Fast nonnegative least squares through flexible Krylov subspaces

FAST NONNEGATIVE LEAST SQUARES THROUGH FLEXIBLE KRYLOV SUBSPACES*

SILVIA GAZZOLA † AND YVES WIAUX†

Abstract. Constrained least squares problems arise in a variety of applications, and many iterative methods are already available to compute their solutions. This paper proposes a new efficient approach to solve nonnegative linear least squares problems. The associated KKT conditions are leveraged to form an adaptively preconditioned linear system, which is then solved by a flexible Krylov subspace method. The new method can be easily applied to image reconstruction problems affected by both Gaussian and Poisson noise, where the components of the solution represent nonnegative intensities. Theoretical insight is given, and numerical experiments and comparisons are displayed in order to validate the new method, which delivers results of equal or better quality than many state-of-the-art methods for nonnegative least squares solvers, with a significant speedup.

Key words. nonnegative least squares, flexible Krylov methods, conjugate gradient, image reconstruction, Gaussian noise, Poisson noise.

AMS subject classifications. 65F08, 65F10, 65F22, 65N20.

1. Introduction.

Let us consider the constrained linear least squares problem

$$\min_{x\geq 0} \Phi(x)\,, \quad \Phi(x) := \|b - Ax\|_2^2\,, \tag{1}$$

where $A \in \mathbb{R}^{M\times N}$, and the constraint $x \geq 0$ on $x \in \mathbb{R}^N$ is intended component-wise. This problem typically arises in imaging applications, where the entries (pixels) of x represent light intensities, which are nonnegative. Common examples are 2D image deblurring and reconstruction problems. When dealing with the former, one seeks to restore a blurred image. The blur is assumed to be known, and it depends on the application; for instance, when dealing with astronomical images acquired by ground-based telescopes, the blur is caused by atmospheric turbulence. When dealing with image reconstruction problems, the goal is to compute the image of an object given a set of projections thereof. Since the vector $b \in \mathbb{R}^M$ in (1) represents collected measurements, it is usually affected by (unknown) random noise η, i.e., $b = b^{ex} + \eta$. Image deblurring and reconstruction problems are also ill-posed, and some regularization should be considered in order to compute a meaningful approximation of the solution x^{ex} of the consistent exact system

$$Ax^{ex} = b^{ex}, \tag{2}$$

cf. [14]. The nonnegativity constraints in (1) can be considered as a form of regularization, since information about the problem is incorporated into the model. Nonnegatively constrained least squares also arise naturally in the class of Fourier imaging applications, where the measurements are acquired in the Fourier domain. Key examples are magnetic resonance imaging in medicine [17], and radio-interferometric imaging in astronomy [9], which typically require the solution of under-determined linear systems. Another strong information to enforce into imaging applications is sparsity, i.e., often only a small number of pixels (or, more in general, expansion coefficients with

*Submitted to the editors DATE.

Funding: This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC, grants EP/M019306/1 and EP/M008843/1)

†Institute of Sensors, Signals and Systems. School of Engineering and Physical Sciences. Heriot-Watt University, Edinburgh, UK. (S.Gazzola, Y.Wiaux@hw.ac.uk).

respect to a sparsity basis) are nonzero. The authors of [7] prove that, if the measurement matrix A in (2) satisfies some assumptions and if x^{ex} is sufficiently sparse, then requiring nonnegativity is equivalent to requiring sparsity (e.g., minimizing the ℓ_1 norm of x). Because of the multi-dimensional nature of imaging problems, the quantities in (1) are typically large-scale. In particular, with the advent of next-generation radio telescopes such as the Square Kilometre Array, solvers for (1) must be able to handle data sizes of the order of Terabytes or more: this implies that the employed algorithms should be scalable and extremely efficient, with an overall low computational cost, usually measured as total number of matrix-vector products.

In developing our theory, and in our experiments, problems affected by Gaussian white noise are mainly taken into account. Some possible extensions to include problems affected by both Gaussian and Poisson noise are outlined. The latter is a realistic model when dealing with astronomical images acquired by charge-coupled-devices (cf. [2], and the references therein). Following the derivations in [2], the generic noise model is given by

$$b = \text{Poisson}(Ax^{ex}) + \text{Poisson}(\beta\mathbf{1}) + \text{Normal}(\mathbf{0}, \sigma^2 I)\,, \tag{3}$$

where $\mathbf{1} = [1, \dots, 1]^T \in \mathbb{R}^M$, $\mathbf{0} = [0, \dots, 0]^T \in \mathbb{R}^M$, and I is the identity matrix of order M. After some statistical approximations, the noisy problem associated with (2) can be expressed as

