemented either and, certainly, this is the case in which it is recommendable to incorporate

the constraints $Ax = b$ to the objective function, as LANCELOT does (see [7, 14]). In the second case, the Hessian of the objective function is the one that makes the factorization (4) impossible. In this case, the factorization of

$$\begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix}$$

is probably affordable, and the spectral gradient method based on $B_k \equiv I$ can be comparatively efficient. In an extreme case, the problem is unconstrained ($m = 0$), a situation where the spectral gradient technique has proved to be effective.

Finally, a few words must be said with respect to the penalty strategy used here. There exist at least two completely different strategies for dealing with the bounds. The most classical one is the active set strategy described, for example, in [12] and many other text books and papers. In the last decade, interior point algorithms like the ones described in [22] became increasingly popular. The association of interior point (logarithmic barrier) algorithms with Newtonian directions seems to be very effective. However, this does not seem to be the case when Newtonian directions cannot be computed at all. In this case, we feel that the modified exponential penalization is a good alternative although much research is necessary in order to balance the termination criterion of the inner algorithm with the global convergence criterion.

References

[1] J. Barzilai and J. M. Borwein [1988], Two point step size gradient methods, *IMA Journal of Numerical Analysis* 8, pp. 141-148.

[2] D. P. Bertsekas [1995], *Nonlinear Programming*, Athena Scientific, Belmont, MA.

[3] R. H. Bielschowsky, A. Friedlander, F. A. M. Gomes, J. M. Martínez and M. Raydan [1997], An adaptive algorithm for bound constrained quadratic minimization, *Investigación Operativa* 7, pp. 67-102.

[4] E. G. Birgin, I. Chambouleyron and J. M. Martínez [1999], Estimation of the optical constants and the thickness of thin films using unconstrained optimization. To appear in *Journal of Computational Physics.*

[5] E. G. Birgin and Y. G. Evtushenko [1998], Automatic differentiation and spectral projected gradient methods for optimal control problems, *Optimization Methods and Software* 10, pp. 125–146.

[6] E. G. Birgin, J. M. Martínez and M. Raydan [1997], Nonmonotone Spectral Projected Gradient Methods on Convex Sets, Technical Report, Department of Applied Mathematics, University of Campinas. Submitted to *SIAM Journal on Optimization.*

[7] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1991], A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds, *SIAM Journal on Numerical Analysis* 28, pp. 545 - 572.

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

[9] J. E. Dennis Jr. and R. B. Schnabel [1983], *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs.

[10] I. S. Duff and J. K. Reid [1982], MA27 - a set of Fortran subroutines for solving sparse symmetric sets of linear equations, Technical Report R-10533, Computer Science and System Division, AERE Harwell, Oxford, England.

[11] A. Friedlander, J. M. Martínez and M. Raydan [1995], A new method for large-scale box constrained convex quadratic minimization problems, *Optimization Methods and Software* 5, pp. 57-74.

[12] P. E. Gill, W. Murray and M. H. Wright [1981], *Practical optimization*, Academic Press, London, England.

[13] N. I. M. Gould [1991] An algorithm for large-scale quadratic programming, *IMA Journal of Numerical Analysis* 11, pp. 299-324.

[14] N. Krejić, J. M. Martínez, M. P. Mello and E. A. Pilotta [1999], Validation of an augmented Lagrangian algorithm with a Gauss-Newton Hessian approximation using a set of hard-spheres problems. Submitted to *Journal of Computational Optimization and Applications.*

[15] L. Grippo, F. Lampariello and S. Lucidi [1986], A nonmonotone line search technique for Newton's method, *SIAM Journal on Numerical Analysis* 23, pp. 707-716.

[16] F. Luengo, M. Raydan, W. Glunt and T.L. Hayden [1996], Preconditioned spectral gradient method for unconstrained optimization problems, Technical Report R.T. 96-08, Computer Science Department, Universidad Central de Venezuela, Venezuela. Submitted to *SIAM Journal on Scientific Computing.*

[17] J. M. Martínez [1999], BOX-QUACAN and the implementation of Augmented Lagrangian algorithms for minimization with inequality constraints, to appear in *Computational and Applied Mathematics.*

[18] F. H. Murphy [1974], A class of exponential penalty functions, *SIAM Journal on Control* 12, pp. 679–687.

[19] M. Raydan [1993], On the Barzilai and Borwein choice of steplength for the gradient method, *IMA Journal of Numerical Analysis* 13, pp. 321-326.

[20] M. Raydan [1997], The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, *SIAM Journal on Optimization* 7, pp. 26-33.

[21] R. J. Vanderbei [1994], LOQO: An interior point code for quadratic programming, Technical Report SOR-94-15, Princeton University.

[22] R. J. Vanderbei and D. F. Shanno [1997]: An interior point algorithm for nonconvex nonlinear programming, Technical Report SOR-97-21, Statistics and Operations Research, Princeton University.

Capítulo 4

Um método secante multipoint com memória limitada para otimização com canalizações

Resumo

No presente trabalho é apresentado um novo algoritmo para problemas de minimização com canalizações de uma função diferenciável. O modo de tratar as restrições ativas é similar ao usado em alguns dos novos *solvers* quadráticos. São usadas aproximações *multipoint* secante simétricas das Hessianas. Dado que estas aproximações não forçam que as aproximações sejam definidas positivas é possível acumular, e usar eficientemente, informação de curvatura negativa. São apresentados resultados numéricos.

A limited-memory multipoint secant method for bound constrained optimization

Oleg Burdakov * José Mario Martínez † Elvio A. Pilotta ‡

March 15, 2000

Abstract

A new algorithm for box-constrained minimization of differentiable functions is introduced. The way for dealing with active constraints is similar to the one used by some of the recently introduced quadratic solvers. Memoryless multipoint symmetric secant Hessian approximations are used. Since these approximations do not force positive definiteness, it is possible to accumulate, and use efficiently, negative curvature information. Numerical experiments are given.

Keywords: box constrained minimization, active set methods, projected gradients, trust regions, multipoint symmetric secant methods.

*Computing Center, Russian Academy of Sciences, 40, Vavilov Street, 117967 Moscow GSP-1, Russia. The work of this author was done while he visited the University of Campinas under support of FAPESP (Grant 1197-11730-5).

†Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil. This author was supported by PRONEX-Optimization, FAPESP (Grant 90-3724-6), CNPq and FAEP-UNICAMP. E-Mail: martinez@ime.unicamp.br

‡Facultad de Matemática, Astronomía y Física, FaMAF-UNC, Universidad Nacional de Córdoba, Ciudad Universitaria (5000), Argentina. This author was supported by PRONEX-Optimization, FAPESP (Grant 90-3724-6), CNPq and FAEP-UNICAMP. E-Mail: pilotta@mate.uncor.edu

1 Introduction

The problem considered in this paper is

$$\text{Minimize } f(x) \text{ subject to } x \in \Omega, \tag{1}$$

where

$$\Omega = \{x \in I\!R^n \mid \ell \leq x \leq u\}, \tag{2}$$

$\ell, u \in I\!R^n$, $\ell < u$ and $f : I\!R^n \to I\!R$ has continuous first partial derivatives.

This is a very important problem in practical optimization. On one hand, many physical, engineering and industrial problems admit mathematical models of type (1). On the other hand, one of the most effective approaches for solving general constrained optimization problems, based on augmented Lagrangians, relies on effective algorithms for solving (1). See [11, 12, 29]. Finally, recent work on complementarity and variational inequalities reduce these problems to box-constrained minimization problems in an efficient way. See [1] and references therein.

All practical methods for solving (1) are iterative. Given $x^k \in \Omega$, many methods compute a quadratic model of f, whose gradient at x^k coincides with the gradient of f, and whose Hessian is an approximation of the Hessian of f. At least three different (but related) ways of dealing with this approximation have been considered in recent literature.

1. In [8] the quadratic model is used to compute a "Cauchy point" and this point is used to decide the face to which the first trial point will belong. The quadratic model is (approximately) minimized on that face, giving the first trial point. The effective new point x^{k+1} is computed using a line-search procedure. A related guessing-active-set strategy was proposed in [23]. See [32].

2. In [11, 28] the first trial point must belong to a trust-region $\|x - x^k\|_\infty \leq \delta$. Since the intersection of this trust-region with Ω is an $n-$dimensional box, the Cauchy point is used to decide the face of this box to which the first trial point must belong. If a sufficient decrease condition fails to hold the size of the trust-region is reduced, and the process is repeated.

3. In [27] the box-trust-region approach is also used but, instead of restricting the search of the trial point to a face determined by the Cauchy point, the whole box is explored using a specific algorithm for box-constrained quadratic minimization. See [2, 26, 27].

In the algorithms introduced in this paper we also use quadratic models, but the way of dealing with constraints differ from the ones described above. Roughly speaking, our proposal here is to deal with constraints in the same way the quadratic solvers [2, 16, 26] do. This means that an algorithm for unconstrained minimization on the current face is used, until a separate indicator says that this is not worthwhile anymore. In this case, the face is abandoned along a direction defined in [24, 25, 26] for convex minimization. For this direction, interesting physical interpretations were given in [16]. See also [17, 18, 19, 20, 21, 22]. Moreover, when, in the unconstrained search process within a face, the algorithm hits a bound, several new constraints are added to the trial point in order to avoid the costly process of adding one constraint per iteration.

The main motivation for this approach is to allow one to try conservative strategies for dealing with faces of the feasible region. In fact, recent numerical studies [15] showed that, for box-constrained quadratic minimization, conservative strategies are more efficient than strategies that abandon the current face with mild criteria. In the presence of dual-degenerate solutions conservative strategies tend to avoid zigzagging between the optimal face and a face with larger dimension.

In an extreme case, the algorithm will be able to explore a face until a minimizer on that face is found, or a lower-dimensional face is reached. In other words, a single parameter will cause the algorithm to be more or less persistent in the process of exploring a fixed face. In general, conservative strategies will be more conservative that those described in [8, 11, 27], whereas liberal strategies tend to agree with some of them.

In already published algorithms, different ways of computing the Hessians of the quadratic models are considered. The true Hessian of f and limited memory BFGS and SR1 quasi-Newton approximations are the best known alternatives. See [8, 11]. An interesting Gauss-Newton Hessian approximation for augmented Lagrangians has been tested in [29]. In many cases the true Hessian is very costly or difficult to compute and finite-difference computations (even using the sparse schemes of [10]) are also time-consuming. In these cases, the truncated-Newton approach, where each 'Hessian $\times$ vector' product is replaced by an incremental quotient can be used, but, since each of these products involves an additional gradient evaluation, this alternative can also be inefficient. Moreover, in this approach, information about the Hessian matrix obtained at the current iteration is not used on the next one.

