Nonsmooth Newton's Method: Some Structure Exploitation

Alberto De Marchi[0000−0002−3545−6898] and Matthias Gerdts[0000−0001−8674−5764]

Department of Aerospace Engineering
Bundeswehr University Munich
Werner-Heisenberg-Weg 39
85577 Neubiberg, Germany
alberto.demarchi@unibw.de
matthias.gerdts@unibw.de

Abstract. We investigate real asymmetric linear systems arising in the search direction generation in a nonsmooth Newton's method. This applies to constrained optimisation problems via reformulation of the necessary conditions into an equivalent nonlinear and nonsmooth system of equations. We propose a strategy to exploit the problem structure. First, based on the sub-blocks of the original matrix, some variables are selected and ruled out for a posteriori recovering; then, a smaller and symmetric linear system is generated; eventually, from the solution of the latter, the remaining variables are obtained. We prove the method is applicable if the original linear system is well-posed. We propose and discuss different selection strategies. Finally, numerical examples are presented to compare this method with the direct approach without exploitation, for full and sparse matrices, in a wide range of problem size.

Keywords: Structure Exploitation · Linear Algebra · Nonsmooth Newton's Method · Nonlinear Optimization.

1 Introduction

In this paper, we consider the real square nonsymmetric possibly large sparse linear system

$$\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{A}^\top & \boldsymbol{C}^\top \\ \boldsymbol{A} & & \\ -\boldsymbol{S}\boldsymbol{C} & & \boldsymbol{T} \end{bmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{g} \\ \boldsymbol{h} \end{pmatrix} \tag{1}$$

where $\boldsymbol{Q} \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{A} \in \mathbb{R}^{n_a \times n_x}$, $\boldsymbol{C} \in \mathbb{R}^{n_c \times n_x}$ and $\boldsymbol{S}, \boldsymbol{T} \in \mathbb{R}^{n_c \times n_c}$ are given matrices and $\boldsymbol{f} \in \mathbb{R}^{n_x}$, $\boldsymbol{g} \in \mathbb{R}^{n_a}$, $\boldsymbol{h} \in \mathbb{R}^{n_c}$ are given vectors (n_x, n_a, n_c being some positive integers); $\boldsymbol{S}$ and $\boldsymbol{T}$ are non-zero diagonal matrices. The contribution of this paper is the exploitation of the structure in problem (1) and its transformation into a smaller symmetric linear system, with a saddle-point structure, from which the solution to (1) can be easily recovered. In particular, two stages are discussed. First, a reduction step generates a smaller linear system and a way

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

to recover eliminated variables from the solution of this reduced system. This step exploits the fact that matrix $\boldsymbol{T}$ is diagonal, and then symbolically solve for (some of) the variables in $\boldsymbol{z}$. Several different reduction strategies are discussed and compared. The second step aims at rewriting the linear system in a symmetric form, allowing to adopt solvers for symmetric systems, which are usually more time and memory efficient. Despite these advantages, some computational overhead is needed, especially in the reduction step, which might introduce a break-even point, that is, this exploitation may pay off, e.g., in terms of computational time, only under certain conditions, e.g., large instances. In fact, an optimal reduction strategy might exists, depending on the specific properties of the problem; indeed, it may even depend on the specific values of the entries. Throughout the paper, we investigate the influence of the reduction strategy on the performance of the aforementioned two-steps exploitation; however, a detailed optimization of the reduction policy is beyond the scope of this paper. We point out that the proposed method could be combined with constraint-reduction approaches as, e.g., those presented in [5,12].

Once the original problem (1), say $\boldsymbol{V}\boldsymbol{d} = \boldsymbol{r}$ for brevity, has been transformed, a reduced symmetric linear system, say $\hat{\boldsymbol{V}}\hat{\boldsymbol{d}} = \hat{\boldsymbol{r}}$, is to be solved. To this end, any method can be adopted. The choice may depend on the problem, in particular on its size, fill-in, sparsity pattern, accuracy requirements and memory constraints, availability of good preconditioners, and so on. Within this work, we compare the effectiveness and the limitations of the proposed method for structure exploitation when a direct solver is adopted to tackle the linear system.

1.1 Motivation

Linear systems with the form (1) arise, e.g., from nonlinear complementarity problems [6], nonlinear optimization problems with inequality constraints [7] and discretized optimal control problems with state and control constraints [8,9]. Usually, these are reformulated through the Karush-Kuhn-Tucker (KKT) necessary optimality conditions, then equivalently transformed into a nonlinear system of equations with the so called NCP-functions [16] and finally solved with a nonsmooth version of Newton's method [14]. Some globalization strategies [9,11]

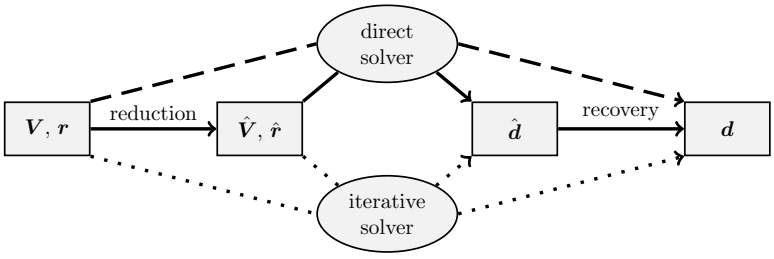

Fig. 1. Solution diagram: direct methods with (solid) and without (dashed) structure exploitation are compared; iterative methods (dotted) are not considered here.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

and results in functions spaces [18] have been reported. It has been shown that, for some NCP-functions, this approach is equivalent to a primal-dual active set strategy [10]. Indeed, different NCP-functions exhibit different properties and might affect the convergence behaviour [1,16]. With reference to the problem (1), matrices $\boldsymbol{Q}$, $\boldsymbol{A}$ and $\boldsymbol{C}$ can be considered as iterate-dependent linear-quadratic approximations of an underlying nonlinear problem, while diagonal matrices $\boldsymbol{S}$ and $\boldsymbol{T}$ originate from the NCP-function adopted and vectors $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are the residuals of the aforementioned nonlinear system of equations.