$$\underbrace{b - \beta\mathbf{1}}_{=:\, b_\beta} = Ax^{ex} + \text{Normal}(\mathbf{0}, \underbrace{\text{diag}(Ax^{ex} + \beta\mathbf{1} + \sigma^2\mathbf{1})}_{=:\, C_\eta})\,, \tag{4}$$

where the terms Ax^{ex} and $\beta\mathbf{1}$ in the above random variables originate from the Poisson terms on the right-hand side of (3). The entries of A are assumed to be nonnegative (this is typically true for imaging problems), so that the diagonal elements of C_η in (4) are positive. In particular, when only Gaussian noise is involved, equation (4) reduces to

$$b = Ax^{ex} + \text{Normal}(\mathbf{0}, \sigma^2 I)\,, \tag{5}$$

and the associated nonnegative problem is (1). In general, the nonnegative least squares problem associated with the formulation (4) reads as

$$\min_{x \geq 0} \Phi_C(x), \quad \Phi_C(x) := \|C_\eta^{-1/2}(b_\beta - Ax)\|_2^2\,, \tag{6}$$

where a power of the covariance matrix is basically incorporated as left preconditioner for (1). Note that, since C_η is a function of Ax^{ex}, one should consider an approximation (cf. again [2]).

1.1. Related works. Over the last decades, many methods have been derived to iteratively solve problem (1). The most basic ones are the so-called gradient projection methods that, at each iteration, combine a descent step in the direction of the negative gradient of $\Phi(x)$ with a projection onto the nonnegative orthant. Some examples include the projected Landweber (or Richardson) method, the projected Cimmino method, and the projected Steepest Descent method (NNSD): these methods differ in the way the step-size is set, and in the possible use of fixed preconditioners (cf. [5] and the references therein). In the optimization literature, these methods can be incorporated into the framework of the so-called "iterative shrinkage thresholding algorithms" (ISTA), and a well-known accelerated version of them, dubbed "FISTA" [4], will be considered in the following sections.

Another class of methods for the solution of nonnegative least squares stems from the enforcement of the Karush-Kuhn-Tucker (KKT) conditions. For problem (1), the KKT conditions are necessary and sufficient for optimality, and they can be compactly expressed as

$$XA^T(Ax-b)=0\,,\quad \text{where}\quad X=\operatorname{diag}(x)\,,\quad x\geq 0\,,\quad A^T(Ax-b)\geq 0\,, \tag{7}$$

cf. [6, §6.8] and [20]. The authors of [13, 20] remark that the conditions in (7) can be also obtained by reformulating (1) as a convex unconstrained least squares problem with the component-wise reparametrization $x=e^z$, and by imposing the stationarity condition on the gradient of the objective function computed by the chain rule: this is immediate from the fact that

$$\nabla_z\Phi(x)=\operatorname{diag}(x)\nabla_x\Phi(x)=XA^T(Ax-b)\,. \tag{8}$$

However, this is not the point of view adopted in this paper. It should be underlined that the first condition in (7) is a nonlinear system of equations with respect to x, and different approaches for its solution are already available in the literature, cf. [2, 16, 20]. All of them can be expressed as fixed-point iterations of the form

$$x_m=x_{m-1}+\alpha_{m-1}\underbrace{X^{(m)}A^T(b-Ax_{m-1})}_{=:\,d_{m-1}}\,,\quad \text{where}\quad X^{(m)}=\operatorname{diag}(x_{m-1})\,. \tag{9}$$

Note that the vector d_{m-1} is the negative gradient $-\nabla_z\Phi(x)$ computed in x_{m-1}, so that these methods still descend along the direction of the gradient. However, with respect to the usual gradient descent methods applied to solve unconstrained least squares problems, the step length α_{m-1} in the above equation should be somewhat bounded in order to impose nonnegativity of the approximate solution at each iteration. For this reason, in [2, 20] these methods are named Modified Residual Norm Steepest Descent (MRNSD) algorithms.