On the other hand, quasi-Newton Hessian approximations (see, e.g., [14]) are able to accumulate such kind of information. These approximations involve only one gradient evaluation per iteration, but the quadratic model does not fit with the true function so well, as true-Hessian models do. In the large-scale case, most known quasi-Newton Hessian approximations, which do not preserve sparsity, cannot be completely stored and, so, limited-memory alternatives have been developed. See [8, 9].

Our limited memory approach will be based on the multipoint symmetric secant approximations of the Hessian matrix proposed in [4]. These approximations are an extension of the classical multipoint secant scheme (see [30, 33] and references therein) with the advantage that they use the symmetry of the Hessian matrix in a natural way. The idea is that the Hessian approximation should be such that the gradient of the quadratic model agrees with the gradient of f at some previous m points. However, since this objective conflicts with the symmetry, the most "fresh" information carried by the gradient values is privileged. Finally, the tendency to instability of the sequential secant approach is overcome with the approach of [5, 6, 7]. Since multipoint symmetric secant schemes generate better Hessian approximations, comparing e.g. to BFGS, they fit well with the conservative procedures for dealing with active constraints considered in this research.

The organization of this paper is as follows. The general model algorithm, which is independent of the choice of Hessian approximations, is described in Section 2, where basic global convergence theorems are also proved. In Section 3 we describe the multipoint secant scheme, which is used within the faces. In Section 4 we discuss some implementations details. The numerical results are given in Section 5. Finally, the conclusions are presented in Section 6.

2 Global convergence framework

As in [26], let us divide the feasible set Ω into disjoint open faces, as follows. For all $I \subset \{1, 2, \ldots, n, n+1, n+2, \ldots, 2n\}$, we define

$$F_I = \{x \in \Omega \mid x_i = \ell_i \text{ if } i \in I,\ x_i = u_i \text{ if } n+i \in I,\ \ell_i < x_i < u_i \ \text{ otherwise}\}.$$

Let $[F_I]$ denote the smallest linear manifold that contains F_I, and S_I denote the subspace obtained by the parallel translation of $[F_I]$. If $x \in F_I$, the orthogonal projection of $-\nabla f(x)$ on S_I will be called "negative internal gradient" and denoted $g_I(x)$.

The "negative chopped gradient" (see [16, 26]) $g_C(x)$ is defined by

$$[g_C(x)]_i = -\frac{\partial f}{\partial x_i}(x)$$

if

$$x_i = \ell_i \quad \text{and} \quad \frac{\partial f}{\partial x_i}(x) < 0$$

or

$$x_i = u_i \quad \text{and} \quad \frac{\partial f}{\partial x_i}(x) > 0.$$

In all the other cases (for example, if $\ell_i < x_i < u_i$) the $i-$th component of $g_C(x)$ is defined to be 0.

Observe that

$$g_I(x) \perp g_C(x)$$

and that, defining $g_P(x) = g_I(x) + g_C(x)$, the stationary points of (1) are the points that satisfy $g_P(x) = 0$. The vector $g_P(x)$ is called (non-continuous) negative projected gradient. In spite of the lack of continuity of $g_P(x)$ it is easy to see that $x^k \to x$ and $\|g_P(x^k)\| \to 0$ imply that $g_P(x) = 0$.

The main algorithm presented in this paper is an active set method with a special procedure for dropping constraints. It calls a Sub-algorithm for a minimization on current face. For a given $\bar{F}_I$, the Sub-algorithm generates iterates $\{x^k, x^{k+1}, x^{k+2}, \ldots\} \subset \bar{F}_I$, which are assumed to meet the following requirements:

A1. For all $k \in I\!N$, if x^{k+1} is computed by the Sub-algorithm, then $f(x^{k+1}) < f(x^k)$.
A2. If $\{x^k, x^{k+1}, x^{k+2}, \ldots\} \subset F_I$ is a set of infinitely many iterates computed by the Sub-algorithm, then $g_I(x^k) \to 0$.
A3. The Sub-algorithm terminates when it generates an iterate that belongs to the boundary $\bar{F}_I - F_I$.

The main model algorithm is the one given below. ($\|\cdot\|$ will be the Euclidean norm throughout the paper, although many times it can be replaced by an arbitrary norm on $I\!R^n$.)

Algorithm 2.1

Assume that $x^0 \in \Omega$ is an arbitrary initial point, $\eta \in (0,1)$, $0 < \sigma_{min} \leq \sigma_{max} < \infty$ and $\theta \in (0,1)$. Let F_I be the face that contains the current iterate x^k. The new iterate x^{k+1} is computed as follows.
Step 1: If $\|g_P(x^k)\| = 0$ terminate the execution of the algorithm. (x^k is stationary.) If

$$\frac{\|g_C(x^k)\|}{\|g_P(x^k)\|} \geq \eta, \tag{3}$$

compute x^{k+1} at Step 2. Otherwise, compute x^{k+1} using the Sub-algorithm.
Step 2: Choose $\sigma_k \in [\sigma_{min}, \sigma_{max}]$. Let α_{max} be the maximum value of α such that the segment $[x^k, x^k + \alpha g_C(x^k)]$ is contained in Ω. Set $\alpha \leftarrow \min\{\sigma_k, \alpha_{max}\}$. If

$$f(x^k + \alpha g_C(x^k)) \leq f(x^k) - \theta\alpha\|g_C(x^k)\|^2 \tag{4}$$

set $\alpha_k = \alpha$, $x^{k+1} = x^k + \alpha_k g_C(x^k)$ and finish the iteration. Else, choose $\alpha \leftarrow \alpha_{new} \in [0.1\alpha, 0.9\alpha]$ and repeat the test (4).

To prove global convergence, we assume in the next theorem that $\nabla f(x)$ satisfies a Lipschitz condition: There exists $L > 0$ such that

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\| \tag{5}$$

for all $x, y \in \Omega$. This implies that

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2 \tag{6}$$

for all $x, y \in \Omega$.

Theorem 2.1. *Algorithm 2.1 is well defined, and every limit point of its iterates is a stationary point for problem (1).*
Proof. Let us call

$$K_1 = \{k \in I\!N \mid \|g_C(x^k)\|/\|g_P(x^k)\| \geq \eta\}.$$

The proof that the algorithm is well defined consists on showing that, for all $k \in K_1$, after a finite number of reductions of α, the condition (4) is satisfied. In fact, by (6), we have that for all $\alpha \geq 0$,

$$f(x^k + \alpha g_C(x^k)) \leq f(x^k) - \alpha\|g_C(x^k)\|^2 + \frac{\alpha^2 L}{2}\|g_C(x^k)\|^2.$$

This implies that (4) holds for $\alpha \leq \frac{2(1+\theta)}{L}$. Therefore, the new iteration is well defined.

Moreover, the α accepted at Step 2 of Algorithm 2.1 is bounded below by the positive

$$\bar{\alpha} = \min\{\sigma_{min}, \frac{2(1+\theta)}{10L}\}.$$

Therefore, at Step 2, we have

$$f(x^k) - f(x^{k+1}) \geq \theta\bar{\alpha}\eta\|g_P(x^k)\|^2. \tag{7}$$

Since $f(x^{k+1}) \leq f(x^k)$ for all $k \in I\!N$, (7) implies that either K_1 is finite or

$$\sum_{k \in K_1} \|g_P(x^k)\|^2 < \infty. \tag{8}$$

In the infinite case, (8) implies that $g_P(x^k) \to 0$ for $k \in K_1$. Therefore, every limit point of $\{x^k, k \in K_1\}$ is stationary.

If K_1 is finite, there exists $k_0 \in I\!N$ and a face F_I such that $x^k \in F_I$ for all $k \geq k_0$. Therefore, x^{k+1} is computed by the Sub-algorithm for all $k \geq k_0$. So, by Axiom A2, $\lim_{k\to\infty} \|g_I(x^k)\| = 0$. But, for all $k \geq k_0$, the inequality (3) does not

hold. Therefore, $\lim_{k\to\infty} \|g_P(x^k)\| = 0$. As before, this means that every limit point of $\{x^k\}$ is stationary. QED

As we saw before, the stationary points of our problem are characterized by $g_P(x) = 0$. If x is a stationary point such that $x_i = \ell_i$ (or $x_i = u_i$) and $\frac{\partial f}{\partial x_i} = 0$ we say that x is *degenerate*. In the following theorem we prove that, if degenerate points do not exist, the algorithm identifies the active constraints at the cluster points in a finite number of iterations.

Theorem 2.2. *Assume that all the stationary points of (1) are nondegenerate. Then, there exists $I \subset \{1, 2, \ldots, 2n\}$ such that all the limit points of the sequence generated by Algorithm 2.1 are stationary and belong to F_I. Moreover, there exists $k_0 \in I\!N$ such that $x^k \in F_I$ for all $k \geq k_0$*

Proof. Let us prove first that Step 2 cannot be executed infinitely many times. We proceed by contradiction. If the iterate x^{k+1} is computed at Step 2 infinitely many times, there exists a constraint that is abandoned infinitely many times. Without loss of generality, assume that this constraint is $x_i = \ell_i$. So, there exists an infinite set $K_1 \subset I\!N$ such that

$$x_i^k = \ell_i, \quad x_i^{k+1} > \ell_i, \tag{9}$$

$$\frac{\partial f}{\partial x_i}(x^k) < 0, \tag{10}$$

and x^{k+1} is computed at Step 2 for all $k \in K_1$. Let x^* be a limit point of $\{x^k, k \in K_1\}$. By the proof of Theorem 2.1, x^* is stationary and, by (9) and (10), $x_i^* = \ell_i$ and $\frac{\partial f}{\partial x_i}(x^*) \leq 0$. Now, since x^* is stationary, $\frac{\partial f}{\partial x_i}(x^*) \geq 0$. So, x^* is degenerate, which contradicts the hypothesis. Therefore, we proved that there exists $k_0 \in I\!N$ and $I \subset \{1, 2, \ldots, 2n\}$ such that $x^k \in F_I$ for all $k \geq k_0$. This implies that, for all $k \geq k_0$, x^{k+1} is computed by the internal unconstrained algorithm. So, by Axiom A2, $g_I(x^k) \to 0$. Since (3) does not hold for all $k \geq k_0$ this implies that $g_P(x^k) \to 0$ and, all the limit points are stationary. Suppose now that x^* is a limit point. By continuity, since $g_I(x^k) \to 0$, we have that

$$\frac{\partial f}{\partial x_i}(x^*) = 0$$

for all i such that $i \notin I$ and $n + i \notin I$. Since x^* is nondegenerate, this implies that $\ell_i < x_i^* < u_i$. Therefore, $x^* \in F_I$, as we wanted to prove. QED.