Example Let us consider a quadratic program (QP) with linear equality and inequality constraints. Hence, we seek an $\boldsymbol{x} \in \mathbb{R}^{n_x}$ minimizing $\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{q}^\top \boldsymbol{x}$, subject to constraints $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{a}$, $\boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{c}$, where $\boldsymbol{Q}$, $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{q}$, $\boldsymbol{a}$, $\boldsymbol{c}$ are given matrices and vectors, respectively. Here the inequalities are understood componentwise. Linear constraints ensure that regularity conditions are met, then the KKT conditions are necessary for optimality; these read:

$$\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{q} + \boldsymbol{A}^\top \boldsymbol{\lambda} + \boldsymbol{C}^\top \boldsymbol{\mu} = \boldsymbol{0} \tag{2a}$$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{a} \tag{2b}$$

$$\boldsymbol{0} \leq \boldsymbol{\mu} \quad \perp \quad \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{c} \tag{2c}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ denote the Lagrange multipliers. In (2c), inequality and complementarity constraints hold componentwise. Let us consider an NCP-function $\varphi : \mathbb{R}^2 \to \mathbb{R}$, e.g., the original or the penalized Fischer-Burmeister function [1,7], which by definition satisfies

$$\varphi(a, b) = 0 \quad \Leftrightarrow \quad 0 \leq a \perp b \geq 0 \tag{3}$$

for any pair (a, b). Thanks to this property, the KKT system (2) is equivalently rewritten as a nonlinear system $\boldsymbol{\psi}(\boldsymbol{z}) = \boldsymbol{0}$, collecting vector $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^{n_z}$, $n_z := n_x + n_a + n_c$, and with vector-valued function $\boldsymbol{\psi} : \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ defined by

$$\boldsymbol{\psi}(\boldsymbol{z}) = \begin{pmatrix} \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{q} + \boldsymbol{A}^\top \boldsymbol{\lambda} + \boldsymbol{C}^\top \boldsymbol{\mu} \\ \boldsymbol{A}\boldsymbol{x} - \boldsymbol{a} \\ \varphi(\boldsymbol{c} - \boldsymbol{C}\boldsymbol{x}, \boldsymbol{\mu}) \end{pmatrix} \tag{4}$$

Here the NCP-function φ applies componentwise. A (globalized) possibly nonsmooth Newton's-type method generates a sequence $\{\boldsymbol{z}^k\}$ through the recurrence $\boldsymbol{z}^{k+1} = \boldsymbol{z}^k + \alpha^k \boldsymbol{d}^k$, $k = 0, 1, 2, \ldots$, where the step length $\alpha^k > 0$ is determined, e.g., by a line-search procedure of Armijo's type and the search direction $\boldsymbol{d}^k$ is the solution of the linear equation $\boldsymbol{V}^k \boldsymbol{d} = -\boldsymbol{\psi}(\boldsymbol{z}^k)$ [7,8,9,11,14]. The matrix $\boldsymbol{V}^k$ is an element of the Clarke's generalized Jacobian (that is the convex hull of the Bouligand differential [2]) of $\boldsymbol{\psi}$ at $\boldsymbol{z}^k$, namely $\boldsymbol{V}^k \in \partial\boldsymbol{\psi}(\boldsymbol{z}^k)$ [8,11]. The NCP-function φ is the only element in (4) which can possibly make function $\boldsymbol{\psi}$ nonsmooth. Hence, from (4), one obtains

$$\boldsymbol{V}^k = \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{A}^\top & \boldsymbol{C}^\top \\ \boldsymbol{A} & & \\ -\boldsymbol{S}^k \boldsymbol{C} & & \boldsymbol{T}^k \end{bmatrix} \tag{5}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

with diagonal matrices $\boldsymbol{S}^k = \operatorname{diag}\left(s_1^k, s_2^k, \ldots, s_{n_c}^k\right)$ and $\boldsymbol{T}^k = \operatorname{diag}\left(t_1^k, t_2^k, \ldots, t_{n_c}^k\right)$, whose entries are pairwise coupled via the (possibly generalized) differential of the NCP-function φ. Defining $\boldsymbol{v}^k := \boldsymbol{c} - \boldsymbol{C}\boldsymbol{x}^k$ the inequality constraint violation at the k-th iterate, for $i = 1, 2, \ldots, n_c$, the coupling for the i-th inequality constraint reads [1,7,8]:

$$\left(s_i^k, t_i^k\right) \in \partial\varphi\left(v_i^k, \mu_i^k\right) \tag{6}$$

We remark that matrices $\boldsymbol{S}^k$ and $\boldsymbol{T}^k$ are diagonal because the NCP-function φ applies componentwise in (4). Then, the linear system $\boldsymbol{V}^k\boldsymbol{d} = -\boldsymbol{\psi}(\boldsymbol{z}^k)$ to compute the search direction $\boldsymbol{d}^k$ corresponds exactly to problem (1).

1.2 Outline

This work is organized as follows. Section 2 outlines the structure exploitation strategy and introduces an underlying assumption. In Sections 2.1 and 2.2 the two main steps are developed and discussed. Section 3 validates the proposed approach numerically, for both full and sparse matrices, with different reduction strategies, showing effectiveness and limitations of the proposed algorithm. Section 4 concludes the paper and presents ideas for future research.