When considering problems affected by both Gaussian and Poisson noise, a scheme similar to (9) can be applied to solve problem (6). Namely, the iterates are updated as

$$x_m=x_{m-1}+\alpha_{m-1}X^{(m)}A^TC_\eta^{-1}(b_\beta-Ax_{m-1})\,,\quad \text{with}\quad X^{(m)}=\operatorname{diag}(x_{m-1})\,, \tag{10}$$

and different methods originate from different approximations of the diagonal covariance matrix C_η. The authors of [2] outline two strategies: the first one consists of taking

$$C_\eta=\operatorname{diag}(b+\sigma^2\mathbf{1})\,, \tag{11}$$

and the corresponding method (10) is called weighted MRNSD (WMRNSD) algorithm; the second one considers a step-dependent C_η, defined as

$$C_\eta^{(m)}=\operatorname{diag}(Ax_{m-1}+\beta\mathbf{1}+\sigma^2\mathbf{1})\,, \tag{12}$$

and the corresponding method (10) is called k-weighted MRNSD (KWMRNSD). Many numerical experiments available in the literature show that the class of the MRNSD methods is efficient and reliable. However, as usual when considering gradient descent methods, the rate of convergence is quite slow. The authors of [2, 20] use some (additional) specific left preconditioners L to accelerate the convergence of the MRNSD methods (PMRNSD), so that A in (9) and (10) is replaced by the matrix $L^{-1}A$.

Krylov subspace methods are well-known iterative solvers that are commonly employed to regularize unconstrained least squares problems of the form

$$\min_{x\in\mathbb{R}^N} \Phi(x)\,, \tag{13}$$

where $\Phi(x)$ is defined as in (1), or the normal equations

$$A^TAx = A^Tb \tag{14}$$

associated with it, cf. [11, 14]. To keep the derivations simpler, only problems affected by Gaussian noise are considered at this stage. At the mth iteration of a projection method, an approximation x_m of a solution of (13) is computed by imposing x_m to belong to an m-dimensional approximation subspace $\mathcal{A}_m$, and additional constraints on the corresponding residual $r_m := b - Ax_m$ or A^Tr_m. Given an initial guess x_0 for the solution of (13), a Krylov method is a projection method whose approximation subspace $\mathcal{A}_m$ is of the form $x_0 + \mathcal{K}_m(\widetilde{A}, \widetilde{r}_0)$, where $\mathcal{K}_m(\widetilde{A}, \widetilde{r}_0)$ is a Krylov subspace defined as

$$\mathcal{K}_m(\widetilde{A}, \widetilde{r}_0) = \text{span}\{\widetilde{r}_0, \widetilde{A}\widetilde{r}_0, \ldots, (\widetilde{A})^{m-1}\widetilde{r}_0\}\,, \tag{15}$$

and is assumed to be of dimension m. Different Krylov methods are obtained by varying the conditions on r_m or A^Tr_m, the matrix $\widetilde{A}$, and the vector $\widetilde{r}_0$ in $\mathcal{K}_m(\widetilde{A}, \widetilde{r}_0)$: every square matrix linked to A, and any vector linked to r_0 can be potentially used (cf. [22, Chaper 6]). CG (Conjugate Gradient), CGLS (CG for Least Squares problems), GMRES (Generalized Minimal Residual), and RR-GMRES (Range Restricted GMRES) are among the most popular Krylov subspace methods. Various theoretical considerations and many numerical experiments available in the literature show that Krylov methods are much more efficient than the gradient descent methods for the solution of (13). Indeed, when considering image deblurring problems, some Krylov methods can compute a good regularized solution in only a few iterations, i.e., requiring only a few matrix-vector products with A and/or A^T. Unfortunately, Krylov subspace methods cannot be straightforwardly adopted to handle constrained problems in general, and problem (1) in particular.

The following considers only a version of the well-known CGLS method (cf. [6, §7.4] and [22, §6.7, §8.3]), which can be applied to the normal equations (14). At the mth iteration, the condition

$$x_m = \arg\min_{x\in x_0+\mathcal{A}_m} \Phi(x) \tag{16}$$

is imposed, with approximation subspace $\mathcal{A}_m = \mathcal{K}_m(A^TA, A^Tr_0)$. One way of deriving CGLS is to first generate an orthonormal basis $\{v_1, \ldots, v_m\}$ for the subspace $\mathcal{K}_m(A^TA, A^Tr_0)$ for increasing values of m, by means of a Gram-Schmidt-like orthonormalization process. Taking $V_m = [v_1, \ldots, v_m] \in \mathbb{R}^{N\times m}$ and considering the tridiagonal matrix $T_m = V_m^T(A^TA)V_m$, an approximation x_m for a solution of (13) satisfying the condition (16) is obtained by solving the projected system

$$T_my_m = \|r_0\|_2e_1\,, \quad e_1 = [1, 0, \ldots, 0]^T \in \mathbb{R}^m, \tag{17}$$

and by taking $x_m = x_0 + V_my_m$ [22, Chapter 6]. An analogous approach can be also adopted to derive the other Krylov subspace methods listed above. Figure 1 provides a typical example of the difficulties encountered when trying to enforce nonnegativity within Krylov methods. In this case, the CGLS method is extremely efficient to solve an unconstrained 1D deblurring problem.