3 The limited-memory multipoint secant method

The multipoint secant method is going to be used as Sub-algorithm at Step 1 of Algorithm 2.1. Given $x^k \in F_I$ (not satisfying (3)), a symmetric Hessian approximation $B_k \in I\!R^{n\times n}$ (which is going to be a low-rank modification of a multiple of the Identity matrix) and a trust region radius δ_k, Algorithm 3.1 shows how $x^{k+1} \in \bar{F}_I$ is obtained. The method is interrupted (and restarted in a new face) if $x^{k+1} \in \bar{F}_I - F_I$. To simplify the notation, suppose that the face F_I is the interior of Ω. The extension to a general F_I is straightforward.

Algorithm 3.1
Step 1. Consider the problem

$$\text{Minimize } Q(z) \equiv \frac{1}{2} z^T B_k z + \langle \nabla f(x^k), z \rangle. \tag{11}$$

Starting with $z^0 = 0$, apply the conjugate gradient algorithm to (11) until it generates z^j, for which one of the following criteria is satisfied:

1. z^j violates the constraints given by

$$\|z^j\|_\infty \leq \delta_k, \;\; \ell \leq x^k + z^j \leq u. \tag{12}$$

2. The conjugate gradient method generates a direction along which $Q(z)$ tends to $-\infty$.

3. The gradient of the quadratic $Q(z)$ is null at z^j.

In the first case, compare the value of $Q(z)$ at the projection of z^j on the region (12) with the value of $Q(z)$ at the further point from z^j that satisfies (12) on the segment $[z^{j-1}, z^j]$. Define z_{trial} as the argument of the minimum between these two values.

In the second case, assuming that z^{j-1} is the last computed vector by the conjugate gradient algorithm and that d^{j-1} is the generated direction, proceed as in the previous case replacing z^j by $z^{j-1} + Md^{j-1}$, where M is a large positive number.

In the third case, define $z_{trial} = z^j$.

Step 2. Define

$$x_{trial} = x^k + z_{trial},$$

$$Pred = -Q(z_{trial}), \;\; Ared = f(x^k) - f(x^k + z_{trial}).$$

If $Ared < 0.1Pred$, replace $\delta_k \leftarrow 0.5\|z_{trial}\|_\infty$ and go to Step 1.
If $0.1Pred \leq Ared \leq Pred$ define $x^{k+1} = x_{trial}$, $\delta_{k+1} = \delta_k$.
If $Ared > Pred$ define $x^{k+1} = x_{trial}$, $\delta_{k+1} = 3\delta_k$.

Many general theories can be invoked to prove that, under boundedness assumptions on $\|B_k\|$, Algorithm 3.1 satisfies the axioms A1–A3. In particular, see [11]. The main goal of this research is to show that, if we generate the matrices B_k using secant multipoint formulae, the combination of Algorithms 2.1 and 3.1 is quite efficient.

Let us now describe the multipoint secant generation of the matrices B_k.

Denote $s^k = x^{k+1} - x^k$, $y^k = g^{k+1} - g^k$. Suppose that the vectors $s^0, \ldots, s^{n-1}$ have been generated somehow, and that they are linearly independent. The ideal aim would be to construct a Hessian approximation $B^n \in \mathbb{R}^{n\times n}$ such that

$$(B^n)^T = B^n, \tag{13}$$

$$B^n s^i = y^i, \quad i = 0, \ldots, n-1. \tag{14}$$

However, in general, this is impossible, because the system of $n(n-1)/2 + n^2$ equations (13) and (14) in n^2 unknown elements of B^n is overdetermined. The information about the symmetry of the Hessian matrix conflicts here with the information carried by the pairs $\{s^i, y^i\}$. The idea of the sequential symmetric secant methods introduced in [4] is to release in a natural way relations (14) in order to have B^n well defined. It can be done non-uniquely. The uniqueness can be achieved by ranging the pairs $\{s^i, y^i\}$ in accordance with the reliability of the information that they carry. For example, for $i > j$, one can consider $\{s^i, y^i\}$ as more reliable for the Hessian approximation than $\{s^j, y^j\}$, because the first pair was computed at a more recent iteration, and therefore it carries more fresh information. Then, in the process of constructing the Hessian approximation B^n, it is natural to use the pairs $\{s^i, y^i\}$ sequentially for $i = n-1, n-2, \ldots, 0$. For any current $\{s^i, y^i\}$, we can ignore the part of its information, which conflicts with the more fresh information $\{s^j, y^j\}$, $j = n-1, \ldots, i-1$, that have already been used.

To clarify this idea, suppose for the moment, that the vectors s^{n-i}, $i = 1, \ldots, n$, are parallel to the coordinate axes e^i. Then the first column and the first row of the Hessian matrix can be approximated by the standard finite-difference formula as $y^{n-1}/\|s^{n-1}\|$. The second column and row, in their parts outside the first column and row, are approximated by $y^{n-2}/\|s^{n-2}\|$, and so on. To fill in the nonfilled part of the ith row and column, the components $y_j^{n-i}/\|s^{n-i}\|$, $j = i, \ldots, n$, are used (see Fig. 1).

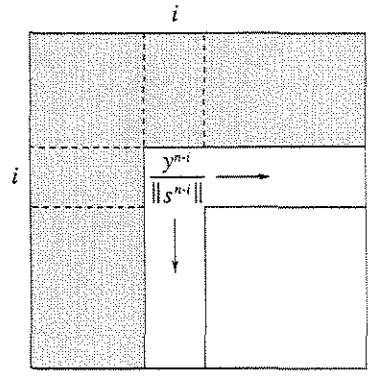


Figure 1: *Symmetric secant approximation of the Hessian matrix.*

In the general case of arbitrary vectors s^{n-i}, the space can be linearly transformed so that, in the new space, the vectors $\tilde{s}^{n-i}$ are parallel to the new coordinate axes $\tilde{e}^i$. Then the described approach can be used to approximate the Hessian matrix in the new space. After returning back to the original space, we get the approximation

$$B^n = S^{-T}\text{sym}(S^T Y)S^{-1}, \tag{15}$$

where $S = [s^0, \ldots, s^{n-1}]$, $Y = [y^0, \ldots, y^{n-1}] \in I\!R^{n\times n}$, and for any matrix A, the symmetrization operation is defined as

$$(\text{sym}A)_{ij} = \begin{cases} A_{ij}, & i \geq j, \\ A_{ji}, & \text{otherwise.} \end{cases}$$

Note that $B^n = f''$, if $f(x)$ is quadratic. If not, the multipoint secant approximation (15) gives a good approximation to $f''(x^n)$, provided that the matrix S is "safely" nonsingular (see [4]).

Let us compare the approximation (15) with the one $B^n = YS^{-1}$ given by the classic multipoint secant method [33]. In the new subspace, where $\tilde{s}^{k-i} \parallel \tilde{e}^i$, $i = 1, \ldots, n$, it is easy to see for each element of approximation, how "fresh" is the information involved in its computation. Comparing these two approximations, say, row by row (see Fig. 2), one can see that the symmetric one uses more "fresh" information comparing to the classic one, which uses in each row all spectrum of information, from the most "fresh" to the "oldest". Such comparison of (15) with the symmetric versions of the secant method proposed in [35] leads to the same conclusions. This is the reason why in the symmetric case ($f''^T = f''$), (15) generates better approximations, and why this approach is applied here in the limited-memory framework. An important property of (15) is that B^n can be obtained from any initial $B^0 \in I\!R^{n\times n}$ as the result of n sequential updatings by the rank-two formula

$$B^{k+1} = B^k + \frac{(y^k - B^k s^k)(c^k)^T + c^k(y^k - B^k s^k)^T}{\langle s^k, c^k\rangle} - \frac{\langle y^k - B^k s^k, s^k\rangle c^k (c^k)^T}{\langle s^k, c^k\rangle^2}, \tag{16}$$

where c^k is any vector in $I\!R^n$, such that

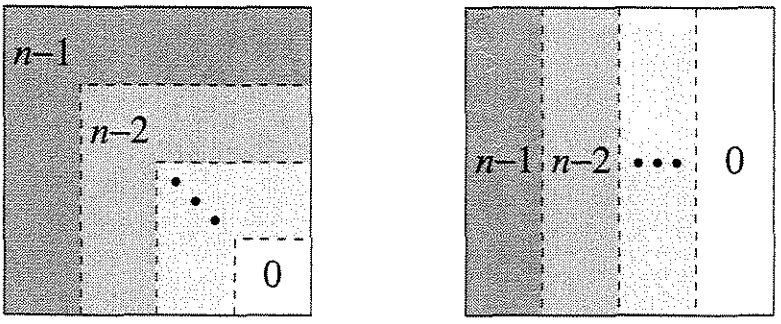


Figure 2: *The symmetric (left) and the classic (right) secant approximations with indication, for each element, the iteration at which its pair* $\{s, y\}$ *was computed.*

$$\langle c^k, s^i \rangle = 0, \quad 0 \leq i < k, \tag{17}$$

$$\langle c^k, s^k \rangle \neq 0. \tag{18}$$

The sequence $\{B^k\}_0^n$ is well defined by (16)-(18) in the sense that there is no break-down for all $k = 0, \ldots, n-1$. Though we assume from now on that B^0 is symmetric, some of the further assertions don't require this assumption.

