Inner solvers for interior point methods for large scale nonlinear programming *

Silvia Bonettini[1], Emanuele Galligani[1], Valeria Ruggiero[2]
[1] *Dipartimento di Matematica, Università di Modena e Reggio Emilia*
Via Campi 213/b, 41100 Modena, Italy
[2] *Dipartimento di Matematica – Sede Distaccata, Università di Ferrara*
Via Saragat 1, Blocco B, 44100 Ferrara, Italy

Technical Report n. 64, May 2005
Dipartimento di Matematica, Università di Modena e Reggio Emilia

Abstract

This paper deals with the solution of nonlinear programming problems arising from elliptic control problems by an interior point scheme.
At each step of the scheme, we have to solve a large scale symmetric and indefinite system; inner iterative solvers, with adaptive stopping rule, can be used in order to avoid unnecessary inner iterations, especially when the current outer iterate is far from the solution.
In this work, we analyze the method of multipliers and the preconditioned conjugate gradient method as inner solvers for interior point schemes. We discuss on the convergence of the whole approach, on the implementation details and we report results of a numerical experimentation on a set of large scale test problems arising from the discretization of elliptic control problems. A comparison with other interior point codes is also reported.

Keywords: Large scale nonlinear programming, interior point method, method of multipliers, preconditioned conjugate gradient method.

1 Introduction

This work is concerned with the numerical solution of large scale nonlinear programming problems with interior point method. In this work, we present

*This research was supported by the Italian Ministry for Education, University and Research (MIUR), FIRB Project RBAU01JYPN: "Parallel Nonlinear Numerical Optimization PN^2O" (http://dm.unife.it/pn2o/).
E–mail addresses: bonettini.silvia@unimo.it (S. Bonettini), galligani@unimo.it (E. Galligani), rgv@unife.it (V. Ruggiero).

effective iterative inner solvers for the solution of the perturbed system that occurs at each step of the interior point scheme. In Section 4, three iterative solvers are described.
In particular, the first one consists into applying the method of multipliers to the perturbed system, since it can be seen as the optimality conditions of a linear–quadratic programming problem; at each iteration of the iterative method of the multipliers we solve a symmetric positive definite system by the efficient library routine of Ng and Peyton implementing Cholesky factorization ([63]).
The second solver uses the conjugate gradient method to solve the perturbed system with an indefinite preconditioner introduced by Lukšan in [65]; exploiting the block structure of the preconditioning matrix, at each step of the preconditioned conjugate gradient method, we solve, by Ng and Peyton routine, a symmetric positive definite system.
The third solver also uses conjugate gradient method with Lukšan preconditioner, but the symmetric indefinite system that occurs at each step of the preconditioned conjugate gradient method, is solved by a routine implementing a Cholesky–like factorization with a regularization technique; this routine, introduced in [11] and available on the website *http://dm.unife.it/blkfclt*, is called BLKFCLT.

These iterative solvers are inserted into an interior point scheme. The chosen interior point scheme is the one which uses the *Newton iteration* and the reduction of the *damping parameter* (sections 2 and 3). This scheme permits a simple implementation and the convergence is assured in the framework of inexact Newton methods, even if iterative inner solvers are used (subsection 5.2). Furthermore, in this context, a nonmonotone interior point method can be introduced (see [9]).
In Section 5 (subsections 5.3 and 5.4), we analyse with examples, convergence failures of the interior point scheme using Newton iteration, indicating how to detect the failure, and then, we study the Newton iteration by analysing the behaviour of the continuous solution, in a neighbourhood of singular points, of the correspondent differential problem. A classification of singular points is also reported here.
In Section 6, we evaluate the effectiveness of the whole scheme, the damped Newton interior point method with the inner solvers, as presented in subsection 5.1.
The test problems are described in [71], [67] and [68], and they are large and sparse nonlinear programming problems that arise from finite difference discretization of elliptic boundary or distributed control problems.
These nonlinear problems have quadratic objective function, weakly nonlinear[1] equality constraints and box constraints; the involved matrices have a *PDE–type* structure and interior point methods are particularly suited.

[1] A continuously differentiable mapping $\boldsymbol{F}(\boldsymbol{u})$ is said a weakly nonlinear mapping if it has the form $\boldsymbol{F}(\boldsymbol{u}) = A\boldsymbol{u} + \boldsymbol{G}(\boldsymbol{u})$ where A is a matrix and $\boldsymbol{G}(\boldsymbol{u})$ is a continuously differentiable mapping. A weakly nonlinear system $\boldsymbol{F}(\boldsymbol{u}) = 0$, where $\boldsymbol{F}(\boldsymbol{u})$ is a nonlinear weakly nonlinear mapping ([96]), arises from the discretization of many classical semilinear elliptic partial differential equations by the finite difference or finite element methods.

In the experiments, we compare the whole scheme with the three solvers and with a direct library routine solver. Moreover, a comparison, in terms of CPU time, with other interior point codes is reported and we also consider results of the nonmonotone interior point method introduced in [9] with the third, and most efficient, inner solver.
The results highlights the efficiency of BLKFCLT routine.

2 Interior point framework

Consider the following nonlinear programming problem

$$\begin{array}{l} \min f(\boldsymbol{x}) \\ \boldsymbol{g}_1(\boldsymbol{x}) = 0 \\ \boldsymbol{g}_2(\boldsymbol{x}) \geq 0 \\ P_l \boldsymbol{x} \geq \boldsymbol{l} \\ P_u \boldsymbol{x} \leq \boldsymbol{u} \end{array} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $f(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{g}_1(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^{neq}$, $\boldsymbol{g}_2(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^m$, $\boldsymbol{l} \in \mathbb{R}^{nl}$, $\boldsymbol{u} \in \mathbb{R}^{nu}$, $P_l \in \mathbb{R}^{nl \times n}$, $P_u \in \mathbb{R}^{nu \times n}$. P_l (P_u) is given by the rows of the identity matrix whose row indices are equal to those of the entries of $\boldsymbol{x}$ which are bounded below (above). If $x_i \geq l_j$ for some i and, simultaneously, $x_i \leq u_h$, we assume $l_j \neq u_h$. This hypothesis means that there are no fixed variables. On the other hand, in this case, the problem can be reduced by eliminating them.
We assume that $f(\boldsymbol{x})$, $\boldsymbol{g}_1(\boldsymbol{x})$, $\boldsymbol{g}_2(\boldsymbol{x})$ are twice continuously differentiable and that standard assumptions for a constrained nonlinear programming problems hold ([64, Chapt. 10]). We are interested in the case where (1) is a large nonconvex problem. We assume that the first and second derivatives of the objective function and constraints are available.
By introducing slack variables, the problem (1) can be rewritten as

$$\begin{array}{l} \min f(\boldsymbol{x}) \\ \boldsymbol{g}_1(\boldsymbol{x}) = 0 \\ \boldsymbol{g}_2(\boldsymbol{x}) - \boldsymbol{s} = 0 \\ P_l \boldsymbol{x} - \boldsymbol{l} - \boldsymbol{r}_l = 0 \\ P_u \boldsymbol{x} - \boldsymbol{u} + \boldsymbol{r}_u = 0 \\ \boldsymbol{s} \geq 0, \boldsymbol{r}_l \geq 0, \boldsymbol{r}_u \geq 0 \end{array} \tag{2}$$

whose Karush–Kuhn–Tucker optimality conditions are

$$\begin{array}{lcll} \boldsymbol{\alpha} & \equiv & \nabla f(\boldsymbol{x}) - \nabla \boldsymbol{g}_1(\boldsymbol{x}) \boldsymbol{\lambda}_1 - \nabla \boldsymbol{g}_2(\boldsymbol{x}) \boldsymbol{\lambda}_2 - P_l^t \boldsymbol{\lambda}_l + P_u^t \boldsymbol{\lambda}_u & = 0 \\ \boldsymbol{\varepsilon} & \equiv & -\boldsymbol{g}_1(\boldsymbol{x}) & = 0 \\ \boldsymbol{\beta} & \equiv & -\boldsymbol{g}_2(\boldsymbol{x}) + \boldsymbol{s} & = 0 \\ \boldsymbol{\gamma} & \equiv & -P_l \boldsymbol{x} + \boldsymbol{l} + \boldsymbol{r}_l & = 0 \\ \boldsymbol{\delta} & \equiv & P_u \boldsymbol{x} - \boldsymbol{u} + \boldsymbol{r}_u & = 0 \\ \boldsymbol{\theta} & \equiv & \Lambda_2 S \boldsymbol{e}_m & = 0 \\ \boldsymbol{\zeta} & \equiv & \Lambda_l R_l \boldsymbol{e}_{nl} & = 0 \\ \boldsymbol{\eta} & \equiv & \Lambda_u R_u \boldsymbol{e}_{nu} & = 0 \end{array} \tag{3}$$

with

$$\boldsymbol{s}, \boldsymbol{r}_l, \boldsymbol{r}_u \geq 0; \qquad \boldsymbol{\lambda}_2, \boldsymbol{\lambda}_l, \boldsymbol{\lambda}_u \geq 0$$

where $\boldsymbol{s}, \boldsymbol{\lambda}_2 \in \mathbb{R}^m$, $\boldsymbol{r}_l, \boldsymbol{\lambda}_l \in \mathbb{R}^{nl}$, $\boldsymbol{r}_u, \boldsymbol{\lambda}_u \in \mathbb{R}^{nu}$ and $\Lambda_2 = \text{diag}(\boldsymbol{\lambda}_2)$; $\Lambda_l = \text{diag}(\boldsymbol{\lambda}_l)$; $\Lambda_u = \text{diag}(\boldsymbol{\lambda}_u)$; $S = \text{diag}(\boldsymbol{s})$; $R_l = \text{diag}(\boldsymbol{r}_l)$; $R_u = \text{diag}(\boldsymbol{r}_u)$.
The vector $\boldsymbol{e}_N$ indicates the vector of N components whose values are equal to 1.
Here $\nabla f(\boldsymbol{x})$ denotes the gradient of $f(\boldsymbol{x})$, $\nabla \boldsymbol{g}_1(\boldsymbol{x})$ and $\nabla \boldsymbol{g}_2(\boldsymbol{x})$ are the transpose of the Jacobian matrices of $\boldsymbol{g}_1(\boldsymbol{x})$ and $\boldsymbol{g}_2(\boldsymbol{x})$ respectively.
Let us indicate $\tilde{\boldsymbol{s}} = (\boldsymbol{s}^t, \boldsymbol{r}_l^t, \boldsymbol{r}_u^t)^t \in \mathbb{R}^p$, $\tilde{\boldsymbol{w}} = (\boldsymbol{\lambda}_2^t, \boldsymbol{\lambda}_l^t, \boldsymbol{\lambda}_u^t)^t \in \mathbb{R}^p$ and $p = m + n_l + n_u$; the *primal–dual system* (3) can be written as

$$\begin{array}{c} \boldsymbol{H}(\boldsymbol{v}) = 0 \\ \tilde{\boldsymbol{s}} \geq 0; \tilde{\boldsymbol{w}} \geq 0 \end{array} \tag{4}$$

where

$$\boldsymbol{v} = \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda}_1 \\ \left.\begin{matrix} \boldsymbol{\lambda}_2 \\ \boldsymbol{\lambda}_l \\ \boldsymbol{\lambda}_u \end{matrix}\right\} \tilde{\boldsymbol{w}} \\ \left.\begin{matrix} \boldsymbol{s} \\ \boldsymbol{r}_l \\ \boldsymbol{r}_u \end{matrix}\right\} \tilde{\boldsymbol{s}} \end{pmatrix} \quad \text{and} \quad \boldsymbol{H}(\boldsymbol{v}) = \begin{pmatrix} \left.\begin{matrix} \boldsymbol{\alpha} \\ \boldsymbol{\varepsilon} \\ \boldsymbol{\beta} \\ \boldsymbol{\gamma} \\ \boldsymbol{\delta} \end{matrix}\right\} \boldsymbol{H_1(v)} \\ \boldsymbol{\theta} \\ \boldsymbol{\zeta} \\ \boldsymbol{\eta} \end{pmatrix}$$

The Jacobian matrix of $\boldsymbol{H}$ is the matrix

$$H'(\boldsymbol{v}) = \begin{pmatrix} Q & -\nabla \boldsymbol{g}_1(\boldsymbol{x}) & -\nabla \boldsymbol{g}_2(\boldsymbol{x}) & -P_l^t & P_u^t & 0 & 0 & 0 \\ -\nabla \boldsymbol{g}_1(\boldsymbol{x})^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\nabla \boldsymbol{g}_2(\boldsymbol{x})^t & 0 & 0 & 0 & 0 & I & 0 & 0 \\ -P_l & 0 & 0 & 0 & 0 & 0 & I & 0 \\ P_u & 0 & 0 & 0 & 0 & 0 & 0 & I \\ 0 & 0 & S & 0 & 0 & \Lambda_2 & 0 & 0 \\ 0 & 0 & 0 & R_l & 0 & 0 & \Lambda_l & 0 \\ 0 & 0 & 0 & 0 & R_u & 0 & 0 & \Lambda_u \end{pmatrix}$$

where $Q = \nabla^2 f(\boldsymbol{x}) - \sum_1^{neq} \lambda_{1,i} \nabla^2 g_{1,i}(\boldsymbol{x}) - \sum_1^m \lambda_{2,i} \nabla^2 g_{2,i}(\boldsymbol{x})$ is the Hessian matrix of the Lagrangian function of the problem (2); $\nabla^2 f(\boldsymbol{x})$, $\nabla^2 g_{1,i}(\boldsymbol{x})$, $(i = 1, ..., neq)$, $\nabla^2 g_{1,j}(\boldsymbol{x})$, $(j = 1, ..., m)$, are the Hessian matrices of $f(\boldsymbol{x})$, $g_{1,i}(\boldsymbol{x})$, $(i = 1, ..., neq)$ and $g_{1,j}(\boldsymbol{x})$, $(j = 1, ..., m)$, respectively.
If we solve the system (4) with the Newton's method, at each iteration k we have to compute the vector $\Delta \boldsymbol{v}^{(k)}$ which is the solution of the Newton equation

$$H'(\boldsymbol{v}^{(k)}) \Delta \boldsymbol{v} = -\boldsymbol{H}(\boldsymbol{v}^{(k)}) \tag{5}$$

When we consider the last p equations of the system (5), the ones related to the complementarity conditions $\tilde{S}\tilde{W}\boldsymbol{e}_p = 0$, it can be observed that if, at an iteration k, $\tilde{s}_i^{(k)} = 0$ (or $\tilde{w}_i^{(k)} = 0$), then $\tilde{s}_i^{(j)} = 0$ (or $\tilde{w}_i^{(j)} = 0$), for all iterations

j with $j > k$. It means that if the iterate reaches the boundary of the feasible region, it sticks on the boundary even if it is far from the solution.
In order to avoid this disadvantage, the idea of interior point method (see e.g. [97] for one of the last survey papers) is to perturb the system (4) only in the last p equations and to generate a sequence of iterates $\{\boldsymbol{v}^{(k)}\}$ satisfying the perturbed system

$$\begin{array}{l} \boldsymbol{H}(\boldsymbol{v}) = \rho_k \tilde{\boldsymbol{e}} \\ \tilde{\boldsymbol{s}} > 0; \tilde{\boldsymbol{w}} > 0 \end{array} \tag{6}$$

and the Karush–Kuhn–Tucker conditions (4) only in the limit. The perturbation parameter ρ_k tends to 0 when k diverges. Here $\tilde{\boldsymbol{e}} = (0^t_{n+neq}, \boldsymbol{e}^t_p)^t$.
By introducing a "measure" $\mathcal{M}$ of the system (6) expressed by the vector $\boldsymbol{H}_{\rho_k}(\boldsymbol{v}) = \boldsymbol{H}(\boldsymbol{v}) - \rho_k \tilde{\boldsymbol{e}}$ (for example $\mathcal{M}(\boldsymbol{H}_{\rho_k}(\boldsymbol{v})) = \|\boldsymbol{H}_{\rho_k}(\boldsymbol{v})\|)^2$, we can write a general scheme for the whole class of the interior point methods:

1. Choose the initial guess $\boldsymbol{v}^{(0)}$ such that $\tilde{\boldsymbol{s}}^{(0)}, \tilde{\boldsymbol{w}}^{(0)} > 0$, the stopping tolerance $tol > 0$, the measure $\mathcal{M}$; $k = 0$;
2. While $\mathcal{M}(\boldsymbol{H}_0(\boldsymbol{v}^{(k)})) \geq tol$
   2a. Choose the perturbation parameter ρ_k and the inner tolerance tol_{ρ_k};
   2b. Compute a new point $\boldsymbol{v}^{(k+1)}$ such that:

$$\begin{array}{l} \mathcal{M}(\boldsymbol{H}_{\rho_k}(\boldsymbol{v}^{(k+1)})) < tol_{\rho_k} \\ (\tilde{\boldsymbol{s}}^{(k+1)}, \tilde{\boldsymbol{w}}^{(k+1)}) > 0 \end{array}$$

   2c. Set $k = k + 1$

The scheme described above includes a wide class of methods; it allows many choices for $\mathcal{M}$, for the perturbation parameter ρ_k and for the method used to compute the new point at the step 2b. Barrier function methods (see e.g. [39, §12.1]) can also be described by such a scheme. Some recent interior point methods for nonlinear programming are reported in the papers [36], [82], [99], [41], [91] [16], [5], [87], [19], [15], [3], [86], [94].
In Table 1 we report some results, obtained by Hans Mittelmann at the Arizona State University [70], of a comparison among interior point codes for nonlinear programming: the LOQO algorithm[3], version 6.2, (see [91], [83], [7]), the KNITRO algorithms[4], version 3.1, (see [16], [15]) with the direct (KNITRO–D) or with the iterative (KNITRO–I) solution ([73]) of inner subproblems and IPOPT algorithm[5], (see [92], [94]).
In the table, "it" and "sec" indicate the number of iterations and the time expressed in seconds respectively, the symbols "*" and "m" denote an algorithm failure and a memory failure. The codes have been carried out in order to obtain the same precision on the solution.

[2] Here and in the following, the vector norm $\|\cdot\|$ indicates the Euclidean norm
[3] *http://www.orfe.princeton.edu/~loqo/*
[4] *http://www.ziena.com/knitro.html*
[5] *http://www.coin-or.org/Ipopt/*

	LOQO		KNITRO–D		KNITRO–I		IPOPT	
	it	sec	it	sec	it	sec	it	sec
P1-3-199	39	108	19	72	12	94	25	276
P1-3-299	*	*	20	278	12	322	20	1143
P1-3-399	*	*	21	786	15	1020	28	3618
P1-3-499	*	*	22	1585	14	1754	22	7374
P1-3-599	*	*	*	*	16	2876	m	m
P2-6-99	131	51	34	17	45	33	91	29
P2-6-199	143	427	44	180	41	263	74	302
P2-6-299	*	*	41	674	101	1637	113	1670
P2-6-399	*	*	40	1829	109	4693	90	3518
P2-6-499	*	*	42	3498	*	*	88	7034

Table 1: Comparison on elliptic control test problems ([70])

For the description of the test problems see Table 4 in the section of the numerical experiments.
See [72] for a comparison of interior point codes and active set sequential quadratic programming codes on CUTE collection test problems.

3 An interior point method as inexact Newton scheme

3.1 Inexact Newton methods

Consider the case of systems of nonlinear equations:

$$\boldsymbol{F}(\boldsymbol{u}) = 0$$

where $\boldsymbol{F} : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable; if we denote $F'(\boldsymbol{u})$ the Jacobian matrix of $\boldsymbol{F}$ at $\boldsymbol{u}$, which we suppose nonsingular, the inexact Newton method introduced in [23] has the form:

For $k = 0, 1, ...$ *until the convergence do*

- *Find the step* $\boldsymbol{\Delta} u^{(k)}$ *which satisfies*

$$F'(\boldsymbol{u}^{(k)})\Delta\boldsymbol{u} = -\boldsymbol{F}(\boldsymbol{u}^{(k)}) + \boldsymbol{r}^{(k)} \qquad \textit{where} \quad \frac{\|\boldsymbol{r}^{(k)}\|}{\|\boldsymbol{F}(\boldsymbol{u}^{(k)})\|} \leq \eta_k \tag{7}$$

- *Set* $\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \Delta\boldsymbol{u}^{(k)}$

Here, $\boldsymbol{u}^{(0)}$ is a suitable initial guess, the *forcing term* $\eta_k \in [0, 1)$ is a measure of the inexactness of the solution of Newton equation

$$F'(\boldsymbol{u}^{(k)})\Delta\boldsymbol{u} = -\boldsymbol{F}(\boldsymbol{u}^{(k)}) \tag{8}$$

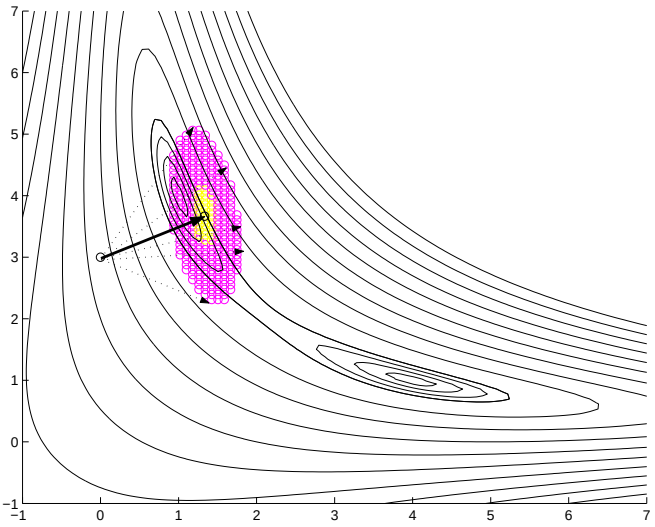


Figure 1: Newton step, inexact Newton and nonmonotone inexact Newton regions

We observe that inexact Newton methods include the class of Newton–iterative methods (e.g. see [76], [84], [57], [43], [18]), where an iterative method is used to compute an approximation of the solution of the Newton equation (8) with stopping criteria given in (7) and the class of quasi–Newton methods (e.g. see [79], [24]) when the *a posteriori* Dennis–Moré condition holds ([57, Theor. 7.1.1]).
A global convergence of inexact Newton method has been introduced (see [35], [79]), by requiring, at each iteration, a prescribed reduction of the norm of $\boldsymbol{F}$.
Thus, the **fundamental convergence theorem** of inexact Newton method [79, Theor. 6.7] can state as follow:

- if a sequence $\{\boldsymbol{u}^{(k)}\}$ satisfying the residual and the norm conditions

$$\|F'(\boldsymbol{u}^{(k)})(\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}) + \boldsymbol{F}(\boldsymbol{u}^{(k)})\| \leq \eta_k \|\boldsymbol{F}(\boldsymbol{u}^{(k)})\| \tag{9}$$

$$\|\boldsymbol{F}(\boldsymbol{u}^{(k+1)})\| \leq \xi_k \|\boldsymbol{F}(\boldsymbol{u}^{(k)})\| \tag{10}$$

  (with $0 \leq \eta_k \leq \bar{\eta} < 1$, $0 < \xi_k \leq \bar{\xi} < 1$) has a limit point $\boldsymbol{u}^*$ where $F'(\boldsymbol{u}^*)$ is nonsingular, then $\{\boldsymbol{u}^{(k)}\}$ converges to $\boldsymbol{u}^*$ and $\boldsymbol{F}(\boldsymbol{u}^*) = 0$.

