Chapter 4
Krylov-Subspace Methods for Quadratic Hypersurfaces: a Grossone–based Perspective

Giovanni Fasano

Abstract We study the role of the recently introduced *infinite* number grossone, to deal with two renowned Krylov-subspace methods for symmetric (possibly indefinite) linear systems. We preliminarily explore the relationship between the Conjugate Gradient (CG) method and the Lanczos process, along with their specific role of yielding tridiagonal matrices which retain large information on the original linear system matrix. Then, we show that on one hand there is not immediate evidence of an advantage from embedding grossone within the Lanczos process. On the other hand, coupling the CG with grossone shows clear theoretical improvements. Furthermore, reformulating the CG iteration through a grossone-based framework allows to encompass also a certain number of Krylov-subspace methods relying on conjugacy among vectors. The last generalization remarkably justifies the use of a grossone-based reformulation of the CG to solve also indefinite linear systems. Finally, pairing the CG with the algebra of grossone easily provides relevant geometric properties of quadratic hypersurfaces.

1.1 Introduction

We consider the iterative solution of indefinite linear systems by Krylov-subspace methods. After a preliminary analysis, where a couple of renowned methods are briefly detailed and compared, we directly focus on those algorithms based on the generation of conjugate vectors, and we disregard those methods which rely on generating Lanczos vectors.

More specifically, we analyze the behaviour of the Conjugate Gradient (CG) method in case of degeneracy, since it yields relevant implications when solving symmetric linear systems within Nonconvex Optimization problems. In this regard, the current literature on Krylov-subspace methods (see e.g. [27]) reports plenty of

Giovanni Fasano
University Ca' Foscari of Venice, Department of Management, e-mail: fasano@unive.it

applications in nonlinear programming, where the CG is used and it can possibly prematurely halt on the solution of indefinite linear systems (e.g. Newton's equation for nonconvex problems).

We recall that the CG iteratively computes the sequence $\{x_k\}$, where x_k approximates at step k the solution of the symmetric linear system $Ax = b$, being $A \in \mathbf{R}^{n\times n}$. The stopping rule of the CG is based on a Ritz-Galerkin condition, i.e. the norm of the current residual $r_k = b - Ax_k$ is checked, in order to evaluate the quality of the current approximate solution x_k. Unexpectedly, the unfortunate choice of the initial iterate x_1 may cause a premature undesired stop of the CG on specific indefinite linear systems. As well known, the last drawback may have a direct dramatic impact on optimization frameworks: a so called gradient-related direction cannot be computed and possibly inefficient arrangements need to be considered. When a premature stop of the CG occurs it corresponds to an unexpected numerical failure: namely a division by a small quantity is involved. This situation is usually addressed in the literature as a *pivot breakdown*, and corresponds to the fact that the steplength along the current search direction selected by the CG tends to be unbounded. As a consequence, the CG stops beforehand and the current iterate x_k may be far from a solution of the linear system (i.e. the quantity $\|r_k\|$ might be significantly nonzero).

This paper specifically addresses the pivot breakdown of the CG, from a perspective suggested by the recent introduction of the numeral *grossone* [37]. We urge to remark that a comprehensive description of the grossone-based methodology can be found in [42], and it should be stressed that it is not formally related to *non-standard analysis* (see [43]).
Our perspective is definitely unusual for the CG, since the literature of the last decades has mainly focused on its performance and stability, rather than on the way to recover its iteration in the indefinite case. Nevertheless, we are convinced that a proper investigation of the ultimate reasons of CG degeneracy might pursue a couple of essential tasks:

- to recover the degeneracy and *provide gradient-related directions* within optimization frameworks;
- to generate *negative curvature directions*, that allow convergence of optimization methods to solutions satisfying second order necessary optimality conditions.

As regards the organization of the paper, in Sect. 1.2 we detail similarities and dissimilarities of two well known Krylov-subspace methods: namely the CG and the Lanczos process. In Sect. 1.3 we describe one of the main conclusions in the current paper, i.e. the use of grossone with the CG can help overcoming problems of degeneracy in the indefinite case. Sect. 1.4 contains details on the second relevant contribution of this paper, namely the use of grossone for the iterative computation of negative curvature directions in large scale (unconstrained) optimization frameworks, where the objective function is twice continuously differentiable. Finally, a section of conclusions will complete the paper.

As regards the symbols adopted in the paper, we use $\mathbf{R}^p$ to represent the set of the real p-vectors, while for the sake of simplicity $\|x\|$ is used to indicate the Euclidean norm of the vector x, in place of $\|x\|_2$. Given the n-real vectors x and y,

Table 1.1 The CG method for solving (1.1) when $A \succ 0$.

The Conjugate Gradient (CG) method

Step 1**:** $k = 1\ ,\ x_1 \in \mathbf{R}^n\ ,\ r_1 = b - Ax_1\ ,\ p_1 = r_1$.
Step k**:** If $r_k = 0$ then STOP, else

$$x_{k+1} = x_k + \theta_k p_k, \qquad \theta_k = \frac{r_k^T p_k}{p_k^T A p_k},$$
$$r_{k+1} = r_k - \theta_k A p_k,$$
$$p_{k+1} = r_{k+1} + \beta_k p_k, \qquad \beta_k = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k} = \frac{\|r_{k+1}\|^2}{\|r_k\|^2},$$
$$k \leftarrow k+1 \text{ repeat } \textbf{Step } k.$$

End if

with $x^T y$ we indicate their standard inner product in $\mathbf{R}^n$. The symbol $f \in C^\ell(\mathcal{A})$ indicates that the function f is ℓ times continuously differentiable on the set $\mathcal{A}$. The symbol $B \succ 0$ (respectively $B \succeq 0$) indicates that the square matrix B is positive definite (respectively semidefinite). Finally, $\lambda_M(A)$ (respectively $\lambda_m(A)$) represents the largest (respectively smallest) eigenvalue of the square matrix A.

1.2 The CG method and the Lanczos process for matrix tridiagonalization

Let us consider the solution of the *symmetric* linear system

$$Ax = b, \qquad A \in \mathbf{R}^{n\times n}, \tag{1.1}$$

where the matrix A is possibly *indefinite* and *nonsingular*. As long as A in (1.1) is *positive definite*, the CG method [26] iteratively provides a tridiagonalization of it (see also [22]). A general description of the CG method for solving (1.1) is reported in Table 1.1 [25], where $r_{k+1} = b - Ax_{k+1}$ and the sequences $\{r_i\}$ and $\{p_i\}$ are such that after $k+1$ iterations:

$$\begin{aligned} r_i^T r_j &= 0, \quad i \neq j \leq k+1, \quad \text{(orthogonality among } \{r_i\}\text{)},\\ r_i^T p_j &= 0, \quad j < i \leq k+1,\\ p_i^T A p_j &= 0, \quad i \neq j \leq k+1, \quad \text{(conjugacy among } \{p_i\}\text{)}. \end{aligned}$$