It can be easily shown by analogy with [7] that formulas (16)-(18) generate symmetric Hessian approximations that satisfy for all $k = 0, \ldots, n-1$ the following equations

$$(S^k)^T B^{k+1} S^k = \mathrm{sym}((S^k)^T Y^k), \tag{19}$$

$$(s^k)^T B^{k+1} S^k = s^T Y^k, \quad \forall s \perp S^k, \tag{20}$$

$$(S^k)^T B^{k+1} s = (Y^k)^T s, \quad \forall s \perp S^k, \tag{21}$$

where $S^k = [s^0, \ldots, s^k]$, $Y^k = [y^0, \ldots, y^k] \in \mathbb{R}^{n \times (k+1)}$. These equations imply

$$B^{k+1} s^k = y^k. \tag{22}$$

Note that the vector c^k is not uniquely defined by (17) and (18). The uniqueness can be obtained, if we assume that B^{k+1} is the solution to the following problem

$$\text{Minimize} \qquad \|B - B^k\|_F, \tag{23}$$

$$\begin{aligned} \text{subject to:} \quad & (S^k)^T B S^k = \text{sym}((S^k)^T Y^k), \\ & (s^k)^T B S^k = s^T Y^k, \quad \forall s \perp S^k, \\ & (S^k)^T B s = (Y^k)^T s, \quad \forall s \perp S^k, \end{aligned}$$

where $\|\cdot\|_F$ is the Frobenius matrix norm, and B^k is supposed to satisfy equations similar to (19) and (20). The solution to this problem is unique, and it is given by formula (16) with

$$c^k = [I - S^{k-1}((S^{k-1})^T S^{k-1})^{-1}(S^{k-1})^T]s^k.$$

This means that the sequence $\{c^k\}_0^{n-1}$ results from the Gram-Schmidt orthogonalization process applied to the sequence $\{s^k\}_0^{n-1}$. Denoting

$$C^k = \left[\frac{c_0}{\|c_0\|}, \ldots, \frac{c_k}{\|c_k\|}\right] \in I\!R^{n\times(k+1)},$$

we see that $(C^k)^T C^k = I$ and

$$c^k = [I - C^{k-1}(C^{k-1})^T]s^k. \tag{24}$$

This choice of c^k assures that the equation

$$s^T B^k s = s^T B^0 s, \quad \forall s \perp S^{k-1} \tag{25}$$

holds for all $k = 1, \ldots, n$. Note that the sequence of approximations B^k is uniquely defined by (19)-(21) and (25). Our limited-memory approach will be essentially based on this property.

In the limited-memory methods, the Hessian matrix is approximated by a low-rank modification of a simple matrix B^0. In the next theorem, we present the multipoint symmetric secant approximations in the form that will be useful for implementation in the framework of the limited-memory approach. For the simplicity, the upper indices of S^k and Y^k will be omited.

Theorem 3.1. *Let $S = [s^0, \ldots, s^k] \in I\!R^{n\times(k+1)}$ be a full-rank matrix. Suppose that the matrices $B^1, \ldots, B^{k+1}$ are generated by formulas (16) and (24). Then for any $B^0 \in I\!R^{n\times n}$,*

$$\begin{aligned} B^{k+1} &= (I - S(S^TS)^{-1}S^T)B^0(I - S(S^TS)^{-1}S^T) \\ &+ \begin{bmatrix} S & Y \end{bmatrix} \begin{bmatrix} -(S^TS)^{-1}\text{sym}(Y^TS)(S^TS)^{-1} & (S^TS)^{-1} \\ (S^TS)^{-1} & 0 \end{bmatrix} \begin{bmatrix} S^T \\ Y^T \end{bmatrix}, \end{aligned} \tag{26}$$

where $Y = [y^0, \ldots, y^k] \in I\!R^{n\times(k+1)}$.

Proof. Let $S_\perp \in \mathbb{R}^{n\times(n-k-1)}$ be any matrix such that

$$S_\perp^T S_\perp = I \text{ and } S_\perp^T S = 0.$$

Then equations (19)-(21) and (25) can be written as

$$\begin{bmatrix} S^T \\ S_\perp^T \end{bmatrix} B^{k+1} \begin{bmatrix} S & S_\perp \end{bmatrix} = \begin{bmatrix} \mathrm{sym}(S^T Y) & Y^T S_\perp \\ S_\perp^T Y & S_\perp^T B^0 S_\perp \end{bmatrix}. \tag{27}$$

By the assumption, the matrix $[S\ S_\perp] \in \mathbb{R}^{n\times n}$ is nonsingular. Then

$$\begin{bmatrix} S & S_\perp \end{bmatrix}^{-1} = \begin{bmatrix} (S^T S)^{-1} S \\ S_\perp^T \end{bmatrix}.$$

Therefore, with the use of the evident relations

$$S_\perp S_\perp^T = I - S(S^T S)^{-1} S^T,$$

$$\mathrm{sym}(Y^T S) - S^T Y - Y^T S = -\mathrm{sym}(S^T Y),$$

formula (26) can be easily derived from (27). QED

In the limited-memory methods, the initial Hessian approximation is usually chosen as $B^0 = \gamma I$, where the positive scalar γ may depend on k. For this important special case, formula (26) can be written as

$$\begin{aligned} B^{k+1} &= \gamma I \\ &+ \begin{bmatrix} S & Y \end{bmatrix} \begin{bmatrix} -W\mathrm{sym}(Y^T S)W - \gamma W & W \\ W & 0 \end{bmatrix} \begin{bmatrix} S^T \\ Y^T \end{bmatrix}, \end{aligned} \tag{28}$$

where $W = (S^T S)^{-1} \in \mathbb{R}^{(k+1)\times(k+1)}$. The middle matrix is of the size $2(k+1) \times 2(k+1)$. For $k \ll n$, this means that the matrix B^{k+1} is a low-rank correction of γI. It is the most essential property of the limited-memory methods.

4 Some implementation features

This section deals with some specific implementation choices. We present the results of computer experiments in the next section.

Update of matrix S at the k-th algorithm. After computing the new point x_{k+1} we have to update the matrix S of the multipoint secant method. So, given the matrix $S_{k-1} \in \mathbb{R}^{n\times p}$ and the current search direction we have to decide whether

we define the matrix S_k simply adding the vector s_k to S_{k-1} as new column or it will be necessary re-construct the matrix S_k in another suitable way.

Suppose that $S_{k-1} = Q_{k-1}R_{k-1}$, where $Q_{k-1} \in \mathbb{R}^{n\times p}$ is an orthogonal matrix and $R_{k-1} \in \mathbb{R}^{p\times p}$ is an upper triangular matrix. The matrix S_{k-1} is composed of some of the previous search directions s_i, although the columns of S_{k-1} are not supposed to be ordered by the iteration number i. Let m_1 be the maximal number of columns related to the limited memory size. Let m_2 be the parameter that does not allow to have in S_k vectors s_i with $i < k - m_2$ (we called such vectors *too old*). Now, the S_{k-1} matrix is updated carefully using the following algorithm:

Algorithm 4.1: Update the matrix S_{k-1} at k-iteration

For simplicity, we denote $s_c = s_k, S_c = S_{k-1}, Q_c = Q_{k-1}, R_c = R_{k-1}, S = S_k, Q = Q_k$ and $R = R_k$.

Step 1: if S_c has a too old vector at the last column then set $p = p - 1$. That is, we exclude the last column in S_c and Q_c, and we exclude the last column and the last row in R_c.

Step 2: If $p < m_1$ and there is not too old vector in S_c then set

$$r = Q_c^T s_c, \quad q = s_c - Q_c r, \quad \rho = \|q\|, \quad q = q/\rho \quad \text{and } r_* = s_c^T q.$$

If $\rho > \omega\|s_c\|$ then

$$\text{set } p = p + 1, \quad S = [S_c\ s_c], \quad Q = [Q_c\ q] \text{ and } R = \begin{bmatrix} R_c & r \\ 0 & r_* \end{bmatrix} \text{ STOP.}$$

Step 3: set $p = 1, S = [s_c], Q = [s_c/\|s_c\|]$ and $R = [\|s_c\|]$.

Step 4: check one by one the vectors s_i that compose the columns of S_c, in decreasing order of i, do:

While $i \geq k - m_2$ and $p \neq m_1$, then set

$$r = Q_c^T s_i, \quad q = s_i - Qr, \quad \rho = \|q\|, \quad q = q/\rho \quad \text{and } r_* = s_i^T q.$$

If $\rho > \omega\|s_i\|$ then

$$\text{set } p = p + 1, \quad S = [S_c\ s_i], \quad Q = [Q\ q] \text{ and } R = \begin{bmatrix} R & r \\ 0 & r_* \end{bmatrix}.$$

Computation of α at Step 3 in Algorithm 2.1. Recall that in Algorithm 2.1 we define the step length α in the chopped direction g_C as the minimum between σ_k and α_{max}, where $\sigma_k \in [\sigma_{min}, \sigma_{max}]$. To take into account the approximated second order information at this step we adopted the spectral choice (see [34], [31]):

$$\sigma_k = \begin{cases} \max\left(\sigma_{\min}, \min\left(\sigma_{\max}, \frac{s_k^T y_k}{s_k^T B_k s_k}\right)\right) & \text{if } s_k^T y_k > 0 \\ \sigma_{\min} & \text{else .} \end{cases}$$

where B_k gives the second order information (see (11)).

Initial Hessian approximation. At the beginning of the iterations in each face, the initial Hessian approximation is chosen as $B^0 = \gamma I$, where we set the constant γ equal to the absolute value of the objective function at the first approximation in that face. (See [14]). If γ is less than a tolerance ε we set $\gamma = 1.0$, that is, we choose $B^0 = I$.

Stopping criterion at Step 1 in Algorithm 3.1. The conjugate gradient is applied until the new approximation violates the constraints or the algorithm generates a direction with negative curvature or the gradient of the quadratic is null at this approximation. The last one option means that we stopped if the the 2-norm of the quadratic approximation at the new approximation was less than or equal to 0.1 times the 2-norm of the quadratic approximation at the initial approximation, that is, zero.

5 Numerical experiments

Algorithm 2.1, with the Sub-algorithm 3.1 and the multipoint secant generation of the matrices B_k define an implementable algorithm for box-constrained minimization of differentiable functions. A particular implementation also depends of some parameters. We adjusted them using a small set of test problems. The default parameters of our implementation are:

- $\varepsilon = 10^{-5}$, tolerance for 2-norm of projected gradient g_P.
- $\sigma_{min} = 10^{-3}$, $\sigma_{max} = 10^3$, the safeguards interval for the spectral parameter σ.
- $m_1 = 5$, maximal number of columns related to the limited memory size.
- $m_2 = 5$, integer that does not allow to have too old vectors in the S matrix.
- $\delta = 0.5$, the trust region radius for Algorithm 3.1.
- $\omega = 0.1$, tolerance used in Algorithm 4.1.
- $\eta = 0.9$, tolerance used in Algorithm 2.1 to quit the face or not.
- $\theta = 10^{-4}$, parameter used for the line search in Algorithm 2.1.