Different strategies to implement condition (10) are analyzed in [35].
Furthermore, a nonmonotone inexact Newton method has been also introduced in [9], where both the conditions (9) and (10) have been relaxed.
In Figure 1, the arrow denotes the Newton step for the problem $\boldsymbol{F}(\boldsymbol{u}) = 0$, with

$$\boldsymbol{F}(\boldsymbol{u}) = \begin{pmatrix} u_1 + u_2 - 5 \\ u_1 u_2 - 4 \end{pmatrix}$$

while the smaller light grey region contains the points of the region allowed by the inexact Newton step, and the larger dark grey region contains the points of the region of the nonmonotone inexact Newton step. Thus, a coarser accuracy in the solution of the Newton equation and larger step sizes are allowed in a nonmonotone choice.

3.2 A Newton interior point method

The *Newton interior point method* for nonlinear programming (1) is obtained when the step 2b of the scheme of the previous section is performed by applying Newton's method to the perturbed system (6); that is at each iteration k, we compute the solution $\Delta \boldsymbol{v}^{(k)}$ of the *perturbed Newton equation*

$$H'(\boldsymbol{v}^{(k)})\Delta \boldsymbol{v} = -\boldsymbol{H}(\boldsymbol{v}^{(k)}) + \rho_k \tilde{\boldsymbol{e}} \tag{11}$$

Let us define $\rho_k = \sigma_k \mu_k$, where $\sigma_k \in [\sigma_{\min}, \sigma_{\max}] \subset (0,1)$; if the following condition holds

$$\mu_k \leq \mu_k^{(2)} \equiv \frac{\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|}{\sqrt{p}}, \tag{12}$$

then the solution $\Delta \boldsymbol{v}^{(k)}$ of the system (11) is a descent direction for $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ ([28, p. 77]) and, from (11), it satisfies the residual condition (9) of the inexact Newton method that is rewritten as

$$\|H'(\boldsymbol{v}^{(k)})\Delta \boldsymbol{v}^{(k)} + \boldsymbol{H}(\boldsymbol{v}^{(k)})\| \leq \eta_k \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \tag{13}$$

with the forcing term $\eta_k = \sigma_k \leq \sigma_{\max} < 1$.

Furthermore, it is easy to prove that $\mu_k^{(1)} \equiv \frac{\tilde{\boldsymbol{s}}^{(k)\,t} \tilde{\boldsymbol{w}}^{(k)}}{p} \leq \mu_k^{(2)}$, where $\mu_k^{(1)}$ is the usual choice of the perturbation parameter in the interior point method and is strictly connected with the notion of adherence to the central path (e.g. see [36]). Then, the choice of the perturbation parameter

$$\mu_k \in [\mu_k^{(1)}, \mu_k^{(2)}] \tag{14}$$

assures that $\Delta \boldsymbol{v}^{(k)}$ satisfies the residual condition of the inexact Newton method and it is a descent direction for $\|\boldsymbol{H}(\boldsymbol{v})\|^2$.

At the same time, the range of values of the perturbation parameter is enlarged in order to avoid *stagnation* of the current iterate on the boundary of the non-negative orthant $(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{w}}) \geq 0$ that occurs when the value of $\mu_k^{(1)}$ is too small and we are far away from the solution (see Section 5 in [29]).

When the size of the system (11) is large, the computation of the exact solution can be too expensive and then the system (11) can be solved *approximately*. We denote again by $\Delta \boldsymbol{v}^{(k)}$ the approximate solution of system (11). If the coefficient matrix has a special structure, an iterative scheme can exploit this feature. Nevertheless, the use of an iterative solver determines the necessity to state an adaptive termination rule so that the accuracy in solving the inner system depends on the quality of the current iterate of the outer method in order to avoid unnecessary inner iterations when we are far from the solution.

Then, we can apply an inner iterative scheme to the perturbed Newton equation (11) until the final inner residual

$$\boldsymbol{r}^{(k)} = H'(\boldsymbol{v}^{(k)})\Delta\boldsymbol{v}^{(k)} + \boldsymbol{H}(\boldsymbol{v}^{(k)}) - \sigma_k\mu_k\tilde{\boldsymbol{e}} \tag{15}$$

satisfies the condition

$$\|\boldsymbol{r}^{(k)}\| \leq \delta_k\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \tag{16}$$

That is, we introduce a further perturbation. Obviously, the choice $\delta_k = 0$ means the system (11) is solved exactly.
It is possible to prove (see [10, Theor. 1]), that, if $\sigma_k \in (0, \sigma_{\max}] \subset (0, 1)$, $\delta_k \in [0, \delta_{\max}] \subset [0, 1)$ and $\sigma_{\max} + \delta_{\max} < 1$, then, the vector $\Delta\boldsymbol{v}^{(k)}$, that satisfies (15) and (16), is a descent direction for $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ and it satisfies the residual condition (13) with the forcing term $\eta_k = \sigma_k + \delta_k$.

The new iterate $\boldsymbol{v}^{(k+1)}$ can be obtained by a globally convergent modification of Newton's method such as a line search technique or a trust region approach (e.g. see [25, §6.3, 6.4]).
We consider the Newton line–search interior point method (or damped Newton interior point method), which computes the new iterate as follows

$$\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)} + \alpha_k\Delta\boldsymbol{v}^{(k)} \tag{17}$$

where the *damping* parameter α_k has to satisfy the feasibility of the new iterate and appropriate path–following conditions (*centrality conditions*) and has to guarantee a *sufficient decrease* of a *merit function*, for example of $\|\boldsymbol{H}(\boldsymbol{v})\|^2$.
In order to satisfy all the conditions, the damping parameter α_k is determined by the following sequence of steps:

1. feasibility condition means that all the iterates $\boldsymbol{v}^{(k)}$ have to belong to the feasible region

$$\{\boldsymbol{v} \in \mathbb{R}^{n+neq+2p} \quad \text{s.t.} \quad \tilde{s}_i > 0 \quad \text{and} \quad \tilde{w}_i > 0 \quad \forall\, i = 1, ..., p\}.$$

   So, if $\Delta\tilde{s}_i^{(k)} < 0$ (or $\Delta\tilde{w}_i^{(k)} < 0$), ${\alpha_k}^{(1)}$ will be chosen such that $\tilde{s}_i^{(k+1)} > 0$ (or $\tilde{w}_i^{(k+1)} > 0$);

2. centrality conditions are expressed by the nonnegativity of the following functions introduced in [36] (see also [74, p. 402]):

$$\varphi(\alpha) \equiv \min_{i=1,p}\left(\tilde{S}^{(k)}(\alpha)\tilde{W}^{(k)}(\alpha)\boldsymbol{e}_p\right) - \gamma_k\tau_1\left(\frac{\tilde{\boldsymbol{s}}^{(k)}(\alpha)^t\tilde{\boldsymbol{w}}^{(k)}(\alpha)}{p}\right) \geq 0 \tag{18}$$

$$\psi(\alpha) \equiv \tilde{\boldsymbol{s}}^{(k)}(\alpha)^t\tilde{\boldsymbol{w}}^{(k)}(\alpha) - \gamma_k\tau_2\|\boldsymbol{H}_1(\boldsymbol{v}^{(k)}(\alpha))\| \geq 0 \tag{19}$$

   where $\tilde{\boldsymbol{s}}^{(k)}(\alpha) = \tilde{\boldsymbol{s}}^{(k)} + \alpha\Delta\tilde{\boldsymbol{s}}^{(k)}$ and $\tilde{\boldsymbol{w}}^{(k)}(\alpha) = \tilde{\boldsymbol{w}}^{(k)} + \alpha\Delta\tilde{\boldsymbol{w}}^{(k)}$; $\gamma_k \in [\frac{1}{2}, 1)$.

At each iterate we choose $\tilde{\alpha}_k$ such that conditions (18)–(19) are satisfied $\forall \alpha \in (0, \tilde{\alpha}_k] \subseteq (0, 1]$; then ${\alpha_k}^{(2)} = \min\{\tilde{\alpha}_k, {\alpha_k}^{(1)}\}$.

In order to satisfy inequalities (18) and (19) in the initial iterate, we have $\tau_1 \leq \frac{\min_{i=1,p}\left(\tilde{S}^{(0)}\tilde{W}^{(0)}\boldsymbol{e}_p\right)}{(\frac{\tilde{\boldsymbol{s}}^{(0)\,t}\tilde{\boldsymbol{w}}^{(0)}}{p})}$, and $\tau_2 \leq \frac{\tilde{\boldsymbol{s}}^{(0)\,t}\tilde{\boldsymbol{w}}^{(0)}}{\|\boldsymbol{H}_1(\boldsymbol{v}^{(0)})\|}$, where we assume $\tilde{\boldsymbol{s}}^{(0)} > 0$, $\tilde{\boldsymbol{w}}^{(0)} > 0$.

Practically, we set

$$\tau_1 = \min\left(0.99, \frac{10^{-7} \cdot \min_{i=1,p}\left(\tilde{S}^{(0)}\tilde{W}^{(0)}\boldsymbol{e}_p\right)}{0.5 \cdot (\frac{\tilde{\boldsymbol{s}}^{(0)\,t}\tilde{\boldsymbol{w}}^{(0)}}{p})}\right);\ \tau_2 = 10^{-7} \cdot \frac{\tilde{\boldsymbol{s}}^{(0)\,t}\tilde{\boldsymbol{w}}^{(0)}}{\|\boldsymbol{H}_1(\boldsymbol{v}^{(0)})\|} \tag{20}$$

3. a sufficient decrease of the merit function $\|\boldsymbol{H}(\boldsymbol{v})\|^2$, can be obtained implementing the Armijo backtracking procedure ([6]) as in [36] or the one in [28]

   - *Set* $\beta \in (0,1)$, $\theta \in (0,1)$, $\alpha = {\alpha_k}^{(2)}$, $t = 0$;
   - *while* $\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha\Delta\boldsymbol{v}^{(k)})\| > (1 - \beta\alpha(1 - (\sigma_k + \delta_k)))\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$ *or* $t < t_{\max}$
     $\alpha \leftarrow \theta\alpha$; $t \leftarrow t + 1$;
     endwhile

We observe that, the first centrality condition, $\varphi(\alpha) \geq 0$, keeps the iterates $\boldsymbol{v}^{(k)}(\alpha) = \boldsymbol{v}^{(k)} + \alpha\Delta\boldsymbol{v}^{(k)}$ far from the boundary of the feasible region, while the second, $\psi(\alpha) \geq 0$, forces the sequence $\{{\tilde{\boldsymbol{s}}^{(k)}}^t\tilde{\boldsymbol{w}}^{(k)}\}$ to converge to zero slower than the sequence $\{\|\boldsymbol{H}_1(\boldsymbol{v}^{(k)})\|\}$.
If the backtracking procedure terminates after $\bar{t}$ steps, we denote as α_k the last value α in the backtracking rule, i.e. $\alpha_k = \theta^{\bar{t}}\alpha_k^{(2)}$.
Let us consider the following:

P1. Suppose that $\alpha_k^{(1)}$ is bounded below by a scalar greater than zero, say $\alpha^{(1)}$, and that also $\alpha_k^{(2)}$ is bounded below by a scalar greater than zero, say $\tilde{\alpha}$. Thus, we denote $\alpha^{(2)} = \min\{\alpha^{(1)}, \tilde{\alpha}\} > 0$.
P2. Suppose that the backtracking rule terminates after a finite number of steps, then α_k is bounded below by a positive scalar, say $\breve{\alpha} > 0$.
We have the following result.
If P1 and P2 hold (see subsection 5.2), and denoting $\bar{\alpha} = \min\{\alpha^{(2)}, \breve{\alpha}\} > 0$, then we have that $\alpha_k \geq \bar{\alpha} > 0$ and that the vector $\alpha_k\Delta\boldsymbol{v}^{(k)}$ satisfies the norm condition of the inexact Newton method (10) with $\bar{\xi} = (1 - \beta\bar{\alpha}(1 - (\sigma_{\max} + \delta_{\max}))) < 1$.
Moreover, from (12), (15) and (16), it is easy to prove that the vector $\alpha_k\Delta\boldsymbol{v}^{(k)}$, $k \geq 0$, also satisfies the residual condition of the inexact Newton method (13) with the forcing term $\eta_k = 1 - \alpha_k(1 - (\sigma_k + \delta_k)) \leq 1 - \bar{\alpha}(1 - (\sigma_{\max} + \delta_{\max})) \equiv \bar{\eta} < 1$.

In the following, in the related section 5.2, we report convergence results in [10] of the Newton line–search interior point method based on convergence results of inexact Newton methods.
Furthermore, in [9], the author extends the interior point method to nonmonotone choices in the context of the nonmonotone inexact Newton method. Indeed, by allowing the choice of the parameter μ_k in the larger interval, respect to the one in (14)

$$\mu_k \in \left[\frac{\tilde{\boldsymbol{s}}^{(k)\,t} \tilde{\boldsymbol{w}}^{(k)}}{p}, \frac{\|\boldsymbol{H}(\boldsymbol{v}^{(\ell(k))})\|}{\sqrt{p}} \right]$$

if the direction $\Delta \boldsymbol{v}^{(k)}$, computed by solving approximately the Newton equation (11), satisfies the condition

$$\|\boldsymbol{r}^{(k)}\| \leq \delta_k \|\boldsymbol{H}(\boldsymbol{v}^{(\ell(k))})\|$$

then, such direction is a nonmonotone inexact Newton step with forcing term equal to $\delta_k + \sigma_k$. A nonmonotone backtracking rule is also introduced

$$\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha_k \Delta \boldsymbol{v}^{(k)})\| \leq (1 - \alpha_k \beta (1 - (\delta_k + \sigma_k)) \|\boldsymbol{H}(\boldsymbol{v}^{(\ell(k))})\|.$$

Here, we still denote by $\{\boldsymbol{v}^{(k)}\}$ the nonmonotone interior point sequences, and by $\{\boldsymbol{r}^{(k)}\}$ the sequence of the residual in (15); the vector $\boldsymbol{v}^{(\ell(k))}$ indicates an element of the sequence $\{\boldsymbol{v}^{(k)}\}$ such that

$$\|\boldsymbol{H}(\boldsymbol{v}^{(\ell(k))})\| \equiv \max_{0 \leq j \leq \min(M,k)} \|\boldsymbol{H}(\boldsymbol{v}^{(k-j)})\|$$

where $k - \min(M, k) \leq \ell(k) \leq k$. Here $M \in \mathbb{N}$ is called *memory* or *degree of nonmonotonicity*.
We observe that the nonmonotone choices involve three crucial issues: the perturbation parameter, the inner adaptive stopping criterion and the backtracking rule. The first two choices influence the direction itself, while a less restrictive backtracking rule allows to retain larger stepsizes than in the monotone case.
Convergence properties and numerical experiences of this nonmonotone interior point method are also investigated in [9].

4 Iterative solvers for interior point iteration

We focus our attention to the solution of the linear system (11) that, by omitting the iteration index k, can be written

$$\begin{cases} Q\Delta\boldsymbol{x} - \nabla\boldsymbol{g}_1(\boldsymbol{x})\Delta\boldsymbol{\lambda}_1 - \nabla\boldsymbol{g}_2(\boldsymbol{x})\Delta\boldsymbol{\lambda}_2 - P_l^t\Delta\boldsymbol{\lambda}_l + P_u^t\Delta\boldsymbol{\lambda}_u &=& -\boldsymbol{\alpha} \\ -\nabla\boldsymbol{g}_1(\boldsymbol{x})^t\Delta\boldsymbol{x} &=& -\boldsymbol{\varepsilon} \\ -\nabla\boldsymbol{g}_2(\boldsymbol{x})^t\Delta\boldsymbol{x} + \Delta\boldsymbol{s} &=& -\boldsymbol{\beta} \\ -P_l\Delta\boldsymbol{x} + \Delta\boldsymbol{r}_l &=& -\boldsymbol{\gamma} \\ +P_u\Delta\boldsymbol{x} + \Delta\boldsymbol{r}_u &=& -\boldsymbol{\delta} \\ S\Delta\boldsymbol{\lambda}_2 + \Lambda_2\Delta\boldsymbol{s} &=& -\boldsymbol{\theta} + \rho\boldsymbol{e}_m \\ R_l\Delta\boldsymbol{\lambda}_l + \Lambda_l\Delta\boldsymbol{r}_l &=& -\boldsymbol{\zeta} + \rho\boldsymbol{e}_{nl} \\ R_u\Delta\boldsymbol{\lambda}_u + \Lambda_u\Delta\boldsymbol{r}_u &=& -\boldsymbol{\eta} + \rho\boldsymbol{e}_{nu} \end{cases}$$

From the complementarity equations we can deduce

$$\Delta\tilde{\boldsymbol{s}} = \begin{pmatrix} \Delta\boldsymbol{r}_l \\ \Delta\boldsymbol{r}_u \\ \Delta\boldsymbol{s} \end{pmatrix} = \begin{pmatrix} \Lambda_l^{-1}[-R_l\Delta\boldsymbol{\lambda}_l - \boldsymbol{\zeta} + \rho\boldsymbol{e}_{nl}] \\ \Lambda_u^{-1}[-R_u\Delta\boldsymbol{\lambda}_u - \boldsymbol{\eta} + \rho\boldsymbol{e}_{nu}] \\ \Lambda_2^{-1}[-S\Delta\boldsymbol{\lambda}_2 - \boldsymbol{\theta} + \rho\boldsymbol{e}_m] \end{pmatrix}$$

and then

$$\Delta\tilde{\boldsymbol{w}} = \begin{pmatrix} \Delta\boldsymbol{\lambda}_l \\ \Delta\boldsymbol{\lambda}_u \\ \Delta\boldsymbol{\lambda}_2 \end{pmatrix} = \begin{pmatrix} R_l^{-1}[-\Lambda_l P_l\Delta\boldsymbol{x} + \Lambda_l\boldsymbol{\gamma} - \boldsymbol{\zeta} + \rho\boldsymbol{e}_{nl}] \\ R_u^{-1}[\Lambda_u P_u\Delta\boldsymbol{x} + \Lambda_u\boldsymbol{\delta} - \boldsymbol{\eta} + \rho\boldsymbol{e}_{nu}] \\ S^{-1}[-\Lambda_2\nabla\boldsymbol{g}_2(\boldsymbol{x})^t\Delta\boldsymbol{x} + \Lambda_2\boldsymbol{\beta} - \boldsymbol{\theta} + \rho\boldsymbol{e}_m] \end{pmatrix}$$

where $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{\lambda}_1$ are the solution of the system in a *condensed form*

$$\begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix}\begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda}_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{q} \end{pmatrix} \tag{21}$$

with

$$\begin{aligned} A = \;& Q + \nabla\boldsymbol{g}_2(\boldsymbol{x})S^{-1}\Lambda_2\nabla\boldsymbol{g}_2(\boldsymbol{x})^t + P_l^t R_l^{-1}\Lambda_l P_l + P_u^t R_u^{-1}\Lambda_u P_u \\ B = \;& -\nabla\boldsymbol{g}_1(\boldsymbol{x}) \\ \boldsymbol{c} = \;& -\boldsymbol{\alpha} - \nabla\boldsymbol{g}_2(\boldsymbol{x})S^{-1}[\Lambda_2\boldsymbol{g}_2(\boldsymbol{x}) + \rho\boldsymbol{e}_m] - P_l^t R_l^{-1}[\Lambda_l(P_l\boldsymbol{x} - \boldsymbol{l}) - \rho\boldsymbol{e}_{nl}] - \\ & -P_u^t R_u^{-1}[\Lambda_u(P_u\boldsymbol{x} - \boldsymbol{u}) + \rho\boldsymbol{e}_{nu}] \\ \boldsymbol{q} = \;& -\boldsymbol{\varepsilon} \end{aligned}$$

The system (21) is symmetric and indefinite and it can be solved by the sparse Bunch–Parlett triangular factorization ([14]), that combines dynamic reordering for sparsity preserving and pivoting technique for numerical stability (see MA27 routine of HSL Library ([54])) or by considering an *inertia–controlling factorization* (see [42], [41], [40]).
Nevertheless, for large scale nonlinear programming, the size of these systems is large and, even if the coefficient matrices are sparse and the sparsity is exploited, the computation of the solution by direct methods can be very expensive in terms of CPU time and storage requirements. Indeed, in the framework of direct methods, much efforts have been performed to avoid the use of MA27 for large scale nonlinear programming problems.
Interior point schemes transform, by elimination techniques as the one above, the symmetric systems (11) in a condensed form and reduce it into a *quasidefinite* form[6] ([90]), such that a Cholesky–like factorization can be obtained. At the beginning of the interior point scheme, the a–priori determination of a sparsity preserving reordering of the coefficient matrix (taking into account only of its structure) and of the symbolic Cholesky factor is carried out. Then, at each