Assume that after m steps the CG stops and $r_{m+1} = 0$ (i.e. a solution to the linear system (1.1) is found), then setting

$$R_m = \left(\frac{r_1}{\|r_1\|} \cdots \frac{r_m}{\|r_m\|}\right) \in \mathbf{R}^{n\times m}, \tag{1.2}$$

$$P_m = \left(\frac{p_1}{\|r_1\|} \cdots \frac{p_m}{\|r_m\|}\right) \in \mathbf{R}^{n\times m}, \tag{1.3}$$

along with

$$L_m = \begin{pmatrix} 1 & & 0 \\ -\sqrt{\beta_1} & \ddots & \\ & \ddots & \ddots \\ 0 & -\sqrt{\beta_{m-1}} & 1 \end{pmatrix} \in \mathbf{R}^{m\times m}, \tag{1.4}$$

and

$$D_m = \begin{pmatrix} \frac{1}{\theta_1} & & 0 \\ & \ddots & \\ & & \ddots \\ 0 & & \frac{1}{\theta_m} \end{pmatrix} \in \mathbf{R}^{m\times m}, \tag{1.5}$$

after an easy computation we have from (1.2)-(1.5)

$$P_m L_m^T = R_m, \tag{1.6}$$

$$AP_m = R_m L_m D_m, \tag{1.7}$$

$$AP_m L_m^T = R_m L_m D_m L_m^T \implies AR_m = R_m T_m^{CG}, \tag{1.8}$$

being $T_m^{CG} = L_m D_m L_m^T \in \mathbf{R}^{m\times m}$ the symmetric tridiagonal matrix

$$T_m^{CG} = \begin{pmatrix} \frac{1}{\theta_1} & -\frac{\sqrt{\beta_1}}{\theta_1} & & & 0 \\ -\frac{\sqrt{\beta_1}}{\theta_1} & \left(\frac{1}{\theta_2}+\frac{\beta_1}{\theta_1}\right) & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \left(\frac{1}{\theta_{m-1}}+\frac{\beta_{m-2}}{\theta_{m-2}}\right) & -\frac{\sqrt{\beta_{m-1}}}{\theta_{m-1}} \\ 0 & & & -\frac{\sqrt{\beta_{m-1}}}{\theta_{m-1}} & \left(\frac{1}{\theta_m}+\frac{\beta_{m-1}}{\theta_{m-1}}\right) \end{pmatrix} \in \mathbf{R}^{m\times m}. \tag{1.9}$$

Remark 1 Relation (1.8) underlies a three-term recurrence among the residuals $\{r_i\}$, being

$$A\frac{r_i}{\|r_i\|} \in span\{r_{i-1}, r_i, r_{i+1}\}, \qquad i \in \{1,\dots,m\}. \tag{1.10}$$

1.2.1 Basics on the Lanczos process

Similarly to the previous section, let us now consider the Lanczos process which is reported in Table 1.2. Unlike the CG method, it was initially conceived to iteratively solve a symmetric eigenvalue problem in the indefinite case [29], so that after m steps it allows to reduce (1.8) into relation

$$AQ_m = Q_m T_m^L, \tag{1.11}$$

Table 1.2 The Lanczos process for the tridiagonalization of (1.1), when A is possibly indefinite.

The Lanczos process

Step 1**:** $k = 0\,,\ v_0 = b \in \mathbf{R}^n\,,\ q_0 = 0\,,\ \delta_0 = \|b\|$.
Step k**:** If $\delta_k = 0$ then STOP, else
$q_{k+1} = \dfrac{v_k}{\delta_k}$,
$k \leftarrow k + 1$,
$\alpha_k = q_k^T A q_k$,
$v_k = (A - \alpha_k I) q_k - \delta_{k-1} q_{k-1}$,
$\delta_k = \|v_k\|$,
repeat **Step** k.
End if

where A is the matrix in (1.1), $Q_m = (q_1 \cdots q_m)$ and T_m^L is again a tridiagonal matrix, such that (Sturm sequence of tridiagonal matrices)

$$\lambda_m(A) \le \lambda_m\left(T_m^L\right) \le \lambda_m\left(T_{m-1}^L\right) \le \cdots \le \lambda_M\left(T_{m-1}^L\right) \le \lambda_M\left(T_m^L\right) \le \lambda_M(A)\,.$$

Moreover, coupling the Lanczos process with a suitable factorization of the matrix T_m^L, the iterative solution of (1.1) can be pursued.

Indeed, given a *symmetric indefinite* matrix A, after $m \ge 1$ steps the Lanczos process similarly to (1.2) generates the directions (the *Lanczos vectors*) $q_1, \ldots, q_m$ satisfying the orthogonality properties

$$q_i^T q_j = 0, \qquad i \ne j \le m.$$

In particular at step $k \le m$ of the iterative procedure, the basis $\{q_1, \ldots, q_k\}$ for the Krylov subspace $\mathcal{K}_k(q_1, A) \doteq span\{q_1, Aq_1, \ldots, A^{k-1}q_1\}$ is generated. Then, as for the CG, the Lanczos process provides a basis of $\mathcal{K}_k(q_1, A)$ which is used to solve the problem

$$\min_{x \in \mathcal{K}_k(q_1, A)} \|Ax - b\|.$$

Hence, since

$$\dim\left[\mathcal{K}_1(q_1, A)\right] < \dim\left[\mathcal{K}_2(q_1, A)\right] < \cdots,$$

in at most n iterations of the CG or the Lanzos process a sufficient information is available to compute the solution of (1.1).

Similarly to the CG (see (1.8)), in case at step m of the Lanczos process we have $q_{m+1} = 0$ (i.e. $\mathcal{K}_m(q_1, A) \equiv \mathcal{K}_{m+1}(q_1, A)$), then relation (1.11) holds, where $T_m^L \in \mathbf{R}^{m \times m}$ is the *tridiagonal matrix*

$$T_m^L \;=\; \begin{pmatrix} \alpha_1 & \delta_1 & & 0 \\ \delta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \delta_{m-1} \\ 0 & & \delta_{m-1} & \alpha_m \end{pmatrix},$$

and a conclusion similar to (1.10) holds, replacing R_m by Q_m and T_m^{CG} by T_m^L. Moreover, at step $k \geq 1$ of the Lanczos process we also have

$$T_k^L \;=\; Q_k^T A Q_k,$$

so that in case the Lanczos process performs n steps, the square matrix Q_n turns to be orthogonal and its columns span $\mathbf{R}^n$.

On the other hand, since A is nonsingular the problem (1.1) is equivalent to compute the stationary point of the quadratic functional

$$q(x) \;=\; \frac{1}{2} x^T A x - b^T x, \tag{1.12}$$

and the Lanczos method can be a natural candidate for its solution, too. Indeed, if the Lanczos process stops at step m (i.e. $\delta_m = 0$), then replacing $x = Q_m z$, with $z \in \mathbf{R}^m$, into (1.12) and recalling that $q_1 = b/\|b\|$, we obtain:

$$\nabla q(z) \;=\; Q_m^T A Q_m z - Q_m^T b \;=\; T_m^L z - \|b\| e_1.$$

Hence, if the solution z^* of the *tridiagonal* system

$$T_m^L z - \|b\| e_1 \;=\; 0\,, \qquad z \in \mathbf{R}^m, \tag{1.13}$$

is available, the point $x^* = Q_m z^*$ is both a solution of the original system (1.1) and a stationary point of (1.12) over the Krylov subspace $\mathcal{K}_m(b, A) = span\{q_1, \ldots, q_m\}$.