2 Structure exploitation

Problem (1), also denoted $\boldsymbol{V}\boldsymbol{d} = \boldsymbol{r}$ for brevity, can be directly solved via any linear algebra package, e.g., MA48 [4], PARDISO [15], SUPERLU [13], `mldivide` in MATLAB [17]. However, we aim at exploiting our knowledge about the structure of matrix $\boldsymbol{V}$; computational effort and achieved accuracy might benefit from this, especially for large-scale and sparse linear systems. Firstly, we notice that matrix $\boldsymbol{V}$ and vector $\boldsymbol{r}$ are often computed blockwise and then assembled. Hence, matrices $\boldsymbol{Q}$, $\boldsymbol{A}$, $\boldsymbol{C}$ and vectors $\boldsymbol{s}$, $\boldsymbol{t}$, $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are here considered as the starting ingredients for solving (1). Overall, two directions are explored, mainly exploiting the diagonal structure of $\boldsymbol{S}$ and $\boldsymbol{T}$. In Section 2.1 a reduction step is discussed, eliminating some variables and introducing a smaller asymmetric linear system. Then, in Section 2.2, this is transformed into a symmetric one which is equivalent. Nonetheless, these operations for reorganizing the linear system and successive recovering of variables constitutes an overhead of computation. This means that these procedures might be worthy only for certain problems, likely large instances with lots of inequality constraints. In Section 3, numerical tests show that this break-even point corresponds to relatively small problem instances (for the tested implementation).

We point out that the proposed method relies on the following assumption on the diagonals of $\boldsymbol{S}$ and $\boldsymbol{T}$; it reads:

Assumption 1 *For $i = 1, 2, \ldots, n_c$, it holds $(s_i, t_i) \neq (0, 0)$.*

This is a mild requirement, in that it corresponds to problem (1) to be well-posed. One can show the following result:

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

Lemma 1. *If problem (1) admits a unique solution, then Assumption 1 holds.*

Proof (by contradiction). Let us assume there exist $\boldsymbol{d}$ unique solution to (1) and i such that $(s_i, t_i) = (0, 0)$. Hence, the row of $\boldsymbol{V}$ corresponding to the i-th inequality constraint consists of zeros only. Then, the matrix $\boldsymbol{V}$ is rank deficient and the problem is undetermined. Two cases are possible, depending on the value of h_i on the right-hand side. If $h_i = 0$, then (1) admits infinitely many solutions, hence solution $\boldsymbol{d}$ is not unique. If $h_i \neq 0$, then the linear system is unsolvable (impossible) and $\boldsymbol{d}$ cannot be a solution. □

Remark 1. Assumption 1 requires a mild condition to be satisfied by the NCP-function φ. For instance, a sufficient condition is that for any given pair $(a, b) \in \mathbb{R}^2$ there exists a pair $(s, t) \in \partial\varphi(a, b)$ such that $(s, t) \neq (0, 0)$; this allows to choose always suitable entries for $\boldsymbol{S}$ and $\boldsymbol{T}$. The Fischer-Burmeister function and the max function, among other NCP-functions, have this property.

Let us denote $\mathcal{I} := \{1, 2, \ldots, , n_c\}$ the index set for the inequality constraints, $\mathcal{I}^0_{\text{re}} := \{i \in \mathcal{I} \mid t_i \neq 0\}$ and $\mathcal{I}^0_{\text{sy}} := \{i \in \mathcal{I} \mid s_i \neq 0\}$ the largest index sets that allow respectively the reduction step and the symmetrization step, discussed below. Thanks to Assumption 1, these satisfy $\mathcal{I}^0_{\text{re}} \cup \mathcal{I}^0_{\text{sy}} = \mathcal{I}$. Let us consider an index subset $\mathcal{I}_{\text{re}} \subseteq \mathcal{I}^0_{\text{re}}$, sufficiently large to satisfy $\mathcal{I}_{\text{re}} \cup \mathcal{I}^0_{\text{sy}} = \mathcal{I}$. Then, for the associated complement $\mathcal{I}_{\overline{\text{re}}} := \mathcal{I} \backslash \mathcal{I}_{\text{re}}$, it holds $\mathcal{I}_{\overline{\text{re}}} \subseteq \mathcal{I}^0_{\text{sy}}$. With this construction, it is possible to apply the reduction step, ruling out a given set $\mathcal{I}_{\text{re}}$ of variables, and subsequently the symmetrization step on the linear system with the remaining variables, namely those in $\mathcal{I}_{\overline{\text{re}}}$.

Remark 2. We stress that in general it is $\mathcal{I}^0_{\text{re}} \cap \mathcal{I}^0_{\text{sy}} \neq \emptyset$ and hence the choice of $\mathcal{I}_{\text{re}}$ is not unique. This suggests there could be an optimal reduction strategy, possibly dependent on $\boldsymbol{V}$ and with some degree of computation awareness. However, this issue is beyond the scope of this paper.

In Section 3, we compare the following definitions of $\mathcal{I}_{\text{re}}$ through numerical investigations:

$$\mathcal{I}^{\text{t}}_{\text{re}} := \left\{ i \in \mathcal{I} \;\middle|\; |t_i| \geq \epsilon \right\} \tag{7a}$$

$$\mathcal{I}^{\text{s}}_{\text{re}} := \left\{ i \in \mathcal{I} \;\middle|\; |s_i| \leq \epsilon \right\} \tag{7b}$$

$$\mathcal{I}^{\text{ts}}_{\text{re}} := \left\{ i \in \mathcal{I} \;\middle|\; |t_i| \geq |s_i| \right\} \tag{7c}$$

where $\epsilon > 0$ is a given, sufficiently small value, introduced as a numerical tolerance in (7a)–(7b). For $\epsilon \to 0^+$, these sets approach the largest and the smallest possible reduction sets, respectively, namely reducing the most and the least of the variables. Instead, the set defined in (7c) represents an arbitrary trade-off, introduced for the sake of comparison; see Fig. 2.