[6] A matrix $\begin{pmatrix} S & V \\ V^T & -U \end{pmatrix}$ is quasidefinite if S and U are symmetric positive definite matrices. A quasidefinite matrix is strongly factorizable, i.e. a Cholesky–like factorization LDL^T (with a diagonal matrix D and a lower triangular matrix L with diagonal elements equal to one) exists for any symmetric permutation of the quasidefinite matrix. The diagonal matrix D has a number of positive (negative) diagonal entries equal to the size of S (U respectively). See [50] for an error analysis of Cholesky–like factorization of quasidefinite matrices.

iteration the factor is computed, without using pivoting technique, saving a lot of CPU time.
The reduction of a coefficient matrix into a quasidefinite form is obtained by a *regularization* technique, consisting in to modify this matrix by adding an appropriate diagonal matrix $\tilde{D}$ (e.g. see [91], [81], [2]). In ([2]) the matrix $\tilde{D}$ is dynamically computed in the implementation of the Cholesky factorization: when a critical pivot is reached, this is perturbed by a small quantity with a convenient sign.
Nevertheless, the use of regularization requires additional recovery procedures and several factorizations; for example to individuate a perturbation as small as possible ([91]) or to implement an iterative refinement if the computed solution of the perturbed system is not satisfactory ([2]).
From a theoretical point of view, an interior point method can be viewed as a nonstationary iterative method of the form

$$\boldsymbol{v}^{(k+1)} = \boldsymbol{\phi}_k(\boldsymbol{v}^{(k)})$$

where $\boldsymbol{\phi}_k$ is the law of the method; thus, an a priori unknown modification of the matrices of the problem, or an a priori unknown bound of the modification of the problem, is equivalent to find a new iterate which is not described by the law of the method; in other words, it is equivalent to interrupt the sequence and to restart from another initial point.
A different approach that avoids modifications of the matrices of the subproblems is to use iterative inner solvers for (21), that exploit the sparsity of the involved matrices, solving *approximately* the inner subproblems, so that unnecessary inner iterations can be avoided when we are far from the solution.
As seen in the previous section, the Newton line–search interior point method combined with an inner iterative solver can be viewed as an inexact Newton method and we can deduce a suitable *adaptive* stopping rule for the inner solver that assures the global convergence and the local superlinear convergence of the whole outer–inner scheme (see [10], [34]).
We remark that, when the solution $\Delta\boldsymbol{v}^{(k)}$ is computed by solving approximately the perturbed Newton equation (11) rewritten in the condensed form (21), the further perturbation that the inner solver introduces on the residual (15), appears only in the first two block–rows. That is, if we partition the residual $\boldsymbol{r}^{(k)}$ commensurately as $\boldsymbol{v}^{(k)}$, we have

$$\boldsymbol{r}^{(k)} = \begin{pmatrix} \boldsymbol{r}_1^{(k)} \\ \boldsymbol{r}_2^{(k)} \\ 0 \\ 0 \end{pmatrix}; \qquad \begin{pmatrix} \boldsymbol{r}_1^{(k)} \\ \boldsymbol{r}_2^{(k)} \end{pmatrix} = \begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda}_1 \end{pmatrix} + \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{q} \end{pmatrix}$$

Here, we remind that A, B, $\boldsymbol{c}$, $\boldsymbol{q}$, $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{\lambda}_1$ are dependent on the outer iteration k.
In the following two subsections, we consider the iterative method of multipliers for the approximate solution of the system (21) that yields at each inner iterate

a symmetric positive definite system of small size, that can be solved by efficient Cholesky codes, and in the other subsection, two different implementation of preconditioned conjugate gradient (PCG) method for system (21) with the preconditioner described in [65] (see also [66]); efficient code for Cholesky factorization are requested in the first version while in the second an efficient code for Cholesky–like factorization is introduced.

4.1 The method of multipliers

Suppose that the matrices A and B of the system (21) satisfy the following conditions:

- B^t is a full row rank matrix;
- A is symmetric and positive definite on the null space of B^t: $\mathcal{N}(B^t) = \{\boldsymbol{x} \in \mathbb{R}^n \ : \ B^t\boldsymbol{x} = 0\}$.

These conditions assure that the matrix

$$M = \begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix} \tag{22}$$

is nonsingular ([64, p. 424]). We note that these assumptions are the standard assumptions for local sequential quadratic programming (SQP) method ([74, p. 531])
Setting $\boldsymbol{y}_1 = \Delta\boldsymbol{x}$ and $\boldsymbol{y}_2 = \Delta\boldsymbol{\lambda}_1$, the system (21), can be viewed as the Lagrange necessary conditions for the minimum point of the following quadratic problem

$$\begin{array}{ll} \min & \frac{1}{2}\boldsymbol{y}_1^t A\boldsymbol{y}_1 - \boldsymbol{c}^t\boldsymbol{y}_1 \\ & B^t\boldsymbol{y}_1 - \boldsymbol{q} = 0 \end{array}$$

This quadratic problem can be solved efficiently by the method of multipliers[7] [8], that consists in updating the dual variable by the rule

$$\boldsymbol{y}_2^{(\nu+1)} = \boldsymbol{y}_2^{(\nu)} + \chi(B^t\boldsymbol{y}_1^{(\nu)} - \boldsymbol{q}) \tag{23}$$

where χ is a positive parameter (penalty parameter) and $\boldsymbol{y}_1^{(\nu+1)}$ minimizes the augmented Lagrangian function of the quadratic problem

$$\mathcal{L}_\chi(\boldsymbol{y}_1, \boldsymbol{y}_2^{(\nu)}) = \frac{1}{2}\boldsymbol{y}_1^t A\boldsymbol{y}_1 - \boldsymbol{y}_1^t\boldsymbol{c} + \boldsymbol{y}_2^{(\nu)^t}(B^t\boldsymbol{y}_1 - \boldsymbol{q}) + \frac{\chi}{2}(B^t\boldsymbol{y}_1 - \boldsymbol{q})^t(B^t\boldsymbol{y}_1 - \boldsymbol{q})$$

This means that $\boldsymbol{y}_1^{(\nu+1)}$ is the solution of the linear system of order n

$$(A + \chi BB^t)\boldsymbol{y}_1 = -B\boldsymbol{y}_2^{(\nu)} + \boldsymbol{c} + \chi B\boldsymbol{q} \tag{24}$$

[7] The method of multipliers [56, Chapt. 5, §10, p. 307], was originally suggested by Hestenes in [55]; an equivalent method motivated from a different viewpoint has been proposed by Powell in [77]. See [64, Chapt. 13] for the dual viewpoint of the method.

[8] In [45] it is showed that the method of multipliers for equality constrained least squares problems is equal to the method of weighting [89] for a particular choice of the starting point.

We remark that if we premultiply for an appropriate matrix the *augmented* system (21), we have

$$\begin{pmatrix} I & \chi B \\ 0 & I \end{pmatrix} \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{pmatrix} = \begin{pmatrix} I & \chi B \\ 0 & I \end{pmatrix} \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{q} \end{pmatrix}$$

then, changing the sign to the last block-row, we have

$$\begin{pmatrix} A + \chi BB^T & B \\ -B^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{c} + \chi B\boldsymbol{q} \\ -\boldsymbol{q} \end{pmatrix}$$

If we split the coefficient matrix of this last system into:

$$\begin{pmatrix} A + \chi BB^T & B \\ -B^T & 0 \end{pmatrix} = D - L - U$$

with

$$D = \begin{pmatrix} A + \chi BB^T & 0 \\ 0 & \frac{1}{\chi} I \end{pmatrix}; \qquad L = \begin{pmatrix} 0 & 0 \\ B^T & 0 \end{pmatrix}; \qquad U = \begin{pmatrix} 0 & -B \\ 0 & \frac{1}{\chi} I \end{pmatrix};$$

and we apply Gauss-Seidel method (SOR–like method in [52] with $\omega = 1$, see also [62]), we obtain the two iteration of the method of multipliers (23)–(24) (see also [51]).
Moreover, we note that, since B^t has full row–rank, the null space of BB^t is equal to the null space of B^t, then the matrix A is positive definite on the null space of BB^t.
Then, by the theorem in ([64, p. 408]), there exists a positive parameter χ^* such that for all $\chi > \chi^*$, the matrix $A + \chi BB^t$ is positive definite.

This last result enables us to solve the system (24) by applying a Cholesky factorization.
In order to choose the parameter χ, we observe that, for any $\boldsymbol{x} \neq 0$, we must have $\boldsymbol{x}^t(A + \chi BB^t)\boldsymbol{x} > 0$. When $B^t\boldsymbol{x} = 0$, we have $\boldsymbol{x}^t A\boldsymbol{x} > 0$.
If $B^t\boldsymbol{x} \neq 0$, $\boldsymbol{x}^t BB^t\boldsymbol{x} > 0$. Then, it follows that

$$\chi > \max(0, \max_{\boldsymbol{x} \notin \mathcal{N}(B^t)} \frac{-\boldsymbol{x}^t A\boldsymbol{x}}{\boldsymbol{x}^t BB^t\boldsymbol{x}})$$

Since $\|A\| \geq (-\boldsymbol{x}^t A\boldsymbol{x})/(\boldsymbol{x}^t\boldsymbol{x})$ for any natural norm and also for the Frobenius norm $\|\cdot\|_F$, and $\boldsymbol{x}^t BB^t\boldsymbol{x}/(\boldsymbol{x}^t\boldsymbol{x}) \geq \tau_{\min}$, where $\tau_{\min}$ is the minimum nonzero eigenvalue of BB^t or of B^tB, we can choose χ as the following value (see also [51]):

$$\chi = \frac{\|A\|_F}{t_{\min}} \tag{25}$$

where $t_{\min}$ is the minimum between 1 and the smallest positive diagonal element of B^tB. Although $t_{\min} \geq \tau_{\min}$, $t_{\min}$ is a good approximation of $\tau_{\min}$ ([46]).

For an analysis of the conditioning of the system (24) and on the behaviour of the method of multipliers with a *normalization matrix* see [46] and [27, §6].
Let us consider the implementation details of the method. We assume that the Hessian matrix Q of the Lagrangian function and the Jacobian matrix B^t of the equality constraints are stored in a column compressed format ([80]).

In the cases of the test problems of the section of the numerical experiments, the inequality constraints are box constraints, then the matrices A and Q have the same structure and they differ only for the diagonal entries.
Thus, at each outer iteration of the interior point method, the method of multipliers requires the computation of the matrix $T = A + \chi BB^t$ and its Cholesky factorization $T = L_n L_n^t$.
The operations related to each inner iteration ν, that is, sparse matrix–vector products $B(-\boldsymbol{y}_2^{(\nu)} + \chi \boldsymbol{q})$ and $B^t \boldsymbol{y}_1^{(\nu)}$ and solution of the triangular systems with coefficient matrices L_n, have a negligible computational complexity.
Thus, the method of multipliers requires:

- for any outer iteration, the computation of the matrix $T = A + \chi BB^t$ and its Cholesky factorization $T = L_n L_n^t$;
- for any inner iteration ν, the sparse matrix–vector products $B(-\boldsymbol{y}_2^{(\nu)} + \chi \boldsymbol{q})$ and $B^t \boldsymbol{y}_1^{(\nu)}$ and the solution of the triangular systems related to L_n and L_n^t.

The computational complexity of any inner iteration is negligible with respect to the operations required at any outer iteration.
When BB^t is sufficiently sparse, in order to save a lot of CPU time, before starting the outer scheme, we can perform a preprocessing procedure that executes the following steps:

- the formation of a data structure for storing the indices of the nonzero entries of the lower triangular part of T: for any nonzero entry of T, in the same data structure we also store the pairs of indices of the elements of B and B^t that give a nonzero contribution in the scalar product forming the entry; this task can be expensive since we have to investigate $\mathcal{O}(n^2/2)$ entries of the lower part of T and for each (i, j) entry, we have to individuate the nonzero pairs among the neq pairs of elements of the i–th and j–th columns of B^T;
- the computation of the symbolic Cholesky factorization of the sparse symmetric and positive definite matrix T by the Fortran package (version 0.3) of Ng and Peyton (included in the package LIPSOL, downloadable from *www.caam.rice.edu/~zhang/lipsol*); the multiple minimum degree reordering of Liu used to minimize the fill–ins in L_n and the supernodal block factorization enables to take advantage of the presence of the cache memory in modern computer architectures ([63]).

Thus, in the results, we consider the time for solving a nonlinear programming problem by interior point scheme combined with the method of multipliers divided in the *preprocessing time* and the time for computing the solution. We

observe that the preprocessing time is dependent on the strategy used to perform the matrix–matrix products needed in the method for computing T.
We refer as IP–MM, the interior point method with the method of multipliers as inner iterative solver.

4.2 Preconditioned conjugate gradient method

Another approach for the solution of the symmetric and indefinite system (21) that occurs at each iteration of an interior point method, is the preconditioned conjugate gradient (PCG) method (see e.g. [8], [20], [30], [31], [49], [65], [66], [58]).
The PCG method, with the preconditioning matrix $\bar{M}$, for the solution of the system (21)

$$M\boldsymbol{y} = \boldsymbol{b}$$

where M is given by (22) and $\boldsymbol{y} = (\Delta\boldsymbol{x}^t, \Delta\boldsymbol{\lambda}_1^t)^t$ and $\boldsymbol{b} = (\boldsymbol{c}^t, \boldsymbol{q}^t)^t$, is expressed as follows ([75, p. 199]):

Set: $\hat{\boldsymbol{r}}^{(0)} = \boldsymbol{b} - M\boldsymbol{y}^{(0)}$; *let* $\boldsymbol{d}^{(0)}$ *be the solution of* $\bar{M}\boldsymbol{d} = \hat{\boldsymbol{r}}^{(0)}$; *set:* $\boldsymbol{p}^{(0)} = \boldsymbol{d}^{(0)}$;

For $\nu = 0, 1, ...$ *until the convergence do*

- $\tilde{\beta}_\nu = {\boldsymbol{d}^{(\nu)}}^t \hat{\boldsymbol{r}}^{(\nu)} / {\boldsymbol{p}^{(\nu)}}^t M\boldsymbol{p}^{(\nu)}$
- $\boldsymbol{y}^{(\nu+1)} = \boldsymbol{y}^{(\nu)} + \tilde{\beta}_\nu \boldsymbol{p}^{(\nu)}$
- $\hat{\boldsymbol{r}}^{(\nu+1)} = \hat{\boldsymbol{r}}^{(\nu)} - \tilde{\beta}_\nu M\boldsymbol{p}^{(\nu)}$
- *let* $\boldsymbol{d}^{(\nu+1)}$ *be the solution of* $\bar{M}\boldsymbol{d} = \hat{\boldsymbol{r}}^{(\nu+1)}$
- $\hat{\beta}_\nu = {\boldsymbol{d}^{(\nu+1)}}^t \hat{\boldsymbol{r}}^{(\nu+1)} / {\boldsymbol{d}^{(\nu)}}^t \hat{\boldsymbol{r}}^{(\nu)}$
- $\boldsymbol{p}^{(\nu+1)} = \boldsymbol{d}^{(\nu+1)} + \hat{\beta}_\nu \boldsymbol{p}^{(\nu)}$

The basic conjugate gradient method results by choosing $\bar{M} = I$, the identity matrix.
The multiplication $M\boldsymbol{p}$ that occurs at each iteration ν, does not require the explicit computation of the matrix A of system (21); indeed, if we set the $n \times p$ matrix $C = (\nabla \boldsymbol{g}_2(\boldsymbol{x}), P_l^t, P_u^t)$, and the $p \times p$ diagonal matrices $\tilde{S} = \mathrm{diag}(\tilde{\boldsymbol{s}})$ and $\tilde{W} = \mathrm{diag}(\tilde{\boldsymbol{w}})$, respectively, then, the matrix A becomes $A = Q + C\tilde{S}^{-1}\tilde{W}C^t$.
The computation of $\boldsymbol{t} = M\boldsymbol{p}$, where $\boldsymbol{p} = (\boldsymbol{p}_1^t, \boldsymbol{p}_2^t)^t$, $\boldsymbol{t} = (\boldsymbol{t}_1^t, \boldsymbol{t}_2^t)^t$ ($\boldsymbol{p}_1, \boldsymbol{t}_1 \in \mathbb{R}^n$, $\boldsymbol{p}_2, \boldsymbol{t}_2 \in \mathbb{R}^{neq}$), can be carried out as

- $\boldsymbol{t}_1 \leftarrow C^t \boldsymbol{p}_1$
- $\hat{\boldsymbol{t}} \leftarrow \tilde{S}^{-1}\tilde{W}\boldsymbol{t}_1$
- $\boldsymbol{t}_1 \leftarrow C\hat{\boldsymbol{t}}$
- $\boldsymbol{t}_1 \leftarrow \boldsymbol{t}_1 + Q\boldsymbol{p}_1 + B\boldsymbol{p}_2$

- $\boldsymbol{t}_2 \leftarrow B^t \boldsymbol{p}_1$

Here $\hat{\boldsymbol{t}}$ is a temporary array of size p.
In this work, we consider the indefinite preconditioner introduced by Lukšan in [65]:

$$\bar{M} = \begin{pmatrix} \bar{A} & B \\ B^t & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ B^t \bar{A}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A} & 0 \\ 0 & -B^t \bar{A}^{-1} B \end{pmatrix} \begin{pmatrix} I & \bar{A}^{-1} B \\ 0 & I \end{pmatrix} \tag{26}$$

where $\bar{A}$ is a positive diagonal approximation of A. We refer to [65] for spectral properties of the conjugate gradient method with this preconditioner.
In this work the diagonal matrix $\bar{A}$ is chosen as $\bar{A} = \mathrm{diag}(\bar{a}_{ii})$ as follows

$$\bar{a}_{ii} = \begin{cases} a_{ii} = q_{ii} + \sum_{j=1}^{p} c_{ij}^2 \tilde{w}_j / \tilde{s}_j & \text{if } a_{ii} > 10^{-8} \\ 1.5 \cdot 10^{-8} & \text{otherwise.} \end{cases} \quad i = 1, ..., n \tag{27}$$

where $\tilde{w}_j$ and $\tilde{s}_j$, $j = 1, ..., p$ are the component of the vectors $\tilde{\boldsymbol{w}}$ and $\tilde{\boldsymbol{s}}$ respectively, the coefficients c_{ij}, $i = 1, ..., n$, $j = 1, ..., p$, are the entries of the $n \times p$ matrix C defined above and q_{ii}, $i = 1, ..., n$, are the diagonal entries of the matrix Q.
At each iteration ν of PCG algorithm, we have to solve the linear system of order $n + neq$,

$$\bar{M} \boldsymbol{d} = \hat{\boldsymbol{r}} \tag{28}$$

We partition the vectors $\boldsymbol{d}$ and $\hat{\boldsymbol{r}}$ as $\boldsymbol{d} = (\boldsymbol{d}_1^t, \boldsymbol{d}_2^t)^t$ and $\hat{\boldsymbol{r}} = (\hat{\boldsymbol{r}}_1^t, \hat{\boldsymbol{r}}_2^t)^t$ respectively, $(\boldsymbol{d}_1, \hat{\boldsymbol{r}}_1 \in \mathbb{R}^n,\ \boldsymbol{d}_2, \hat{\boldsymbol{r}}_2 \in \mathbb{R}^{neq})$.
In this work we consider two different methods to solve the system (28) when the coefficient matrix $\bar{M}$ is defined in (26). The first method exploits the block structure of the matrix (26) while the second one solves directly the system (28) by introducing a regularization technique on the preconditioning matrix $\bar{M}$ in order to assure that it admits a Cholesky–like factorization.
The two different techniques to compute the solution of system (28) produce different performances, especially for large scale problems.
In the first case, at the beginning of the PCG method we compute the symmetric positive definite matrix $T = B^t \bar{A}^{-1} B$ and its Cholesky factorization $T = L_{neq} L_{neq}^t$; then, taking into account of $\bar{M}^{-1}$ from (26), the solution of (28) can be determined by the following procedure

- $\boldsymbol{d}_1 \leftarrow \bar{A}^{-1} \hat{\boldsymbol{r}}_1$
- $\boldsymbol{d}_2 \leftarrow \hat{\boldsymbol{r}}_2 - B^t \boldsymbol{d}_1$
- $\hat{\boldsymbol{t}} \leftarrow -L_{neq}^{-1} \boldsymbol{d}_2$
- $\boldsymbol{d}_2 \leftarrow L_{neq}^{-t} \hat{\boldsymbol{t}}$
- $\boldsymbol{d}_1 \leftarrow \boldsymbol{d}_1 - \bar{A}^{-1} B \boldsymbol{d}_2$

Here $\hat{\boldsymbol{t}}$ is a temporary array of size neq.
As in the implementation of the method of multipliers, when $B^t\bar{A}^{-1}B$ is sufficiently sparse, a preprocessing routine executes:

- the formation of a data structure for storing the information needed to compute the matrix T;
- the determination of the minimum degree reordering of T and of its symbolic Cholesky factor.

For this last part and for computing the elements of L_{neq} we use the package of Ng and Peyton.
We observe that this approach can be more convenient with respect to the IP–MM method at two level:

- at the preprocessing phase, we have to compute the entries of the $neq \times neq$ matrix $B^t\bar{A}^{-1}B$ instead of the ones of the $n \times n$ matrix $A + \chi BB^t$;
- at solution phase, at any iterate of the inner solver, we have to solve positive definite linear systems with coefficient matrix $B^t\bar{A}^{-1}B$ instead of $A + \chi BB^t$ and $neq < n$.

Nevertheless, the formation of data structure phase can be expensive since we have to investigate $\mathcal{O}(neq^2/2)$ entries of the lower part of T and for each (i, j) entry, we have to individuate the nonzero pairs among the n pairs of elements of the i–th and j–th columns of B.
We refer as IP–PCG1, the interior point method with the preconditioned conjugate gradient as inner solver, with preconditioning matrix $\bar{M}$ as in (26), and the solution of the system (28) computed as described above.