The resulting code, named BSS, was compared against the well known package LANCELOT [11, 13], using a set of 20 bound constrained problems from the CUTE [3] collection. In Table 1 we reported the name of the problem, the type of objective function and the number of variables.

All the experiments were run on a SUN Sparc Ultra1. LANCELOT and BSS are formulated in Fortran 77 and were compiled with f77 compiler. In both of cases we used the optimization compiler option "-O".

We used the following default options for LANCELOT:

- `exact-second-derivatives-used`
- `bandsolver-preconditioned-cg-solver-used 5`
- `exact-Cauchy-point-required`
- `solve-bqp-accurately`
- `gradient-tolerance 1.0D-06`
- `constraints-tolerance 1.0D-06`

Problem	Objective function	n
BQPGABIM	quadratic	50
CHEBYQAD	sum of squares	50
DECONVB	sum of squares	61
HARKERP2	quadratic	100
HS110	sum of squares	100
S368	other	100
EXPLIN	other	500
EXPLIN2	other	500
EXPQUAD	other	500
QRTQUAD	other	500
BDEXP	other	10000
CVXBQP1	quadratic	10000
HATFLDC	sum of squares	10000
NONSCOMP	sum of squares	10000
NCVXBQP1	quadratic	10000
NCVXBQP2	quadratic	10000
PENTDI	quadratic	10000
PROBPENL	other	10000
QUDLIN	quadratic	10000
TORSION3	quadratic	14884

Table 1. Test problems.

The numerical results are shown in Table 2. We listed the name of the problem, the number of variables, the final value of the objective function and the CPU time (in seconds) for both of algorithms BSS and LANCELOT. We compare three versions of LANCELOT. For the first one (LAN(1)) we used the
option, for the second one (LAN(2)) we used
option and, finally, in LAN(3) we used
. We get the same solution in all of the test problems using LANCELOT and BSS, except for problem S368 and BDEXP. For S368 the version LAN(2) reached the maximum number of iterations and for BDEXP the version LAN(3) exceded the maximum CPU time allowed.

Problem	n	$f(x)$	CPU Time			
			BSS	LAN(1)	LAN(2)	LAN(3)
BQPGABIM	50	-3.7903D-05	0.06	0.04	10.71	0.11
CHEBYQAD	50	5.3863D-03	306.11	4.56	79.52	4.51
DECONVB	61	8.6383D-03	1.84	0.36	0.77	0.46
HARKERP2	100	-5.0000D-01	0.80	0.94	0.95	0.95
HS110	100	-9.9800D+19	0.01	0.04	0.04	0.04
S368	100	-1.3360D+02	3.88	3.14	(*)425.79	29.87
EXPLIN	500	-1.2523D+07	0.10	0.45	0.47	0.48
EXPLIN2	500	-1.2464D+07	0.65	0.48	0.51	0.52
EXPQUAD	500	-2.6553D+08	0.83	3.07	3.40	3.12
QRTQUAD	500	-2.6553D+08	0.81	3.08	3.48	3.15
BDEXP	10000	3.9288D-03	3.76	8.27	12.82	(**)
CVXBQP1	10000	2.2502D+06	29.88	4.76	4.97	4.87
HATFLDC	10000	6.9494D-11	9.55	4.63	5.01	4.76
NONSCOMP	10000	3.0559D-14	7.84	6.63	5.48	6.27
NCVXBQP1	10000	-1.9855D+10	0.95	8.31	8.80	8.55
NCVXBQP2	10000	-1.3340D+10	9.25	11.94	12.75	11.50
PENTDI	10000	-7.5000D-01	0.56	3.05	6.40	3.17
PROBPENL	10000	1.9998D-08	0.62	27.63	51.45	51.73
QUDLIN	10000	-4.9995D+09	1.09	8.30	8.01	7.81
TORSION3	14884	-1.2138D+00	358.63	17.26	22.58	21.65

Table 2. Performance of BSS versus LANCELOT.
(*): maximum limit of iterations reached.
(**): maximum CPU time reached.

6 Conclusions

Active set methods are among the most traditional tools of constrained optimization. Their appeal come from the fact that they allow the algorithmic designer to take full advantage of previously developed unconstrained optimization techniques. As far as new ideas in unconstrained minimization continue to be introduced, the implementation of active set methods based on those ideas is a natural task.

The unconstrained optimization technique exploited in this paper is the memoryless multipoint symmetric secant scheme. As many other quasi-Newton methods this method exploits well the unconstrained structure. The fulfillment of several secant equations within a given face (or subspace) usually guarantee that Newton-like directions inside that face are produced. On the other hand, since the approximate Hessians so far generated are not necessarily positive definite, a trust-region strategy for global convergence is in order. In this paper we adopted the memoryless approach, thanks to which large problems can be solved. Moreover, a small number of low-rank corrections guarantees that the Hessian approximations possess a small number of different eigenvalues and, so, the conjugate-gradient method is efficient for dealing with the quadratic models.

The comparison with LANCELOT seems to reveal that the method so far introduced is reliable. It is interesting to observe that the new method worked very well in problems where the performance of LANCELOT was rather poor (S368, PROBPENL) whereas LANCELOT was much more efficient in others (CHEBYQAD, CVXBQP1, TORSION3). This seems to indicate that the trust-region strategy of LANCELOT and other box-constrained solvers is complementary to the active-set strategy in the sense that difficult problems for one of them are relatively easy for the other.

As we mentioned in the introduction, one of the main purposes of box-constrained solvers relies on its application as subalgorithms for more general algorithms for nonlinear programming. (See [12, 29]). This will be the object of our practical research in the near future.

References

[1] R. Andreani and J. M. Martínez [1999], Reformulation of variational inequalities on a simplex and the compactification of complementarity problems, to appear in *SIAM Journal on Optimization.*

[2] R. H. Bielschowsky, A. Friedlander, F. M. Gomes, J. M. Martínez and M. Raydan [1998], An adaptive algorithm for bound constrained quadratic minimization, *Investigación Operativa* 7, pp. 67-102.

[3] I. Bongartz, A. R. Conn, N. I. M. Gould and Ph. L. Toint [1995], CUTE: constrained and unconstrained testing environment, *ACM Transactions on Mathematical Software* 21, pp. 123–160.

[4] O. Burdakov [1983], Methods of secant type for systems of equations with symmetric Jacobian matrix, *Numerical Functional Analysis and Optimization* 6, pp. 183–195.

[5] O. Burdakov [1983], Stable versions of the secant method for solving systems of equations, *U.S.S.R. Comput. Math. and Math. Phys.* 23, pp. 1–10.

[6] O. Burdakov [1986], On superlinear convergence of some stable variants of the secant method, *ZAMM* 66, pp. 615–622.

[7] O. Burdakov [1991], Stable symmetric secant methods with restarts, *Cybernetics* 27, pp. 390-396.

[8] R. H. Byrd, P. Lu, J. Nocedal and C. Zhu [1995], A limited memory algorithm for bound constrained minimization, *SIAM Journal on Scientific Computing* 16, pp. 1190–1208.

[9] R. H. Byrd, J. Nocedal and R. B. Schnabel [1994], Representations of quasi-Newton matrices and their use in limited memory methods, *Mathematical Programming* 63, pp. 129–156.

[10] T. F. Coleman and J. J. Moré [1983], Estimation of sparse Jacobian matrices and graph coloring problems, *SIAM Journal on Numerical Analysis* 8, pp. 639-655.

[11] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1988], Global convergence of a class of trust region algorithms for optimization with simple bounds, *SIAM Journal on Numerical Analysis* 25, pp. 433 - 460.

[12] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1991], A globally convergent augmented Lagrangean algorithm for optimization with general constraints and simple bounds, *SIAM Journal on Numerical Analysis* 28, pp. 545–572.

[13] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1992], *LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A)*, Springer Series in Computational Mathematics 17, Springer Verlag.

[14] J. E. Dennis and R. B. Schnabel [1983], *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice-Hall, Englewood Cliffs, N. J.

[15] M.A. Diniz-Ehrhardt, M.A. Gomes-Ruggiero and S.A. Santos [1997], Comparing the numerical performance of two trust-region algorithms for large-scale bound-constrained minimization. *Revista Latinoamericana de Investigación Operativa*, Vol. 7, N. 1–2, pp. 23–54.

[16] Z. Dostál [1997], Box constrained quadratic programming with proportioning and projections, *SIAM Journal on Optimization* 7, pp. 871–887.

[17] Z. Dostál, A. Friedlander, S. A. Santos and J. Malík [1997], Analysis of semicoercive contact problems using symmetric BEM and augmented Lagrangians, *Engineering Analysis with Boundary Elements*, 18, Elsevier, Great Britain, pp. 195–201.

[18] Z. Dostál, A. Friedlander and S. A. Santos [1997], Solution of contact problems using subroutine BOX-QUACAN, *Investigación Operativa*, 7, N. 1–2, ALIO, Rio de Janeiro, Brazil, pp.13–22.

[19] Z. Dostál, A. Friedlander and S. A. Santos [1997], Analysis of block structures by augmented Lagrangians with adaptive precision control, *Proceedings of GEOMECHANICS'96*, ed. Z. Rakowski, A. A. Balkema, Rotterdam, pp.175–180.

[20] Z. Dostál, A. Friedlander and S. A. Santos [1998], Solution of coercive and semicoercive contact problems by FETI domain decomposition, *Contemporary Mathematics 218 - The Tenth International Conference on Domain Decomposition Methods*, Boulder (CO). J. Mandel, C. Farhat, X. Cai eds. pp. 82–93.

[21] Z. Dostál, A. Friedlander and S. A. Santos [1999], Augmented Lagrangians with adaptive precision control for quadratic programming with equality constraints, *Computational Optimization and Applications*, 14, N. 1, pp.1–17.

[22] Z. Dostál, F. A. M. Gomes and S. A. Santos [1999], Duality-based domain decomposition with natural coarse-space for variational inequalities, to appear in *Computational methods in applied Mechanics and Engineering.*

[23] F. Facchinei, J. J. Júdice and J. Soares [1998], An active set Newton algorithm for large scale nonlinear programs with box constraints, *SIAM Journal on Optimization* 8, pp. 158–186.