1.2.2 How the CG and the Lanczos process compare: a path to degeneracy

We urge to give some considerations about the comparison between the CG and the Lanczos process, in the light of possibly introducing the issue of degeneracy for both these algorithms:

- the Lanczos process properly *does not solve* the linear system (1.1); it rather *reformulates* (1.1) into the tridiagonal one (1.13). This means that some further calculations are necessary (i.e. a factorization for the matrix T_m^L) in order to give the explicit solution of (1.13) and then backtracking to a solution of (1.1). The CG (similarly for the CG-based methods in [14, 15, 16, 18] – see the next sections) does not require the last two-step solution scheme, inasmuch as at step k it at once

decomposes the matrix T_k^{CG} and computes x_{k+1} as

$$x_{k+1} \in \underset{x \in \mathcal{K}_k(b,A)}{\operatorname{argmin}} \{\|Ax - b\|\};$$

- since the solution z^* of (1.13) yields the solution $x^* = Q_m z^*$ of (1.1), for the Lanczos process we apparently need to *store the matrix* Q_m, in order to calculate x^*. However, in case we are just interested about computing the solution x^*, the storage of Q_m can be avoided (see e.g. [27], the algorithm SYMMLQ [35] and the algorithm SYMMBK [3]), by means of a suitable recursion. On the other hand, in case the Lanczos process were also asked to provide information on negative curvature directions associated with $q(x)$ in (1.12), at x^*, then the storage of the full rank matrix Q_m seems mandatory (see also [30]) or an additional computational effort is required (see the more recent paper [6]). Both the CG and the CG-based schemes reported in this paper avoid the last additional effort. Hence, our great interest for specifically pairing grossone with conjugacy.

We also recall that the tridiagonal matrices T_m^{CG} and T_m^L are obtained in a similar fashion, by the CG and the Lanczos process, respectively. However, in general neither in case $A \succ 0$ nor in the indefinite case they coincide, as extensively motivated in the paper [17]. Furthermore, the CG explicitly performs the Cholesky-like factorization $T_m^{CG} = L_m D_m L_m^T$ of T_m^{CG} in (1.8), in order to solve the linear system (1.1). The last matrix decomposition always exists when $A \succ 0$; conversely, if A is indefinite this decomposition exists if and only if no *pivot breakdown* occurs, i.e. none of the diagonal entries of D_m is near zero (which causes a premature stop of the CG).

On the contrary, if the Lanczos process is applied it cannot stop beforehand also when A is indefinite, because it does rely on any matrix factorization of T_m^L, meaning that no pivot breakdown can occur (see also [35]). Therefore, the application of the Lanczos process is well-posed in the indefinite case, too. In the next sections we show that the last conclusion motivates the use of grossone to handle pivot breakdown for the CG. Conversely, no immediate application of grossone algebra for the Lanczos process seems advisable, inasmuch as no breakdown opportunity can take place.

1.3 Coupling the CG with Grossone: a marriage of interest

Here we motivate the importance of pairing the CG with grossone, in case the system matrix A in (1.1) is indefinite. We first give a geometric viewpoint of the CG degeneracy (see the next section), then we detail how to recover the last degeneracy using grossone: this yields a general framework, that is used to describe the issue of degeneracy also for several CG-based methods, as detailed in [12].

1.3.1 The geometry behind CG degeneracy

When the CG is applied to solve (1.1), with A indefinite, by Sect. 1.2.2 a possible degenerate or nearly degenerate situation may occur, namely $p_k^T A p_k \approx 0$, with $p_k \neq 0$. This implies a couple of results we report here, that will be suitably reinterpreted in the next sections from an alternative standpoint, using *grossone*.

Observe that when A is positive definite, at any Step k of the CG we have $0 < \lambda_m(A)\|p_k\|^2 \leq p_k^T A p_k$, so that $p_k^T A p_k$ is suitably bounded from below. Conversely, in case A is indefinite (nonsingular), a similar bound does not hold and possibly we might have $p_k^T A p_k = 0$, for a nonzero vector p_k. Furthermore, in order to better analyze the (near) degenerate case, when A is indefinite nonsingular and at Step k we have $|p_k^T A p_k| \geq \varepsilon_k \|p_k\|^2$, $\varepsilon_k > 0$, with $\|p_k\|, \|p_{k+1}\| < +\infty$, then (see also [15]) the angle $\alpha_{k,k+1}$ between the vectors p_k and p_{k+1} satisfies

$$\frac{\pi}{2} - \arccos\left(\frac{\varepsilon_k}{|\lambda_M(A)|}\right) \leq |\alpha_{k,k+1}| \leq \frac{\pi}{2} + \arccos\left(\frac{\varepsilon_k}{|\lambda_m(A)|}\right). \tag{1.14}$$

The two side inequality (1.14) suggests that p_k and p_{k+1} may not become parallel as long as the constant value ε_k is sufficiently bounded away from zero. Conversely when p_k and p_{k+1} tend to be parallel, it implies from (1.14) that ε_k is approaching zero. As special cases, we report in Figs. 1.1 and 1.2 the geometry of the directions when $A \succ 0$ (Fig. 1.1) and A is indefinite (Fig. 1.2), respectively. In Fig. 1.1, when the eccentricity of the ellipse increases, then a (near) degeneracy may occur, but since $A \succ 0$ no degeneracy can be observed, i.e. p_k and p_{k+1} cannot become parallel. On the contrary, in Fig. 1.2 we have A indefinite, so that at Step k of the CG we can experience a degeneracy, with $p_k^T A p_k = 0$ and a premature CG halt. Equivalently, the point x_{k+1} approaches a point at infinity and the norm of $\|p_{k+1}\|$ becomes unbounded; moreover (see [12]), p_k and p_{k+1} tend to become parallel.

1.3.2 A new perspective for CG degeneracy using Grossone

This section details how the recently defined extension of real numbers based on *grossone* (see e.g. [1, 7, 21, 28, 32, 36, 37, 38, 39, 40, 41, 45], along with the related applications in optimization frameworks [2, 4, 5, 8, 9, 10, 11, 12, 23]), can be suitably used to model the CG degeneracy. In particular, we show that:

- adopting grossone algebra within the CG allows to recover the CG degeneracy in the indefinite case;
- coupling the CG with grossone provides results which exactly match the analysis carried on for the CG-based methods in [14, 31];
- our approach confirms the geometry behind CG degeneracy in the indefinite case, as underlined by polarity for quadratic hypersurfaces (see also Fig. 1.2 and [18]).

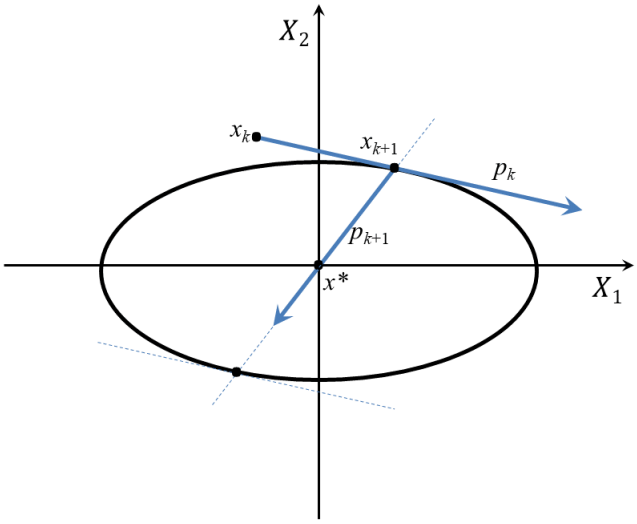

Fig. 1.1 The geometry behind the conjugate directions p_k and p_{k+1}: when $A > 0$ in (1.1) then without loss of generality in (1.14) we have $\varepsilon_k \geq \lambda_m(A) > 0$.

On this purpose, let us consider the computation of the steplength α_k at Step k of Table 1.1. Then, we set

$$p_k^T A p_k = s①, \tag{1.15}$$

where

- $s = O(①^{-1})$ if the Step k is a non-degenerate CG step (i.e. if $p_k^T A p_k \neq 0$),
- $s = O(①^{-2})$ if the Step k is a degenerate CG step (i.e. if $p_k^T A p_k = 0$).