Remark 3. One could think about performing either the reduction or symmetrization step. However, (i) under Assumption 1, once the system is reduced, the symmetrization step is straightforward, inexpensive and likely effective; (ii) the symmetrization step might be impossible without preliminary reduction, depending on the invertibility of $\boldsymbol{S}$.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

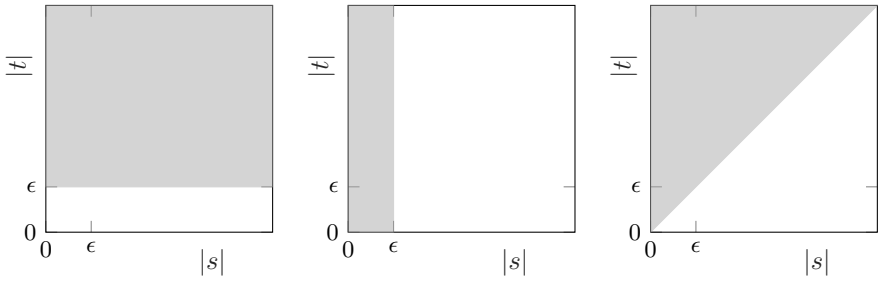

Fig. 2. Reduction sets (7) in the $|s|$-$|t|$ plane: $\mathcal{I}^{\mathrm{t}}_{\mathrm{re}}$ (left), $\mathcal{I}^{\mathrm{s}}_{\mathrm{re}}$ and $\mathcal{I}^{\mathrm{ts}}_{\mathrm{re}}$ (right).

2.1 Reduction

The idea behind the reduction step stems from the observation that problem (1) may be separable, i.e. that it may be possible to compute the value of some variables *a posteriori*, namely once the others are given. In fact, a solution to problem (1) must satisfy

$$\boldsymbol{T}\boldsymbol{z} = \boldsymbol{S}\boldsymbol{C}\boldsymbol{x} + \boldsymbol{h} \tag{8}$$

where matrix $\boldsymbol{T}$ is diagonal. Given an index set $\mathcal{I}_{\mathrm{re}} \subseteq \mathcal{I}^0_{\mathrm{re}}$, it is possible to compute z_i, for every $i \in \mathcal{I}_{\mathrm{re}}$, from (8) once the solution vector $\boldsymbol{x}$ is known. To be sure, let us build matrices $\boldsymbol{T}_{\mathrm{re}} := \mathrm{diag}\,(t_i \mid i \in \mathcal{I}_{\mathrm{re}})$ and $\boldsymbol{T}_{\overline{\mathrm{re}}} := \mathrm{diag}\,(t_i \mid i \notin \mathcal{I}_{\mathrm{re}})$ and define $\boldsymbol{z}_{\mathrm{re}}$ and $\boldsymbol{z}_{\overline{\mathrm{re}}}$ the corresponding vectors of unknown variables which can and cannot be reduced, respectively. Then, partitioning the linear system (1) accordingly with these definitions yields:

$$\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{A}^\top & \boldsymbol{C}_{\mathrm{re}}^\top & \boldsymbol{C}_{\overline{\mathrm{re}}}^\top \\ \boldsymbol{A} & & & \\ -\boldsymbol{S}_{\mathrm{re}}\boldsymbol{C}_{\mathrm{re}} & & \boldsymbol{T}_{\mathrm{re}} & \\ -\boldsymbol{S}_{\overline{\mathrm{re}}}\boldsymbol{C}_{\overline{\mathrm{re}}} & & & \boldsymbol{T}_{\overline{\mathrm{re}}} \end{bmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z}_{\mathrm{re}} \\ \boldsymbol{z}_{\overline{\mathrm{re}}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{g} \\ \boldsymbol{h}_{\mathrm{re}} \\ \boldsymbol{h}_{\overline{\mathrm{re}}} \end{pmatrix} \tag{9}$$

where matrices $\boldsymbol{C}_{\mathrm{re}}$, $\boldsymbol{C}_{\overline{\mathrm{re}}}$, $\boldsymbol{S}_{\mathrm{re}}$, $\boldsymbol{S}_{\overline{\mathrm{re}}}$ and vectors $\boldsymbol{h}_{\mathrm{re}}$ and $\boldsymbol{h}_{\overline{\mathrm{re}}}$ are constructed analogously, based on $\mathcal{I}_{\mathrm{re}}$. The matrix $\boldsymbol{T}_{\mathrm{re}}$ is nonsingular, by definition, and then, from (9), one can formally solve for $\boldsymbol{z}_{\mathrm{re}}$, obtaining

$$\boldsymbol{z}_{\mathrm{re}} = \boldsymbol{T}_{\mathrm{re}}^{-1}\left(\boldsymbol{S}_{\mathrm{re}}\boldsymbol{C}_{\mathrm{re}}\boldsymbol{x} + \boldsymbol{h}_{\mathrm{re}}\right) \ , \tag{10}$$

whose evaluation is straightforward because $\boldsymbol{T}_{\mathrm{re}}$ is diagonal. Plugging (10) back into (9) leads to a smaller linear system, after rearrangements, without reduced variables $\boldsymbol{z}_{\mathrm{re}}$, namely:

$$\begin{bmatrix} \hat{\boldsymbol{Q}} & \boldsymbol{A}^\top & \boldsymbol{C}_{\overline{\mathrm{re}}}^\top \\ \boldsymbol{A} & & \\ -\boldsymbol{S}_{\overline{\mathrm{re}}}\boldsymbol{C}_{\overline{\mathrm{re}}} & & \boldsymbol{T}_{\overline{\mathrm{re}}} \end{bmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z}_{\overline{\mathrm{re}}} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{f}} \\ \boldsymbol{g} \\ \boldsymbol{h}_{\overline{\mathrm{re}}} \end{pmatrix} \tag{11}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

where matrix $\hat{\boldsymbol{Q}}$ and vector $\hat{\boldsymbol{f}}$ are defined by:

$$\hat{\boldsymbol{Q}} := \boldsymbol{Q} + \boldsymbol{C}_{\text{re}}^{\top} \boldsymbol{T}_{\text{re}}^{-1} \boldsymbol{S}_{\text{re}} \boldsymbol{C}_{\text{re}} \tag{12a}$$

$$\hat{\boldsymbol{f}} := \boldsymbol{f} - \boldsymbol{C}_{\text{re}}^{\top} \boldsymbol{T}_{\text{re}}^{-1} \boldsymbol{h}_{\text{re}} \tag{12b}$$