[24] A. Friedlander and J. M. Martínez [1989], On the Numerical Solution of Bound Constrained Optimization Problems, *RAIRO Operations Research*, 23, pp. 319–341.

[25] A. Friedlander and J. M. Martínez [1992], New Algorithms for Maximization of Concave Functions with Box Constraints, *RAIRO Operations Research*, 26, pp. 209–236.

[26] A. Friedlander and J. M. Martínez [1994], On the maximization of a concave quadratic function with box constraints, *SIAM Journal on Optimization* 4, pp. 177–192.

[27] A. Friedlander, J. M. Martínez and S. A. Santos [1994], A new trust region algorithm for bound constrained minimization, *Applied Mathematics and Optimization* 30, pp. 235–266.

[28] C. J. Lin and J. J. Moré [1999], Newton's method for large bound-constrained optimization problems, *Siam Journal on Optimization 9*, pp. 1100–1127.

[29] N. Krejić, J. M. Martínez, M. P. Mello and E. A. Pilotta [1998], Validation of an augmented Lagrangian algorithm with a Gauss-Newton Hessian approximation using a set of hard-spheres problems, to appear in *Computational Optimization and Applications.*

[30] J. M. Martínez [1979], Three new Algorithms based on the sequential secant method, *BIT* 19, pp. 236–243.

[31] J. M. Martínez, E. A. Pilotta, M. Raydan [1999], Spectral gradient method for linearly constrained optimization, Technical Report 22/99, IMECC, UNICAMP, Campinas, Brazil.

[32] J. J. Moré and D. J. Thuente [1994], Line search algorithms with guaranteed sufficient decrease, *ACM Transactions on Mathematical Software*, 20, pp. 286–307.

[33] J.M. Ortega and W. Rheinboldt [1970], *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York.

[34] M. Raydan [1993], On the Barzilai and Borwein choice of steplength for the gradient method, *SIAM Journal on Optimization* 7, pp. 26–33.

[35] R. B. Schnabel [1983], Quasi-Newton methods using multiple secant equations, Technical report CU-CS-247-83, Department of Computer Science, University of Colorado, Boulder, CO.

[36] C. Zhu, R. H. Byrd, P. Lu and J. Nocedal [1997], L-BFGS-B-Fortran subroutines for large-scale bound constrained optimization, *ACM Transactions on Mathematical Software* 23, N. 4, pp. 550-560.

Conclusões Gerais

No presente trabalho estudamos o problema de minimizar uma função geral sujeita a restrições do ponto de vista da resolução numérica do problema.

A idéia dos métodos baseados no Lagrangeano Aumentado para programação não linear para problemas de grande porte ficou consolidada com os trabalhos de Conn, Gould and Toint no algoritmo LANCELOT. No novo algoritmo ALBOX apresentamos algumas diferenças com essa formulação para a resolução do subproblema quadrático. A principal diferença está relacionada em explorar uma caixa de região de confiança no lugar de usar a informação do ponto de Cauchy aproximado. Outra diferença, que mostrou ser muito eficiente computacionalmente, foi o uso de uma simplificação convexa da Hessiana verdadeira do modelo quadrático. O objetivo de usar ALBOX não é somente mostrar que ele pode resolver problemas da vida real senão também fazer comparações frente a outros códigos. Este último é de grande utilidade pois assim é possível ajustar parâmetros e testar algumas variantes e alternativas do mesmo código. O resultado desta comparação parece indicar que ALBOX pode ser usado como um algoritmo competitivo para a resolução de problemas de programação não linear.

Apresentamos um algoritmo modelo de tipo restauração inexata para um problema de minimização com restrições não lineares e variáveis em Ω, um conjunto fechado e convexo do espaço n-dimensional geral. Pelo fato de ser um algoritmo modelo admite diferentes implementações, e a eficiência do algoritmo dependerá dos métodos escolhidos para realizar cada uma das fases. Como na prática estamos interessados no caso em que Ω seja um politopo, usando normas convenientes na formulação dos problemas de cada fase, é possível usar qualquer algoritmo ou pacote disponível para problemas de minimização com restrições lineares para problemas de grande porte. Na nossa implementação usamos MINOS [67] na fase de minimização. Os resultados numéricos obtidos usando o HSP mostraram a eficiência da nossa implementação assim como da estratégia de restauração inexata.

Apresentamos um algoritmo para minimização com restrições lineares e canalizações usando um método Quase-Newton onde a matriz de atualização é obtida

usando uma escolha espectral associada a um quociente de Rayleigh e combinada com uma busca linear não monótona. Pela estrutura do problema foi possível usar técnicas de fatorações esparsas. Na nossa implementação usamos o código MA27 [39]. O uso de fatorações esparsas pode ser decissivo na performance de um algoritmo sempre que seja possível explorar a estrutura de esparsidade da matriz das restrições. Além disso, o fato de não fatorar a matriz KKT em cada iteração é uma importante vantagem do nosso algoritmo, o que pode ser observado nos resultados numéricos. Como aconteceu em outros trabalhos baseados no gradiente espectral a busca linear não monótona mostrou-se mais eficiente que a busca linear monótona. Por outro lado, a penalização exponencial modificada parece ser uma boa alternativa em relação às clássicas estratégias de penalização quando direções Newtonianas não podem ser calculadas.

No último Capítulo consideramos um método para um problema de minimização com canalizações. O método explora a estratégia de restrições ativas com um esquema *multipoint* secante simétrico com memória limitada. Dado que as aproximações Quase-Newton não são necessariamente definidas positivas usamos uma estratégia de região de confiança para a convergência global do algoritmo. A memória limitada é uma metodologia adequada para resolver problemas de grande porte. Além disso, atualizações com correções de posto pequeno asseguram que as aproximações das Hessianas têm um número pequeno de autovalores diferentes e portanto o método dos gradientes conjugados é eficiente para tratar com modelos quadráticos. Os resultados numéricos e as comparações mostraram que o método apresentado é confiável.

Finalmente, e utilizando o que aprendemos, estudamos e experimentamos podemos sugerir algumas futuras linhas de pesquisa, por exemplo:

- continuar trabalhando em métodos de tipo Lagrangeano Aumentado e testar com um conjunto maior de problemas testes o qual permitirá ajustar alguns parâmetros assim como ter uma maior compreensão do comportamento destes métodos. Em particular, testar com problemas de grande porte.

- analisar outras alternativas para resolver os problemas de minimização das fases de restauração e minimização no Algoritmo do Capítulo 2.

- considerando os bons resultados obtidos usando a penalização exponencial sería interesante estudar cuidadosamente o critério de parada das iterações internas e relação com o critério de convergência global do algoritmo completo do Capítulo 3. Por outro lado outras estratégias para tratar as restrições de canalizações poderiam ser consideradas.

- continuar fazendo experimentos numéricos com o algoritmo do Capítulo 4 e comparar numericamente com algum outro algoritmo que use memória limitada.

Referências Bibliográficas

[1] J. Abadie and J. Carpentier [1968], *Generalization of the Wolfe reduced-gradient method to the case of nonlinear constraints*, Optimization, Edited by R. Fletcher, Academic Press, New York, pp. 37–47.

[2] R. Andreani and J. M. Martínez [1999], Reformulation of variational inequalities on a simplex and the compactification of complementarity problems, to appear in *SIAM Journal on Optimization.*

[3] J. Barzilai and J. M. Borwein [1988], Two point step size gradient methods, *IMA Journal of Numerical Analysis* 8, pp. 141–148.

[4] D. P. Bertsekas [1995], *Nonlinear Programming*, Athena Scientific, Belmont, MA.

[5] R. H. Bielschowsky [1996], *Nonlinear programming algorithms with dynamic definition of near-feasibility: theory and implementations*, Tese de Doutorado, IMECC-UNICAMP, Campinas.

[6] R.H. Bielschowsky, A. Friedlander, F.M. Gomes, J.M. Martínez and M. Raydan [1997], An adaptative algorithm for bound constrained quadratic minimization. *Investigación Operativa* 7, pp. 67–102.

[7] E. G. Birgin, I. Chambouleyron and J. M. Martínez [1999], Estimation of the optical constants and the thickness of thin films using unconstrained optimization. To appear in *Journal of Computational Physics.*

[8] E. G. Birgin and Y. G. Evtushenko [1998], Automatic differentiation and spectral projected gradient methods for optimal control problems, *Optimization Methods and Software* Vol. 10, pp. 125–146.

[9] E. G. Birgin, J. M. Martínez and M. Raydan [1997], Nonmonotone Spectral Projected Gradient Methods on Convex Sets, Technical Report, Department of Applied Mathematics, University of Campinas. Submitted to *SIAM Journal on Optimization.*

[10] I. Bongartz, A. R. Conn, N. I. M. Gould and Ph. L. Toint [1995], CUTE: constrained and unconstrained testing environment, *ACM Transactions on Mathematical Software* Vol. 21, pp. 123–160.

[11] O. Burdakov [1983], Methods of secant type for systems of equations with symmetric Jacobian matrix, *Numerical Functional Analysis and Optimization* Vol. 6, pp. 183–195.

[12] O. Burdakov [1983], Stable versions of the secant method for solving systems of equations, *U.S.S.R. Comput. Math. and Math. Phys.* Vol. 23, pp. 1–10.

[13] O. Burdakov [1986], On superlinear convergence of some stable variants of the secant method, *ZAMM* Vol. 66, pp. 615–622.

[14] O. Burdakov [1991], Stable symmetric secant methods with restarts, *Cybernetics* Vol. 27, pp. 390-396.

[15] R. H. Byrd [1987], Robust trust region methods for constrained optimization, Contributed presentation, SIAM Conference on Optimization, Houston, Texas.

[16] R. H. Byrd, J. Ch. Gilbert and J. Nocedal [1996], A trust region method based on interior point techniques for nonlinear programming, Technical Report OTC 96/02, Optimization Technology Center, Argonne National Laboratory and Northwestern University, Illinois.

[17] R. H. Byrd, P. Lu, J. Nocedal and C. Zhu [1995], A limited memory algorithm for bound constrained minimization, *SIAM Journal on Scientific Computing* Vol. 16, pp. 1190–1208.

[18] R. H. Byrd, J. Nocedal and R. B. Schnabel [1994], Representations of quasi-Newton matrices and their use in limited memory methods, *Mathematical Programming* Vol. 63, pp. 129–156.