In the last setting, following the standard Landau-Lifsitz notation we indicate with the symbol $O(①^{-2})$ *a term containing powers of* ① *at most equal to* -2. Observe that in the last case, standard results for grossone imply that the finite part of $p_k^T A p_k$ equals zero (or equivalently $p_k^T A p_k$ is infinitesimal). To large extent, the grossone-based expression on the righthand side of (1.15) can be further generalized; nevertheless, the setting (1.15) both seems simple enough and adequate to prove that the axioms and the basic algebra of grossone are well-suited to detail the behaviour of the CG, in the degenerate case.

In particular, a remarkable aspect of our approach is that using grossone to cope with CG degeneracy does not require to alter the scheme in Table 1.1, which is therefore almost faithfully applied '*as is*'. This represents an undoubted advantage with respect to the CG-based methods (namely *planar methods*) in [14, 15, 16, 25, 31], that indeed need to suitably rearrange the CG iteration in order to dodge degeneracy. The consequence of introducing grossone in Table 1.1 is analyzed in the next Sect. 1.3.3, where in case of CG degeneracy at Step k, the expressions of the coefficients and vectors at Step k explicitly depend on ① and its powers.

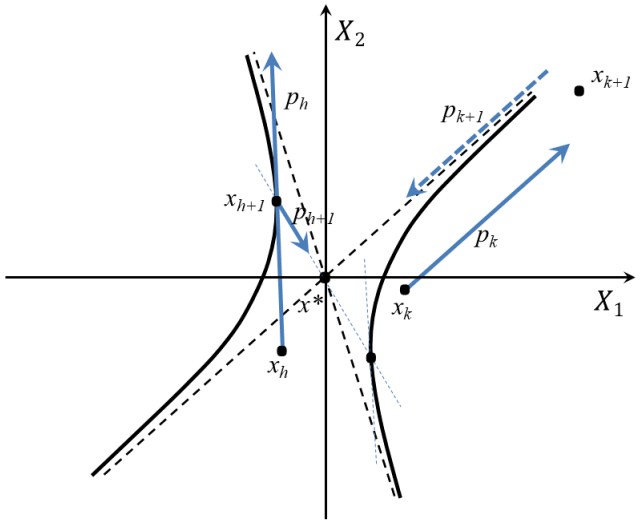

Fig. 1.2 The geometry of conjugate directions with A indefinite nonsingular in (1.1): since $|p_h^T A p_h|$ is sufficiently bounded away from zero then p_h and p_{h+1} are conjugate and *do not* tend to become parallel. Conversely, when $p_k^T A p_k \approx 0$ then p_k and p_{k+1} tend to become parallel.

1.3.3 Grossone for the degenerate Step k of the CG

From Table 1.1, the position (1.15) and the properties of the CG, when A is indefinite and *the Step k is degenerate* we have

$$r_{k+1} = r_k - \alpha_k A p_k = r_k - \frac{\|r_k\|^2}{s①} A p_k, \tag{1.16}$$

so that after a few arrangements (see [12])

$$p_{k+1} = r_{k+1} + \beta_k p_k = -\beta_{k-1} p_{k-1} - \frac{\|r_k\|^2}{s①} A p_k + \frac{\|r_k\|^2 \|A p_k\|^2}{s^2 ①^2} p_k. \tag{1.17}$$

We highlight that the CG degeneracy implies $s①$ to be infinitesimal, so that $\|p_{k+1}\|$ tends to be unbounded. The last result matches the geometric perspective reported in Sect. 1.3.1. Then, from (1.17) and the orthogonality/conjugacy conditions among vectors generated by the CG we can also infer

$$r_i^T r_j = 0, \qquad p_i^T A p_j = 0, \qquad \forall i \neq j,$$

along with

$$p_{k+1}^T A p_{k+1} = \frac{\|r_k\|^4}{s^2①^2}(Ap_k)^T A(Ap_k) - \frac{\|r_k\|^4\|Ap_k\|^4}{s^3①^3} + O(①), \tag{1.18}$$

(we recall that $O(①)$ in (1.18) sums up powers of ① equal to +1 and 0), and

$$r_{k+1}^T p_{k+1} = -\|r_k\|^2 + \frac{\|r_k\|^4\|Ap_k\|^2}{s^2①^2}. \tag{1.19}$$

From (1.16), (1.18)–(1.19) and recalling that when $p_k^T A p_k$ is infinitesimal so does $s①$, we obtain after some computation

$$r_{k+2} = r_k - \frac{\|r_k\|^2}{\|Ap_k\|^2}A(Ap_k) - \beta_{k-1}\frac{s①}{\|Ap_k\|^2}Ap_{k-1} + O(①^{-1}). \tag{1.20}$$

A noteworthy consequence of (1.16)–(1.17) and (1.20) is that in practice

- $r_1, \ldots, r_k$ are *independent* of ①,
- r_{k+1} and p_{k+1} heavily *depend* on ①,
- r_{k+2} is *independent* of negative powers of $s①$.

Thus, the geometric drawback detailed in Sect. 1.3.1, i.e. the CG degeneracy in the indefinite case, can be bypassed by exploiting the simple grossone algebra and neglecting the (infinitesimal) term with $s①$ in (1.20). This leaves the steps in the CG scheme of Table 1.1 fully unchanged.

Similarly, as regards the computation of the search direction p_{k+2}, by (1.16), (1.17) and (1.20) we have after some arrangements

$$p_{k+2} = r_k - \frac{\|r_k\|^2}{\|Ap_k\|^2}A(Ap_k) + \frac{\|r_{k+2}\|^2}{\|r_k\|^2}p_k - \beta_{k-1}\frac{s①}{\|Ap_k\|^2}Ap_{k-1} + O(①^{-1}). \tag{1.21}$$

Thus, similarly to r_{k+2}, in case of CG degeneracy also p_{k+2} is independent of negative powers of ① (i.e. equivalently $\|p_{k+2}\| < +\infty$, so that grossone algebra is able to bypass the degeneracy, recovering the CG iteration). After some computations it is also not difficult to verify that the vectors $r_{k+1}, p_{k+1}, r_{k+2}, p_{k+2}$ in (1.16), (1.17), (1.20) and (1.21) satisfy the standard CG properties

$$\begin{cases} r_{k+2}^T r_i = 0, & i = 1, \ldots, k+1, \\ p_{k+2}^T A p_i = 0, & i = 1, \ldots, k+1. \end{cases} \tag{1.22}$$

An additional remarkable comment from (1.21) is that, neglecting the terms which contain powers of $s①$ larger or equal to 1 (i.e. neglecting infinitesimals in (1.21)), the vector p_{k+2} coincides with the one obtained in Algorithm CG_Plan of [14] (a similar result holds considering the algorithm by Luenberger in [31], too). Therefore, the use of grossone to cope with a CG degeneracy at Step k does not simply recover the theory and the results in [14] and [31], but *it also retrieves the same scaling of the generated search directions*, which is a so relevant issue for large scale problems.
Also note that the expression of p_{k+1} in (1.17) explicitly includes negative powers of

Table 1.3 The $CG_{①}$ algorithm for solving the symmetric indefinite linear system $Ax = b$. In a practical implementation of Step k of $CG_{①}$, the test $p_k^T A p_k \neq 0$ may be replaced by the inequality $|p_k^T A p_k| \geq \varepsilon_k \|p_k\|^2$, with $\varepsilon_k > 0$ small.