The other method to compute the solution of the system (28), avoids the computation of the matrix–matrix product $B^T\bar{A}^{-1}B$.
We observe that the matrix $\bar{M}$ can be factorized in a Cholesky–like form

$$L_{n+neq} D L_{n+neq}^t, \tag{29}$$

where L_{n+neq} is a lower triangular matrix with diagonal entries equal to one and D is a nonsingular diagonal matrix. In order to reduce the fill–ins in the lower triangular factor, we can perform a minimum degree reordering of the matrix $\bar{M}$. But, it is not assured that the symmetrically permuted matrix $P\bar{M}P^t$ can be factorized in the Cholesky–like form.
Nevertheless, we can obtain a factorization in the form (29) if we use for the matrix $\bar{M}$ the regularization technique described in [2]; in other words, instead of using the preconditioner $\bar{M}$, we compute the factorization of

$$\bar{\bar{M}} = \bar{M} + \begin{pmatrix} R_1 & 0 \\ 0 & -R_2 \end{pmatrix}$$

where R_1 and R_2 are nonnegative diagonal matrices such that $P\bar{\bar{M}}P^T$ admits a factorization of the form (29). The computation of R_1 and R_2 can

be obtained during the factorization procedure. If a pivot d_i is too small ($|d_i| < 10^{-15} \max_{j<i} |d_j|$), we put $d_i = \sqrt{eps}$ if $1 \leq i \leq n$, or $d_i = -\sqrt{eps}$ if $n+1 \leq i \leq n+neq$, where eps is the machine precision.
The dynamic computation of the elements of R_1 and R_2 reduces the perturbation to a minimum. This approach is used in [8] for linear and quadratic programming problems with equality and box constraints.
The Cholesky–like factorization of $\bar{M}$ can be obtained by a modification of the Ng and Peyton package. In particular, we modify the subroutine PCHOL such that we compute $L_{n+neq} D L_{n+neq}^T$ with diagonal elements of L_{n+neq} equal to 1. Consequently, it is necessary to construct suitable subroutines (MMPYM and SMXPYM) to update the blocks of the factor L_{n+neq}, and to modify the subroutine BLKSVT for the computation of the solution of the system

$$L_{n+neq} D L_{n+neq}^t \boldsymbol{d} = \hat{\boldsymbol{r}}$$

The routines for performing the minimum degree reordering, for determining the supernodes and for the computation of the symbolic factor are unchanged. Consequently, the effectiveness of the package of Ng and Peyton due to a suitable use of the cache memory is maintained. This package, called BLKFCLT, introduced in [11], is downloadable from the website *http://dm.unife.it/blkfclt*.
We refer as IP–PCG2, the interior point method with the preconditioned conjugate gradient as inner solver, with preconditioning matrix $\bar{M}$ as in (26), and the solution of the system (28) computed by the package BLKFCLT.

5 Analysis of the convergence

5.1 Formulation of the algorithm

In this subsection we state the damped Newton interior point algorithm as described in the previous sections.
We refer this algorithm as IP–MM, IP–PCG1, IP–PCG2, depending on we use as inner solver the method of multipliers with Ng–Peyton routine, the conjugate gradient method with Lukšan preconditioner and Ng–Peyton routine or the conjugate gradient method with Lukšan preconditioner and BLKFCLT routine, respectively,
We also refer as IP–MA27, when we solve the inner system by the direct method implemented in MA27 routine.
The algorithm can be formulated as follows:

- set $\boldsymbol{v}^{(0)}$ s.t. $\tilde{\boldsymbol{s}}^{(0)} > 0$ and $\tilde{\boldsymbol{w}}^{(0)} > 0$;
- set the backtracking parameters $\beta, \theta \in (0,1)$ and the centrality conditions parameters τ_1 and τ_2 as in (20) with $\gamma_k = \frac{1}{2}$; set the tolerance $\epsilon_{\text{exit}} > 0$;
- for $k = 0, 1, ...$ until stopping rule is satisfied, do:
  - set $\mu_k \in [\mu_k^{(1)}, \mu_k^{(2)}]$, $\delta_k \in [0,1)$; $\sigma_k > \delta_k(1 + \frac{1}{2}\tau_2)$ with $\sigma_k + \delta_k < 1$;

– compute the solution $\Delta \boldsymbol{v}^{(k)}$ by solving the system (21) with a direct or an iterative process. In this last case, the stopping rule satisfies condition (16), that is:

$$\|\boldsymbol{r}\| \leq \max(5\epsilon_{\text{exit}}, \delta_k \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|)$$

where $\boldsymbol{r}$ is the inner current residual and it is computed as in (15);

– find α such that $\tilde{\boldsymbol{s}} = \tilde{\boldsymbol{s}}^{(k)} + \alpha \Delta \tilde{\boldsymbol{s}}^{(k)} > 0$ and $\tilde{\boldsymbol{w}} = \tilde{\boldsymbol{w}}^{(k)} + \alpha \Delta \tilde{\boldsymbol{w}}^{(k)} > 0$ (feasibility conditions), i.e. ($\hat{\theta} < 1$):

$$\alpha \equiv \alpha_k^{(1)} = \min \left(\min \left(\min_{\Delta \tilde{s}_i^{(k)} < 0} \frac{-\tilde{s}_i^{(k)}}{\Delta \tilde{s}_i^{(k)}}, \min_{\Delta \tilde{w}_i^{(k)} < 0} \frac{-\tilde{w}_i^{(k)}}{\Delta \tilde{w}_i^{(k)}} \right) \hat{\theta}, 1 \right) \quad (30)$$

– reduce eventually the parameter $\alpha_k^{(1)}$, by multiplying by a positive factor $\breve{\theta} < 1$, until the centrality conditions (18)–(19) are satisfied[9]. Denote $\alpha_k^{(2)}$ the obtained value;

– apply backtracking procedure [28]:
set $\alpha = \alpha_k{}^{(2)}$; $t = 0$;
while $\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha \Delta \boldsymbol{v}^{(k)})\| > (1 - \beta\alpha(1 - (\sigma_k + \delta_k)))\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$
or $t < t_{\max}$
$\alpha \leftarrow \theta\alpha$; $t \leftarrow t + 1$;
endwhile
Denote α_k the value of α at the final backtracking step;

– $\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)} + \alpha_k \Delta \boldsymbol{v}^{(k)}$

Here, the outer iterations stop when the outer residual $\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$ satisfies

$$\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \leq 10^{-8}$$

or when (see [91])

$$\frac{|\text{gap}|}{1 + |\text{gap}|} \leq 10^{-8}$$

where *gap* is the difference between the primal function $f(\boldsymbol{x})$ and the dual function

$$\begin{aligned} d(\boldsymbol{x}, \boldsymbol{\lambda}_1, \tilde{\boldsymbol{w}}) \quad = \quad & f(\boldsymbol{x}) - \boldsymbol{\lambda}_2^t \boldsymbol{g}_2(\boldsymbol{x}) - \boldsymbol{\lambda}_1^t \boldsymbol{g}_1(\boldsymbol{x}) + \boldsymbol{l}^t \boldsymbol{\lambda}_l - \boldsymbol{u}^t \boldsymbol{\lambda}_u - \nabla f(\boldsymbol{x})^t \boldsymbol{x} + \\ & + \begin{pmatrix} \boldsymbol{\lambda}_1^t & \boldsymbol{\lambda}_2^t \end{pmatrix} \begin{pmatrix} \nabla \boldsymbol{g}_1(\boldsymbol{x})^t \\ \nabla \boldsymbol{g}_2(\boldsymbol{x})^t \end{pmatrix} \boldsymbol{x} \end{aligned}$$

[9] In the experiments, we use an adaptive rule for $\hat{\theta}$ as in [4]; that is, if the result of the minimum in (30) is less then 1 we have

$$\hat{\theta} = \max\left(0.8, \min(0.9995, 1 - 100(\tilde{\boldsymbol{s}}^{(k)^t} \tilde{\boldsymbol{w}}^{(k)}))\right)$$

otherwise, if $\Delta \boldsymbol{v}^{(k)}$ does not bring the new iterate out of the feasible region, we set

$$\hat{\theta} = \max\left(0.8, 1 - 100(\tilde{\boldsymbol{s}}^{(k)^t} \tilde{\boldsymbol{w}}^{(k)})\right)$$

Furthermore, we set $\breve{\theta} = 0.5$.

5.2 Convergence of inexact Newton method for Karush–Kuhn–Tucker systems

The analysis of the convergence of the damped Newton interior point algorithm as described above, can be developed as an analysis of the convergence of inexact Newton method for solving Karush–Kuhn–Tucker systems.
Given $\epsilon \geq 0$, we define

$$\begin{aligned}\Omega(\epsilon) \quad = \quad & \Big\{ \boldsymbol{v} : 0 \leq \epsilon \leq \|\boldsymbol{H}(\boldsymbol{v})\|^2 \leq \|\boldsymbol{H}(\boldsymbol{v}^{(0)})\|^2, \text{ s. t.} \\ & \min_{i=1,p} \left(\tilde{S}\tilde{W}\boldsymbol{e}_p \right) \geq \frac{\tau_1}{2} \left(\frac{\tilde{\boldsymbol{s}}^t \tilde{\boldsymbol{w}}}{p} \right) \\ & \tilde{\boldsymbol{s}}^t \tilde{\boldsymbol{w}} \geq \frac{\tau_2}{2} \|\boldsymbol{H}_1(\boldsymbol{v})\| \Big\} \end{aligned} \tag{31}$$

$\Omega(\epsilon)$ is a closed set.
Let assume that the following conditions hold [33] (see also [36] and [28]):

- C1 in $\Omega(0)$, $f(\boldsymbol{x})$, $\boldsymbol{g}_1(\boldsymbol{x})$, $\boldsymbol{g}_2(\boldsymbol{x})$ are twice continuously differentiable; the gradients of the equality constraints are linearly independent and $H_1'(\boldsymbol{v})$ is Lipschitz continuous;
- C2 the sequences $\{\boldsymbol{x}^{(k)}\}$ and $\{\tilde{\boldsymbol{w}}^{(k)}\}$ are bounded;
- C3 for any $\Omega_{\boldsymbol{s}}$, the matrix $H'(\boldsymbol{v})$ is nonsingular. The set $\Omega_{\boldsymbol{s}}$ is a compact subset of $\Omega(0)$ where $\boldsymbol{s}$ is bounded away from zero.

The condition C3 is equivalent to the condition that the matrix M of (22) is nonsingular for any $\Omega_{\boldsymbol{s}}$.
In general, in literature, the condition C3 is replaced by a sufficient condition to assure that C3 holds.
For example, a sufficient condition is to require that, for any $\Omega_{\boldsymbol{s}}$, the matrix A is symmetric and positive definite on the null space of B^t and B^t is a full row rank matrix. Another sufficient condition is to require that, for any $\Omega_{\boldsymbol{s}}$, the matrices A and $B^t A^{-1} B$ are nonsingular[10].
The boundedness of the sequence $\{\boldsymbol{x}^{(k)}\}$ can be assured by enforcing box constraints $-l_i \leq x_i^{(k)} \leq l_i$ for sufficiently large $l_i > 0,\ i = 1, ..., n$.
The following result in ([10, Theor. 2]), assures the *boundedness of the iteration.*

Theorem 1. Let $\{\boldsymbol{v}^{(k)}\}$ be a sequence generated by the damped Newton interior point method.
If $\boldsymbol{v}^{(k)} \in \Omega(\epsilon)$, $\epsilon > 0$, then

[10]The inverse of the matrix M of (22), by the inversion of a matrix by partitioning [38, p. 161], is given by

$$M^{-1} = \begin{pmatrix} A^{-1} - A^{-1}B(B^tA^{-1}B)^{-1}B^tA^{-1} & A^{-1}B(B^tA^{-1}B)^{-1} \\ (B^tA^{-1}B)^{-1}B^tA^{-1} & -(B^tA^{-1}B)^{-1} \end{pmatrix}$$

(a) the sequences $\{\tilde{\boldsymbol{s}}^{(k)^t}\tilde{\boldsymbol{w}}^{(k)}\}$ and $\{\tilde{s}_i^{(k)}\tilde{w}_i^{(k)}\}$, $i = 1, \dots p$, are bounded above and below away from zero for any $k \geq 0$; the sequence $\{\|\boldsymbol{H}_1(\boldsymbol{v}^{(k)})\|\}$ is bounded above for any $k \geq 0$;

(b) if C1 and C2 hold, then $\{\boldsymbol{v}^{(k)}\}$ is bounded above and $\tilde{\boldsymbol{s}}^{(k)}$ and $\tilde{\boldsymbol{w}}^{(k)}$ are componentwise bounded away from zero;

(c) if C1, C2 and C3 hold, then the sequence of matrices $\{H'(\boldsymbol{v}^{(k)})\}$ is bounded and then, also the sequence of $\{H'(\boldsymbol{v}^{(k)})^{-1}\}$ is bounded;

(d) if C1, C2 and C3 hold, and if $\sigma_k + \delta_k < 1$, then the sequence $\{\Delta\boldsymbol{v}^{(k)}\}$ is bounded.

Hence, as a consequence of Theorem 1, we see that the *damping parameter is uniformly bounded away from zero.*
Since the final value of the damping parameter is obtained after satisfaction of the feasibility, the path–following conditions and the backtracking reduction, we separate the analysis into three steps.

1. (*Feasibility*) It is easy to see that $\alpha_k^{(1)}$ in (30) is bounded away from zero, i.e. $\alpha_k^{(1)} \geq \alpha^{(1)} > 0$, since, for any iteration k, $\tilde{s}_i^{(k)}$ and $\tilde{w}_i^{(k)}$ are bounded away from zero and $\Delta\tilde{s}_i^{(k)}$ and $\Delta\tilde{w}_i^{(k)}$ are bounded above, for $i = 1, ..., p$.

2. (*Path–following*) The following result in [10, Theor. 3] assures the existence of two positive numbers $\hat{\alpha}_k^{(2)}$ and $\check{\alpha}_k^{(2)}$ such that the centrality functions $\varphi(\alpha)$ and $\psi(\alpha)$ are nonnegative for $\alpha \in (0, \hat{\alpha}_k^{(2)}]$ and for $\alpha \in (0, \check{\alpha}_k^{(2)}]$ respectively.

   Theorem 2. Let $\{\boldsymbol{v}^{(k)}\}$ be a sequence generated by the damped Newton interior point method. Let us also assume $\sigma_k \in [\sigma_{\min}, \sigma_{\max}] \subset (0, 1)$ and $\delta_k \in [0, \delta_{\max}] \subset [0, 1)$, and

   $$\sigma_k > \delta_k(1 + \gamma_k\tau_2) \tag{32}$$

   Then, if $\varphi^{(k)}(0) \geq 0$, there exists a positive number $\hat{\alpha}_k^{(2)} > 0$, such that $\varphi^{(k)}(\alpha) \geq 0$ for all $\alpha \in (0, \hat{\alpha}_k^{(2)}]$.

   Then, if $\psi^{(k)}(0) \geq 0$, there exists a positive number $\check{\alpha}_k^{(2)} > 0$, such that $\psi^{(k)}(\alpha) \geq 0$ for all $\alpha \in (0, \check{\alpha}_k^{(2)}]$.

   From result in Theorem 1 (see the proof of [10, Theor. 3]), we have that

   $$\min\{\hat{\alpha}_k^{(2)}, \check{\alpha}_k^{(2)}, 1\} \in (0, 1] \geq \tilde{\alpha} > 0$$

   Then, $\alpha_k^{(2)} \geq \alpha^{(2)} = \min\{\alpha^{(1)}, \tilde{\alpha}\} > 0$.

   For sake of completeness, we report the result in [44] (also in [10]) that shows that the strict feasibility of the initial vectors $\tilde{\boldsymbol{s}}^{(0)} > 0$ and $\tilde{\boldsymbol{w}}^{(0)} > 0$ is sufficient to guarantee the nonnegativity of the centrality functions $\varphi(\alpha)$ and $\psi(\alpha)$ at each iterate k.

Theorem 3. Let $\varphi(\alpha)$ and $\psi(\alpha)$ be the centrality functions defined in (18) and (19); set τ_1 and τ_2 as in (20) and let be given a sequence of parameters $\{\gamma_k\}$ with

$$1 > \gamma_0 \geq \gamma_1 \geq ... \geq \gamma_k \geq \gamma_{k+1} \geq ... \geq \frac{1}{2}.$$

Furthermore, since $\tilde{\boldsymbol{s}}^{(0)} > 0$, $\tilde{\boldsymbol{w}}^{(0)} > 0$, then

$$\begin{aligned} \varphi^{(k)}(\alpha) &\geq 0 \qquad \text{for all} \quad \alpha \in (0, \hat{\alpha}_k^{(2)}] \\ \psi^{(k)}(\alpha) &\geq 0 \qquad \text{for all} \quad \alpha \in (0, \check{\alpha}_k^{(2)}] \end{aligned}$$

for any $k = 0, 1, ...$

3. (*Backtracking*) As shown in [10, Theor. 4], under the hypotheses of theorems 1 and 2, the *while loop* of the backtracking procedure of the damped Newton interior point algorithm, terminates in a finite number of steps (see also [35, Lemma 5.1]).

Thus, by P1 and P2, the damping parameter α_k is bounded below away from zero and the damped Newton interior point step, $\alpha_k \Delta \boldsymbol{v}^{(k)}$ satisfies the norm and the residual conditions of inexact Newton method.
Hence, we report here the convergence theorem for the damped Newton interior point method, based on the fundamental convergence theorem of inexact Newton method ([79, Theor. 6.7]).

Theorem 4. Suppose that the assumptions C1, C2 and C3 hold. Suppose that $\sigma_k \in [\sigma_{\min}, \sigma_{\max}] \subset (0, 1)$, $\delta_k \in [0, \delta_{\max}] \subset [0, 1)$, $\sigma_{\max} + \delta_{\max} < 1$ and $\sigma_k > \delta_k(1 + \gamma_k \tau_2)$. Then, the damped Newton interior point algorithm, with $\epsilon_{\text{exit}} = 0$, generates a sequence $\{\boldsymbol{v}^{(k)}\}$ such that:

(i) the sequence $\{\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|\}$ converges to zero and each limit point of the sequence $\{\boldsymbol{v}^{(k)}\}$ satisfies the Karush–Kuhn–Tucker conditions (3) for (1) and (2);

(ii) if the sequence $\{\boldsymbol{v}^{(k)}\}$ converges to $\boldsymbol{v}^*$ with $H'(\boldsymbol{v}^*)$ nonsingular matrix, $\sigma_k = \mathcal{O}(\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|^{\zeta})$, $0 < \zeta < 1$, and $\delta_k = \mathcal{O}(\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|)$, then there exists an index $\bar{k}$ such that $\alpha_k = 1$ for $k \geq \bar{k}$. Thus, the damped Newton interior point method has a superlinear local convergence.

Proof. (i) We have that the sequence $\{\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|\}$ is monotone, nonincreasing and bounded. Hence, this sequence has a limit $H^* \in \mathbb{R}$. If $H^* = 0$, we have the result. Suppose, by contradiction, that $H^* > 0$. Then the sequence $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$ with $\epsilon = (H^*)^2 > 0$. Since $\{\boldsymbol{v}^{(k)}\}$ is bounded above, then it possesses limit points (Bolzano–Weierstrass Theorem [1, p. 54]). Let $\boldsymbol{v}^*$ be one of these limit points. Then, there is a subsequence of $\{\boldsymbol{v}^{(k)}\}$ that converges to $\boldsymbol{v}^*$. Denoting this converging subsequence by $\{\boldsymbol{v}^{(k_i)}\}$, we have that $\boldsymbol{v}^{(k_i)} \to \boldsymbol{v}^*$ as $k_i \to \infty$. Since $\boldsymbol{H}(\boldsymbol{v})$ is continuous, it follows that $\boldsymbol{H}(\boldsymbol{v}^{(k_i)}) \to \boldsymbol{H}(\boldsymbol{v}^*)$ and $\|\boldsymbol{H}(\boldsymbol{v}^{(k_i)})\| \to \|\boldsymbol{H}(\boldsymbol{v}^*)\|$. But $\|\boldsymbol{H}(\boldsymbol{v}^{(k_i)})\| \to H^*$. Therefore, $\|\boldsymbol{H}(\boldsymbol{v}^*)\| = H^*$.