[19] M. R. Celis, J. E. Dennis and R. A. Tapia [1985], A trust-region strategy for nonlinear equality constrained optimization, Numerical Optimization, Edited by P. Boggs, R. Byrd and R. Schnabel, SIAM Publications, Philadelphia, Pennsylvania, pp. 71–82.

[20] T. F. Coleman and J. J. Moré [1983], Estimation of sparse Jacobian matrices and graph coloring problems, *SIAM Journal on Numerical Analysis* Vol. 8, pp. 639–655.

[21] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1988], Global convergence of a class of trust region algorithms for optimization with simple bounds, *SIAM Journal on Numerical Analysis* 25, pp. 433 - 460.

[22] A.R. Conn, N.I.M. Gould and Ph.L. Toint [1991], A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, Vol. 28, pp. 545–572.

[23] A.R. Conn, N.I.M. Gould and Ph.L. Toint [1992], *LANCELOT: a Fortran Package for Large-Scale Nonlinear Optimization (Release A)*. Springer-Verlag, Berlin.

[24] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1988], Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis* Vol. 25, pp. 433–460.

[25] J.H. Conway and N.J.C. Sloane [1988]. *Sphere Packings, Lattices and Groups.* Springer-Verlag, New York.

[26] R. S. Dembo, S. C. Eisenstat and T. Steihaug [1982], *Inexact Newton methods*, SIAM Journal on Numerical Analysis, Vol. 19, pp. 400-408.

[27] J. E. Dennis and M. El-Alem [1996], A trust-region algorithm for solving feasibility problems, Technical Report, Department of Applied and Computational Mathematics, Rice University, Houston, Texas.

[28] J. E. Dennis, M. El-Alem and M. C. Maciel [1997], *A global convergence theory for general trust-region-based algorithms for equality constrained optimization, SIAM Journal on Optimization* Vol. 7, pp. 177-207.

[29] J. E. Dennis and R. B. Schnabel [1983], *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice Hall, Englewood Cliffs, NJ.

[30] K. Devlin [1994], *Mathematics: The Science of Patterns.* Scientific American Library, New York.

[31] M.A. Diniz-Ehrhardt, M.A. Gomes-Ruggiero and S.A. Santos [1997], Comparing the numerical performance of two trust-region algorithms for large-scale bound-constrained minimization. *Revista Latinoamericana de Investigación Operativa*, Vol. 7, N. 1–2, pp. 23–54.

[32] Z. Dostál [1997], Box constrained quadratic programming with proportioning and projections, *SIAM Journal on Optimization* Vol. 7, pp. 871–887.

[33] Z. Dostál, A. Friedlander and S. A. Santos [1997], Solution of contact problems using subroutine BOX-QUACAN, *Investigación Operativa*, Vol. 7, N. 1–2, ALIO, Rio de Janeiro, Brazil, pp. 13–22.

[34] Z. Dostál, A. Friedlander and S. A. Santos [1997], Analysis of block structures by augmented Lagrangians with adaptive precision control, *Proceedings of GEOMECHANICS'96*, ed. Z. Rakowski, A. A. Balkema, Rotterdam, pp. 175–180.

[35] Z. Dostál, A. Friedlander and S. A. Santos [1998], Solution of coercive and semicoercive contact problems by FETI domain decomposition, *Contemporary Mathematics 218 - The Tenth International Conference on Domain Decomposition Methods*, Boulder (CO). J. Mandel, C. Farhat, X. Cai eds. pp. 82–93.

[36] Z. Dostál, A. Friedlander and S. A. Santos [1999], Augmented Lagrangians with adaptive precision control for quadratic programming with equality constraints, *Computational Optimization and Applications*, Vol. 14, N. 1, pp. 1–17.

[37] Z. Dostál, A. Friedlander, S. A. Santos and J. Malík [1997], Analysis of semicoercive contact problems using symmetric BEM and augmented Lagrangians, *Engineering Analysis with Boundary Elements*, 18, Elsevier, Great Britain, pp. 195–201.

[38] Z. Dostál, F. A. M. Gomes and S. A. Santos [1999], Duality-based domain decomposition with natural coarse-space for variational inequalities, to appear in *Computational methods in applied Mechanics and Engineering.*

[39] I. S. Duff and J. K. Reid [1982], MA27 - a set of Fortran subroutines for solving sparse symmetric sets of linear equations, Technical Report R-10533, Computer Science and System Division, AERE Harwell, Oxford, England.

[40] M. El-Alem [1995], A robust trust region algorithm with a nonmonotonic penalty parameter scheme for constrained optimization, *SIAM Journal on Optimization*, Vol. 5, pp. 348–378.

[41] F. Facchinei, J. J. Júdice and J. Soares [1998], An active set Newton algorithm for large scale nonlinear programs with box constraints, *SIAM Journal on Optimization*netsc 8, pp. 158–186.

[42] A. Friedlander and J. M. Martínez [1989], On the Numerical Solution of Bound Constrained Optimization Problems, *RAIRO Operations Research*, 23, pp. 319–341.

[43] A. Friedlander and J. M. Martínez [1992], New Algorithms for Maximization of Concave Functions with Box Constraints, *RAIRO Operations Research*, 26, pp. 209–236.

[44] A. Friedlander and J.M. Martínez [1994], On the maximization of a concave quadratic function with box constraints. *SIAM Journal on Optimization*, Vol. 4, pp. 177–192.

[45] A. Friedlander, J.M. Martínez and S.A. Santos [1994], A new trust-region algorithm for bound constrained minimization. *Applied Mathematics and Optimization*, Vol. 30, pp. 235–266, 1994.

[46] A. Friedlander, J. M. Martínez and M. Raydan [1995], A new method for large-scale box constrained convex quadratic minimization problems, *Optimization Methods and Software* 5, pp. 57-74.

[47] D. M. Gay [1983], A trust-region approach to linearly constrained optimization, Proceedings of the Dundee Biennial Conference on Numerical Analysis.

[48] P. E. Gill, W. Murray and M. H. Wright,[1981], *Practical optimization*, Academic Press, London, England.

[49] F. M. Gomes, M. C. Maciel and J.M. Martínez [1999], Nonlinear programming algorithms using trust regions and augmented Lagrangians with nonmonotone penalty parameters. *Mathematical Programming* Vol. 84, pp. 161–200.

[50] N. I. M. Gould [1991], An algorithm for large-scale quadratic programming. *IMA Journal of Numerical Analysis* Vol. 11, pp. 299–324.

[51] L. Grippo, F. Lampariello and S. Lucidi [1986], A nonmonotone line search technique for Newton's method, *SIAM Journal on Numerical Analysis* Vol. 23, pp. 707–716.

[52] W. W. Hager [1993], Analysis and implementation of a dual algorithm for constraint optimization. *Journal of Optimization Theory and Applications* Vol. 79, pp. 427–462.

[53] W. W. Hager [1987], Dual techniques for constraint optimization. *Journal of Optimization Theory and Applications* Vol. 55, pp. 37–71.

[54] M. Lalee, J. Nocedal and T. Plantenga [1993], On the implementation of an algorithm for large-scale equality constrained optimization. Technical Report N.A.M. 08, E.E.C.S. Dept., Northwestern University.

[55] L. S. Lasdon [1982], Reduced gradient methods, *Nonlinear Optimization 1981*, edited by M. Powell, Academic Press, New York, pp. 235–242.

[56] C. J. Lin and J. J. Moré [1999], Newton's method for large bound-constrained optimization problems, *Siam Journal on Optimization 9*, pp. 1100–1127.

[57] F. Luengo, M. Raydan, W. Glunt and T.L. Hayden [1996], Preconditioned spectral gradient method for unconstrained optimization problems, Technical Report R.T. 96-08, Computer Science Department, Universidad Central de Venezuela, Venezuela. Submitted to *SIAM Journal on Scientific Computing.*

[58] N. Maculan, P. Michelon and J. MacGregor Smith [1996], Bounds on the kissing numbers in $\mathbb{R}^n$: mathematical programming formulations. Technical report, COPPE, University of Rio de Janeiro, Brazil.

[59] J. M. Martínez [1979], Three new Algorithms based on the sequential secant method, *BIT* 19, pp. 236–243.

[60] J.M. Martínez [1997], Augmented Lagrangians and the resolution of packing problems. Technical Report 08/97, Institute of Mathematics, University of Campinas, Brazil.

[61] J.M. Martínez [1998], A Two-Phase Model Algorithm with global convergence for Nonlinear Programming. to appear in *Journal of Optimization Theory and Applications*, Vol. 96.

J. M. Martínez [1999], BOX-QUACAN and the implementation of Augmented Lagrangian algorithms for minimization with inequality constraints, to appear in *Computational and Applied Mathematics.*

[62] J. M. Martínez and S. A. Santos [1995], A trust region strategy for minimization on arbitrary domains, *Mathematical Programming*, Vol. 68, pp. 267–302.

[63] A. Miele, H. Y. Huang and J. C. Heideman, [1969],Sequential gradient-restoration algorithm for the minimization of constrained functions, ordinary and conjugate gradient version, *Journal of Optimization Theory and Applications*, Vol. 4, pp. 213–246.

[64] A. Miele, A. V. Levy and E. E. Cragg [1971], Modifications and extensions of the conjugate-gradient restoration algorithm for mathematical programming problems, *Journal of Optimization Theory and Applications*, Vol. 7, pp. 450–472.

[65] A. Miele, E. M. Sims and V. K. Basapur [1983], Sequential Gradient-Restoration algorithm for mathematical programming problem with inequality constraints, Part 1, Theory, Rice University, Aero-Astronautics Report N. 168.

[66] J. J. Moré and D. J. Thuente [1994], Line search algorithms with guaranteed sufficient decrease, *ACM Transactions on Mathematical Software*, 20, pp. 286–307.

[67] R. B. Murtagh and M. A. Saunders [1978], Large-scale linearly constrained optimization, *Mathematical Programming*, Vol. 14, pp. 41–72.

[68] B. A. Murtagh and M. A. Saunders [1995], MINOS 5.4 User's guide, Technical Report SOL 83-20 R, Stanford University.

[69] F. H. Murphy [1974], A class of exponential penalty functions, *SIAM Journal on Control* Vol. 12, pp. 679–687.