$CG_{①}$: the CG method coupled with grossone

Data: Set $k = 1$, $x_1 \in \mathbf{R}^n$, $r_1 = b - Ax_1$, $s = O(①^{-2})$.
If $\|r_1\| = 0$, then STOP. Else, set $p_1 = r_1$.
Step k: If $\|p_k\|$ is finite (bounded) and $p_k^T A p_k \neq 0$ then compute
$\alpha_k = r_k^T p_k / p_k^T A p_k$, $x_{k+1} = x_k + \alpha_k p_k$, $r_{k+1} = r_k - \alpha_k A p_k$.
If $\|r_{k+1}\| = 0$, then STOP.
Elseif $\|p_k\|$ is finite (bounded) set $p_k^T A p_k = s①$ and compute
$r_{k+1} = r_k - \|r_k\|^2/(s①) A p_k$.
Else compute $\alpha_k = r_k^T p_k / p_k^T A p_k$,
$x_{k+1} = x_k + \alpha_k p_k$, $r_{k+1} = r_k - \alpha_k A p_k$.
If the finite part of r_{k+1} satisfies $\|r_{k+1}\| = 0$, then STOP.
Endif
Set $\beta_k = -r_{k+1}^T A p_k / p_k^T A p_k = \|r_{k+1}\|^2 / \|r_k\|^2$, and
$p_{k+1} = r_{k+1} + \beta_k p_k$, $k = k + 1$.
Go to **Step** k.

$s①$, showing that to large extent it can be assimilated to a vector with an unbounded norm, in accordance with Fig. 1.2, where x_{k+1} is a point at infinity.

Now, let us compute the iterate x_{k+2}, to verify to which extent using grossone may recover the CG iteration in case of degeneracy at Step k. By Table 1.1 and [12], and using

$$x_{k+2} \;=\; x_k + \alpha_k p_k + \alpha_{k+1} p_{k+1}$$

we obtain the final expression

$$x_{k+2} = x_k + \frac{\|r_k\|^2}{\|Ap_k\|^2} Ap_k - \frac{\|r_k\|^2}{\|Ap_k\|^4} (Ap_k)^T A(Ap_k) p_k + \; O(①^{-1}), \qquad (1.23)$$

which perfectly matches the expression of x_{k+2} computed in [14] and [31], as long as $O(①^{-1})$ is neglected. Therefore, in case of CG degeneracy at Step k, using grossone does not simply recover the residuals and search directions as in (1.22), but it also recovers the iterate x_{k+2}, since it is *independent of grossone*. Table 1.3 gives a formal description of $CG_{①}$, i.e. the CG method where ① is introduced in case of degeneracy, while Proposition 1 summarizes the results in the current section.

Proposition 1 *Let be given the indefinite linear system $Ax = b$, with $A \in \mathbf{R}^{n\times n}$ and n large. Suppose the $CG_{①}$ method in Table 1.3 is applied for its solution, using the position* (1.15)*. In case at Step $k \leq n$ the quantity $|p_k^T A p_k|$ is bounded away from zero, then the vectors generated by $CG_{①}$ exactly preserve the same properties of the corresponding vectors generated by the CG method in Table 1.1. Conversely, in case at Step k we have $p_k^T A p_k \approx 0$, then the vectors x_{k+2} in* (1.23) *and p_{k+2}*

in (1.21) *computed by the* $CG_{①}$ *differ by infinitesimals from the corresponding vectors computed by the CG_Plan in [14] (or by the algorithm in [31]).*

1.4 Large scale (unconstrained) optimization problems: the need of negative curvatures

The solution of indefinite linear systems like (1.1) is almost ubiquitous in both constrained and unconstrained optimization frameworks. E.g. the iterative solution of the following problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1.24}$$

where f is twice continuously differentiable and n is large, requires in a unified framework *(i)* to solve the associated Newton's equation (first order methods)

$$\nabla^2 f(x_j)\ s = -\nabla f(x_j), \tag{1.25}$$

and *(ii)* to identify those minima among the stationary points (second order methods – see also [6]). The task *(ii)* is often accomplished by selecting promising *negative curvature directions* for the function f at the current iterate x_j. In particular, for the sake of clarity here we restrict our attention to the Truncated Newton methods, that represent an efficient class of iterative methods to solve (1.24). Among them, the second order methods often rely on the theory in the seminal papers [33, 34], in order to assess algorithms generating negative curvature directions and converging to solutions where the Hessian matrix is positive semidefinite.

1.4.1 A theoretical path to the assessment of negative curvature directions

On the guidelines of the previous section, and with reference to [34], a sequence $\{d_j\}$ of effective negative curvature directions can be generated in accordance with the following assumption.

Assumption 1 Let us consider the optimization problem (1.24), with $f \in C^2(\mathbf{R}^n)$; the nonascent directions in the sequence $\{d_j\}$ are bounded and satisfy (see also [30])

(a) $\nabla f(x_j)^T d_j \le 0,\ \ d_j^T \nabla^2 f(x_j) d_j \le 0,$

(b) if $\lim_{j\to\infty} d_j^T \nabla^2 f(x_j) d_j = 0$ then $\lim_{j\to\infty} \min\left\{0,\ \lambda_m\left[\nabla^2 f(x_j)\right]\right\} = 0.$ □

In practice, (a) in Assumption 1 claims that at x_j the nonascent vector d_j cannot be a positive curvature direction for f. Conversely, condition (b) prevents the asymptotic convergence of the iterative algorithm to a region of concavity for the objective function. Evidently, on convex problems eventually the solutions of (1.24)

both fulfill Newton's equation and satisfy second order stationarity conditions, without requiring the computation of negative curvature directions.
We remark that even in case the current iterate x_j is far from a stationary point, the use of the negative curvature direction d_j may considerably enhance efficiency in Truncated Newton methods. The latter fact was clearly evidenced in [19, 24, 30], and follows by considering at x_j the quadratic expansion along the vector d

$$q_j(d) = f(x_j) + \nabla f(x_j)^T d + \frac{1}{2} d^T \nabla^2 f(x_j) d,$$

which implies for the directional derivative of $q_j(d)$

$$\nabla q_j(d)^T d = \nabla f(x_j)^T d + d^T \nabla^2 f(x_j) d.$$

Thus, $\nabla q_j(d)^T d$ may strongly decrease both when d is of descent for f at x_j and is a negative curvature direction for f at x_j.

On large scale problems, we highlight that computing effective negative curvature directions for f at x_j, fulfilling Assumption 1, is a challenging issue, but it may become an easier task as long as *proper factorizations* of $\nabla^2 f(x_j)$ are available. Indeed, suppose at x_j the nonsingular matrices $M_j \in \mathbf{R}^{n\times k}$ and $C_j, Q_j, B_j \in \mathbf{R}^{k\times k}$ are available such that

$$M_j^T \nabla^2 f(x_j) M_j = C_j, \qquad C_j = Q_j B_j Q_j^T. \tag{1.26}$$

Then, for $y \in \mathbf{R}^k$ and assuming that $w \in \mathbf{R}^k$ is an eigenvector of B_j associated with the negative eigenvalue $\lambda < 0$, relations (1.26) yield

$$\begin{aligned}(M_j y)^T \nabla^2 f(x_j)(M_j y) = y^T \left[M_j^T \nabla^2 f(x_j) M_j \right] y \; &= \; y^T C_j y \\ &= (Q_j^T y)^T B_j (Q_j^T y) \; = \; w^T B_j w \; = \; \lambda \|w\|^2 \; < \; 0,\end{aligned}$$

so that $d_j = M_j y$ represents a negative curvature direction for f at x_j. Furthermore, if λ is the *smallest* negative eigenvalue of B_j, then $M_j y$ also represents an eigenvector of $\nabla^2 f(x_j)$ associated to it.