This implies that $\boldsymbol{v}^*$ belongs to $\Omega(\epsilon)$, $\epsilon > 0$; then, the matrix $H'(\boldsymbol{v}^*)$ is invertible. Consequently by the Theorem 6.7 in [79, p. 70] (see also Theorem 6.1 in [35]), we deduce that $\boldsymbol{H}(\boldsymbol{v}^*) = 0$. This contradicts our assumptions that $H^* > 0$. Hence, the sequence $\{\boldsymbol{H}(\boldsymbol{v}^{(k)})\}$ must converge to zero.
Moreover, the limit point $\boldsymbol{v}^*$ satisfies $\boldsymbol{H}(\boldsymbol{v}^*) = 0$ and $(\tilde{\boldsymbol{s}}^*, \tilde{\boldsymbol{w}}^*) \geq 0$, i.e., $\boldsymbol{v}^*$ satisfies the KKT conditions for the problem (1).
(ii) See part (c) of the proof of Theorem 5 in [10] (see also [32]). □

5.3 On the global convergence

In this subsection, we briefly make some remarks on cases of a global convergence failure of the damped Newton interior point method. Obviously, when it happens, we have that at least one of the sufficient conditions C1–C3 for the convergence is not satisfied.
We will check some small examples.
Let us consider, for instance, the example in [93] where it is stressed that algorithms which uses Newton direction could fail:

$$\begin{aligned} &\min x \\ &x^2 \geq -a \\ &x \geq b \end{aligned}$$

As pointed out in [33, Example 3.1], when the initial point is taken as $x^{(0)} = -3$, $\tilde{\boldsymbol{w}}^{(0)} = \boldsymbol{e}_2$ ($a = -1$, $b = 1$), the damped Newton interior point method generates a sequence which is not convergent to the optimal solution $x^* = 1$. In this case, in [33], it has been observed that the sufficient condition on the boundedness of the sequence of the inequalities' multipliers $\{\tilde{\boldsymbol{w}}^{(k)}\}$ is not satisfied.
Indeed, the values of $\tilde{\boldsymbol{w}}^{(k)}$ increase and the values of $\tilde{\boldsymbol{s}}^{(k)}$ become very small with respect to $(\tilde{\boldsymbol{s}}^{(k)^t}\tilde{\boldsymbol{w}}^{(k)})/p$. This is a case where the sequence generated by the damped Newton interior point iteration tends to a solution which does not belong to the feasible region.
On the other hand, if we start with $x^{(0)} = 3$, $\tilde{\boldsymbol{w}}^{(0)} = \boldsymbol{e}_2$, the sequence $\{\tilde{\boldsymbol{w}}^{(k)}\}$ is bounded and the algorithm converges to the solution.
Thus, an increasing of the values of the sequence $\{\tilde{\boldsymbol{w}}^{(k)}\}$ is a detection that the sequence of the solution could not converge and then, another choice of the initial point is recommended.
Moreover, we observe that the sufficient condition (C4) in [36] of linear independence of the gradients of the active constraints is violated here, either if we start from a *bad* or a *good* initial point. Thus, the condition on the boundedness of the inequalities' multipliers is more general respect to the condition (C4) in [36].
An example of large scale nonlinear programming problem for which the damped Newton interior point method gives the same behaviour is still given in [33, Example 3.2].
A way to compute a suitable starting point could be the one to execute some steps of the gradient projection method (e.g. see [64, §11.4, p. 330]), before to

ν	k	x	s	r_l	λ_2	λ_l	$\tilde{\boldsymbol{s}}^t\tilde{\boldsymbol{w}}/2$	$\|\boldsymbol{H}(\boldsymbol{v})\|$
0	-	3.0	1.0	1.0	1.0	1.0	1.0	10.58
1	-	0.69	1.38	0.63	$1.0 \cdot 10^{-5}$	1.31	0.41	2.3
2	0	1.58	0.90	0.12	$1.0 \cdot 10^{-5}$	1.11	$6.7 \cdot 10^{-2}$	0.76
-	1	1.46	0.95	0.34	0.26	0.22	0.16	0.33
-	2	1.16	0.19	0.11	0.31	0.25	$4.3 \cdot 10^{-2}$	0.17
-	3	1.05	$3.8 \cdot 10^{-2}$	$3.3 \cdot 10^{-2}$	0.34	0.26	$1.1 \cdot 10^{-2}$	$6.3 \cdot 10^{-2}$
-	4	1.01	$7.6 \cdot 10^{-3}$	$8.0 \cdot 10^{-3}$	0.36	0.27	$2.4 \cdot 10^{-3}$	$1.6 \cdot 10^{-2}$
-	5	1.00	$1.5 \cdot 10^{-3}$	$1.7 \cdot 10^{-3}$	0.36	0.27	$5.0 \cdot 10^{-4}$	$3.5 \cdot 10^{-3}$
-	6	1.00	$1.5 \cdot 10^{-4}$	$1.7 \cdot 10^{-4}$	0.37	0.27	$5.1 \cdot 10^{-5}$	$3.6 \cdot 10^{-4}$

Table 2: Polyalgorithm for Example 3.1 in [33]

start with the damped Newton interior point method.
In Table 2, we report the number of iterations and the values of the primal and dual variables of the sequence generated by the *polyalgorithm* composed by the gradient projection method, with Armijo backtracking procedure for the merit function $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ and, by the damped Newton interior point method for the Example 3.1 in [33].
The result shows that only two iterations ν of the gradient projection method are necessary *to enter* in a *good* region[11] for the damped Newton interior point method.

Finally, we consider the Example 3 of nonlinear programs in [17]

$$\begin{array}{l} \min \frac{1}{3}(x-1)^3 + x \\ x \geq 0 \end{array}$$

where it is stressed that Newton iteration with decrease of the merit function $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ fails, since the Newton direction becomes orthogonal to the gradient of the merit function.
If we indicate ϑ_k the angle between the opposite of the direction of the damped Newton interior point method and the gradient of the least squares merit function $\Upsilon(\boldsymbol{v}) = \|\boldsymbol{H}(\boldsymbol{v})\|^2$, at the iteration k, in [43] is shown that,

$$\cos\vartheta_k \geq \frac{1-\eta_k}{2K(H'(\boldsymbol{v}^{(k)}))} \tag{33}$$

where $K(H'(\boldsymbol{v}^{(k)})) = \|H'(\boldsymbol{v}^{(k)})\| \cdot \|H'(\boldsymbol{v}^{(k)})^{-1}\|$ indicates the condition number of the Jacobian matrix at the point $\boldsymbol{v}^{(k)}$ and η_k is the forcing term.

[11] We refer *good region* as the set of starting points for which the method generates a sequence of point convergent to a solution. This is the definition of *attraction basin* in [79, p. 43].
For example, in the case of nonlinear systems, when the function satisfies a Lipschitz continuous condition, the bound of the attraction basin for Newton method depends on the Lipschitz constant ([79, p. 44]). When the function satisfies an *affine invariant* condition, see [26, p. 88] and [43] for bounds of attraction basins of Newton and inexact Newton methods, respectively. In this section, we do not develop an analogous analysis for nonlinear systems with restriction on the sign of some variables such as the Karush–Kuhn–Tucker systems.

We report, here, the simple proof of (33). Indeed, by squaring (13) we obtain:

$$2(\Delta \boldsymbol{v}^{(k)})^t H'(\boldsymbol{v}^{(k)})^t \boldsymbol{H}(\boldsymbol{v}^{(k)}) + \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|^2 + \|H'(\boldsymbol{v}^{(k)})\Delta \boldsymbol{v}^{(k)}\|^2 \leq \eta_k^2 \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|^2$$

Hence

$$\Delta \boldsymbol{v}^{(k)^t} \nabla \Upsilon(\boldsymbol{v}^{(k)}) = 2(\Delta \boldsymbol{v}^{(k)})^t H'(\boldsymbol{v}^{(k)})^t \boldsymbol{H}(\boldsymbol{v}^{(k)}) \leq (\eta_k^2 - 1)\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|^2$$

But,

$$\begin{aligned} \|\Delta \boldsymbol{v}^{(k)}\| &= \|H'(\boldsymbol{v}^{(k)})^{-1} \left((\boldsymbol{H}(\boldsymbol{v}^{(k)}) + H'(\boldsymbol{v}^{(k)})\Delta \boldsymbol{v}^{(k)}) - \boldsymbol{H}(\boldsymbol{v}^{(k)}) \right) \| \\ &\leq \|H'(\boldsymbol{v}^{(k)})^{-1}\| \left(\|\boldsymbol{H}(\boldsymbol{v}^{(k)}) + H'(\boldsymbol{v}^{(k)})\Delta \boldsymbol{v}^{(k)}\| + \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \right) \\ &\leq \|H'(\boldsymbol{v}^{(k)})^{-1}\|(\eta_k + 1)\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \end{aligned}$$

and

$$\|\nabla \Upsilon(\boldsymbol{v}^{(k)})\| = \|2H'(\boldsymbol{v}^{(k)})^t \boldsymbol{H}(\boldsymbol{v}^{(k)})\| \leq 2\|H'(\boldsymbol{v}^{(k)})\|\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$$

Thus

$$\begin{aligned} \cos \vartheta_k &= \frac{-\nabla \Upsilon(\boldsymbol{v}^{(k)})^t \Delta \boldsymbol{v}^{(k)}}{\|\nabla \Upsilon(\boldsymbol{v}^{(k)})\|\|\Delta \boldsymbol{v}^{(k)}\|} \\ &\geq \frac{1 - \eta_k^2}{2\|H'(\boldsymbol{v}^{(k)})\|\|H'(\boldsymbol{v}^{(k)})^{-1}\|(1 + \eta_k)} = \frac{1 - \eta_k}{2K(H'(\boldsymbol{v}^{(k)}))} \end{aligned}$$

then, we have (33).

If, it happens that, the matrix $H'(\boldsymbol{v}^{(k)})$ becomes *nearly* singular, then condition C3 does not hold and then, the condition number might be not bounded since the sequences $\{\|H'(\boldsymbol{v}^{(k)})\|\}$ and/or $\{\|H'(\boldsymbol{v}^{(k)})^{-1}\|\}$ are not bounded. This is the case of this example, as shown in Table 3. The starting point of the example is $x = 2$, $\lambda_l = 1$ and $r_l = 1$ and the used backtracking rule is the one in [28].
The results of tables 2 and 3 have been obtained by a Matlab code which solves directly the inner linear system that occurs at each Newton iteration. The forcing term η_k is equal to $\eta_k = 1 - \alpha_k(1 - \sigma_k)$.
In the following subsection, we consider an alternative explanation of the approaching of the Newton iteration to singular points.

5.4 On Newton's iteration and singular points

In the analysis of the damped Newton interior point method, we point out some remarks regarding the reasons of global convergence failures of Newton's iteration which can be explained by considering the differential equation associated to the nonlinear problem.
Indeed, in order to extend the domain of convergence of iterative methods for the solution of systems of nonlinear equations, many authors proposed methods which associated a differential equation to the nonlinear system, and found

k	x	$\|\boldsymbol{H}(\boldsymbol{v})\|$	α	$1-\eta$	$\cos\vartheta$	$K(H'(\boldsymbol{v}))$
0	2	1.73205	0.80	0.48	0.76	2.88
1	1.30667	0.844158	$5.3\cdot10^{-7}$	$3.2\cdot10^{-7}$	$2.7\cdot10^{-4}$	9590.65
2	0.49322	0.825476	$3.3\cdot10^{-6}$	$2.0\cdot10^{-6}$	$8.5\cdot10^{-4}$	3082.29
3	0.49535	0.825476	$3.1\cdot10^{-5}$	$1.8\cdot10^{-5}$	$2.0\cdot10^{-3}$	1309.26
4	0.50103	0.825475	$2.0\cdot10^{-4}$	$1.2\cdot10^{-4}$	$6.5\cdot10^{-3}$	398.28
5	0.51249	0.825473	$9.4\cdot10^{-4}$	$5.6\cdot10^{-4}$	$1.5\cdot10^{-2}$	168.49
6	0.53582	0.825406	$3.8\cdot10^{-3}$	$2.3\cdot10^{-3}$	$3.2\cdot10^{-2}$	80.27
7	0.58411	0.825049	$1.4\cdot10^{-2}$	$8.4\cdot10^{-3}$	$6.0\cdot10^{-2}$	42.08
8	0.68787	0.823723	$2.0\cdot10^{-2}$	$1.2\cdot10^{-2}$	$8.7\cdot10^{-2}$	28.22
9	0.80922	0.821058	$2.7\cdot10^{-3}$	$1.6\cdot10^{-3}$	$4.7\cdot10^{-2}$	51.66
10	0.84640	0.817953	$5.1\cdot10^{-4}$	$3.1\cdot10^{-4}$	$1.7\cdot10^{-2}$	139.76
11	0.86634	0.817029	$2.6\cdot10^{-6}$	$1.6\cdot10^{-6}$	$1.4\cdot10^{-3}$	1713.55
12	0.86504	0.81698	$3.1\cdot10^{-8}$	$1.8\cdot10^{-8}$	$1.3\cdot10^{-4}$	18551.1
13	0.86488	0.81698	$8.5\cdot10^{-10}$	$5.1\cdot10^{-10}$	$2.9\cdot10^{-5}$	83680.8
14	0.86490	0.81698	$1.3\cdot10^{-10}$	$7.9\cdot10^{-11}$	$8.9\cdot10^{-6}$	$2.69\cdot10^{5}$
15	0.86491	0.81698	$8.7\cdot10^{-13}$	$5.2\cdot10^{-13}$	$9.4\cdot10^{-7}$	$2.56\cdot10^{6}$

Table 3: Newton interior point method for Example 3 in [17]

the roots of the nonlinear systems by integrating numerically the differential equation (see e.g. [21], [48], [22], [98], [88], [100], [59], [60], [37]).
Let us consider the following Cauchy problem for nonlinear differential equations:

$$\dot{\boldsymbol{u}}(t) = -\Phi(\boldsymbol{u}(t))\boldsymbol{F}(\boldsymbol{u}(t)) \qquad 0 \le t < \infty; \qquad \boldsymbol{u}(0) = \boldsymbol{u}^{(0)} \tag{34}$$

If we apply Euler's method to the differential equation (34) with $\Phi(\boldsymbol{u}(t)) = F'(\boldsymbol{u}(t))^{-1}$ then we obtain the damped Newton method where the damping parameter is the time step Δt.
Thus, if we set $\Phi(\boldsymbol{u}(t)) = F'(\boldsymbol{u}(t))^{-1}$, then the differential problem (34) describes the *continuous Newton method*; furthermore, if we set $\Phi(\boldsymbol{u}(t)) = I$ or $\Phi(\boldsymbol{u}(t)) = F'(\boldsymbol{u}(t))^t$ then we have the method of the most rapid slope (continuous steepest descent method) or the continuous gradient method respectively. Here I denotes the identity operator.
Continuous analogs of iterative methods can usually be obtained easier. If a convergence theorem is proved for a continuous method, that is, for the Cauchy problem for a differential equation, then, we can construct various finite difference schemes for the solution of this Cauchy problem. These difference schemes give discrete methods for the solution of $\boldsymbol{F}(\boldsymbol{u}) = 0$. For instance, the method of Euler and Runge–Kutta can be used. Applications and modifications of continuous methods have been and are deeply studied.
Let us consider the following nonlinear system

$$\boldsymbol{F}(\boldsymbol{u}) \equiv \begin{pmatrix} u_1 \\ 10u_1/(u_1+0.1) + 2u_2^2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{35}$$

This example has been firstly proposed by Powell in [78], where the author proved that the Newton's method, starting from the initial point $\boldsymbol{u}^{(0)} = (3,1)^t$, does not converge to the unique solution $\boldsymbol{u}^* = (0,0)^t$, when the step length is chosen as the local minimizer of the least squares function $1/2\|\boldsymbol{F}(\boldsymbol{u})\|^2$.

Indeed, the iterative process leads to the point $\hat{\boldsymbol{u}} = (1.8016, 0.0000)^t$, which is neither a solution of the system (35) nor a stationary point of the least squares merit function, as showed in [17]. Moreover, the vector $\boldsymbol{F}(\hat{\boldsymbol{u}})$ does not belong to the null space of the Jacobian matrix of $\boldsymbol{F}$,

$$F'(\boldsymbol{u}) = \begin{pmatrix} 1 & 0 \\ 1/(u_1+0.1)^2 & 4u_2 \end{pmatrix}$$

computed in the points of the $(u_1, 0)^t$, where it is singular.
This behaviour of Newton's iteration can be easily explained if we consider the associated differential problem (34) with $\Phi(\boldsymbol{u}(t)) = F'(\boldsymbol{u}(t))^{-1}$, that is

$$F'(\boldsymbol{u}(t))^{-1} = \begin{pmatrix} 1 & 0 \\ -1/4u_2(t)(u_1(t)+0.1)^2 & 1/(4u_2(t)) \end{pmatrix}$$

The system (34) for the problem (35) is given by

$$\begin{aligned} \frac{du_1}{dt}(t) &= -u_1(t) \\ \frac{du_2}{dt}(t) &= \frac{u_1(t)}{4u_2(t)(u_1(t)+0.1)^2} - \frac{1}{4u_2(t)}\left(\frac{10u_1(t)}{u_1(t)+0.1} + 2u_2(t)^2\right) \end{aligned}$$

By calculations, it yields to the equation

$$\frac{du_2(t)}{du_1(t)} = \frac{1}{4u_2(t)}\left(-\frac{1}{(u_1(t)+0.1)^2} + \frac{10}{u_1(t)+0.1} + \frac{2u_2(t)^2}{u_1(t)}\right)$$

whose solution, dependent on the initial point $\boldsymbol{u}(0) = (u_1(0), u_2(0))^t \equiv (u_1^{(0)}, u_2^{(0)})^t$, is given by

$$u_2(t) = \pm\sqrt{u_1(t)\left(-\frac{5}{u_1(t)+0.1} + \frac{5}{u_1^{(0)}+0.1} + \frac{{u_2^{(0)}}^2}{u_1^{(0)}}\right)} \tag{36}$$

In Figure 2, we indicate the solution curves (36) in the phase plane (u_1, u_2) (left figure) and an enlargement of the phase plane where the circles indicates the first three iterates of Newton's method when the step length is chosen as local minimizer of the least squares function (right figure).
We note that the Newton step $\Delta\boldsymbol{u}^{(k)} = -F'(\boldsymbol{u}^{(k)})^{-1}\boldsymbol{F}(\boldsymbol{u}^{(k)})$ is tangent to the curve (36) at the curve point $\boldsymbol{u}^{(k)}$.
A same behaviour of Newton's iteration, we obtain when we use Armijo or Eisenstat and Walker ([35]) backtracking strategies.
Indeed, we observe that, at the first iterate, the step is shortened by a the value of the damping parameter approximately equal to 0.395 and this leads the next iterate very close to the u_1 axis, where the Jacobian matrix is singular.
This yields a Newton step almost orthogonal to the axis and that the first local minimizer of the merit function along this direction is very close to the previous iterate, such that the step length is of order 10^{-3}.

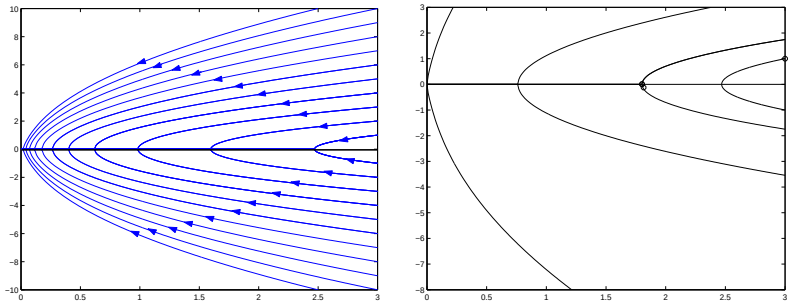


Figure 2: Powell example: solution curves and first three Newton's iterates

At the next iterate the situation is the same, and the sequence sticks to one of the solution curves and stagnates in a neighborhood of the point $\hat{\boldsymbol{u}}$.
This behaviour can be explained following [47], by defining the set of the singular points

$$\Omega = \{\boldsymbol{u} \in \mathbb{R}^n \,:\, F'(\boldsymbol{u}) \text{ is singular}\}$$

and a Lyapunov's function $V(\boldsymbol{u})$ which is a continuous and differentiable function, always positive, except in Ω where it is equal to zero. For instance, we can set

$$V(\boldsymbol{u}) = \frac{1}{2}\left(\det F'(\boldsymbol{u})\right)^2$$

Here $\det F'(\boldsymbol{u})$ denotes the determinant of the Jacobian matrix $F'(\boldsymbol{u})$.
If we have $\dot{V}(\boldsymbol{u}) < 0$ in some domain containing Ω, then the function $V(\boldsymbol{u})$ is decreasing, with respect to the variable t, on the solution curves of the differential equation. Thus, the function $V(\boldsymbol{u})$, with increasing values of t, will be able to become arbitrarily small on $\boldsymbol{u}(t)$, that is to say that, the distance between $\boldsymbol{u}(t)$ and Ω will become arbitrarily small. Thus the solutions of the differential equation, which are defined both at the left and at the right of Ω, are directed, from the right or from the left, towards the points of Ω. These points of Ω are called *end points* ([53, p. 61]). This is the case of the Powell example.
Indeed, we can see that in the example, we have

$$\dot{V}(\boldsymbol{u}) = \nabla_{\boldsymbol{u}} V(\boldsymbol{u})^t \left(\frac{du_1}{dt}, \frac{du_2}{dt}\right)^t = -4\frac{10u_1(t)^2 + 2u_2(t)^2(u_1(t)+0.1)^2}{(u_1(t)+0.1)^2} \leq 0$$

If $\dot{V}(\boldsymbol{u}) > 0$ in some domain containing Ω, then the function $V(\boldsymbol{u})$, with increasing values of t, will be arbitrarily large on $\boldsymbol{u}(t)$. Thus the solutions of the differential equation, which are defined both at the left and at the right of Ω, are diverging from the points of Ω. These points of Ω are called *initial points* ([53, p. 61]).
Now we consider the example in [47].

