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*

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 an 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 analyse the method of multipliers and the preconditioned conjugate gradient method as inner solvers for interior point schemes. We discuss the convergence of the whole approach, the implementation details and report the results of 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 an interior point method. In particular, we present three effective iterative inner solvers for the solution of the perturbed system that occurs at each step of the interior point scheme.

[*]This research was supported by the Italian Ministry for Education, University and Research (MIUR) projects: FIRB Project: "Parallel Nonlinear Numerical Optimization PN^2O" (grant n. RBAU01JYPN, http://dm.unife.it/pn2o/) and COFIN/PRIN04 Project "Numerical Methods and Mathematical Software for Applications" (grant n. 2004012559, http://www.math.unifi.it/ brugnano/Cofin2004/).
E–mail addresses: bonettini.silvia@unimo.it (S. Bonettini), galligani@unimo.it (E. Galligani), rgv@unife.it (V. Ruggiero).

The first one consists of 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 method of the multipliers we solve a symmetric positive definite system using the efficient library routine of Ng and Peyton that implements Cholesky factorization ([36]).
The second solver is the conjugate gradient method combined with the indefinite preconditioner introduced by Lukšan in [38]; exploiting the block structure of the preconditioning matrix, at each step of the preconditioned conjugate gradient method, we solve, using the Ng and Peyton routine, a symmetric positive definite system.
The third solver also uses the 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 that implements a Cholesky–like factorization with a regularization technique; this routine, introduced in [7] and available on the website *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 simple implementation and 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 approach can be introduced (see [5]).
In subsection 5.3 we analyse with examples convergence failures of the interior point scheme using the Newton iteration, indicating how to detect the failure.
In Section 6, we evaluate the effectiveness of the whole scheme, as in subsection 5.1, on a set of large and sparse nonlinear programming problems that arises from finite difference discretization of elliptic boundary or distributed control problems ([44], [40] and [41]).
These nonlinear problems have quadratic objective function, weakly nonlinear[1] equality constraints and box constraints; the involved matrices have a *PDE–type* structure.
In the experiments, we compare the whole scheme with the three solvers and a direct library routine solver. Moreover, a comparison, in terms of CPU time, with other interior point codes is reported and we also consider the results of the nonmonotone interior point method introduced in [5] with the third, and most efficient, inner solver. The results highlight the efficiency of the BLKFCLT routine.

[1]A continuously differentiable mapping $\boldsymbol{F}(\boldsymbol{u})$ is said to be 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 ([57]), arises from the discretization of many classical semilinear elliptic partial differential equations by the finite difference or finite element methods.

2 Interior point framework

Consider the following nonlinear programming problem

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

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 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 $l_j < u_h$. 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 ([37, Chapt. 10]). We are interested in the case where (1) is a large nonconvex problem and the first and second derivatives of the objective function and the constraints are available.
By introducing slack variables, the problem (1) can be rewritten as

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

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 $\Lambda_2 = diag(\boldsymbol{\lambda}_2)$; $\Lambda_l = diag(\boldsymbol{\lambda}_l)$; $\Lambda_u = diag(\boldsymbol{\lambda}_u)$; $S = diag(\boldsymbol{s})$; $R_l = diag(\boldsymbol{r}_l)$; $R_u = diag(\boldsymbol{r}_u)$.
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

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

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) using 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 drawback, the idea of the interior point method ([21]) is to perturb the system (4) only in the last p equations and generate a sequence of iterates $\{\boldsymbol{v}^{(k)}\}$ satisfying the perturbed system

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

and the Karush–Kuhn–Tucker conditions (4) only in the limit. The perturbation parameter ρ_k tends towards 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 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 the method used to compute the new point at step 2b. Barrier function methods (see e.g. [22, §12.1]) and most of the recent interior point methods (see [58] for one of the last survey papers and the references therein) can also be described by such a scheme.

3 An interior point method as an inexact Newton scheme

The *Newton interior point method* for nonlinear programming (1) is obtained when step 2b of the scheme in 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*

$$H'(\boldsymbol{v}^{(k)})\Delta\boldsymbol{v} = -\boldsymbol{H}(\boldsymbol{v}^{(k)}) + \rho_k \tilde{\boldsymbol{e}} \tag{7}$$

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

$$\mu_k \leq \mu_k^{(2)} \equiv \frac{\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|}{\sqrt{p}}, \tag{8}$$

then the solution $\Delta\boldsymbol{v}^{(k)}$ of the system (7) is a descent direction for $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ ([14]). The system (7) can also be viewed as one step of an inexact Newton scheme ([12], [20]) applied to the exact system $\boldsymbol{H}(\boldsymbol{v}) = 0$, starting from $\boldsymbol{v}^{(k)}$. Indeed, the solution $\Delta\boldsymbol{v}^{(k)}$ of the system (7) satisfies the residual condition of the inexact Newton method that is written as

$$\|H'(\boldsymbol{v}^{(k)})\Delta\boldsymbol{v}^{(k)} + \boldsymbol{H}(\boldsymbol{v}^{(k)})\| \leq \eta_k \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \tag{9}$$

[2]Here and subsequently, the vector norm $\|\cdot\|$ indicates the Euclidean norm