The most renowned Krylov-subspace methods for symmetric indefinite linear systems (i.e. SYMMLQ, SYMMBK, CG, Planar-CG methods [15, 16, 17]) are all able to provide the factorizations (1.26) (i.e. they fulfill item (a) in Assumption 1) when applied to Newton's equation at x_j. Nevertheless, fulfilling also (b) and the *boundedness* of the sequence of negative curvature directions is definitely a less trivial task (see e.g. the counterexample in Sect. 4 of [34]).

In this regard, we highlight that, under mild assumptions, by the use of grossone (namely CG$_{①}$ in Table 1.3) we can easily yield an implicit Hessian matrix factorization as in (1.26), fulfilling both (a) and (b), as well as the boundedness of the negative curvature directions $\{d_j\}$ in Assumption 1. To accomplish this last task we need the next result (the proof follows from Lemma 4.3 in [34] and Theorem 3.2 in [20]).

Lemma 1 *Let problem* (1.24) *be given with* $f \in C^2(\mathbf{R}^n)$*, and consider any iterative method for solving* (1.24)*, which generates the sequence* $\{x_j\}$*. Let the level set* $\mathcal{L}_0 = \{x \in \mathbf{R}^n \ : \ f(x) \leq f(x_0)\}$ *be compact, being any limit point* $\bar{x}$ *of* $\{x_j\}$ *a stationary point for* (1.24)*, with* $\left|\lambda[\nabla^2 f(\bar{x})]\right| > \bar{\lambda} > 0$*. Suppose n iterations of a Newton-Krylov method are performed to solve Newton's equation* (1.25) *at iterate* x_j*, so that the decompositions*

$$R_j^T \nabla^2 f(x_j) R_j = T_j, \qquad T_j = L_j B_j L_j^T \tag{1.27}$$

are available. Moreover, suppose $R_j \in \mathbf{R}^{n\times n}$ *is orthogonal,* $T_j \in \mathbf{R}^{n\times n}$ *has the same eigenvalues of* $\nabla^2 f(x_j)$*, with at least one negative eigenvalue, and* $L_j, B_j \in \mathbf{R}^{n\times n}$ *are nonsingular. Let* $\bar{z}$ *be the unit eigenvector corresponding to the smallest eigenvalue of* B_j*, and let* $\bar{y} \in \mathbf{R}^n$ *be the (bounded) solution of the linear system* $L_j^T y = \bar{z}$*. Then, the vector* $d_j = R_j \bar{y}$ *is bounded and satisfies Assumption 1.*

However, three main insidious drawbacks arise from Lemma 1:

- both computing the eigenvector $\bar{z}$ of B_j and solving the linear system $L_j^T y = \bar{z}$ might not be considered *easy tasks*;
- the vector $\bar{y}$ should be provably *bounded* (equivalently $|\det(L_j)|$ should be bounded away from zero, for any $j \geq 1$);
- at iterate x_j the Newton-Krylov method adopted to solve (1.25) possibly does not perform exactly n iterations.

1.4.2 CG_① for the computation of negative curvature directions

Though the third issue raised in the end of the previous section remains of great theoretical interest, in practice when the sequence $\{x_j\}$ is approaching a stationary point, then we typically observe that Newton-Krylov methods tend to perform a large number of iterations to solve (1.25). On the contrary, the first two issues in the end of the previous section may be definitely more challenging, since they can be tackled only by a few Krylov-subspace methods, due to the structure and the complexity of the generated matrices L_j and B_j in Lemma 1. In our approach (see also [11] for more complete justifications), we can provably show that the use of $\mathrm{CG}_{①}$ can fruitfully fulfill all the hypotheses of Lemma 1. In particular, in the current section we detail how to couple the Krylov-subspace method in [14] with $\mathrm{CG}_{①}$, in the light of complying with the hypotheses of Lemma 1. Broadly speaking, observe that the Krylov-subspace method in [14] is basically a CG-based algorithm which performs exactly the CG iterations, as long as no degeneracy is experienced. On the contrary, in case at Step k the breakdown condition $p_k^T A p_k \approx 0$ occurs, a so called *planar step* is carried on to equivalently replace the Steps k and $k+1$ of the CG. Hence the taxonomy of *planar method*.

Assume without loss of generality that the Krylov-subspace method in [14] has performed n steps. Moreover, for the sake of simplicity hereafter in this section we

drop the dependency of matrices on the iterate subscript j. After some computation the following matrices are generated by the method in [14] (see also [13])

$$L = \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ 0 & L_{32} & L_{33} \end{pmatrix}, \qquad B = \begin{pmatrix} B_{11} & 0 & 0 \\ 0 & B_{22} & 0 \\ 0 & 0 & B_{33} \end{pmatrix},$$

where

$$L_{11} = \begin{pmatrix} 1 & & \\ -\sqrt{\beta_1} & \ddots & \\ & \ddots & 1 \end{pmatrix}, \; L_{21} = \begin{pmatrix} 0 \cdots & -\sqrt{\beta_{k-1}} \\ 0 \cdots & 0 \end{pmatrix}, \; L_{22} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{1.28}$$

$$L_{32} = \begin{pmatrix} -\sqrt{\beta_k \beta_{k+1}} & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix}, \; L_{33} = \begin{pmatrix} 1 & & \\ -\sqrt{\beta_{k+2}} & \ddots & \\ & \ddots & 1 \\ & -\sqrt{\beta_{n-1}} & 1 \end{pmatrix}, \tag{1.29}$$

and

$$B_{11} = \begin{pmatrix} \frac{1}{\alpha_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\alpha_{k-1}} \end{pmatrix}, \quad B_{22} = \begin{pmatrix} 0 & \sqrt{\beta_k} \\ \sqrt{\beta_k} & e_{k+1} \end{pmatrix}, \quad B_{33} = \begin{pmatrix} \frac{1}{\alpha_{k+2}} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\alpha_n} \end{pmatrix}, \tag{1.30}$$

such that

$$AR = RT, \qquad T = LBL^T, \tag{1.31}$$

being the matrix $R \in \mathbf{R}^{n \times n}$ orthogonal with

$$R = \left(\frac{r_1}{\|r_1\|} \; \cdots \; \frac{r_n}{\|r_n\|} \right), \tag{1.32}$$

and $r_{k+1} = Ap_k$, while $T \in \mathbf{R}^{n \times n}$ is tridiagonal. Moreover, the quantities $\{\alpha_i\}$, $\{\beta_i\}$, e_{k+1} in (1.28)-(1.30) are suitable scalars (being in particular $e_{k+1} = (Ap_k)^T A(Ap_k)/\|Ap_k\|^2$). We also recall that $\beta_i > 0$, for any $i \geq 1$.

Furthermore, to simplify our analysis, the above matrices L and B are obtained applying the method in [14], assuming that it performed all CG steps, with the exception of only one planar iteration (namely the k-th iteration), corresponding to have indeed $p_k^T Ap_k \approx 0$. Then, our approach ultimately consists to introduce the numeral grossone, to exploit a suitable matrix factorization in place of (1.31), such that Lemma 1 is fulfilled. To this purpose, let us consider again the algorithm CG$_①$ in Table 1.3 (see also [12]), and assume that at Steps k and $k+1$ it generated the

Table 1.4 Correspondence between quantities/vectors computed by the algorithm in [14] (*left*) and the algorithm $\mathrm{CG}_{①}$ in [12] (*right*).