The larger the set $\mathcal{I}_{\text{re}}$, the more reduced variables, the smaller the obtained linear system (11). In turn, the computation of $\hat{\boldsymbol{Q}}$ may be costly, involving a matrix-matrix multiplication, Eq. 12a. Also, for sparse problems, the fill-up of matrix $\hat{\boldsymbol{Q}}$ may become significant. These drawbacks suggest there might be a trade-off in the reduction step, and hence an optimal reduction strategy, as pointed out in Remark 2.

2.2 Symmetrization

Linear systems with a symmetric matrix can be solved more efficiently, in terms of time and memory. In order to get a symmetric matrix out of (11), it would suffice to left-multiply the rows associated with inequality constraints, namely with $\boldsymbol{z}_{\overline{\text{re}}}$, by the inverse of $-\boldsymbol{S}_{\overline{\text{re}}}$. As discussed above, it is $\mathcal{I}_{\overline{\text{re}}} \subseteq \mathcal{I}_{\text{sy}}^{0}$, hence the matrix $\boldsymbol{S}_{\overline{\text{re}}}$ is nonsingular, by construction; moreover, its inversion is straightforward, since it is diagonal. Then, the reduced symmetric linear system $\hat{\boldsymbol{V}}\hat{\boldsymbol{d}} = \hat{\boldsymbol{r}}$ reads:

$$\begin{bmatrix} \hat{\boldsymbol{Q}} & \boldsymbol{A}^{\top} & \boldsymbol{C}_{\overline{\text{re}}}^{\top} \\ \boldsymbol{A} & & \\ \boldsymbol{C}_{\overline{\text{re}}} & & -\boldsymbol{S}_{\overline{\text{re}}}^{-1} \boldsymbol{T}_{\overline{\text{re}}} \end{bmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z}_{\overline{\text{re}}} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{f}} \\ \boldsymbol{g} \\ -\boldsymbol{S}_{\overline{\text{re}}}^{-1} \boldsymbol{h}_{\overline{\text{re}}} \end{pmatrix} \tag{13}$$

The matrix $\hat{\boldsymbol{V}}$ is symmetric and smaller than $\boldsymbol{V}$; the vector $\hat{\boldsymbol{d}}$ collects the unknowns corresponding to optimization variables ($\boldsymbol{x}$), equality constraints' multipliers ($\boldsymbol{y}$) and not-reduced inequality constraints' multipliers ($\boldsymbol{z}_{\overline{\text{re}}}$).

Remark 4. In (12)–(13), the matrix-matrix products $\boldsymbol{T}_{\text{re}}^{-1}\boldsymbol{S}_{\text{re}}$, $\boldsymbol{S}_{\overline{\text{re}}}^{-1}\boldsymbol{T}_{\overline{\text{re}}}$ and the matrix-vector products $\boldsymbol{T}_{\text{re}}^{-1}\boldsymbol{h}_{\text{re}}$, $\boldsymbol{S}_{\overline{\text{re}}}^{-1}\boldsymbol{h}_{\overline{\text{re}}}$ can be evaluated as entry-wise vector-vector products. In fact, this is possible because matrices $\boldsymbol{S}_{\text{re}}$, $\boldsymbol{S}_{\overline{\text{re}}}$, $\boldsymbol{T}_{\text{re}}$ and $\boldsymbol{T}_{\overline{\text{re}}}$ are diagonal. Furthermore, one can exploit this feature by choosing a specific multiplication ordering, aiming at the lowest possible computational complexity.

3 Numerical results

This Section reports and discusses the results obtained from a MATLAB [17] implementation of Algorithm 1, considering Remark 4. The plain code (as well as a Julia 1.0 and a Python 3.6 implementation) are publicly available [3].

We are interested in comparing the computation time for solving problem (1), through direct methods, with and without the proposed structure exploitation method, see Fig. 1 above. Also, we investigate how it is affected by the problem size $N := n_x + n_a + n_c$ and the relative number of equality and inequality constraints, $\alpha := n_a/n_x$ and $\gamma := n_c/n_x$, respectively.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

Algorithm 1 Abstract structure-exploiting linear solver.

Input: $\boldsymbol{Q}$, $\boldsymbol{A}$, $\boldsymbol{C}$, $\boldsymbol{s}$, $\boldsymbol{t}$, $\boldsymbol{f}$, $\boldsymbol{g}$, $\boldsymbol{h}$; ϵ
Output: $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$