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 perturbation parameter in the interior point method and is strictly connected with the notion of adherence to the central path (e.g. see [21]). Then, the choice of the perturbation parameter

$$\mu_k \in [\mu_k^{(1)}, \mu_k^{(2)}] \tag{10}$$

assures that $\Delta\boldsymbol{v}^{(k)}$ satisfies the residual condition of the inexact Newton method and 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 [15]).
When the size of the system (7) is large, the computation for the exact solution can be too expensive and then it seems convenient the system (7) to be solved *approximately*. We denote $\Delta\boldsymbol{v}^{(k)}$ again as the approximate solution of the system (7). 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.
Then, we can apply an inner iterative scheme to the perturbed Newton equation (7) until the final inner residual

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

satisfies the condition

$$\|\boldsymbol{r}^{(k)}\| \leq \delta_k \|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \tag{12}$$

That is, we introduce a further perturbation. Obviously, the choice $\delta_k = 0$ means the system (7) is solved exactly.
It is possible to prove (see [6, 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 $\Delta\boldsymbol{v}^{(k)}$, which satisfies (11) and (12), is a descent direction for $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ and satisfies the residual condition (9) 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. In particular, we consider a Newton line–search interior point method (or a damped Newton interior point method) that computes the new iterate as follows

$$\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)} + \alpha_k \Delta\boldsymbol{v}^{(k)} \tag{13}$$

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* in a *merit function*, for example of the least squares merit function.

In order to satisfy all these conditions, the damping parameter α_k is determined by the following sequence of steps:

1. feasibility condition: 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 $\Delta\tilde{w}_i^{(k)} < 0$), ${\alpha_k}^{(1)}$ is chosen such that $\tilde{s}_i^{(k+1)} > 0$ (or $\tilde{w}_i^{(k+1)} > 0$);

2. centrality conditions: they are expressed by the nonnegativity of the following functions introduced in [21] (see also [47, p. 402]):

$$\varphi(\alpha) \equiv \min_{i=1,p}\left(\tilde{S}^{(k)}(\alpha)\tilde{W}^{(k)}(\alpha)\boldsymbol{e}_p\right) - \gamma_k\tau_1\left(\frac{\tilde{\boldsymbol{s}}^{(k)}(\alpha)^t\tilde{\boldsymbol{w}}^{(k)}(\alpha)}{p}\right) \geq 0 \quad (14)$$

$$\psi(\alpha) \equiv \tilde{\boldsymbol{s}}^{(k)}(\alpha)^t\tilde{\boldsymbol{w}}^{(k)}(\alpha) - \gamma_k\tau_2\|\boldsymbol{H}_1(\boldsymbol{v}^{(k)}(\alpha))\| \geq 0 \quad (15)$$

   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 $\tilde{\alpha}_k$ as large as possible such that conditions (14)–(15) 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 the inequalities (14) and (15) at the starting iteration, we have $\tau_1 \leq \frac{\min_{i=1,p}\left(\tilde{S}^{(0)}\tilde{W}^{(0)}\boldsymbol{e}_p\right)}{\left(\frac{\tilde{\boldsymbol{s}}^{(0)t}\tilde{\boldsymbol{w}}^{(0)}}{p}\right)}$, 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

$$\tau_1 = \min\left(0.99, \frac{10^{-7}\cdot\min_{i=1,p}\left(\tilde{S}^{(0)}\tilde{W}^{(0)}\boldsymbol{e}_p\right)}{0.5\cdot\left(\frac{\tilde{\boldsymbol{s}}^{(0)t}\tilde{\boldsymbol{w}}^{(0)}}{p}\right)}\right); \quad \tau_2 = 10^{-7}\cdot\frac{\tilde{\boldsymbol{s}}^{(0)t}\tilde{\boldsymbol{w}}^{(0)}}{\|\boldsymbol{H}_1(\boldsymbol{v}^{(0)})\|} \quad (16)$$