The function $\boldsymbol{F}(\boldsymbol{u})$ is defined as

$$\boldsymbol{F}(\boldsymbol{u}) = \begin{pmatrix} u_1 + u_2 - 5 \\ u_1 u_2 - 4 \end{pmatrix}$$

The inverse of the Jacobian of $\boldsymbol{F}(\boldsymbol{u})$ is given by

$$F'(\boldsymbol{u})^{-1} = \frac{1}{u_1 - u_2} \begin{pmatrix} u_1 & -1 \\ -u_2 & 1 \end{pmatrix}$$

and the solution of $\boldsymbol{F}(\boldsymbol{u}) = 0$ are $\boldsymbol{u}^* = (1,4)^t$ and $\tilde{\boldsymbol{u}}^* = (4,1)^t$.
By proceeding as before, the associated differential system (34), with initial conditions $\boldsymbol{u}(0) = (u_1(0), u_2(0))^t \equiv (u_1^{(0)}, u_2^{(0)})^t$, is:

$$\begin{aligned} \frac{du_1}{dt}(t) &= -\frac{1}{u_1(t) - u_2(t)} \left(u_1(t)^2 - 5u_1(t) + 4 \right) \\ \frac{du_2}{dt}(t) &= -\frac{1}{u_1(t) - u_2(t)} \left(-u_2(t)^2 + 5u_2(t) - 4 \right) \end{aligned}$$

In the phase plane we have

$$\frac{du_2(t)}{du_1(t)} = -\frac{(u_2(t) - 4)(u_2(t) - 1)}{(u_1(t) - 4)(u_1(t) - 1)}$$

Set

$$K = \frac{u_1^{(0)} - 1}{u_1^{(0)} - 4} \frac{u_2^{(0)} - 1}{u_2^{(0)} - 4}$$

the solutions of the differential system are a continuum of hyperbolas

$$(1 - K)u_1(t)u_2(t) + (4K - 1)(u_1(t) + u_2(t)) + (1 - 16K) = 0$$

For $K = 0$ and $K = \infty$, the hyperbolas degenerates in two pairs of linear functions $(u_1(t) - 1)(u_2(t) - 1) = 0$ and $(u_1(t) - 4)(u_2(t) - 4) = 0$, respectively.
In Figure 3, the solution curves in the phase plane are plotted and the arrows indicates the direction of the trajectories for increasing values of t.
In this example, the set Ω of singular points is formed by the line $u_1(t) = u_2(t)$ which is also plotted in Figure 3.
Defining the Lyapunov's function as before, and let $\mathcal{A}$ and $\mathcal{B}$ be the points of Ω, $(1,1)$ and $(4,4)$ respectively; we can deduce that all the points of Ω included between $\mathcal{A}$ and $\mathcal{B}$ (except $\mathcal{A}$ and $\mathcal{B}$) are initial points, while the ones outside the interval $[\mathcal{A}, \mathcal{B}]$ are end points.
The points $\mathcal{A}$ and $\mathcal{B}$ are singular points for which the system

$$F'(\boldsymbol{u})\Delta\boldsymbol{u} = -\boldsymbol{F}(\boldsymbol{u})$$

where $F'(\boldsymbol{u})$ and $\boldsymbol{F}(\boldsymbol{u})$ are computed in $\mathcal{A}$ or in $\mathcal{B}$, has solution.
In this case, the derivative of $V(\boldsymbol{u})$ on the solutions of the differential equation has nonconstant sign when it crosses Ω and therefore the paths which approach

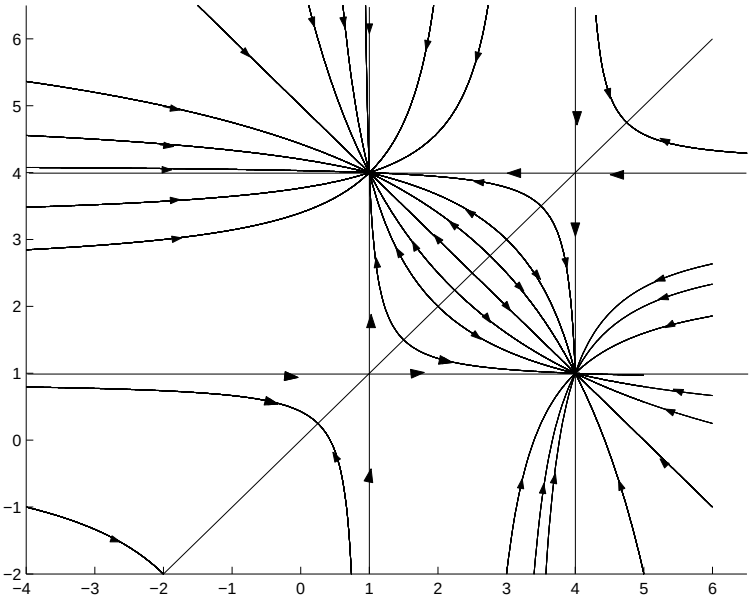


Figure 3: Solution curves and singular points of the example in [47]

from the left, can go away from it on the right. That was to be expected because we can extend the solution in Ω and therefore there must be continuity in the direction of the paths when Ω is crossed (see [47]).
We call $\mathcal{A}$ and $\mathcal{B}$ *cross points*.
We remark, that in the points close to Ω, the differential system becomes *stiff*[12], and then it is difficult to follow the trajectories of the solutions by Euler's method, that is by Newton's method.
Moreover, we observe that if we follow the trajectories of the solutions in the points of the region defined by

$$\mathcal{S} = \{(u_1, u_2) \,:\, u_1 \geq 4,\, u_2 \geq 4\} \cup \{(u_1, u_2) \,:\, u_1 \leq 1,\, u_2 \leq 1\}$$

it is not possible to reach any of the solutions $\boldsymbol{u}^*$ or $\tilde{\boldsymbol{u}}^*$.
We also observe that, when we reach end points of Ω following a trajectory of the solution of the differential system, it is not possible to go away from Ω. To avoid this drawback, Branin, in [13] (see also [85]), has suggested to change the differential system (34), where $\Phi(\boldsymbol{u}(t)) = F'(\boldsymbol{u}(t))^{-1}$, with

$$\dot{\boldsymbol{u}}(t) = F'(\boldsymbol{u}(t))^{-1}\boldsymbol{F}(\boldsymbol{u}(t)) \qquad 0 \leq t < \infty; \qquad \boldsymbol{u}(0) = \boldsymbol{u}^{(0)} \tag{37}$$

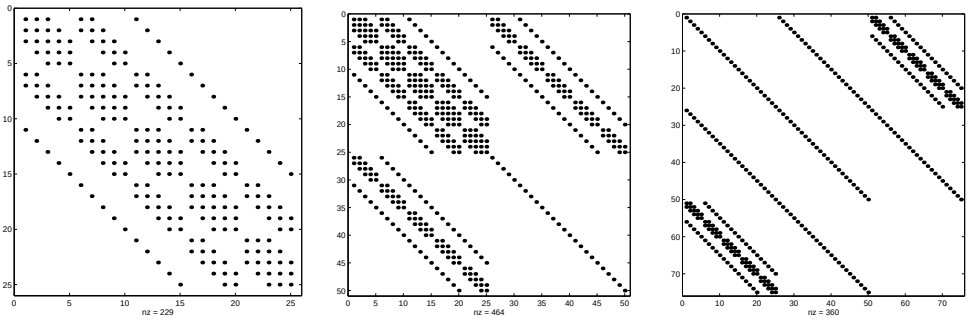
In this way it is possible to go away from Ω because end points become initial points. By the way, it is necessary to keep in mind that other points can become

[12] See e.g. [61, p. 228] for definition of *stiffness*.

end points for the system (37), but they can be extraneous to the differential system (34).

6 Numerical experiments

In order to evaluate the effectiveness of the damped Newton interior point method with different inner solvers, Fortran 90 codes, implementing the method, have been carried out on a workstation HP zx6000 with an Intel Itanium2 processor 1.3 GHz with 2Gb of RAM and they have been compiled with a "+O3" optimization level of the HP compiler.
We consider a set of nonlinear programming test problems arising from the discretization with finite difference formulae of boundary and distributed elliptic control problems ([67], [68], [71], see also [69]).
In these optimal control problems the functional is quadratic, the partial differential system is given by a weakly nonlinear elliptic equation with Dirichlet and/or Neumann conditions on the unit square $(0,1) \times (0,1)$ and the state and the control variables are subject to box constraints. It yields to a nonlinear programming problem with quadratic functional, nonlinear equality constraints and box constraints. The Hessian of the objective function and the Jacobian of the constraints are large and sparse.
Furthermore, the matrix $\nabla \boldsymbol{g}_2(\boldsymbol{x}) S^{-1} \Lambda_2 \nabla \boldsymbol{g}_2(\boldsymbol{x})^t + P_l^t R_l^{-1} \Lambda_l P_l + P_u^t R_u^{-1} \Lambda_u P_u$ is diagonal and the computation of this matrix in the block A of (21) is inexpensive.
In Table 4, we report the references of the test problems shown in the tables.
The symbol N denotes the number of grid points of the discretization along each axis. We remark that the size of any problem depends on the value of N.
In tables 5 and 6, we report the number of primal variables n, the number of equality neq constraints, the numbers nl and nu of variables bounded below and above, the number of nonzero entries nnzjac and nnzhess of the matrices B and Q respectively.
In the experiments, we set the starting point of the damped Newton interior point method as follows: the initial values for the multipliers and for the slack variables are set to 1 while the value $x_i^{(0)}$, $i = 1, ..., n$ are set equal to zero if the i–th component x_i is a free variable, equal to $(u_i + l_i)/2$ if x_i is bounded above and below, and equal to $u_i - 1$ or $l_i + 1$ if x_i is bounded above or below respectively. Only for the test problem P2-6, the first $n/2$ initial values for $\boldsymbol{x}^{(0)}$ are set equal to 6 and the last $n/2$ initial values are set equal to 1.8, as suggested by Mittelmann in [71]. Moreover, for the codes employing an iterative method as inner solver, the initial value of the inner iterations has been fixed equal to the null vector.
All the results in this section have been obtained with the choice $\mu_k = \mu_k^{(1)} = \tilde{\boldsymbol{s}}^{(k)^t} \tilde{\boldsymbol{w}}^{(k)}/p$.
Furthermore, the maximum value of inner iterations has been set equal to 15 for the IP–MM code, to neq for IP–PCG1 and to $n + neq$ for IP–PCG2.
We set the backtracking parameters $\theta = 0.5$, $\beta = 10^{-4}$ while the forcing term



matrix $A + \chi BB^t$ matrix B^tB preconditioner $\bar{M}$

Figure 4: Sparsity pattern of the matrices factorized by the codes

parameters σ_k, δ_k are chosen as in the following: set

$$\delta_{\max} = \frac{0.8}{1 + 0.5\frac{\tau_2\sqrt{2}}{\min(1,\tau_2)}};\ \sigma_{\max} = \frac{\delta_{\max} 0.5\tau_2\sqrt{2}}{\min(1,\tau_2)} \cdot 1.1$$

we have for the initial value $\delta_0 = \min(\delta_{\max}, 0.8 \cdot \|\boldsymbol{H}(\boldsymbol{v}^{(0)})\|)$ and for the iterations k, $k > 1$, we have

$$\delta_k = \min\left(\delta_{\max}, \max\left(5.0 \cdot 10^{-5}, \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|, 0.5 \cdot \frac{\|\boldsymbol{H}_1(\boldsymbol{v}^{(k)})\|}{\|\boldsymbol{H}_1(\boldsymbol{v}^{(k-1)})\|}\right)\right)$$

The forcing term σ_k is chosen of the same order of δ_k as

$$\sigma_k = \min\left(\sigma_{\max}, \max\left(1.1 \cdot \frac{0.5\tau_2\delta_k\sqrt{2}}{\min(1,\tau_2)}, 0.01\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|\right)\right)$$

In the three inner solvers, an explicit computation of the matrices $A + \chi BB^t$, $B^t\bar{A}^{-1}B$ and of the preconditioner $\bar{\bar{M}}$ is needed for the factorization. As explained in Section 4, for the IP–MM and IP–PCG1 codes the structure of matrices $A + \chi BB^t$ and $B^t\bar{A}^{-1}B$ respectively, is computed with a preprocessing routine. This preprocess is not needed for the code IP–PCG2.
In Figure 4, we report, for the test problem P2-6 with N= 5, the sparsity pattern of the matrices $A+\chi BB^t$, B^tB and $\bar{M}$ that have to be factorized for the IP–MM, IP–PCG1 and IP–PCG2 codes respectively.
In Table 7, the number of nonzero entries of one triangular part (including the diagonal elements) of the matrices $A + \chi BB^t$, $B^t\bar{A}^{-1}B$ and $\bar{M}$ is reported in the columns "nnz-mat1", "nnz-mat2" and "nnz-mat3" respectively, while the number of nonzero entries of the Cholesky factor is listed in the columns "L-mat1", "L-mat2" and "L-mat3".

From Table 7, it can be observed that the number of nonzero entries in the Cholesky factor is quite similar in the three cases: the matrices $A + \chi BB^t$ and $B^t \bar{A}^{-1} B$ have a density at most equal to 0.1%, while the ratio of the nonzero entries of the Cholesky factor and of the lower triangular part of $A + \chi BB^t$ is at most equal to 15.3.
In tables 8 and 9, we report the minimum values of the objective functional in the optimal control problems obtained by IP–PCG2; the difference with respect to the minimum values obtained with the other solvers are not significant, they differs of a factor of 10^{-8} and they are according to the values reported in [67], [68], [71].
In tables 10, 11 and 12 we evaluate the effectiveness of the different versions of the code implementing the damped Newton interior point method with the iterative inner solvers with $\epsilon_{\text{exit}} = 10^{-8}$, while in Table 13 there are the results related to the code with the direct inner solver MA27. The method is referred as IP–MA27.
The number of outer iterations ("it"), the total number of inner iterations ("inn") and the execution time in seconds ("time") are reported.
In IP–MM and IP–PCG1 codes the CPU time is divided in the time required by the preprocessing routine in the computation of the matrices structure ("prep") and in the time needed for the the computation of the iterations ("iter").
We consider that an algorithm fails when the backtracking procedure produces a damping parameter smaller than 10^{-8} (we use the symbol "*"). The symbol "m" indicates a memory failure.
From the numerical experiments, we can draw the following remarks:

- the number of inner iterations per outer iteration is very small for all the three inner solvers; we observe that in the experiments sometimes it happens that the total number of inner iterations is less than the number of outer iterations. This can happen in any code that uses an iterative inner solver with an adaptive stopping rule. Indeed, for some values of $\boldsymbol{v}^{(0)}$, it can happen that the inner stopping rule (16) with $\boldsymbol{r}^{(k)}$ as in (15) and $k = 0$, is satisfied without the necessity of inner iterations, since $\|\boldsymbol{H}(\boldsymbol{v}^{(0)}\|$ is large. Nevertheless, even if $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{\lambda}_1$ are unchanged, $\Delta\tilde{\boldsymbol{s}}$ and $\Delta\tilde{\boldsymbol{w}}$ change and then, in the subsequent iteration, $\|\boldsymbol{H}(\boldsymbol{v})\|$ decreases until *to cause* the necessity of iterations of the inner solver. In other words, for some values of $\boldsymbol{v}^{(0)}$, the iterate of the interior point method moves only along the directions of $\tilde{\boldsymbol{s}}$ and $\tilde{\boldsymbol{w}}$; this can arise for some initial iterations;

- the most expensive computational task for the codes IP–MM and IP–PCG1 is the preprocessing phase; we can also notice that the preprocessing time of IP–PCG1 code is smaller than the one of IP–MM code, since the size of the matrix to preprocess is neq respect to the size of the matrix to preprocess for IP–MM which is n and $neq < n$. This gain in terms of time is more significant when the test problem arises from distributed control problems, since in such cases, the number of equality constraints is a half of the number of the variables. On the other hand, beside a "heavy"

preprocessing phase, the time for the computation of the iterations is very fast. This feature could be exploited when different problems with the same structure or the same problem with different parameters have to be solved in sequence;

- in terms of total time, the most effective code is the IP–PCG2, which does not require the preprocessing phase and needs almost the same number of outer and inner iterations than the version IP–PCG1. Thus, since at each iteration of the conjugate gradient, the IP–PCG2 code has to solve a system with the matrix $\bar{M}$ of order $n + neq$ while the IP–PCG1 code has to solve systems of order neq with Ng and Peyton routine, we point out the efficiency of the BLKFCLT routine;

- another feature of IP–PCG2 code is that it requires a relatively little memory storage; this allows us to solve very large scale problems up to one million primal variables.

The results of Tables 14, 15, 16, 17, 18 show the performances of IP–PCG2 code ($\epsilon_{\text{exit}} = 10^{-8}$) and of the direct and iterative versions of KNITRO (version 4.0.2), with different values of the tolerance parameter ("opttol"). We report the computed minimum $f(\boldsymbol{v}^{(it)})$, the value of $\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$, the execution time in seconds ("time"), the number of outer iterations ("it"). For IP–PCG2" code and the iterative version of KNITRO we also report the total number of inner iterations ("inn").
The notation 1e-6 denotes 10^{-6}. For these experiments, IP–PCG2 use the same input AMPL models of the discretized PDE problems as KNITRO. The primal variables are initialized the same way as they are in the AMPL models. Then the execution time and the number of iterations of IP-PCG2 are different with respect those reported in tables 10, 11 and 12.
This comparison highlights the good stability and efficiency of IP–PCG2 method on this kind of test problems for large scale problems.
Finally, we report in Table 19 the results on the distributed control problem P2-7 of the IP–PCG2 algorithm with the nonmonotone choices for the additive inner stopping rule and for the backtracking rule, as seen in subsection 4.2; different values of degree of nonmonotonicity M are examined. Obviously, for $M = 1$, we have the monotone algorithm. From Table 19, we observe that an improvement of efficiency of the method can be obtained by using $M > 1$, for example, in this case $M = 4$.

Acknowledgements. The authors are very grateful to Hans Mittelmann for fruitful discussions and for the results of Table 1.

References

[1] Apostol T.M.; Mathematical Analysis, Second Edition, Addison–Wesley Publ. Reading MA, 1974.

Test problems	References
P1-1	Example 5.5 in [67]
P1-2	Example 5.6 in [67]
P1-3	Example 5.7 in [67]
P1-4	Example 5.8 in [67]
P1-5	Example 5.1 in [67]
P1-6	Example 5.2 in [67]
P1-7	Example 5.3 in [67]
P1-8	Example 5.4 in [67]
P1-9	Example 4.1 in [71] with $\alpha = 0.005$
P1-10	Example 4.1 in [71] with $\alpha = 0$
P2-1	Example 1 in [68]
P2-2	Example 2 in [68]
P2-3	Example 3 in [68]
P2-4	Example 4 in [68]
P2-5	Example 5 in [68]
P2-6	Example 4.2 in [71] with $a(x) = 7 + 4\sin(2\pi x_1 x_2)$, $M = 1$, $K = 0.8$, $b = 1$, $u_1 = 1.7$, $u_2 = 2$, $\psi(x) = 7.1$
P2-7	Example 4.2 in [71] with $a(x) = 7 + 4\sin(2\pi x_1 x_2)$, $M = 0$, $K = 1$, $b = 1$, $u_1 = 2$, $u_2 = 6$, $\psi(x) = 4.8$

Table 4: Description of the test problems

[2] Altman A. , Gondzio J.; *Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization*, Optim. Meth. Software 11–12 (1999), 275–302.

[3] Argáez M., Tapia R.A.; *On the global convergence of a modified augmented lagrangian linesearch interior–point Newton method for nonlinear programming*, J. Optim. Theory Appl. 114 (2002), 1–25.

[4] Argáez M., Tapia R., Velázquez L.; *Numerical comparisons of path–following strategies for a primal–dual interior–point method for nonlinear programming*, J. Optim. Theory Appl. 114 (2002), 255–272.

[5] Armand P., Gilbert J.C., Jan–Jégou S.; *A feasible BFGS interior point algorithm for solving convex minimization problems*, SIAM J. Optim. 11 (2000), 199–222.

[6] Armijo L.; *Minimization of functions having Lipschitz–continuous first partial derivatives*, Pacific J. Math. 16 (1966), 1–3.

[7] Benson H.Y., Shanno D.F., Vanderbei R.J.; *Interior–point methods for nonconvex nonlinear programming: Jamming and numerical testing*, Math. Program. 99 (2004), 35–48.