$\mathcal{I}_{\text{re}} \leftarrow \boldsymbol{s}, \boldsymbol{t}, \epsilon$; // reduction strategy, Eq. 7
$\boldsymbol{C}_{\text{re}}, \boldsymbol{s}_{\text{re}}, \boldsymbol{t}_{\text{re}}, \boldsymbol{h}_{\text{re}}, \boldsymbol{C}_{\overline{\text{re}}}, \boldsymbol{s}_{\overline{\text{re}}}, \boldsymbol{t}_{\overline{\text{re}}}, \boldsymbol{h}_{\overline{\text{re}}} \leftarrow \boldsymbol{C}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{h}, \mathcal{I}_{\text{re}}$; // partitioning
$\hat{\boldsymbol{Q}} \leftarrow \boldsymbol{Q}, \boldsymbol{C}_{\text{re}}, \boldsymbol{t}_{\text{re}}, \boldsymbol{s}_{\text{re}}$; // Eq. 12a
$\hat{\boldsymbol{f}} \leftarrow \boldsymbol{f}, \boldsymbol{C}_{\text{re}}, \boldsymbol{t}_{\text{re}}, \boldsymbol{h}_{\text{re}}$; // Eq. 12b
$\hat{\boldsymbol{V}} \leftarrow \hat{\boldsymbol{Q}}, \boldsymbol{A}, \boldsymbol{C}_{\overline{\text{re}}}, \boldsymbol{s}_{\overline{\text{re}}}, \boldsymbol{t}_{\overline{\text{re}}}$; // Eq. 13
$\hat{\boldsymbol{r}} \leftarrow \hat{\boldsymbol{f}}, \boldsymbol{g}, \boldsymbol{s}_{\overline{\text{re}}}, \boldsymbol{h}_{\overline{\text{re}}}$; // Eq. 13
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}_{\overline{\text{re}}} \leftarrow \hat{\boldsymbol{V}}, \hat{\boldsymbol{r}}$; // linear system
$\boldsymbol{z}_{\text{re}} \leftarrow \boldsymbol{C}_{\text{re}}, \boldsymbol{s}_{\text{re}}, \boldsymbol{t}_{\text{re}}, \boldsymbol{h}_{\text{re}}, \boldsymbol{x}$; // recovering, Eq. 10
$\boldsymbol{z} \leftarrow \boldsymbol{z}_{\text{re}}, \boldsymbol{z}_{\overline{\text{re}}}, \mathcal{I}_{\text{re}}$; // assembling

A problem instance consists of matrices $\boldsymbol{Q}$, $\boldsymbol{A}$, $\boldsymbol{C}$ and vectors $\boldsymbol{s}$, $\boldsymbol{t}$ (the diagonal of $\boldsymbol{S}$ and $\boldsymbol{T}$, respectively), $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$. In the case of full matrices, starting from given values of N, α and γ, an instance is generated as follows:

$$
\begin{aligned}
n_x &= \left\lfloor \frac{N}{1+\alpha+\gamma} \right\rfloor \\
n_a &= \lfloor \alpha n_x \rfloor \\
n_c &= N - n_x - n_a \\
\bar{Q}_{ij} &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_{\text{x}}\ ,\ j = 1,\ldots,n_{\text{x}} \\
\boldsymbol{Q} &= \frac{1}{2}\left(\bar{\boldsymbol{Q}} + \bar{\boldsymbol{Q}}^{\top}\right) \\
A_{ij} &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_{\text{a}}\ ,\ j = 1,\ldots,n_{\text{x}} \\
C_{ij} &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_c\ ,\ j = 1,\ldots,n_{\text{x}} \\
\rho_i &\sim \sqrt{\mathcal{U}(0,1)} && i = 1,\ldots,n_c \\
\theta_i &\sim \mathcal{U}(0,2\pi) && i = 1,\ldots,n_c \\
s_i &= 1 + \rho_i \cos\theta_i && i = 1,\ldots,n_c \\
t_i &= 1 + \rho_i \sin\theta_i && i = 1,\ldots,n_c \\
f_i &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_{\text{x}} \\
g_i &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_{\text{a}} \\
h_i &\sim \mathcal{N}(0,1) && i = 1,\ldots,n_c
\end{aligned}
$$

where $\mathcal{N}(\mu,\sigma)$ denotes the normal continuous probability distribution with mean value μ and standard deviation σ, and $\mathcal{U}(a,b)$ the uniform distribution with support in $[a,b]$. Entries of $\boldsymbol{S}$ and $\boldsymbol{T}$ are pairwise coupled in that they are sampled from a disk in the s-t plane, centered in $(1,1)$ with unitary radius, with uniform probability distribution. This setting is motivated by and mimics the generalized differential of the Fischer-Burmeister function [8]. Both, the direct and the structure-exploiting methods setup the linear system starting from these inputs. Notice that the reduced approach does not build $\boldsymbol{V}$ nor $\boldsymbol{r}$, but their reduced

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

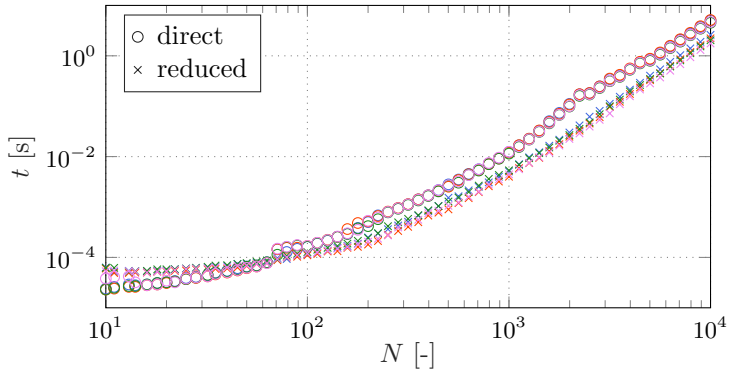

Fig. 3. Execution time: direct approach (t_{dir}, dot) and reduced approach (t_{red}, cross) with reduction set $\mathcal{I}^{\text{t}}_{\text{re}}$; full matrices. Median value.