3. sufficient decrease in the merit function $\|\boldsymbol{H}(\boldsymbol{v})\|^2$: it can be obtained by implementing the Armijo backtracking procedure as in [21] or the one in [14]

```
• Set $\beta \in (0,1)$, $\theta \in (0,1)$, $\alpha = {\alpha_k}^{(2)}$;
• while $\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha\Delta\boldsymbol{v}^{(k)})\| > (1 - \beta\alpha(1 - (\sigma_k + \delta_k)))\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$
      $\alpha \leftarrow \theta\alpha$
  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 region defined by the bound constraints, 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 $\alpha_k \geq \bar{\alpha} > 0$ and the vector $\alpha_k\Delta\boldsymbol{v}^{(k)}$ satisfies the norm condition of the inexact Newton method:

$$\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha_k\Delta\boldsymbol{v}^{(k)})\| \leq \xi_k\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$$

with $0 < \xi_k \leq \bar{\xi} < 1$ and $\bar{\xi} = (1 - \beta\bar{\alpha}(1 - (\sigma_{\max} + \delta_{\max}))) < 1$.
Moreover, from (8), (11) and (12), 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 (9) 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 subsection 5.2, we report the convergence results for the Newton line–search interior point method described in Section 3 (see [6]). They are based on convergence results of inexact Newton methods.
Furthermore, in the context of the nonmonotone inexact Newton method ([5]), choices for α_k can be used in the interior point scheme, that allow a nonmonotone behavior of the merit function.
Indeed, let $\boldsymbol{v}^{(\ell(k))}$ be the element of the nonmonotone interior point 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.*
If we choose the parameter μ_k in the larger interval with respect to the one in (10)

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

then the direction $\Delta\boldsymbol{v}^{(k)}$, computed by approximately solving the system (7), satisfies the condition

$$\|H'(\boldsymbol{v}^{(k)})\Delta\boldsymbol{v}^{(k)} + \boldsymbol{H}(\boldsymbol{v}^{(k)})\| \leq (\sigma_k + \delta_k)\|\boldsymbol{H}(\boldsymbol{v}^{(\ell(k))})\|$$

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

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

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 larger stepsizes than in the monotone case to be retained.
Convergence properties and numerical experiences of this nonmonotone interior point method are investigated in [5].

4 Iterative solvers for interior point iteration

We focus our attention on the solution of the linear system (7) that, by omitting the iteration index k, can be written as

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

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 solutions of the system in a *condensed form*

$$\begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix}\begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda}_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{q} \end{pmatrix} \tag{17}$$

with

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

The system (17) is symmetric and indefinite and can be solved by sparse Bunch–Parlett triangular factorization ([8]), that combines dynamic reordering for the sparsity preserving and pivoting technique for numerical stability (such as the algorithm implemented in the MA27 routine of HSL Library ([31])) or by considering an *inertia–controlling factorization* ([23]).
For systems arising from large scale nonlinear programming problems, the use of direct methods can be very expensive and memory consuming. Then, in the framework of direct methods, different approaches have been devised to avoid the use of direct solvers. Some interior point schemes reduce (by elimination techniques like the one above) the symmetric system (7) into a *quasidefinite* form[3] ([53]), that allows a Cholesky–like factorization. This factorization is more convenient since it avoids the use of pivoting techniques (for the numerical stability, see [28]). Furthermore, it enables an a priori determination of a sparsity preserving reordering of the coefficient matrix (taking only its structure into account) and the symbolic Cholesky factor. Then, at each iteration, only the computation of the Cholesky factor has to be performed, saving a lot of CPU time. The reduction of a coefficient matrix into a quasidefinite form can be obtained by a *regularization* technique, consisting of adding a convenient diagonal matrix $\tilde{D}$ (e.g. see [54], [51], [2]) to this matrix. In [2] the matrix $\tilde{D}$ can be dynamically determined throughout the computation of the Cholesky factor: when a critical pivot is reached, this is perturbed by a small quantity with a convenient sign. This also prevents numerical instability.
This regularization approach requires the implementation of additional recovery procedures involving several factorizations; for example to determine a perturbation that is as small as possible ([54]) or to implement an iterative refinement if the computed solution of the perturbed system is not satisfactory ([2]).
A different approach that avoids perturbations of the matrices of the subproblems is the use of iterative inner solvers for (17), that exploit the sparsity of the involved matrices, and*approximately* solve the inner subproblems avoiding unnecessary inner iterations when we are far from the solution (i.e. at the initial outer iterations).
As seen in the previous section, the Newton line–search interior point method (13) 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 local superlinear convergence of the whole outer–inner scheme ([6], [19]).
We remark that, when the solution $\Delta \boldsymbol{v}^{(k)}$ is computed by approximately solving the perturbed Newton equation (7) rewritten in the condensed form (17), the further perturbation that the inner solver introduces on the residual (11), only appears in the first two block–rows. That is, if we partition the residual $\boldsymbol{r}^{(k)}$

[3]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).

commensurately as $\boldsymbol{v}^{(k)}$, we have