[70] S.G. Nash [1985], Preconditioning of truncated-newton methods. *SIAM Journal on Scientific and Statistical Computing*, Vol. 6, pp. 599–616.

[71] E. Omojokun [1989], Trust-region strategies for optimization with nonlinear equality and inequality constraints, PhD Thesis, Department of Computer Science, University of Colorado, Boulder, Colorado.

[72] J.M. Ortega and W. Rheinboldt [1970], *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York.

[73] M. J. D. Powell and Y. Yuan [1991], A trust-region algorithm for equality constrained optimization, *Mathematical Programming*, Vol. 49, pp. 190–211.

[74] M. Raydan [1993], On the Barzilai and Borwein choice of steplength for the gradient method, *IMA Journal of Numerical Analysis* Vol. 13, pp. 321–326.

[75] M. Raydan [1997], The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, *SIAM Journal on Optimization* Vol. 7, pp. 26–33.

[76] M. Rom and M. Avriel [1989], Properties of the sequential gradient-restoration algorithm (SGRA), Part 1: Introduction and comparison with related methods, *Journal of Optimization Theory and Applications*, Vol. 62, pp. 77–98.

[77] M. Rom and M. Avriel [1989], Properties of the sequential gradient-restoration algorithm (SGRA), Part 2: Convergence Analysis, *Journal of Optimization Theory and Applications*, Vol. 62, pp. 99–126.

[78] J. B. Rosen [1960], The gradient projection method for nonlinear programming, Part 1, Linear constraints, *SIAM Journal on Applied Mathematics*, Vol. 8, pp. 181–217.

[79] J. B. Rosen [1961], The gradient projection method for nonlinear programming, Part 2, Nonlinear constraints, *SIAM Journal on Applied Mathematics*, Vol. 9, pp. 514–532.

[80] J. B. Rosen [1978], Two-phase algorithm for nonlinear constraint problems, *Nonlinear Programming 3*, Edited by O. Mangasarian, R. Meyer and S. Robinson, Academic Press, London and New York, pp. 97–124.

[81] E.B. Saff and A.B.J. Kuijlaars [1997], Distributing many points on a sphere. *Mathematical Intelligencer*, Vol. 19, pp. 5–11.

[82] R. B. Schnabel [1983], Quasi-Newton methods using multiple secant equations, Technical report CU-CS-247-83, Department of Computer Science, University of Colorado, Boulder, CO.

[83] R. J. Vanderbei [1994], LOQO: An interior point code for quadratic programming, Technical Report SOR-94-15, Princeton University.

[84] R. J. Vanderbei and D. F. Shanno [1997], An interior point algorithm for nonconvex nonlinear programming, Technical Report SOR-97-21, Statistics and Operations Research, Princeton University.

[85] M. H. Wright, *Interior point methods for constrained optimization*, Acta Numerica 1992, pp. 341–407,1992.

[86] C. Zhu, R. Byrd, P. Lu and J. Nocedal [1997], L-BFGS-B-Fortran subroutines for large-scale bound constrained optimization, *ACM Transactions on Mathematical Software* Vol. 23, N. 4, pp. 550-560.

Apêndice

Neste apêndice incluimos a formulação do Hard Sphere Problem (HSP) em formato SIF (Standard Input Format) usado para comparar os algoritmos dos Capítulos 1 e 2 com LANCELOT. O SIF é a linguagem formal para formular problemas de programação não linear e ingressar dados os dados do problema que requer LANCELOT.

Atualmente existem mais de mil problemas escritos neste formato no sistema CUTE (Constrained and Unconstrained Testing Environment), dos mesmos autores do LANCELOT, e que pode ser obtida via internet.

A seguir apresentamos a nossa formulação do HSP, chamada `KISSING.SIF`, que foi incluida no CUTE em Outubro de 1998.

```
**************************
* SET UP THE INITIAL DATA *
**************************

NAME          KISSING

*   Problem: KISSING NUMBER PROBLEM
*
*   Source: This problem is associated to the family of Hard-Spheres
*   problem. It belongs to the family of sphere packing problems, a
*   class of challenging problems dating from the beginning of the
*   17th century which is related to practical problems in Chemistry,
*   Biology and Physics. It consists on maximizing the minimum pairwise
*   distance between NP points on a sphere in \R^{MDIM}.
*   This problem may be reduced to a nonconvex nonlinear optimization
*   problem with a potentially large number of (nonoptimal) points
*   satisfying optimality conditions. We have, thus, a class of problems
*   indexed by the parameters MDIM and NP, that provides a suitable
*   set of test problems for evaluating nonlinear programming codes.
*   After some algebric manipulations, we can formulate this problem as
*
*                           Minimize z
*
*                           subject to
*
*     z \geq <x_i, x_j> for all different pair of indices i, j
*
*                           ||x_i||^2 = 1    for all i = 1,...,NP
*
*     The goal is to find an objective value less than 0.5 (This means
*     that the NP points stored belong to the sphere and every distance
*     between two of them is greater than 1.0).
*
*     Obs: the starting point is aleatorally chosen although each
*     variable belongs to [-1.,1.].
*
*
*     References:
*     [1] "Validation of an Augmented Lagrangian algorithm with
*          a Gauss-Newton Hessian approximation using a set of
*          Hard-Spheres problems", N. Krejic, J. M. Martinez,
```

```
*           M. Mello and E. A. Pilotta, Tech. Report RP 29/98,
*           IMECC-UNICAMP, Campinas, 1998.
*      [2] "Inexact-Restoration Algorithm for Constrained
*           Optimization", J. M. Martinez and E. A. Pilotta,
*           Tech. Report, IMECC-UNICAMP, Campinas, 1998.
*      [3]  "Sphere Packings, Lattices and Groups", J. H. Conway
*           and N. J. C. Sloane, Springer-Verlag, NY, 1988.
*
*
*      SIF input: September 29, 1998
*         Jose Mario Martinez
*         Elvio Angel Pilotta
*

*   classification LQR2-RN-V-V

******************************************************************

*   Number of points: NP >= 12

 IE NP                    12
*IE NP                    13
*IE NP                    14
*IE NP                    15
*IE NP                    22
*IE NP                    23
*IE NP                    24
*IE NP                    25
*IE NP                    26
*IE NP                    27
*IE NP                    37
*IE NP                    38
*IE NP                    39
*IE NP                    40
*IE NP                    41
 IE NP                    42

*   Dimension: MDIM >= 3

 IE MDIM                  3
*IE MDIM                  4
```

```
*IE MDIM                  5

*   Other useful parameters.

 IA N-         NP          -1
 IE 1                      1

VARIABLES

 DO I          1                       NP
 DO J          1                       MDIM
 X  X(I,J)
 OD J
 OD I
 X  Z

GROUPS

 XN OBJ        Z           1.0

*   Inequality constraints.

 DO I          1                       N-
 IA I+         I           1
 DO J          I+                      NP
 XL IC(I,J)    Z           -1.0
 ND

*   Equality constraints.

 DO I          1                       NP
 XE EC(I)
 ND

CONSTANTS

 DO I          1                       NP
 X  KISSING    EC(I)        1.0
 ND

BOUNDS
```

```
 DO I          1                        NP
 DO J          1                        MDIM
 XR KISSING    X(I,J)
 ND

 XR KISSING    Z

START POINT

 XV KISSING    X1,1       -0.10890604
 XV KISSING    X1,2        0.85395078
 XV KISSING    X1,3       -0.45461680
 XV KISSING    X2,1        0.49883922
 XV KISSING    X2,2       -0.18439316
 XV KISSING    X2,3       -0.04798594
 XV KISSING    X3,1        0.28262888
 XV KISSING    X3,2       -0.48054070
 XV KISSING    X3,3        0.46715332
 XV KISSING    X4,1       -0.00580106
 XV KISSING    X4,2       -0.49987584
 XV KISSING    X4,3       -0.44130302
 XV KISSING    X5,1        0.81712540
 XV KISSING    X5,2       -0.36874258
 XV KISSING    X5,3       -0.68321896
 XV KISSING    X6,1        0.29642426
 XV KISSING    X6,2        0.82315508
 XV KISSING    X6,3        0.35938150
 XV KISSING    X7,1        0.09215152
 XV KISSING    X7,2       -0.53564686
 XV KISSING    X7,3        0.00191436
 XV KISSING    X8,1        0.11700318
 XV KISSING    X8,2        0.96722760
 XV KISSING    X8,3       -0.14916438
 XV KISSING    X9,1        0.01791524
 XV KISSING    X9,2        0.17759446
 XV KISSING    X9,3       -0.61875872
 XV KISSING    X10,1      -0.63833630
 XV KISSING    X10,2       0.80830972
 XV KISSING    X10,3       0.45846734
 XV KISSING    X11,1       0.28446456
```

```
 XV KISSING   X11,2      0.45686938
 XV KISSING   X11,3      0.16368980
 XV KISSING   X12,1      0.76557382
 XV KISSING   X12,2      0.16700944
 XV KISSING   X12,3     -0.31647534

ELEMENT TYPE
 EV PROD      X                      Y
 EV QUA       V

ELEMENT USES

* Inequality constraints.

 DO I         1                      N-
 IA I+        I          1
 DO J         I+                     NP
 DO K         1                      MDIM
 XT A(I,J,K)  PROD
 ZV A(I,J,K)  X                      X(I,K)
 ZV A(I,J,K)  Y                      X(J,K)
 ND

* Equality constraints.

 DO I         1                      NP
 DO K         1                      MDIM
 XT B(I,K)    QUA
 ZV B(I,K)    V                      X(I,K)
 ND

GROUP USES

* Inequality constraints.

 DO I         1                      N-
 IA I+        I          1
 DO J         I+                     NP
 DO K         1                      MDIM
 XE IC(I,J)   A(I,J,K)
 ND
```

```
* Equality constraints.

 DO I          1                          NP
 DO K          1                          MDIM
 XE EC(I)      B(I,K)
 ND

OBJECT BOUND

*   Solution
*XL SOLUTION             4.47214D-01

ENDATA

************************
* SET UP THE FUNCTION *
* AND RANGE ROUTINES  *
************************

ELEMENTS      KISSING
INDIVIDUALS

* Product of 2 elemental variables.

 T  PROD

 F                      X * Y
 G  X                   Y
 G  Y                   X
 H  X          Y        1.0

* Square of an elemental variables.

 T  QUA

 F                      V * V
 G  V                   V + V
 H  V          V        2.0

ENDATA
```