and symmetric counterpart $\hat{\boldsymbol{V}}$ and $\hat{\boldsymbol{r}}$. In our implementation, the direct method builds $\boldsymbol{V}$ and $\boldsymbol{r}$ and then adopts the `mldivide` routine to solve $\boldsymbol{V}\boldsymbol{d} = \boldsymbol{r}$. Instead, for the reduced approach (with full matrices), the `linsolve` routine is adopted and explicitly informed that matrix $\hat{\boldsymbol{V}}$ is symmetric. The problem size N varies between 10 and 10^4 for full and between 10^3 and $2 \cdot 10^4$ for sparse matrices. For each problem size, a set of 100 problem instances are generated (only 10 if $N > 10^4$), checked for ill-conditioning and eventually solved. The composition of constraints is chosen to be $(\alpha, \gamma) \in \{(0, 1), (0, 1.5), (0.5, 1), (0.5, 1.5)\}$ (colored in blue, red, green and violet, respectively). The index sets defined in (7) are adopted and compared, with the tolerance $\epsilon = 10^{-3}$. Sparse matrices are generated in such a way that they approximately have 10 entries for each row; this makes the number of nonzero entries to increase linearly and not quadratically with the problem size N.

The computation time for the direct and reduced case are depicted in Fig. 3, considering full matrices and the (large) index set $\mathcal{I}^{\text{t}}_{\text{re}}$. This gives an idea about the adopted implementation and computing hardware; also, one can guess the computational complexity of the underlying algorithm for solving a linear system. As expected in Section 2, the overhead due to partitioning, reducing and recovering, introduces a break-even point, at around $N = 60$ (for $\mathcal{I}^{\text{t}}_{\text{re}}$ and $\mathcal{I}^{\text{ts}}_{\text{re}}$, but not for $\mathcal{I}^{\text{s}}_{\text{re}}$); hence, the reduced approach is not beneficial only for small-sized problems. This and other considerations can be drawn based on Fig. 4, where it is depicted the (median value of the) ratio of the execution time with the reduced approach, t_{red}, with the different reduction strategies, over the direct one, t_{dir}. Therein, the break-even point corresponds to the unitary ratio; also, the additional computational burden is significant for low values of N. For large N, instead, the ratio decreases to approximately one-half for $\mathcal{I}^{\text{t}}_{\text{re}}$ and $\mathcal{I}^{\text{ts}}_{\text{re}}$, while for $\mathcal{I}^{\text{s}}_{\text{re}}$ it stays around the unit. As one could expect, the reduction set $\mathcal{I}^{\text{s}}_{\text{re}}$ is not as effective as $\mathcal{I}^{\text{t}}_{\text{re}}$ and $\mathcal{I}^{\text{ts}}_{\text{re}}$ because it does not benefit very much from the reduction step, in that it eliminates only few variables. The relative number

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

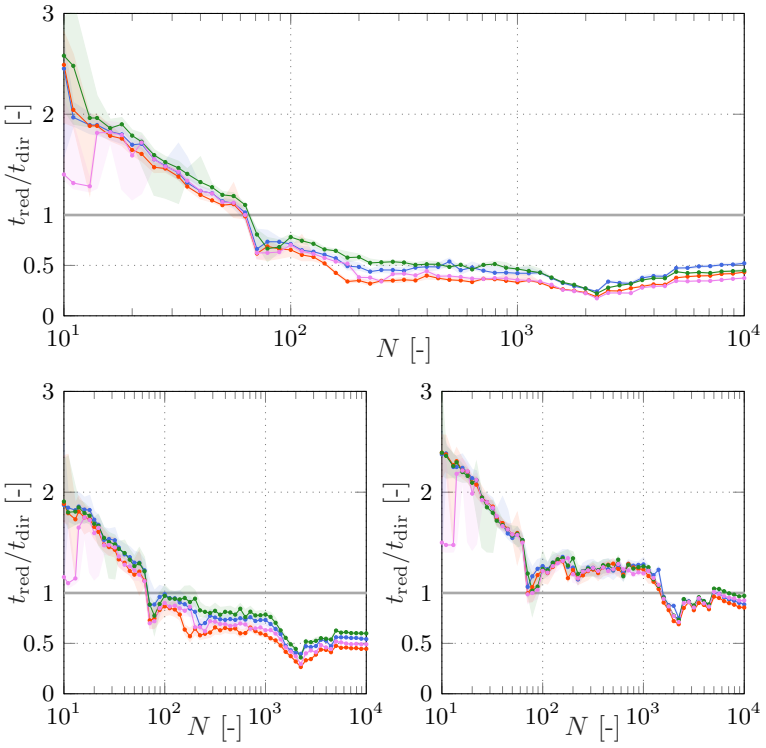

Fig. 4. Ratio of execution time: $\mathcal{I}^{\text{t}}_{\text{re}}$ (top), $\mathcal{I}^{\text{ts}}_{\text{re}}$ (bottom left), $\mathcal{I}^{\text{s}}_{\text{re}}$ (bottom right); full matrices. Median value (line) and 9%-91% quantiles (filled area).

of constraints has an impact on the execution time but it does not drastically affect the overall behaviour (for both, full and sparse case). In fact, all else being equal, either decreasing the number of equalities and increasing the number of inequalities reduces the execution time ratio, meaning that the reduced approach is more effective and worthy for (large) problems with many inequality constraints. For what concerns the case of sparse matrices, similar observations are valid, see Fig. 5. In order to show the distribution of the results obtained from the executed tests, along with the median value, the 9% and 91% quantiles are also reported. For full matrices, Fig. 4, the distribution is relatively narrow, while for sparse matrices, Fig. 5, the results are relatively scattered. Thus, we argue the sparsity pattern greatly affects the computation time. Nevertheless, for relatively large sparse matrices, the ratio $t_{\text{red}}/t_{\text{dir}}$ approaches one-half and promisingly decreases.