$$\boldsymbol{r}^{(k)} = \begin{pmatrix} \boldsymbol{r}_1^{(k)} \\ \boldsymbol{r}_2^{(k)} \\ 0 \\ 0 \end{pmatrix}; \qquad \begin{pmatrix} \boldsymbol{r}_1^{(k)} \\ \boldsymbol{r}_2^{(k)} \end{pmatrix} = \begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda}_1 \end{pmatrix} + \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{q} \end{pmatrix} \tag{18}$$

Here, remember 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 two different iterative methods and discuss their implementation in the interior point scheme.

4.1 The method of multipliers

Suppose that the matrices A and B of the system (17) 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

$$M = \begin{pmatrix} A & B \\ B^t & 0 \end{pmatrix} \tag{19}$$

is nonsingular ([37, p. 424]). We note that these assumptions are the standard assumptions for the local sequential quadratic programming (SQP) method ([47, p. 531])
The system (17) can be viewed as the Lagrange necessary conditions for the minimum point of the following quadratic problem

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

This quadratic problem can be solved efficiently using the method of multipliers [4].
Starting from $\Delta\boldsymbol{\lambda}_1^{(0)} = 0$ and $\Delta\boldsymbol{x}^{(0)} = 0$, the method consists of updating the dual variable from the rule

$$\Delta\boldsymbol{\lambda}_1^{(\nu+1)} = \Delta\boldsymbol{\lambda}_1^{(\nu)} + \chi(B^t\Delta\boldsymbol{x}^{(\nu)} - \boldsymbol{q}) \tag{20}$$

[4] The method of multipliers [33, Chapt. 5, §10, p. 307], was originally suggested by Hestenes in [32]; an equivalent method motivated from a different viewpoint has been proposed by Powell in [48]. See [37, Chapt. 13] for the dual viewpoint of the method.
In [25] it is shown that the method of multipliers for equality constrained least squares problems is equal to the method of weighting [52] for a particular choice of the starting point.

where χ is a positive parameter (penalty parameter) and $\Delta\boldsymbol{x}^{(\nu)}$ minimises the augmented Lagrangian function of the quadratic problem

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

This means that $\Delta\boldsymbol{x}^{(\nu)}$ is the solution of the linear system of order n

$$(A + \chi BB^t)\boldsymbol{\Delta x} = -B\Delta\boldsymbol{\lambda}_1^{(\nu)} + \boldsymbol{c} + \chi B\boldsymbol{q} \tag{21}$$

We remark that if we premultiply the *augmented system* (17) for an appropriate matrix, we have

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

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

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

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 apply the Gauss-Seidel method (SOR–like method in [30] with $\omega = 1$, see also [35]), we obtain the two iterations of the method of multipliers (20)–(21) (see also [29]).
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 and therefore the matrix A is positive definite on the null space of BB^t.
Then, from the theorem in ([37, 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 (21) by applying a Cholesky factorization.
Even though we do not have an analytical way of determining 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 $\boldsymbol{x}^tA\boldsymbol{x} > 0$.
If $B^t\boldsymbol{x} \neq 0$, $\boldsymbol{x}^tBB^t\boldsymbol{x} > 0$. Then, it follows that

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

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:

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

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 an approximation of $\tau_{\min}$ ([26]).
As pointed out in [29], this tentative value may be a good choice, since it is fairly close to the minimum of the condition number of $A + \chi BB^t$ with respect to χ.
Furthermore, in order to avoid the value of χ being too small (the matrix is not positive definite) or too large (too ill–conditioned system), it is convenient to use safeguards:

$$\chi = \min(\max(\chi_{\min}, \frac{\max\{\|A\|_F, 1\}}{t_{min}}), \chi_{\max}) \tag{22}$$

For the test problems considered in the section *Numerical Experiments*, $\chi_{\min} = 10^7$ and $\chi_{\max} = 10^8$.
For an analysis of the conditioning of the system (21) and on the behaviour of the method of multipliers with a *normalization matrix* see [26] and [13, §6].
Let us consider the implementation details of the method, assuming that the matrices Q and B^t are stored in a column compressed format ([50]).

In the cases of the test problems in the numerical experiments section, the inequality constraints are box constraints, then the matrices A and Q have the same structure and differ only in the diagonal entries.
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 $B(-\Delta\boldsymbol{\lambda}_1^{(\nu)} + \chi\boldsymbol{q})$ and $B^t\Delta\boldsymbol{x}^{(\nu)}$ 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, executing 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 the $\mathcal{O}(n^2/2)$ entries of the lower part of T and for each (i,j) entry, we have to identify 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 minimise the fill–ins in L_n and the supernodal block factorization enables us to take advantage of the presence of the cache memory in modern computer architecture ([36]).

Thus, in the numerical results, the time for solving a nonlinear programming problem using the interior point scheme combined with the method of multipliers can be subdivided into two parts: 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 denote 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 (17) that occurs at each iteration of an interior point method, is the preconditioned conjugate gradient (PCG) method (see e.g. [4], [11], [16], [17], [27], [38], [39], [34]).
The PCG method, with the preconditioning matrix $\bar{M}$, for the solution of the system (17)

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

where M is given by (19), $\boldsymbol{y} = (\Delta\boldsymbol{x}^t, \Delta\boldsymbol{\lambda}_1^t)^t$ and $\boldsymbol{b} = (\boldsymbol{c}^t, \boldsymbol{q}^t)^t$ requires at any iteration the computation of the matrix–vector product $M\boldsymbol{p}$ and the solution of the linear system $\bar{M}\boldsymbol{d} = \hat{\boldsymbol{r}}$ ([47, p. 118])