	N	n	neq	nu	nl	nnzjac	nnzhess
P1-1	99	10593	10197	10593	39204	50193	10593
	199	41193	40397	41193	158404	200393	41193
	299	91793	90597	91793	357604	450593	91793
	399	162393	160797	162393	636804	800793	162393
	499	252993	250997	252993	996004	1250993	252993
	599	363593	361197	363593	1435204	1801193	363593
P1-2	99	10593	10197	10593	39204	50193	10197
	199	41193	40397	41193	158404	200393	40397
	299	91793	90597	91793	357604	450593	90597
	399	162393	160797	162393	636804	800793	160797
	499	252993	250997	252993	996004	1250993	250997
	599	363593	361197	363593	1435204	1801193	361197
P1-3	99	10593	10197	10593	39204	50193	10593
	199	41193	40397	41193	158404	200393	41193
	299	91793	90597	91793	357604	450593	91793
	399	162393	160797	162393	636804	800793	162393
	499	252993	250997	252993	996004	1250993	252993
	599	363593	361197	363593	1435204	1801193	363593
P1-4	99	10593	10197	10593	39204	50193	9801
	199	41193	40397	41193	158404	200393	39601
	299	91793	90597	91793	357604	450593	89401
	399	162393	160797	162393	636804	800793	159201
	499	252993	250997	252993	996004	1250993	249001
	599	363593	361197	363593	1435204	1801193	358801
P1-5	99	10197	9801	10197	396	49005	10197
and	199	40397	39601	40397	796	198005	40397
P1-7	299	90597	89401	90597	1196	447005	90597
	399	160797	159201	160797	1596	796005	160797
	499	250997	249001	250997	1996	1245005	250997
	599	361197	358801	361197	2396	1794005	361197
P1-6	99	10197	9801	10197	396	49005	9801
and	199	40397	39601	40397	796	198005	39601
P1-8	299	90597	89401	90597	1196	447005	89401
	399	160797	159201	160797	1596	796005	159201
	499	250997	249001	250997	1996	1245005	249001
	599	361197	358801	361197	2396	1794005	358801
P1-9	119	14637	14518	14280	14637	71519	3840
	179	32757	32578	32220	32757	161279	8460
	279	78957	78678	78120	78957	390879	20160
	379	145157	144778	144020	145157	720479	36870
	479	231357	230878	229920	231357	1150079	58560
	579	337557	336978	335820	337557	1679679	85260
P1-10	119	14637	14518	14280	14637	71519	3721
	179	32757	32578	32220	32757	161279	8281
	279	78957	78678	78120	78957	390879	19881
	379	145157	144778	144020	145157	720479	36491
	479	231357	230878	229920	231357	1150079	58081
	579	337557	336978	335820	337557	1679679	84681

Table 5: Description of the test problems: boundary control problems

	N	n	neq	nu	nl	nnzjac	nnzhess
P2-1	99	19602	9801	19602	9801	59202	19602
	199	79202	39601	79202	39601	238402	79202
	299	178802	89401	178802	89401	537602	178802
	399	318402	159201	318402	159201	956802	318402
	499	498002	249001	498002	249001	1496002	498002
P2-2	99	19602	9801	19602	9801	59202	9801
	199	79202	39601	79202	39601	238402	39601
	299	178802	89401	178802	89401	537602	89401
	399	318402	159201	318402	159201	956802	159201
	499	498002	249001	498002	249001	1496002	249001
P2-3	99	19998	10197	19602	9801	59598	19602
and	199	79998	40397	79202	39601	239198	79202
P2-4	299	179998	90597	178802	89401	538798	178802
	399	319998	160797	318402	159201	958398	318402
	499	499998	250997	498002	249001	1497998	498002
P2-5	99	19998	10197	19602	9801	59598	10197
	199	79998	40397	79202	39601	239198	40397
	299	179998	90597	178802	89401	538798	90597
	399	319998	160797	318402	159201	958398	160797
	499	499998	250997	498002	249001	1497998	250997
P2-6	99	19602	9801	19602	9801	58410	39204
	199	79202	39601	79202	39601	236810	158404
	299	178802	89401	178802	89401	535210	357604
	399	318402	159201	318402	159201	953610	636804
	499	498002	249001	498002	249001	1492010	996004
P2-7	99	19602	9801	19602	9801	58410	29403
	199	79202	39601	79202	39601	236810	118803
	299	178802	89401	178802	89401	535210	268203
	399	318402	159201	318402	159201	953610	477603
	499	498002	249001	498002	249001	1492010	747003

Table 6: Description of the test problems: distributed control problems

	N	nnz-mat1	L-mat1	nnz-mat2	L-mat2	nnz-mat3	L-mat3
P1-1,P1-2,	99	70783	622759	69991	621571	60786	718637
P1-3,P1-4	199	281583	3181444	279195	3179056	241586	3416032
	299	632383	8374469	628795	8370881	542386	9084296
	399	1123183	16252152	1118395	16247364	9631186	20102932
	499	1753983	26855490	1747995	26849502	1503986	28784753
	599	2524783	41135305	2517595	41128117	2164786	43488232
P1-5,P1-6,	99	69595	621571	67619	619595	59202	716261
P1-7,P1-8	199	279195	3179056	275219	3175080	238401	3411256
	299	628795	8370881	622819	8364905	537602	9011520
	399	1118395	16247364	1110419	16239388	956802	20093356
	499	1747995	26849502	1738019	26839526	1496002	28772777
	599	2517595	41128117	2505619	41116141	2155202	43473654
P1-9,P1-10	119	100315	945546	99720	944951	86156	1029560
	179	226075	2541572	225180	2540677	194036	2733190
	279	547675	7167732	546280	7166337	469836	8619291
	379	1009275	14501957	1007380	14500062	865636	15396152
	479	1610875	24901311	1608480	24898916	1381436	26203761
	579	2352475	37810473	2349580	37807578	2017236	48288922
P2-1,P2-2,	99	126029	715465	67619	619595	78012	735071
P2-3	199	512029	3409660	275219	3175080	316012	3488866
	299	1158029	8900195	622819	8364905	714012	9253530
	399	2064029	20090160	1110419	16239388	1272012	20405866
	499	3230029	28768781	1738019	26839526	1990012	29266787
P2-4,P2-5	99	128401	717837	69595	621571	79596	737447
	199	516801	3414432	517993	3179056	319196	3493642
	299	1165201	8907367	1166993	8370881	718796	9260706
	399	2073601	20099732	2075994	16247364	1278396	20418142
	499	3242001	28780753	3244994	26849502	3244994	29278763
P2-6,P2-7	99	72816	715465	67619	619595	78012	735071
	199	295620	3409660	510837	3175080	316012	3488866
	299	1158029	8900195	622819	8364905	714012	9079001
	399	2064029	20090160	1110419	16239388	1272012	20408566
	499	3230029	28768781	1738019	26839526	1990012	29266787

Table 7: Nonzero entries of the matrices and of the Cholesky factors

Boundary control problems

	N	minimum		N	minimum
P1-1	99	0.55224625	P1-6	99	0.09669507
	199	0.55436881		199	0.10044221
	299	0.55507372		299	0.10170115
	399	0.55542568		399	0.10233242
	499	0.55580371		499	0.10271175
	599	0.55577731		599	0.10296487
P1-2	99	0.01507867	P1-7	99	0.32100965
	199	0.01560172		199	0.32812152
	299	0.01577842		299	0.33050688
	399	0.01586721		399	0.33170235
	499	0.01592062		499	0.33242047
	599	0.01595628		599	0.33289956
P1-3	99	0.26416255	P1-8	99	0.24917848
	199	0.26728343		199	0.25587655
	299	0.26832628		299	0.25812464
	399	0.26884799		399	0.25925143
	499	0.26916120		499	0.25992872
	599	0.26937006		599	0.26038061
P1-4	99	0.16553111	P1-9	119	0.25908196
	199	0.16778056		179	0.25305430
	299	0.16854245		279	0.24894250
	399	0.16892441		379	0.24705220
	499	0.16915379		479	0.24596630
	599	0.16930712		579	0.24526176
P1-5	99	0.19651967	P1-10	119	0.15741541
	199	0.20077162		179	0.15128350
	299	0.20219539		279	0.14694570
	399	0.20290856		379	0.14492090
	499	0.20333682		479	0.14375680
	599	0.20362247		579	0.14299524

Table 8: Minimum values of the objective functional: boundary control problems

Distributed control problems

	N	minimum		N	minimum
P2-1	99	0.06216164	P2-5	99	0.05266390
	199	0.06442591		199	0.05293239
	299	0.06519262		299	0.05302628
	399	0.06557820		399	0.05307458
	499	0.06581034		499	0.05310603
P2-2	99	0.05644747	P2-6	99	-6.57642757
	199	0.05869688		199	-6.62009226
	299	0.05946010		299	-6.63464408
	399	0.05984417		399	-6.64192346
	499	0.06007572		499	-6.64629219
P2-3	99	0.11026306	P2-7	99	-18.73618438
	199	0.11026872		199	-18.86331163
	299	0.11026969		299	-18.90575104
	399	0.11027035		399	-18.92698093
	499	0.11027047		499	-18.93972227
P2-4	99	0.07806386			
	199	0.07842594			
	299	0.07854995			
	399	0.07861255			
	499	0.07865054			

Table 9: Minimum values of the objective functional: distributed control problems

		IP–MM			IP–PCG1			IP–PCG2	
	N	it(inn)	prep+iter	time	it(inn)	time(prep+iter)	time	it(inn)	time
P1-1	99	29(32)	2.22+2.03	4.25	37(30)	2.21+2.46	4.67	37(72)	5.24
	199	54(59)	36.38+22.87	59.25	45(37)	35.81+18.31	54.12	45(95)	38.9
	299	181(186)	206.35 +246.8	453.15	52(47)	197.29+68.09	256.38	52(116)	156.49
	399	327(341)	833.79+961.08	1794.92	58(53)	758.23+174.82	933.05	58(137)	493.07
	499	501(527)	1933.8+2768.7	4702.5	63(59)	1635.64+341.65	1977.37	63(158)	845.76
	599	*	*	*	66(62)	1902.21+701.21	2603.55	66(181)	1377.61
P1-2	99	34(34)	2.2+2.3	4.6	35(39)	2.3+2.4	4.7	35(37)	4.5
	199	40(40)	37.8+17.3	55.1	41(45)	37.8+17.3	55.1	41(41)	32.6
	299	55(56)	172.8+73.5	246.3	51(55)	201.8+68.13	269.9	50(52)	140.3
	399	143(144)	558.1+420.8	979	58(64)	655.9+171.6	827.5	59(60)	441.7
	499	197(198)	1418.3+1076.1	2494.4	66(76)	1621.4+364+9	1986.4	70(73)	829.1
	599	242(243)	2997.4+2367.5	5364.9	74(87)	3456.5+714.5	4171.1	79(89)	1560.2
P1-3	99	21(23)	3.02+1.46	4.49	29(36)	2.22+2.05	4.28	28(79)	4.31
	199	26(27)	47.83+10.87	58.71	33(42)	45.87+14.46	60.35	33(91)	30.02
	299	39(45)	162.15+52.88	215.03	36(47)	194.79+49.7	243.52	37(109)	115.84
	399	36(39)	831.0+105.29	936.34	39(54)	617.47+117.91	735.42	38(120)	312.78
	499	65(87)	2062.11+360.03	2422.22	42(55)	1522.11+232.93	1755.12	41(146)	535.14
	599	*	*	*	44(60)	3928.54+427.12	3928.54	43(159)	925.84
P1-4	99	31(31)	2.2+2.1	4.3	33(42)	2.3+2.3	4.6	31(44)	4.2
	199	38(98)	34.5+18.9	53.5	40(51)	36.6+17.2	53.8	38(59)	31.5
	299	41(58)	172.7+56.3	228.9	45(60)	189.9+61.3	251.3	40(59)	114.7
	399	43(45)	560.3+125.3	685.7	49(66)	603.6+147.2	750.8	45(67)	341.7
	499	*	*	*	51(67)	1499.3+283.7	1783.1	50(76)	601.7
	599	*	*	*	69(82)	3621.0+662.8	4284	82(153)	1660.4
P1-5	99	28(28)	2.1+1.9	4	31(31)	2.1+2.1	4.2	28(34)	3.6
	199	33(33)	33.8+13.6	47.4	37(37)	35.7+15.2	50.9	32(42)	26
	299	40(55)	169.0+54.8	223.8	41(41)	194.9+53.9	248.8	36(50)	99.6
	399	45(75)	548.0+137.4	685.4	44(45)	751.0+128.1	879.1	39(62)	298.3
	499	49(79)	1380.6+275.1	1655.8	46(47)	1583.6+248.5	1832.1	43(73)	520.2
	599	51(126)	2978.9+534.4	3513.3	48(50)	3336.1+456.7	3792.9	46(77)	925.3

Table 10: Numerical results: boundary control problems P1-1–P1-5

		IP–MM			IP–PCG1			IP–PCG2	
	N	it(inn)	prep+iter	time	it(inn)	prep+iter	time	it(inn)	time
P1-6	99	30(30)	2.1+2.0	4.1	35(37)	2.1+2.4	4.5	30(39)	3.9
	199	33(33)	33.2+13.6	46.7	37(41)	35.4+15.4	50.8	32(41)	25.8
	299	40(55)	168.3+54.3	222.6	41(47)	231.6+55.8	287.4	37(54)	102.8
	399	46(61)	546.9+136.7	683.7	44(50)	634.8+129.4	764.2	40(65)	306.1
	499	50(95)	1377.3+288.2	1665.5	47(56)	1587.4+258.6	1846.1	44(75)	533.5
	599	51(126)	2971.5+532.8	3504.5	49(59)	3290.6+470.1	3760.8	46(77)	925.2
P1-7	99	39(39)	2.1+2.7	4.8	42(42)	2.1+2.8	4.9	40(54)	5.2
	199	46(46)	33.2+19.0	52.1	46(46)	35.6+18.9	54.5	45(72)	37.4
	299	64(99)	168.4+89.2	257.6	52(52)	193.6+69.5	263.1	49(87)	138.7
	399	96(141)	549.3+288.5	837.8	55(58)	636.0+160.5	796.5	53(95)	407.9
	499	127(169)	1387.3+703.4	2090.7	57(64)	1583.1+311.9	1895.1	52(94)	630
	599	159(202)	3020.9+1548.1	4569.1	59(69)	3309.4+563.6	3873.1	52(98)	1051.9
P1-8	99	40(40)	2.1+2.7	4.8	43(43)	2.1+2.9	5	41(52)	5.3
	199	48(48)	33.2+20.5	53.7	50(50)	35.4+20.6	56	47(74)	38.8
	299	66(106)	168.9+93.1	292	55(55)	195.3+72.8	268.2	52(90)	146.8
	399	101(156)	546.1+304.3	851.4	60(63)	637.0+174.6	808.7	58(105)	447.2
	499	133(195)	1404.1+745.6	2149.8	62(70)	1589.4+337.4	1926.9	57(105)	693.1
	599	167(364)	3033+1725.1	4758.1	65(76)	3595.4+622.9	4218.4	58(112)	1175.5
P1-9	119	41(41)	4.4+4.3	8.7	45(47)	4.5+4.8	9.3	48(74)	9.6
	179	75(87)	17.1+25.7	42.8	51(56)	23.2+16.9	40.1	46(73)	30.1
	279	63(63)	133.1+69.4	202.5	57(63)	140.7+62.4	203.4	51(90)	136.2
	379	82(97)	448.9+212.5	661.4	62(78)	482.3+161.9	644.2	55(112)	302.1
	479	95(185)	1226.6+523.1	1749.7	65(84)	1255.2+341.1	1596.3	58(129)	638.5
	579	110(275)	2562+997.8	3560	67(96)	2658.7+570.9	3229.6	61(149)	1545.1
P1-10	119	44(44)	4.4+4.6	9	49(52)	4.5+5.2	9.8	44(70)	8.9
	179	56(56)	22.4+18.4	40.8	59(65)	23.2+19.5	42.7	55(89)	36
	279	72(87)	129.1+80.1	209.2	67(79)	140.9+74.3	215.2	63(117)	169.2
	379	101(251)	452.9+290.5	743.5	77(100)	483.6+204.9	688.6	74(165)	408.6
	479	105(360)	1202.8+629.1	1832	81(113)	1235.4+429.9	1665.3	78(190)	864.7
	579	119(374)	2558.6+1123.8	3682.5	86(130)	2660.6+738.9	2299.6	83(224)	2118.1

Table 11: Numerical results: boundary control problems P1-6–P1-10

		IP–MM			IP–PCG1			IP–PCG2	
	N	it(inn)	prep+iter	time	it(inn)	prep+iter	time	it(inn)	time
P2-1	99	23(23)	4.8+2.2	7.1	26(25)	2.5+1.9	4.36	24(23)	3.3
	199	28(193)	123.1+26.4	149.5	28(26)	41.5+12.1	53.7	27(26)	22.6
	299	*	*	*	30(29)	218.3+41.2	259.5	28(27)	81.2
	399	*	*	*	31(56)	706.4+100.9	807.4	29(28)	222
	499	*	*	*	32(69)	2166.8+196.3	2363.2	29(28)	351.8
P2-2	99	28(28)	4.8+2.6	7.5	31(45)	2.5+2.4	4.9	29(28)	3.9
	199	31(166)	78.8+25.8	104.6	33(53)	41.7+15.7	57.4	30(29)	24.74
	299	*	*	*	34(64)	217.7+51.5	269.2	32(31)	92.8
	399	*	*	*	36(95)	704.9+125.7	830.7	33(32)	252.3
	499	*	*	*	37(132)	1741.9+252.1	1994.1	33(32)	399.1
P2-3	99	25(25)	4.8+2.4	7.2	31(26)	2.5+2.2	4.7	25(22)	3.4
	199	31(196)	119.0+27.9	147	33(27)	41.5+14.5	55.7	26(23)	21.7
	299	43(403)	694.4+131.1	825.6	34(28)	218.4+46.1	264.5	28(25)	80.8
	399	89(1184)	2339.9+758.3	3098.2	37(58)	869.4+119.4	988.9	30(27)	229.1
	499	*	*	*	36(61)	1742.25+212.91	1955.3	29(26)	350.23
P2-4	99	24(54)	5.0+2.9	7.9	20(16)	3.08+1.45	4.53	20(38)	3.11
	199	27(237)	80.6+29.4	109.9	21(17)	40.82+9.11	49.94	21(37)	19.02
	299	35(335)	424.2+108.3	532.5	22(18)	227.65+30.02	257.67	22(39)	68.21
	399	36(351)	1480.1+256.3	1736.5	23(19)	752.20+69.30	821.5	23(42)	184.77
	499	*	*	*	23(19)	2119.59+127.88	2247.47	23(42)	290.58
P2-5	99	48(63)	5.0+4.8	9.9	56(152)	2.6+5.2	7.8	47(43)	6.4
	199	68(383)	80.7+58.2	138.9	78(712)	52.7+74.7	127.4	65(61)	53.9
	299	104(1439)	421.5+403.4	825.4	91(1356)	226.9+320.7	547.7	80(77)	230.9
	399	155(2255)	1489.0+1376.1	2865.2	107(1436)	718.6+704.4	1423.1	93(92)	709.7
	499	i	i	i	116(3125)	1798.6+2040.4	3839.1	104(104)	1256
P2-6	99	28(29)	5.77+2.7	8.48	35(70)	2.46+3.03	5.5	34(122)	6.28
	199	48(49)	118.03+25.11	143.17	51(88)	41.25+25.19	66.44	51(178)	53.2
	299	81(111)	686.30+131.49	817.99	56(97)	223.61+85.79	309.41	54(177)	173.82
	399	102(153)	2292.11+477.5	2769.7	71(130)	727.06+239.67	966.73	64(221)	553.78
	499	101(166)	5496.66+699.3	6196.11	62(107)	1849.82+361.12	2210.95	61(209)	823.08
P2-7	99	51(51)	4.8+4.9	9.7	51(90)	2.5+4.2	6.7	35(70)	5.5
	199	62(107)	118.7+35.6	154.3	63(284)	41.4+41.8	83.2	51(88)	45.8
	299	68(188)	684.8+127.4	812.4	70(493)	217.14+164.1	381.29	54(94)	158.7
	399	80(1010)	2299.4+654.9	2954.3	81(1014)	703.2+522.5	1225.8	65(109)	515.9
	499	90(1170)	3808.8+1150.7	4959.6	87(1331)	1733.9+1083.6	2817.7	80(115)	989.4

Table 12: Numerical results: distributed control problems

	IP–MA27		
	N	it	time
P1-1	99	29	27.38
	199	37	349.66
P1-2	199	35	339.1
P1-3	99	24	22.52
	199	27	250.28
P1-4	99	25	22.7
	199	30	269.7
P1-5	99	24	21.7
	199	26	370
P1-6	99	24	23.1
	199	26	258.7
P1-7	99	26	22.1
	199	31	269.1
P1-8	99	27	22.9
	199	33	285.1
P1-9	119	31	48.1
	179	34	406.7
P1-10	119	35	54.6
	179	40	581.5
P2-6	99	25	24.71
	199	26	304.11

Table 13: Control problems with direct inner solver

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$	time	it(inn)
99	IP–PCG2		0.55224625	5.7e-9	9.52	37(33)
	KNITRO–D	1e-6	0.55332954	4e-7	5.44	15
		1e-8	0.55224722	3.9e-9	8.47	24
		1e-9	0.55224625	4.2e-11	9.52	27
	KNITRO–I	1e-6	0.55331458	3.9e-7	8.6	14(93)
		1e-8	0.55224739	8.66e-9	33.12	22(983)
		1e-9	0.55224641	7.02e-10	39.57	25(1210)
199	IP–PCG2		0.5543688	3.6e-9	43.6	45(40)
	KNITRO–D	1e-6	0.55446811	2.2e-8	45.22	18
		1e-8	0.55437617	6.4e-9	52.45	21
		1e-9	0.55436933	1e-9	56.71	23
	KNITRO–I	1e-6	0.554480051	3.1e-7	97.84	15(418)
		1e-8	0.554376509	7.4e-9	242.84	22(1462)
		1e-9	0.554368888	1.9e-10	319.27	27(2054)
299	IP–PCG2		0.55507371	5.1e-9	157.41	52(50)
	KNITRO–D	1e-6	0.57017027	8.1e-7	98.79	10
		1e-8	0.555099009	5.9e-9	212.04	23
		1e-9	0.555073838	2.9e-10	474.95	34
	KNITRO–I	1e-6	0.555405597	3.3e-8	202.48	14(153)
		1e-8	0.555099497	5.55e-9	693.32	21(1665)
		1e-9	0.55507386	7.7e-11	1636.35	28(4472)
399	IP–PCG2		0.555425678	2.8e-9	531.16	58(58)
	KNITRO–D	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m
	KNITRO–I	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m

Table 14: Comparison IP–PCG2 with KNITRO v4.0.2 on test problem P1-1

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$	time	it(inn)
99	IP–PCG2		0.2641625459	3.5e-9	5.9	28(24)
	KNITRO–D	1e-6	0.2642862472	4.5e-7	5.5	14
		1e-8	0.264163747	4.7e-9	7.0	18
		1e-9	0.2641625452	7.2e-11	7.7	20
	KNITRO–I	1e-6	0.2643159559	5.5e-7	7.0	11(38)
		1e-8	0.2641637501	5.0e-9	13.0	15(173)
		1e-9	0.2641625452	5.2e-10	16.3	17(247)
199	IP–PCG2		0.2672834461	9.0e-9	40.0	35(34)
	KNITRO–D	1e-6	0.2676809839	5.6e-7	37.2	14
		1e-8	0.2672858418	5.5e-9	49.5	19
		1e-9	0.2672839122	9.2e-10	54.4	21
	KNITRO–I	1e-6	0.2676583552	8.0e-7	57.8	14(85)
		1e-8	0.2672838028	4.8e-9	100.5	21(231)
		1e-9	0.2672835172	3.7e-10	155.2	23(536)
299	IP–PCG2		0.2683261906	6.6e-10	125.9	37(36)
	KNITRO–D	1e-6	0.2683490951	7.6e-7	174.6	19
		1e-8	0.2683296025	9.0e-9	191.7	21
		1e-9	0.2683262257	2.3e-10	217.3	24
	KNITRO–I	1e-6	0.2694845168	4.0e-7	258.2	19(110)
		1e-8	0.2683272538	6.3e-9	361.6	25(217)
		1e-9	0.2683263348	5.0e-10	494.2	27(488)
399	IP–PCG2		0.26884799937	4.3e-9	352.3	39(34)
	KNITRO–D	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m
	KNITRO–I	1e-6	0.2684751969	3.1e-7	401.9	11(70)
		1e-8	0.2688521643	9.6e-9	523.9	14(127)
		1e-9	0.2688480413	3.6e-10	916.4	17(509)