Algorithm in [14]		$\mathrm{CG}_{①}$ in [12]
$r_i, \quad i = 1, \ldots, k$		$r_i, \quad i = 1, \ldots, k$
r_{k+1}		Ap_k
$r_i, \quad i \geq k+2$		$r_i, \quad i \geq k+2$ (neglecting the terms with $s①$)
$p_i, \quad i = 1, \ldots, k$		$p_i, \quad i = 1, \ldots, k$
p_{k+1}		$\frac{Ap_k}{\|Ap_k\|}$
$p_i, \quad i \geq k+2$		$p_i, \quad i \geq k+2$ (neglecting the terms with $s①$)
$\alpha_i, \quad i = 1, \ldots, k$		$\alpha_i, \quad i = 1, \ldots, k$
$\alpha_i, \quad i \geq k+2$		$\alpha_i, \quad i \geq k+2$ (neglecting the terms with $s①$)
$\beta_i, \quad i = 1, \ldots, k-1$		$\beta_i, \quad i = 1, \ldots, k-1$
β_k		$\frac{\|Ap_k\|^2}{\|r_k\|^2}$
$\beta_i, \quad i \geq k+1$		$\beta_i, \quad i \geq k+1$ (neglecting the terms with $s①$)

coefficients α_k and α_{k+1}. Thus, we have[1]

$$\begin{cases} \dfrac{1}{\alpha_k} = \dfrac{s①}{\|r_k\|^2} \\ \dfrac{1}{\alpha_{k+1}} = -\dfrac{\|Ap_k\|^2}{s①}. \end{cases} \tag{1.33}$$

Moreover, using the equivalence in Table 1.4 between the quantities computed by the algorithm in [14] and CG_①, we can compute the matrices

$$\hat{L} = \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & V_k C_k^{-1} & 0 \\ 0 & \hat{L}_{32} & L_{33} \end{pmatrix}, \qquad \hat{D} = \begin{pmatrix} B_{11} & 0 & 0 \\ 0 & \hat{B}_{22} & 0 \\ 0 & 0 & B_{33} \end{pmatrix}, \tag{1.34}$$

where L_{11}, L_{21}, are defined in (1.28), L_{33} in (1.29), B_{11}, B_{33} in (1.30), and

$$\hat{L}_{32} = \begin{pmatrix} \left(-\sqrt{\beta_k \beta_{k+1}} \quad 0\right) \cdot V_k C_k^{-1} \\ \vdots \\ 0 \end{pmatrix}, \quad \hat{B}_{22} = \begin{pmatrix} \dfrac{1}{\alpha_k s①} & 0 \\ 0 & \dfrac{s①}{\alpha_{k+1}} \end{pmatrix},$$

[1] More correctly, we urge to remark that the expressions (1.33) are obtained neglecting in the quantity α_{k+1} the infinitesimal terms, i.e. those terms containing negative powers of $s①$, that are indeed negligibly small due to the degenerate Step k in CG_①.

with

$$V_k C_k^{-1} = \begin{pmatrix} \frac{\|r_k\|\sqrt{\beta_k \lambda_k}}{\sqrt{\beta_k+\lambda_k^2}} & \frac{\sqrt{-\beta_k \lambda_{k+1}}}{\|Ap_k\|\sqrt{\beta_k+\lambda_{k+1}^2}} \\ \frac{\|r_k\|\lambda_k\sqrt{\lambda_k}}{\sqrt{\beta_k+\lambda_k^2}} & \frac{\lambda_{k+1}\sqrt{-\lambda_{k+1}}}{\|Ap_k\|\sqrt{\beta_k+\lambda_{k+1}^2}} \end{pmatrix}$$

and $\lambda_k,\ \lambda_{k+1}$ are the eigenvalues of B_{22} in (1.30). Thus, in Lemma 1 we have for matrix T_j the novel expression (see also (1.31))

$$T_j = LBL^T = \hat{L}\hat{D}\hat{L}^T.$$

We are now ready to compute at iterate x_j the negative curvature direction d_j which complies with Assumption 1, exploiting the decomposition $T_j = \hat{L}\hat{D}\hat{L}^T$ from Lemma 1. The next proposition, whose proof can be found in [11], summarizes the last result.

Proposition 2 *Suppose n iterations of $CG_{①}$ algorithm are performed to solve Newton's equation* (1.25)*, at iterate x_j, so that the decompositions*

$$R^T \nabla^2 f(x_j) R = T, \qquad T = \hat{L}\hat{D}\hat{L}^T$$

exist, where R is defined in (1.32)*, and $\hat{L}$ along with $\hat{D}$ are defined in* (1.34)*. Let $\hat{z}$ be the unit eigenvector corresponding to the (negative) smallest eigenvalue of $\hat{D}$, and let $\hat{y}$ be the solution of the linear system $\hat{L}^T y = \hat{z}$. Then, the vector $d_j = R\hat{y}$ is bounded and satisfies Assumption 1. In addition, the computation of d_j requires the storage of at most two n-real vectors.*

Observe that the computation of the negative curvature direction d_j requires at most the additional storage of a couple of vectors, with respect to the mere computation of a solution for Newton's equation at x_j. This confirms the competitiveness with respect to the storage required in [20]. Thus, the approach in this paper does not only prove to be applicable to large-scale problems, but it also simplifies the theory in [20]. We remark that the theory in [20] is, to our knowledge, the only proposal in the literature of *iterative computation* of negative curvature directions for large-scale problems, such that

- it does not rely on any re-computation of quantities (as in [24]),
- it does not require any full matrix factorization,
- it does not need any matrix storage.

1.5 Conclusions

We propose an unconventional approach for a twofold purpose, within large-scale nonconvex optimization frameworks. On one hand we consider the efficient solution of symmetric linear systems. On the other hand, our proposal is also able to generate negative curvature directions for the objective function, allowing convergence

towards stationary points satisfying second order necessary optimality conditions. Our idea exploits the simplicity of the algebra associated with the numeral grossone [37], which was recently introduced in the literature.

The theory in this paper also guarantees that the iterative computation of negative curvatures does not need any matrix storage, while preserving convergence. In addition, the proposed approach is independent under multiplication of the function by a positive scaling constant or adding a shifting constant. This is an important property that is specially exploited in global optimization frameworks (see e.g. [44]), where *strongly homogeneous* algorithms are definitely appealing.

Acknowledgements The author is thankful to the Editors of the present volume for their great efforts and constant commitment. The author is also grateful for the support he received by both the National Research Council–Marine Technology Research Institute (CNR-INSEAN), and the National Research Group GNCS (*Gruppo Nazionale per il Calcolo Scientifico*) within INδAM, Istituto Nazionale di Alta Matematica, Italy.