The matrix–vector product $M\boldsymbol{p}$ does not require the explicit computation of the matrix A of system (17); 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 $\tilde{S} = diag(\tilde{\boldsymbol{s}})$ and $\tilde{W} = diag(\tilde{\boldsymbol{w}})$, 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 [38]:

$$\bar{M} = \begin{pmatrix} \bar{A} & B \\ B^t & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ B^t\bar{A}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A} & 0 \\ 0 & -B^t\bar{A}^{-1}B \end{pmatrix} \begin{pmatrix} I & \bar{A}^{-1}B \\ 0 & I \end{pmatrix} \tag{23}$$

where $\bar{A}$ is a positive diagonal approximation of A. We refer to [38] for the spectral properties of the conjugate gradient method with this preconditioner.
The diagonal matrix $\bar{A} = diag(\bar{a}_{ii})$ is chosen as follows

$$\bar{a}_{ii} = \begin{cases} a_{ii} = q_{ii} + \sum_{j=1}^{p} c_{ij}^2 \tilde{w}_j/\tilde{s}_j & \text{if } a_{ii} > 10^{-8} \\ 1.5 \cdot 10^{-8} & \text{otherwise.} \end{cases} \quad i = 1, ..., n \tag{24}$$

where $\tilde{w}_j$ and $\tilde{s}_j$, $j = 1, ..., p$ are the components 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.
The solution of the linear system

$$\bar{M}\boldsymbol{d} = \hat{\boldsymbol{r}} \tag{25}$$

where $\bar{M}$ is given by (23), $\boldsymbol{d} = (\boldsymbol{d}_1^t, \boldsymbol{d}_2^t)^t$ and $\hat{\boldsymbol{r}} = (\hat{\boldsymbol{r}}_1^t, \hat{\boldsymbol{r}}_2^t)^t$, $(\boldsymbol{d}_1, \hat{\boldsymbol{r}}_1 \in \mathbb{R}^n$, $\boldsymbol{d}_2, \hat{\boldsymbol{r}}_2 \in \mathbb{R}^{neq})$ can be carried out in two different ways.
The first method exploits the block structure of the matrix (23) while the second solves the system (25) directly by introducing a regularisation technique on the preconditioning matrix $\bar{M}$, in order to assure that it allows a Cholesky–like factorization.
The two different techniques used to compute the solution of system (25) 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 $\bar{M}^{-1}$ from (23), the solution of (25) 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 its symbolic Cholesky factor.

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

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

Nevertheless, the formation of the data structure phase can be expensive since we have to investigate the $\mathcal{O}(neq^2/2)$ entries of the lower part of T and for each (i, j) entry, we have to identify the nonzero pairs among the n pairs of elements of the i–th and j–th columns of B.
We denote IP–PCG1, the interior point method with the preconditioned conjugate gradient as inner solver, with preconditioning matrix $\bar{M}$ as in (23), and the solution of the system (25) computed as described above.

The other method used to compute the solution of the system (25) uses the property that the matrix $\bar{M}$ can be factorized in a Cholesky–like form

$$L_{n+neq} D L_{n+neq}^t, \tag{26}$$

where L_{n+neq} is a lower triangular matrix with diagonal entries equal to one and D is a nonsingular diagonal matrix. In this way, the computation of the product $B^t \bar{A}^{-1} B$ can be avoided but the factorization applied to $\bar{M}$ can produce many fill–ins. Furthermore, the application of a minimum degree reordering to $\bar{M}$ does not ensure that the symmetrically permuted matrix $P \bar{M} P^t$ can be factorized in the Cholesky–like form.
Nevertheless, using the regularisation technique described in [2] for $\bar{M}$, we can compute the Cholesky–like factorization of a new matrix $\bar{\bar{M}}$ given by

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

where R_1 and R_2 are nonnegative diagonal matrices such that $P \bar{\bar{M}} P^T$ admits a factorization of the form (26). The computation of R_1 and R_2 can be dynamically obtained during the factorization procedure in order to reduce the perturbation. If a pivot d_i is too small ($|d_i| < 10^{-15} \max_{j<i} |d_j|$), we put $d_i = \sqrt{eps}$ if $1 \le i \le n$, or $d_i = -\sqrt{eps}$ if $n + 1 \le i \le n + neq$, where eps is the machine precision.

This approach is used in [4] for linear and quadratic programming problems with equality and box constraints.
The Cholesky–like factorization of $\bar{\bar{M}}$ can be obtained by modifying the Ng and Peyton package that maintains the efficient use of the cache memory. This package, called BLKFCLT, introduced in [7], can be downloaded from the website *dm.unife.it/blkfclt*.
We denote IP–PCG2, the interior point method with the preconditioned conjugate gradient as the inner solver, with preconditioning matrix $\bar{M}$, as in (23), and the solution of the system (25) 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.
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 (16) with $\gamma_k = \frac{1}{2}$; set the tolerance $\epsilon_{\text{exit}} > 0$;
- for $k = 0, 1, \ldots$ until the stopping rule is satisfied:
  - 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 (17) with a direct or an iterative process. In this last case, the stopping rule satisfies the condition (12), 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 is computed as in (18);
  - 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. ($\hat{\theta} < 1$):

$$\alpha \equiv \alpha_k^{(1)} = \min\left(\min\left(\min_{\Delta\tilde{s}_i^{(k)}<0} \frac{-\tilde{s}_i^{(k)}}{\Delta\tilde{s}_i^{(k)}}, \min_{\Delta\tilde{w}_i^{(k)}<0} \frac{-\tilde{w}_i^{(k)}}{\Delta\tilde{w}_i^{(k)}}\right)\hat{\theta}, 1\right) \quad (27)$$

  - if necessary reduce the parameter $\alpha_k^{(1)}$, by multiplying by a positive factor $\breve{\theta} < 1$, until the centrality conditions (14)–(15) are satisfied[5]. Denote $\alpha_k^{(2)}$ the obtained value;

[5]In the experiments, we use an adaptive rule for $\hat{\theta}$ as in [3]; that is, if the result of the minimum in (27) is less then 1 we have

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

– apply the backtracking procedure [14]:
```
set $\alpha = {\alpha_k}^{(2)}$;
while $\|\boldsymbol{H}(\boldsymbol{v}^{(k)} + \alpha\Delta\boldsymbol{v}^{(k)})\| > (1 - \beta\alpha(1 - (\sigma_k + \delta_k)))\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\|$
    $\alpha \leftarrow \theta\alpha$
endwhile
Denote $\alpha_k$ the value of $\alpha$ 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