Table 15: Comparison IP–PCG2 with KNITRO v4.0.2 on test problem P1-3

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$	time	it(inn)
99	IP–PCG2		6.216167657e-2	8.0e-9	5.9	23(22)
	KNITRO–D	1e-6	6.28379652e-2	4.2e-7	4.28	8
		1e-8	6.21720059e-2	4.2e-8	6.95	14
		1e-9	6.21637312e-2	9.1e-10	7.85	16
	KNITRO–I	1e-6	6.290104777e-2	1.9e-7	3.51	4(11)
		1e-8	6.21659812e-2	2.5e-9	6.4	8(30)
		1e-9	6.2162183013e-2	4.1e-10	8.86	10(66)
199	IP–PCG2		6.44262870e-2	8e-9	33.3	25(24)
	KNITRO–D	1e-6	6.519188743e-2	1.3e-7	20.39	6
		1e-8	6.44368448e-2	3.5e-9	36.8	12
		1e-9	6.442826536e-2	2.3e-10	45.2	15
	KNITRO–I	1e-6	6.5203856844	8e-7	16.5	3(8)
		1e-8	6.444127289e-2	3.6e-9	33.7	7(25)
		1e-9	6.44335931e-2	7.8e-10	44.1	9(47)
299	IP–PCG2		6.5193140696e-2	9.3e-9	100.1	26(25)
	KNITRO–D	1e-6	6.5970554302e-2	1.1e-7	66.3	6
		1e-8	6.5269064297e-2	1.8e-10	92.4	9
		1e-9	6.519765283e-2	6.2e-11	159.1	16
	KNITRO–I	1e-6	6.597880027e-2	5.2e-7	52.1	3(8)
		1e-8	6.525495675e-2	1.1e-9	91.7	6(17)
		1e-9	6.519876526e-2	3.0e-10	147.2	10(46)
399	IP–PCG2		6.5578165106e-2	8.8e-10	279.8	28(27)
	KNITRO–D	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m
	KNITRO–I	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m

Table 16: Comparison IP–PCG2 with KNITRO v4.0.2 on test problem P2-1

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$	time	it(inn)
99	IP–PCG2		-6.57642730	9.4e-8	10.1	29(66)
	KNITRO–D	1e-6	-6.57629592	3.9e-7	13.8	18
		1e-8	-6.57640594	8.9e-9	14.8	20
		1e-9	-6.57642744	1.1e-10	16.8	24
	KNITRO–I	1e-6	-6.57585240	6.3e-7	9.9	6(30)
		1e-8	-6.57642667	5.2e-10	42.2	12(494)
		1e-9	-6.57642667	5.2e-10	42.2	12(494)
199	IP–PCG2		-6.62009225	3.9e-8	46.3	27(61)
	KNITRO–D	1e-6	-6.61987115	4.7e-8	63.2	17
		1e-8	-6.62007441	5.6e-9	72.9	20
		1e-9	-6.62009178	3.7e-10	79.5	22
	KNITRO–I	1e-6	-6.61925123	1.2e-7	67.4	6(17)
		1e-8	-6.62009137	2.0e-9	185.2	12(198)
		1e-9	-6.62009225	6.0e-9	984.2	36(1802)[1]
299	IP–PCG2		-6.63464400	5.5e-9	126.3	27(55)
	KNITRO–D	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m
	KNITRO–I	1e-6	-6.63316140	1.3e-7	240.8	6(16)
		1e-8	-6.63462656	6.2e-9	373.5	9(38)
		1e-9	*	*	*	*

Table 17: Comparison IP–PCG2 with KNITRO v4.0.2 on test problem P2-6
[1] Current solution estimate can not be improved by the code.

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$	time	it(inn)
99	IP–PCG2		-18.73614837	5.9e-9	13.9	49(54)
	KNITRO–D	1e-6	-18.73188084	4.8e-7	7.0	13
		1e-8	-18.73611604	9.3e-9	14.8	29
		1e-9	-18.73614224	6.8e-10	17.4	34
	KNITRO–I	1e-6	-18.73221830	4.2e-7	20.5	19(133)
		1e-8	-18.73608419	7.8e-9	33.3	32(214)
		1e-9	-18.73614833	1.8e-10	38.6	38(235)
199	IP–PCG2		-18.86331163	4.9e-9	87.4	60(73)
	KNITRO-D	1e-6	-18.84530188	4.5e-7	38.2	11
		1e-8	-18.86315409	4.0e-9	119.7	37
		1e-9	-18.86328069	8.1e-10	141.0	44
	KNITRO–I	1e-6	-18.84592606	4.5e-7	88.4	14(118)
		1e-8	-18.86315369	4.7e-9	213.1	33(307)
		1e-9	-18.86329639	5.5e-10	248.2	40(339)
299	IP–PCG2		-18.905751265	3.3e-9	278.1	67(93)
	KNITRO–D	1e-6	-18.862397636	5.0e-7	126.7	10
		1e-8	-18.905307698	6.1e-9	361.2	31
		1e-9	-18.905681444	9.7e-10	472.5	41
	KNITRO–I	1e-6	-18.836892321	8.4e-7	309.1	14(153)
		1e-8	-18.905301531	6.1e-9	578.6	28(282)
		1e-9	*	*	*	*
399	IP–PCG2		-18.926980012	2.9e-7	825.2	76(113)
	KNITRO–D	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m
	KNITRO–I	1e-6	m	m	m	m
		1e-8	m	m	m	m
		1e-9	m	m	m	m

Table 18: Comparison IP–PCG2 with KNITRO v4.0.2 on test problem P2-7

		M=1		M=2		M=3		M=4	
	N	it(inn)	time	it(inn)	time	it(inn)	time	it(inn)	time
P2-7	99	35(70)	5.5	39(70)	5.9	42(72)	6.3	25(52)	3.9
	199	51(88)	45.8	54(94)	48.4	62(105)	55.3	28(37)	24.3
	299	54(94)	158.7	57(95)	167.2	73(114)	212.5	32(51)	93.8
	399	65(109)	515.9	79(144)	630.5	32(38)	248.6	35(37)	269.9

Table 19: IP–PCG2 with nonmonotone choices

[8] Bergamaschi L., Gondzio J., Zilli G.; *Preconditioning indefinite systems in interior point methods for optimization*, Comput. Optim. Appl. 28 (2004), 149–171.

[9] Bonettini S.; *A nonmonotone inexact Newton method*, Optim. Meth. Software 20 (2005), 475–491.

[10] Bonettini S., Galligani E., Ruggiero V.; *An inexact Newton method combined with Hestenes multipliers' scheme for the solution of Karush–Kuhn–Tucker systems*, Appl. Math. Comput. 118 (2005), 651–676.

[11] Bonettini S., Ruggiero V.; *Some iterative methods for the solution of a symmetric indefinite KKT system*, to appear on Comput. Optim. Appl. 2006.

[12] Bongartz I., Conn A.R., Gould N.I.M., Toint Ph.L.; *CUTE: Constrained and Unconstrained Testing Environment*, ACM Trans. Mathl. Software 21 (1995), 123–160.

[13] Branin F.H.; *Widely convergent method for finding multiple solutions of simultaneous nonlinear equations*, IBM J. Res. Devel. 16 (1972), 504–522.

[14] Bunch J. R., Parlett B. N.; *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM J. Numer. Anal. 8 (1971), 639–655.

[15] Byrd R.H., Gilbert J.C., Nocedal J.; *A trust region method based on interior–point techniques for nonlinear programming*, Math. Program. 89 (2000), 149–185.

[16] Byrd R.H., Hribar M.E., Nocedal J.; *An interior point algorithm for large scale nonlinear programming*, SIAM J. Optim. 9 (1999), 877–900.

[17] Byrd R.H., Marazzi M., Nocedal J.; *On the convergence of Newton iterations to non–stationary points*, Tech. Rep. OTC 2001/01, Optimization Technology Center, Nortwestern University, Evaston IL, 2001.

[18] Chen Y., Cai D.Y.; *Inexact overlapped block Broyden methods for solving nonlinear equations*, Appl. Math. Comput. 136 (2003), 215–228.

[19] Conn A.R., Gould N.I.M., Orban D., Toint Ph.L.; *A primal–dual trust region algorithm for non–convex nonlinear programming*, Math. Program. 87 (2000), 215–249.

[20] D'Apuzzo M., Marino M.; *Parallel computational issues of an interior point method for solving large bound–constrained quadratic programming problems*, Parallel Computing, 29 (2003), 467–483.

[21] Davidenko D.F.; *On a new method of numerical solution of systems of nonlinear equations*, Dokl. Akad. Nauk SSSR 88 (1953), 601–602 (in Russian).

[22] Davidenko D.F.; *Inversion of matrices by the method of variation of parameters*, Soviet Math. Dokl. 1 (1960), 279–282.

[23] Dembo R.S., Eisenstat S.C., Steihaug T.; *Inexact Newton methods*, SIAM J. Numer. Anal. 19 (1982), 400–408.

[24] Dennis J.E. Jr., Moré J.; *Quasi–Newton methods, motivation and theory*, SIAM Review 19 (1977), 46–89.

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

[26] Deuflhard P., Hohmann A.; Numerical Analysis in Modern Scientific Computing, Second Edition, Springer, New York, 2003.

[27] Durazzi C.; *Numerical solution of discrete quadratic optimal control problems by Hestenes' method*, Rend. Circ. Matem. Palermo, Ser. II, Suppl. 58 (1999), 133–154.

[28] Durazzi C.; *On the Newton interior–point method for nonlinear programming problems*, J. Optim. Theory Appl. 104 (2000), 73–90.

[29] Durazzi C., Galligani E.; *Nonlinear programming methods for solving optimal control problems*, Equilibrium Problems: Nonsmooth Optimization and Variational Inequality Models (F. Giannessi, A. Maugeri, P.M. Pardalos eds.), Nonconvex Optimization and Its Applications 58, Kluwer Academic Publ., Dordrecht, 2001, 71–99.

[30] Durazzi C., Ruggiero V.; *Indefinitely preconditioned conjugate gradient method for large sparse equality and inequality constrained quadratic problems*, Numer. Linear Algebra Appl. 10 (2003), 673–688.

[31] Durazzi C., Ruggiero V.; *Numerical solution of special linear and quadratic programs via a parallel interior–point method*, Parallel Computing 29 (2003), 485–503.

[32] Durazzi C., Ruggiero V.; *A Newton inexact interior–point method for large scale nonlinear optimization problems*, Annali Univ. Ferrara, Sez. VII, Sc. Matem. IL (2003), 333–357.

[33] Durazzi C., Ruggiero V.; *Global convergence of the Newton interior–point method for nonlinear programming*, J. Optim. Theory Appl. 120 (2004), 199–208.

[34] Durazzi C., Ruggiero V., Zanghirati G.; *Parallel interior–point method for linear and quadratic programs with special structure*, J. Optim. Theory Appl. 110 (2001), 289–313.

[35] Eisenstat S.C., Walker H.F.: *Globally convergent inexact Newton methods*, SIAM J. Optim. 4 (1994), 393–422.

[36] El–Bakry A.S., Tapia R.A., Tsuchiya T., Zhang Y.; *On the formulation and theory of Newton interior–point method for nonlinear programming*, J. Optim. Theory Appl. 89 (1996), 507–541.

[37] Evtushenko Y.G.; Numerical Optimization Techniques, Optimization Software Inc., New York, 1985.

[38] Faddeev D.K., Faddeeva V.N., Computational Methods of Linear Algebra, W.H. Freeman & C., San Francisco, 1963.

[39] Fletcher R.; Practical Methods of Optimization, Second Edition, John Wiley & Sons, Chichester, 1987.

[40] Forsgren A.; *Inertia–controlling factorizations for optimization algorithms*, Appl. Numer. Math. 43 (2002), 91–107.

[41] Forsgren A., Gill P.E.; *Primal–dual interior point methods for nonconvex nonlinear programming*, SIAM J. Optim. 8 (1998), 1132–1152.

[42] Forsgren A., Murray W.; *Newton method for large–scale linear equality–constrained minimization*, SIAM J. Matrix Anal. Appl. 14 (1993), 560–587.

[43] Galligani E.: *The Newton–arithmetic mean method for the solution of systems of nonlinear equations*, Appl. Math. Comput. 134 (2003), 9–34.

[44] Galligani E.; *Analysis of the convergence of an inexact Newton method for solving Karush–Kuhn–Tucker systems*, Atti Sem. Matem. Fis. Univ. Modena e Reggio Emilia, LII (2004), 331–368.

[45] Galligani E., Ruggiero V., Zanni L.; *A minimization method for the solution of large symmetric eigenproblems*, Intern. J. Computer Math. 70 (1998), 99–115.

[46] Galligani I., Ruggiero V.; *Numerical solution of equality–constrained quadratic programming problems on vector–parallel computers*, Optim. Meth. Software 2 (1993), 233–247.

[47] Galligani I., Trigiante D.; *Numerical methods for solving large algebraic systems*, Pubblicazioni IAC, 98, Ser. III, Rome, 1974.

[48] Gavurin M.K.; *Nonlinear functional equations and continuous analogies of iterative methods*, Izv. Vyssh. Uchebn. Zaved., Matematika 5 (1958), 18–31 (in Russian).

[49] Gill P.E., Murray D.B., Ponceleon D.B., Saunders M.A.; *Preconditioners for indefinite systems arising in optimization*, SIAM J. Matrix Anal. Appl. 13 (1992), 292–311.

[50] Gill P.E., Saunders M.A., Shinnerl J.R.; *On the stability of Choleski factorization for symmetric quasidefinite systems*, SIAM J. Matrix Anal. Appl. 17 (1996), 35–46.

[51] Golub G.,Greif C.; *On solving block–structured indefinite linear systems*, SIAM J. Sci. Comput. 24 (2003), 2076–2092.

[52] Golub G., Wu X., Yuan J.Y.; *SOR–like methods for augmented system*, BIT 41 (2001), 71–85.

[53] Hahn W.; Stability of Motion, Springer Verlag, Berlin, 1967.

[54] Harwell Subroutine Library; *A Catalogue of Subroutines (HSL 2000)*, AEA Technology, Harwell, Oxfordshire, England (2002).

[55] Hestenes M.R.; *Multiplier and gradient methods*, J. Optim. Theory Appl. 4 (1969), 303–320.

[56] Hestenes M.R.; Optimization Theory. The Finite Dimensional Case, John Wiley & Sons, New York, 1975.

[57] Kelley C.T.; Iterative Methods for Linear and Nonlinear Equations, SIAM, Philadelphia, 1995.

[58] Keller C., Gould N.I.M., Wathen A.J.; *Constraint preconditioners for indefinite linear systems*, SIAM J. Matrix Anal. Appl. 21 (2000), 1300–1317.

[59] Keller H.B.; *Global homotopies and Newton methods*, Recent Advances in Numerical Analysis (C. De Boor, G.H. Golub eds.), Academic Press, New York, 1978, 73–94.

[60] Keller H.B.; Lectures on Numerical Methods in Bifurcation Problems, Springer–Verlag, Heidelberg, 1987.

[61] Lambert J.D.; Computational Methods in Ordinary Differential Equations, John Wiley & Sons, London, 1973.

[62] Li C., Li Z., Shao X., Nie Y., Evans D.J.; *Optimum parameter for the SOR–like method for augmented system*, Intern. J. Computer Math. 81 (2004), 749–763.

[63] Liu J.W., Ng E.G., Peyton B.W.; *On finding supernodes for sparse matrix computations*, SIAM J. Matrix Anal. Appl. 14 (1993), 242–252.

[64] Luenberger D.G.; Linear and Nonlinear Programming, Second Edition, Addison–Wesley Publ., Reading, 1984.

[65] Lukšan L., Vlček J.; *Indefinitely preconditioned inexact Newton method for large sparse equality constrained non–linear programming problems*, Numer. Linear Algebra Appl. 5 (1998), 219–247.

[66] Lukšan L., Matonoha C., Vlček J.; *Interior–point method for non–linear non–convex optimization*, Numer. Linear Algebra Appl. 11 (2004), 431–453.

[67] Maurer H., Mittelmann H.D.; *Optimization techniques for solving elliptic control problems with control and state constraints: Part 1. Boundary control*, Comput. Optim. Appl. 16 (2000), 29–55.

[68] Maurer H., Mittelmann H.D.; *Optimization techniques for solving elliptic control problems with control and state constraints: Part 2. Distributed control*, Comput. Optim. Appl. 18 (2001), 141–160.

[69] Mittelmann H.D.; *Verification of second-order sufficient optimality conditions for semilinear elliptic and parabolic control problems*, Comput. Optim. Appl. 20 (2001), 93–110.

[70] Mittelmann H.D.; Private communication, 2004.

[71] Mittelmann H.D., Maurer H.; *Solving elliptic control problems with interior point and SQP methods: control and state constraints*, J. Comput. Appl. Math. 120 (2000), 175–195.

[72] Morales J.L., Nocedal J., Waltz R.A., Liu G., Goux J.P.; *Assessing the potential of interior methods for nonlinear optimization*, Tech. Rep. OTC 2001/04, Optimization Technology Center, Nortwestern University, Evaston IL, 2001.

[73] Nocedal J., Hribar M.E., Gould N.I.M.; *On the solution of equality constarined quadratic programming problems arising in optimization*, SIAM J. Sci. Comput. 23 (2001), 1375–1394.

[74] Nocedal J., Wright S.J.; Numerical Optimization, Springer, New York, 1999.

[75] Ortega J.M.; Introduction to Parallel and Vector Solution of Linear Systems, Plenum Press, New York, 1988.

[76] Ortega J.M., Rheinboldt W.C.; Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.

[77] Powell M.J.D.; *A method for nonlinear constraints in minimization problems*, Optimization (R. Fletcher ed.), Academic Press, London, 1969, 283–298.

[78] Powell M.J.D.; *A hybrid method for nonlinear equations*, Numerical Methods for Nonlinear Algebraic Equations (P. Rabinowitz ed.), Gordon & Breach, London, 1970, 87–114.

[79] Rheinboldt W.C.; Methods for Solving Systems of Nonlinear Equations, Second Edition, SIAM, Philadelphia, 1998.

[80] Saad Y.; Iterative Methods for Sparse Linear System, PSW Publ. Co., Boston, 1996.

[81] Saunders M., Tomlin J. A., *Solving regularized linear programs using barrier methods and KKT systems*, Tech. Rep. SOL 96–4, Systems Optimization Lab., Dept. Oper. Res., Stanford University, Stanford CA, 1996.

[82] Shanno D.F., Simantiraki E.M.; *Interior point methods for linear and nonlinear programming*, The State of the Art in Numerical Analysis (I.S. Duff, G.A. Watson eds.), Clarendon Press, Oxford, 1997, 339–362.

[83] Shanno D.F., Vanderbei R.J.; *Interior–point methods for nonconvex nonlinear programming: orderings and higher order methods*, Tech. Rep. SOR 99–5, Stats. Oper. Res., Princeton University, Princeton NJ, 1999.

[84] Sherman A.H.; *On Newton–iterative methods for the solution of systems of nonlinear equations*, SIAM J. Numer. Anal. 15 (1978), 755–771.

[85] Smale S.; *A convergent process of price adjustment and global Newton methods*, J. Math. Econ. 3 (1976), 107–120.

[86] Tits A.L., Wächter A., Bakhtiari S., Urban T.J., Lawrence C.T.; *A primal–dual interior–point method for nonlinear programming with strong global and local convergence properties*, SIAM J. Optim. 14 (2003), 173–199.

[87] Ulbrich M., Ulbrich S., Vicente L.N.; *A globally convergent primal–dual interior–point filter method for nonconvex nonlinear programming*, Tech. Rep. TR00–12, Dept. Comput. Appl. Math., Rice University, Houston TX, 2000.

[88] Ul'm S.; *On iterative methods with the successive approximation of the inverse operator*, Izv. Akad. Nauk Estonskoi SSR, Ser. Fizika–Matematika 16 (1967), 403–411 (in Russian).

[89] Van Loan C.; *On the method of weighting for equality–constrained least–squares problems*, SIAM J. Numer. Anal. 22 (1985), 851–864.

[90] Vanderbei R.J.; *Symmetric quasidefinite matrices*, SIAM J. Optim. 5 (1995) 100–113.

[91] Vanderbei R.J., Shanno D.F.; *An interior–point algorithm for nonconvex nonlinear programming*, Comput. Optim. Appl. 13 (1999), 231–252.

[92] Wächter A.; *An interior point algorithm for large–scale nonlinear optimization with applications in processes engineering*, Ph.D. Thesis, Carnegie Mellon University, Pittsburgh PA, 2002.

[93] Wächter A., Biegler L.T.; *Failure of global convergence for a class of interior point methods for nonlinear programming*, Math. Programming 88 (2000), 565–574.

[94] Wächter A., Biegler L.T.; *On the implementation of an interior–point filter line–search algorithm for large–scale nonlinear programming*, Research Rep. RC 23149, IBM T.J. Watson Research Center, Yorktown Heights NY, 2004.

[95] Waltz R.A., Morales J.L., Nocedal J., Orban D.; *An interior algorithm for nonlinear optimization that combines line search and trust region steps*, Tech. Rep. OTC 2003-6, Optimization Technology Center, Northwestern University, Evanston IL, 2003.

[96] Wang D., Bai Z.Z., Evans D.J.; *On the monotone convergence of multisplitting method for a class of systems of weakly nonlinear equations*, Intern. J. Computer Math. 60 (1996), 229–242.

[97] Wright M.; *The interior–point revolution in optimization: hystory, recent developments, and lasting consequences*, Bull. Amer. Math. Soc. 42 (2005), 39–56.

[98] Yacovlev M.N.; *The solution of systems of non–linear equations by a method of differentiation with respect to a parameter*, USSR Comput. Math. Mathl. Phys. 4 (1964), 146–149.

[99] Yamashita H., Yabe H., Tanabe T.; *A globally and superlinearly convergent primal–dual interior point trust region for large scale constrained optimization*, Tech. Rep., Mathematical Systems Inc., Sinjuku–ku, Tokyo, 1998.

[100] Židkov E.P., Puzynin I.V.; *The stabilization method of solving boundary value problems for nonlinear second–order ordinary differential equations*, Soviet Math. Dokl. 8 (1967), 614–616.