References

1. Antoniotti, L., Caldarola, F., Maiolo, M.: Infinite numerical computing applied to Hilbert's, Peano's, and Moore's curves. Mediterranean Journal of Mathematics **17(3)** (2020)
2. Astorino, A., Fuduli, A.: Spherical separation with infinitely far center. Soft Computing **24**, 17751–17759 (2020)
3. Chandra, R.: Conjugate gradient methods for partial differential equations. Ph.D. thesis, Yale University, New Haven (1978)
4. Cococcioni, M., Fiaschi, L.: The Big-M method with the numerical infinite M. Optimization Letters **15(7)** (2021)
5. Cococcioni, M., Pappalardo, M., Sergeyev, Y.D.: Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm. Applied Mathematics and Computation **318**, 298–311 (2018)
6. Curtis, F., Robinson, D.: Exploiting negative curvature in deterministic and stochastic optimization. Math. Program. **176**, 69–94 (1919)
7. D'Alotto, L.: Infinite games on finite graphs using grossone. Soft Computing **55**, 143–158 (2020)
8. De Cosmis, S., De Leone, R.: The use of grossone in mathematical programming and operations research. Applied Mathematics and Computation **218(16)**, 8029–8038 (2012)
9. De Leone, R.: Nonlinear programming and grossone: Quadratic programming and the role of constraint qualifications. Applied Mathematics and Computation **318**, 290–297 (2018)
10. De Leone, R., Egidi, N., Fatone, L.: The use of grossone in elastic net regularization and sparse support vector machines. Soft Computing **24**, 17669–17677 (2020)
11. De Leone, R., Fasano, G., Roma, M., Sergeyev, Y.D.: Iterative grossone-based computation of negative curvature directions in large-scale optimization. Journal of Optimization Theory and Applications **186(2)**, 554–589 (2020)
12. De Leone, R., Fasano, G., Sergeyev, Y.D.: Planar methods and grossone for the conjugate gradient breakdown in nonlinear programming. Computational Optimization and Applications **71(1)**, 73–93 (2018)
13. Fasano, G.: Planar-CG methods and matrix tridiagonalization in large scale unconstrained optimization. In: Di Pillo, G., Murli, A. (eds.) In: High Performance Algorithms and Software for Nonlinear Optimization. Kluwer Academic Publishers, New York (2003)

14. Fasano, G.: Conjugate Gradient (CG)-type Method for the solution of Newton's equation within Optimization Frameworks. Optimization Methods and Software **19(3-4)**, 267–290 (2004)
15. Fasano, G.: Planar-Conjugate Gradient algorithm for Large Scale Unconstrained Optimization, Part 1: Theory. Journal of Optimization Theory and Applications **125(3)**, 523–541 (2005)
16. Fasano, G.: Planar-Conjugate Gradient algorithm for Large Scale Unconstrained Optimization, Part 2: Application. Journal of Optimization Theory and Applications **125(3)**, 543–558 (2005)
17. Fasano, G.: Lanczos Conjugate-Gradient Method and Pseudoinverse Computation on Indefinite and Singular Systems. Journal of Optimization Theory and Applications **132(2)**, 267–285 (2007)
18. Fasano, G.: A Framework of Conjugate Direction Methods for Symmetric Linear Systems in Optimization. Journal of Optimization Theory and Applications **164(3)**, 883–914 (2015)
19. Fasano, G., Lucidi, S.: A nonmonotone truncated Newton-Krylov method exploiting negative curvature directions, for large scale unconstrained optimization. Optimization Letters **3(4)**, 521–535 (2009)
20. Fasano, G., Roma, M.: Iterative computation of negative curvature directions in large scale optimization. Computational Optimization and Applications **38(1)**, 81–104 (2007)
21. Fiaschi, L., Cococcioni, M.: Numerical asymptotic results in game theory using Sergeyev's Infinity Computing. International Journal of Unconventional Computing **14(1)** (2018)
22. Fletcher, R.: Conjugate Gradient Methods for Indefinite Systems. In: Watson G.A. (ed.) Proc. of the Dundee Biennal Conf. on Numerical Analysis. Springer, Berlin Heidelberg New York (1975)
23. Gaudioso, M., Giallombardo, G., Mukhametzhanov, M.S.: Numerical infinitesimals in a variable metric method for convex nonsmooth optimization. Applied Mathematics and Computation **318**, 312–320 (2018)
24. Gould, N., Lucidi, S., Roma, M., Toint, P.: Exploiting negative curvature directions in linesearch methods for unconstrained optimization. Optimization Methods and Software **14**, 75–98 (2000)
25. Hestenes, M.: Conjugate Direction Methods in Optimization. Springer Verlag, New York Heidelberg Berlin (1980)
26. Hestenes, M., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Nat. Bur. Stand. **49**, 409–436 (1952)
27. Higham, N.: Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia (1996)
28. Iavernaro, F., Mazzia, F., Mukhametzhanov, M.S., Sergeyev, Y.D.: Computation of higher order Lie derivatives on the Infinity Computer. Journal of Computational and Applied Mathematics **383** (2021)
29. Lanczos, C.: An Iterative Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators. Journal of Research of the National Bureau of Standards **45(4)**, Research Paper 2133 (1950)
30. Lucidi, S., Rochetich, F., Roma, M.: Curvilinear stabilization techniques for Truncated Newton methods in large scale unconstrained optimization. SIAM J. Optim. **8(4)**, 916–939 (1999)
31. Luenberger, D.G.: Hyperbolic Pairs in the Method of Conjugate Gradients. SIAM Journal on Applied Mathematics **17**, 1263–1267 (1996)
32. Mazzia, F., Sergeyev, Y.D., Iavernaro, F., Amodio, P., Mukhametzhanov, M.S.: Numerical methods for solving ODEs on the Infinity Computer. In: Sergeyev, Y.D., Kvasov, D.E., Dell'Accio, F., Mukhametzhanov, M.S. (eds.) Proc. of the 2nd Intern. Conf. "Numerical Computations: Theory and Algorithms", vol. **1776**, p. 090033. AIP Publishing, New York (2016)
33. McCormick, G.: A modification of Armijo's step-size rule for negative curvature. Mathematical Programming **13(1)**, 111–115 (1977)
34. Moré, J., Sorensen, D.: On the use of directions of negative curvature in a modified Newton method. Mathematical Programming **16**, 1–20 (1979)
35. Paige, C., Saunders, M.: Solution of sparse indefinite systems of linear equations. SIAM J. on Numerical Analysis **12**, 617–629 (1975)
36. Pepelyshev, A., Zhigljavsky, A.: Discrete uniform and binomial distributions with infinite support. Soft Computing **24**, 17517–17524 (2020)

37. Sergeyev, Y.D.: Arithmetic of Infinity. Edizioni Orizzonti Meridionali, CS (2003, 2nd ed. 2013)
38. Sergeyev, Y.D.: Lagrange Lecture: Methodology of numerical computations with infinities and infinitesimals. Rendiconti del Seminario Matematico dell'Università e del Politecnico di Torino **68(2)**, 95–113 (2010)
39. Sergeyev, Y.D.: Higher order numerical differentiation on the Infinity Computer. Optimization Letters **5(4)**, 575–585 (2011)
40. Sergeyev, Y.D.: Computations with grossone-based infinities. In: Calude, C.S., Dinneen, M.J. (eds.) Unconventional Computation and Natural Computation: Proc. of the 14th International Conference UCNC 2015, vol. LNCS **9252**, pp. 89–106. Springer, New York (2015)
41. Sergeyev, Y.D.: Un semplice modo per trattare le grandezze infinite ed infinitesime. Matematica nella Società e nella Cultura: Rivista della Unione Matematica Italiana **8(1)**, 111–147 (2015)
42. Sergeyev, Y.D.: Numerical infinities and infinitesimals: Methodology, applications, and repercussions on two Hilbert problems. EMS Surveys in Mathematical Sciences **4(2)**, 219–320 (2017)
43. Sergeyev, Y.D.: Independence of the grossone-based infinity methodology from non-standard analysis and comments upon logical fallacies in some texts asserting the opposite. Foundations of Science **24(1)** (2019)
44. Sergeyev, Y.D., Kvasov, D.E., Mukhametzhanov, M.S.: On strong homogeneity of a class of global optimization algorithms working with infinite and infinitesimal scales. Communications in Nonlinear Science and Numerical Simulation **59**, 319–330 (2018)
45. Žilinskas, A.: On strong homogeneity of two global optimization algorithms based on statistical models of multimodal objective functions. Applied Mathematics and Computation **218(16)**, 8131–8136 (2012)