$$\|\boldsymbol{H}(\boldsymbol{v}^{(k)})\| \le \epsilon_{\text{exit}}$$

5.2 Convergence of the 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 the inexact Newton method for solving Karush–Kuhn–Tucker systems.
Given $\epsilon \ge 0$, we define

$$\begin{aligned}\Omega(\epsilon) \quad = \quad & \Big\{\boldsymbol{v} : 0 \le \epsilon \le \|\boldsymbol{H}(\boldsymbol{v})\|^2 \le \|\boldsymbol{H}(\boldsymbol{v}^{(0)})\|^2, \text{ s. t.} \\ & \min_{i=1,p}\left(\tilde{S}\tilde{W}\boldsymbol{e}_p\right) \ge \frac{\tau_1}{2}\left(\frac{\tilde{\boldsymbol{s}}^t\tilde{\boldsymbol{w}}}{p}\right) \\ & \tilde{\boldsymbol{s}}^t\tilde{\boldsymbol{w}} \ge \frac{\tau_2}{2}\|\boldsymbol{H}_1(\boldsymbol{v})\|\Big\}\end{aligned} \tag{28}$$

Let us assume that the following conditions hold [18] (see also [21] and [14]):

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 $\Omega_{\tilde{\boldsymbol{s}}}$, the matrix $H'(\boldsymbol{v})$ is nonsingular. The set $\Omega_{\tilde{\boldsymbol{s}}}$ is a compact subset of $\Omega(0)$ where $\tilde{\boldsymbol{s}}$ is bounded away from zero.

The condition C3 is equivalent to the condition that the matrix M of (19) is nonsingular for any $\Omega_{\tilde{\boldsymbol{s}}}$.
In general, in the literature, the condition C3 is replaced by a sufficient condition to ensure that C3 holds.

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 $\breve{\theta} = 0.5$.

For example, a sufficient condition is to require that, for any $\Omega_{\tilde{\boldsymbol{s}}}$, 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 $\Omega_{\tilde{\boldsymbol{s}}}$, the matrices A and $B^tA^{-1}B$ are nonsingular.
The boundedness of the sequence $\{\boldsymbol{x}^{(k)}\}$ can be ensured by enforcing box constraints $-l_i \leq x_i^{(k)} \leq l_i$ for a sufficiently large $l_i > 0,\ i = 1, ..., n$.
The proof of the global convergence of the sequence $\{\boldsymbol{v}^{(k)}\}$ generated by the damped Newton interior point method consists of showing the following: given any $\epsilon > 0$, as long as the iteration sequence $\{\boldsymbol{v}^{(k)}\}$ satisfies

$$\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon),\ \epsilon > 0,$$

then, the step sequence $\{\Delta\boldsymbol{v}^{(k)}\}$ and the steplength sequence $\{\alpha_k\}$ are uniformly bounded above and away from zero, respectively. Following the convergence theory of the inexact Newton methods, we obtain the convergence of the algorithm.
For the sequence $\{\boldsymbol{v}^{(k)}\}$ generated by the damped Newton interior point algorithm, the following statements hold ([21], [6]):

(a) $\Omega(\epsilon)$, $\epsilon \geq 0$, is a closed set;

(b) the sequence $\{\boldsymbol{v}^{(k)}\} \subset \Omega(0)$;

(c) when $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$ with $\epsilon > 0$, the sequences $\{\tilde{\boldsymbol{s}}^{(k)^t}\tilde{\boldsymbol{w}}^{(k)}\}$ and $\{\tilde{s}_i^{(k)}\tilde{w}_i^{(k)}\}$, $i = 1, \ldots p$, are bounded above and below away from zero;

(d) when $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$, with $\epsilon > 0$, then $\{\boldsymbol{v}^{(k)}\}$ is bounded above and the sequences $\{\tilde{s}_i^{(k)}\}$ and $\{\tilde{w}_i^{(k)}\}$ are bounded away from zero;

(e) when $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$, with $\epsilon > 0$, the sequence of matrices $\{H'(\boldsymbol{v}^{(k)})^{-1}\}$ is bounded and, since $\sigma_k + \delta_k < 1$, the sequence of search steps $\{\Delta\boldsymbol{v}^{(k)}\}$ is bounded.

For the proofs of (a), (b), (c), see [21]; for (d), (e) see the proof of Theorem 2 in [6].
Hence, as a consequence, we will 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, path–following conditions and backtracking reduction, we separate the analysis into three steps.

1. (*Feasibility*) It is easy to see that $\alpha_k^{(1)}$ in (27) 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*) When the sequence of the damped Newton interior point method $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$, $\epsilon > 0$, since $\sigma_k \in [\sigma_{\min}, \sigma_{\max}] \subset (0, 1)$ and $\delta_k \in [0, \delta_{\max}] \subset [0, 1)$, and
$$\sigma_k > \delta_k(1 + \gamma_k\tau_2) \tag{29}$$
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 $\alpha \in (0, \check{\alpha}_k^{(2)}]$;
- 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 $\alpha \in (0, \hat{\alpha}_k^{(2)}]$.

For the proof, see the one in Theorem 3 in [6], which runs like that of Lemma 6.3 in [21].

Thus we have
$$\min\{\hat{\alpha}_k^{(2)}, \check{\alpha}_k^{(2)}, 1\} \in (0,1] \geq \tilde{\alpha} > 0$$
Then, $\alpha_k^{(2)} \geq \alpha^{(2)} = \min\{\alpha^{(1)}, \tilde{\alpha}\} > 0$.

In [6, Prop. 1] it is proved 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.

3. (*Backtracking*) As shown in [6, Theor. 4], under the conditions C1, C2 and C3, as long as the iteration sequence $\{\boldsymbol{v}^{(k)}\} \subset \Omega(\epsilon)$, $\epsilon > 0$, the *while loop* of the backtracking procedure of the damped Newton interior point algorithm, terminates in a finite number of steps (see also [20, 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 residual conditions of the inexact Newton method.
Hence, here we report the convergence theorem for the damped Newton interior point method, based on the fundamental convergence theorem of the inexact Newton method ([49, Theor. 6.7]).