These numerical results suggest the set $\mathcal{I}^{\text{t}}_{\text{re}}$ defined in (7a) to be the most effective reduction strategy among those tested. In fact, it generates the smallest linear system and then post-solves the most variables. However, as argued

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

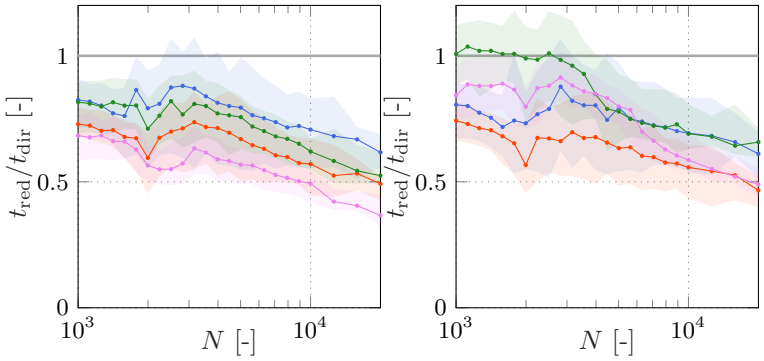

Fig. 5. Ratio of execution time: $\mathcal{I}_{\text{re}}^{\text{t}}$ (top) and $\mathcal{I}_{\text{re}}^{\text{ts}}$ (bottom); sparse matrices. Median value (line) and 9%-91% quantiles (filled area).

in Section 2.1, we claim it might be not the case for (much) larger problem instances, because of the required overhead for the reduction step.

4 Conclusions

This paper proposed and studied a structure-exploiting approach for solving linear systems arising in the context of nonsmooth Newton's method. The applicability of this method was established under well-posedness of the original problem. Numerical examples showed that the developed approach reduces the computational time, with both, full and sparse linear systems. Some of the tested reduction strategies resulted in halving the execution time.

Analogous ideas apply when an iterative linear solver is of choice, e.g., for very large systems; tailored preconditioners are subject of future research. It remains to assess the effectiveness and the drawbacks of the method when embedded into larger routines for numerical optimization. Moreover, it would be interesting to investigate an optimal reduction strategy, possibly computationally aware.

Acknowledgements

A.D.M. heartily thanks his *marafiki* for the unforgettable memories and wishes them a happy marriage.

References

1. Chen, B., Chen, X., Kanzow, C.: A penalized Fischer-Burmeister NCP-function. Mathematical Programming **88**(1), 211–216 (Jun 2000). https://doi.org/10.1007/PL00011375
2. Clarke, F.H.: Optimization and Nonsmooth Analysis. John Wiley & Sons, New York (1983)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32

3. De Marchi, A.: Code supporting "Nonsmooth Newton's method: some structure exploitation" (Nov 2018). https://doi.org/10.5281/zenodo.1486064
4. Duff, I.S., Reid, J.K.: MA48 — a Fortran code for direct solution of sparse unsymmetric linear systems of equations. Tech. Rep. RAL-93-072, Rutherford Appleton Laboratory (Oct 1993)
5. Facchinei, F., Fischer, A., Kanzow, C.: On the accurate identification of active constraints. SIAM Journal on Optimization **9**(1), 14–32 (1998). https://doi.org/10.1137/S1052623496305882
6. Facchinei, F., Kanzow, C.: A nonsmooth inexact Newton method for the solution of large-scale nonlinear complementarity problems. Mathematical Programming **76**(3), 493–512 (Mar 1997). https://doi.org/10.1007/BF02614395
7. Fischer, A.: A special Newton-type optimization method. Optimization **24**, 269–284 (1992). https://doi.org/10.1080/02331939208843795
8. Gerdts, M., Kunkel, M.: A nonsmooth Newton's method for discretized optimal control problems with state and control constraints. Journal of Industrial & Management Optimization **4**(2), 247–270 (2008). https://doi.org/10.3934/jimo.2008.4.247
9. Gerdts, M., Kunkel, M.: A globally convergent semi-smooth Newton method for control-state constrained DAE optimal control problems. Computational Optimization and Applications **48**(3), 601–633 (Apr 2011). https://doi.org/10.1007/s10589-009-9275-0
10. Hintermüller, M., Ito, K., Kunisch, K.: The primal-dual active set strategy as a semismooth Newton method. SIAM Journal on Optimization **13**(3), 865–888 (2002). https://doi.org/10.1137/S1052623401383558
11. Jiang, H.: Global convergence analysis of the generalized Newton and Gauß-Newton methods of the Fischer-Burmeister equation for the complementarity problem. Mathematics of Operations Research **24**(3), 529–543 (1999), http://www.jstor.org/stable/3690647
12. Laiu, M.P., Tits, A.L.: A constraint-reduced MPC algorithm for convex quadratic programming, with a modified active set identification scheme. Computational Optimization and Applications (Mar 2019). https://doi.org/10.1007/s10589-019-00058-0
13. Li, X.S.: An overview of SuperLU: Algorithms, implementation, and user interface. ACM Transactions on Mathematical Software **31**(3), 302–325 (Sep 2005)
14. Qi, L., Sun, J.: A nonsmooth version of Newton's method. Mathematical Programming **58**(1), 353–367 (Jan 1993). https://doi.org/10.1007/BF01581275
15. Schenk, O., Gärtner, K.: Solving unsymmetric sparse systems of linear equations with PARDISO. Future Generation Computer Systems **20**(3), 475–487 (2004). https://doi.org/https://doi.org/10.1016/j.future.2003.07.011
16. Sun, D., Qi, L.: On NCP-functions. Computational Optimization and Applications **13**(1), 201–220 (Apr 1999). https://doi.org/10.1023/A:1008669226453
17. The MathWorks, Inc.: MATLAB Release 2017b, Natick, Massachusetts, United States
18. Ulbrich, M.: Nonsmooth Newton-like Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces. Ph.D. thesis, Technische Universität München (Feb 2002)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22744-9_32