Theorem. 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) 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 from $\{\boldsymbol{v}^{(k_i)}\}$, 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 from Theorem 6.7 in [49, p. 70] (see also Theorem 6.1 in [20]), 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 in Theorem 5 in [6]. $\square$

5.3 On the global convergence

In this subsection, we briefly make some remarks on cases of global convergence failure of the damped Newton interior point method. Obviously, when it happens, 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 [56] where it is stressed that algorithms which use the Newton direction could fail:

$$\begin{array}{l} \min x \\ x^2 \geq -a \\ x \geq b \end{array}$$

As pointed out in [18, 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, as observed in [18], the sufficient condition on the boundedness of the sequence of the inequality 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 towards 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 increase in the values of the sequence $\{\tilde{\boldsymbol{w}}^{(k)}\}$ shows that the sequence of the solution could not converge and then, another choice of initial point is recommended.
Moreover, we observe that the sufficient condition (C4) in [21] of linear independence of the gradients of the active constraints is violated here, either if we start from a *bad* or *good* initial point. Thus, the condition on the boundedness of the inequality multipliers is more general with respect to the condition (C4) in [21].

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 1: Polyalgorithm for Example 3.1 in [18]

An example of large scale nonlinear programming problem, for which the damped Newton interior point method gives the same behaviour, is still given in [18, Example 3.2].
A way to compute a suitable starting point could be to execute some steps of the gradient projection method (e.g. see [37, §11.4, p. 330]), before starting with the damped Newton interior point method.
In Table 1, we report the number of iterations and the values of the primal and dual variables of the sequence generated by the *polyalgorithm* composed of the gradient projection method, with the Armijo backtracking procedure for the merit function $\|\boldsymbol{H}(\boldsymbol{v})\|^2$ and, the damped Newton interior point method for Example 3.1 in [18].
The result shows that only two iterations ν of the gradient projection method are necessary *to enter* a *good* region for the damped Newton interior point method.

Finally, we consider Example 3 of nonlinear programs in [10]

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

where it is stressed that Newton iteration with a decrease in 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 $\|\boldsymbol{H}(\boldsymbol{v})\|^2$, at the iteration k, in [24] it is shown that,

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

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.
If 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 in this

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 2: Newton interior point method for Example 3 in [10]

example, as shown in Table 2. The starting point of the example is $x = 2$, $\lambda_l = 1$ and $r_l = 1$ and the backtracking rule used is the one in [14].
The results of Tables 1 and 2 were obtained using a Matlab code which directly solves the inner linear system that occurs at each Newton iteration. The forcing term η_k is equal to $\eta_k = 1 - \alpha_k(1 - \sigma_k)$.

6 Numerical experiments

In this section we consider a set of nonlinear programming test problems that arise from discretization with finite difference formulae of boundary and distributed elliptic control problems ([40], [41], [44], [42]).
All the problems have 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 $\nabla\boldsymbol{g}_2(\boldsymbol{x})S^{-1}\Lambda_2\nabla\boldsymbol{g}_2(\boldsymbol{x})^t + P_l^tR_l^{-1}\Lambda_lP_l + P_u^tR_u^{-1}\Lambda_uP_u$ is diagonal and the computation of this matrix in block A of (17) is inexpensive.
In Table 3, 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 Table 4, 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 matrices B and Q respectively.
In Table 5 we report some results, obtained by Hans Mittelmann at the Arizona State University [43], of a comparison among interior point codes for nonlinear programming: the LOQO algorithm, version 6.2 ([54]), the KNITRO algorithms, version 3.1 ([9]) with the direct (KNITRO–D) or with the iterative (KNITRO–I) solution ([46]) of inner subproblems and IPOPT algorithm ([55]).

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.
See [45] for a comparison of interior point codes and active set sequential quadratic programming codes on CUTE collection test problems.
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 have been compiled using a "+O3" optimization level of the HP compiler.
In this section we only report the results related to a few test problems; the details of the numerical experiments and a complete set of results for all the test problems in [40], [41] and [44] can be downloaded from *dm.unife.it/blkfclt*.
In the experiments, we set the starting point of the damped Newton interior point method as follows: the initial values for the multipliers and 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 [44]. Moreover, for the codes employing an iterative method as the inner solver, the initial value of the inner iterations was fixed equal to the null vector.
All the results in this section were obtained using the choice $\mu_k = \mu_k^{(1)} = {\tilde{\boldsymbol{s}}^{(k)}}^t \tilde{\boldsymbol{w}}^{(k)}/p$.
Furthermore, the maximum value of inner iterations was 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 parameters σ_k, δ_k were 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 as 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 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 Table 6, 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 6, 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 equal to 0.1% at most, while the ratio of the nonzero entries of the Cholesky factor and the lower triangular part of $A+\chi BB^t$ is equal to 15.3 at most.

In Table 7, we evaluate the effectiveness of the different versions of the code, implementing the damped Newton interior point method with the iterative inner solvers with $\epsilon_{\text{exit}} = 10^{-8}$. 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 into the time required by the preprocessing routine in the computation of the matrix structures ("prep") and 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.
The code with the direct inner solver MA27 (with pivot tolerance equal to 10^{-8}) allows the chosen test problems to be solved with N=99 (27.38 secs and it=29 for P1-1; 22.52 secs and it=24 for P1-3; 24.71 secs and it=25 for P2-6) and with N=199 (349.66 secs and it=37 forP1-1; 250.28 secs and it=27 for P1-3; 304.11 secs and it=26 for P2-6). For N> 299 we observe a memory failure after a few iterates due to the factor fill–ins.
The minimum values of the objective functional computed by the codes differ by a factor of 10^{-8} and are in accordance with the values reported in [40], [41], [44].
From the numerical experiments, we can draw the following remarks:

- the number of inner iterations per outer iteration is very small for all three inner solvers; we observe that in the experiments the total number of inner iterations is sometimes 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 (12) with $\boldsymbol{r}^{(k)}$ as in (18) 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 *causing* 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 occur 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 for the IP–PCG1 code is smaller than the one of IP–MM code, since the size of the matrix to preprocess is neq with 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 half 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 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 $\bar{\bar{M}}$ 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 the IP–PCG2 code is that it requires relatively little memory storage; this allows us to solve very large scale problems up to one million primal variables.

The results in Tables 8, 9, 10, 11 and 12 show the performance of the IP–PCG2 code ($\epsilon_{\text{exit}} = 10^{-8}$) and the direct and iterative versions of KNITRO (version 4.0.2), with different values of the tolerance parameter ("opttol"). We report the computed minimum $f(\boldsymbol{v}^{(it)})$, value of $\|\boldsymbol{H}(\boldsymbol{v}^{(it)})\|$, execution time in seconds ("time") and number of outer iterations ("it"). For the 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 uses the same input AMPL models of the discretized PDE problems as KNITRO. The primal variables are initialized the same way as in the AMPL models. Then the execution time and number of iterations of IP-PCG2 are different with respect to those reported in Table 7.
This comparison highlights the good stability and efficiency of the IP–PCG2 method on this kind of test problem for large scale problems.
Finally, in Table 13 we report 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

Test problems	References
Elliptic boundary control problems	
P1-1	Example 5.5 in [40]
P1-3	Example 5.7 in [40]
Elliptic distributed control problems	
P2-1	Example 1 in [41]
P2-6	Example 4.2 in [44] 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 [44] 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 3: Description of test problems

$M = 1$, we have the monotone algorithm. From Table 13, we observe that an improvement in 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 the results in Table 5 and to the anonymous referee who stimulated us with his comments to improve the paper.

References

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

[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., 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.

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

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

[6] 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. 168 (2005), 651–676.

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

	N	n	neq	nu	nl	nnzjac	nnzhess
P1-1	99	10593	10197	10593	10593	50193	10593
	199	41193	40397	41193	41193	200393	41193
	299	91793	90597	91793	91793	450593	91793
	399	162393	160797	162393	162393	800793	162393
	499	252993	250997	252993	252993	1250993	252993
	599	363593	361197	363593	363593	1801193	363593
P1-3	99	10593	10197	10593	396	50193	10197
	199	41193	40397	41193	796	200393	40397
	299	91793	90597	91793	1196	450593	90597
	399	162393	160797	162393	1596	800793	160797
	499	252993	250997	252993	1996	1250993	250997
	599	363593	361197	363593	2396	1801193	361197
P2-1	99	19602	9801	19602	9801	58410	19602
	199	79202	39601	79202	39601	236810	79202
	299	178802	89401	178802	89401	535210	178802
	399	318402	159201	318402	159201	953610	318402
	499	498002	249001	498002	249001	1492010	498002
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 4: Parameters of test problems

	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 5: Comparison on elliptic control test problems ([43])

	N	nnz-mat1	L-mat1	nnz-mat2	L-mat2	nnz-mat3	L-mat3
P1-1,P1-3	99	70783	622759	69991	621571	60786	718637
	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
P2-1,P2-6,	99	126029	715465	67619	619595	78012	735071
P2-7	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

Table 6: Nonzero entries of the matrices and Cholesky factors

		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
P1-3	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
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
P2-6	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 7: Numerical results: boundary and distributed control problems

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\lVert\boldsymbol{H}(\boldsymbol{v}^{(it)})\rVert$	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 8: Comparison of IP–PCG2 with KNITRO v4.0.2 on test problem P1-1

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\Vert\boldsymbol{H}(\boldsymbol{v}^{(it)})\Vert$	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 9: Comparison of 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 10: Comparison of 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 11: Comparison of IP–PCG2 with KNITRO v4.0.2 on test problem P2-6
[1] Current solution estimate cannot be improved by the code.

N	Solver	opttol	$f(\boldsymbol{x}^{(it)})$	$\lVert \boldsymbol{H}(\boldsymbol{v}^{(it)}) \rVert$	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 12: Comparison of 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 13: IP–PCG2 with nonmonotone choices

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

[9] 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.

[10] 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.

[11] 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.

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

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

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

[15] 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.

[16] 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.

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

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

[19] 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.

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

[21] 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.

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

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

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

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

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

[27] 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.

[28] 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.

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

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

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

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

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

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

[35] 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.

[36] 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.

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

[38] 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.

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

[40] 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.

[41] 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.

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

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

[44] 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.

[45] 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.

[46] 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.

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

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

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

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

[51] 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.

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

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

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

[55] 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.

[56] 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.

[57] 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.

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