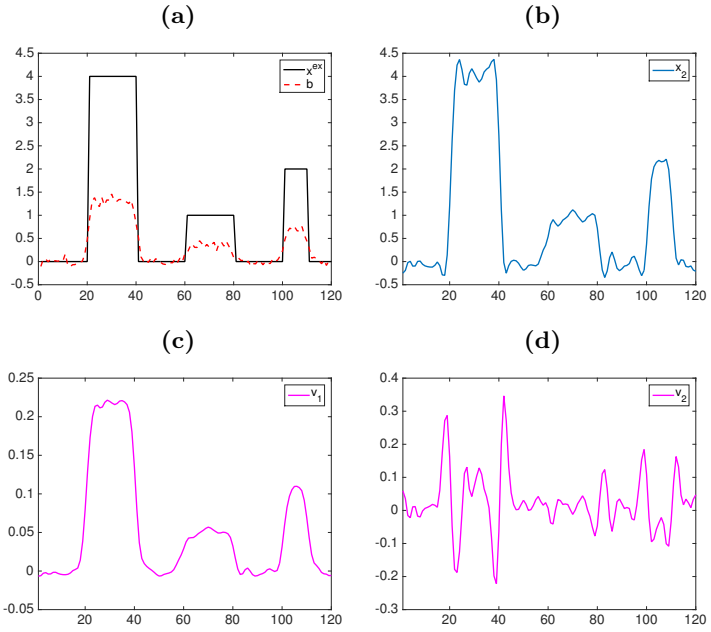

FIG. 1. *1D "image" deblurring and denoising problem. The 1D nonnegative signal x^{ex} is displayed in frame* ***(a)****, along with the vector b affected by Gaussian blur and Gaussian noise. Frame* ***(b)*** *displays the approximation x_2 obtained at the 2nd iteration of the CGLS method. Frames* ***(c)*** *and* ***(d)*** *display the vectors v_1 and v_2, which generate $\mathcal{K}_2(A^TA, A^Tb)$.*

Indeed, after only 2 iterations and with $x_0 = 0$, the behaviour of x^{ex} is quite accurately recovered. Imposing nonnegativity constraints would enhance even more the quality of the reconstruction. However, once V_2 is computed, x_2 is determined by only 2 parameters (i.e., the two entries of y_2 in (17)), and choosing them in order have x_2 nonnegative would heavily modify the overall behavior of the solution. Moreover, trivially trying to project the solution x_2 onto the nonnegative orthant would result in a new approximation not belonging to $\mathcal{K}_2(A^TA, A^Tb)$: the method so obtained would rapidly stagnate, with poor approximation properties (cf. Section 5). To the best of our knowledge, only heuristic approaches have been derived so far to approximate the solution of (1) within a Krylov subspace framework. The strategies in [8, 16] rely on inner-outer iteration cycles. In particular, the author of [16] proposes to solve the nonlinear system in (7) with a modified CGLS method: the matrix X is only updated at the beginning of each outer iteration, so that it is fixed during each inner iteration cycle. The occurrence of a restart is determined by the amount of variations in two consecutive approximations of x^{ex}. The authors of [8] instead propose to employ the CGLS, GMRES, or RRGMRES methods during the inner iterations to solve problems like (13), and restart with the nonnegatively projected last approximation of x^{ex} as soon as the discrepancy principle is satisfied. Although very efficient, this approach can only guarantee nonnegativity at each restart, and not during the inner iterations. Table 1 lists the acronyms associated with the most notable methods introduced so far.

1.2. Contributions. The goal of this paper is to present a reliable, efficient, and still somewhat heuristic new method to enforce nonnegativity within Krylov subspace methods. The new

TABLE 1
Popular methods for the solution of problems (1) or (6), related acronyms, and bibliographic references.

acronym	**method**	**problem**	**reference**
NNSD	Projected Steepest Descent	(1), (6)	[5]
FISTA	Fast iterative shrinkage thresholding algorithms (ISTA)	(1), (6)	[4]
MRNSD	Modified Residual-Norm Steepest Descent	(1)	[20]
PMRNSD	Preconditioned MRNSD	(1)	[20]
WMRNSD	Weighted MRNSD	(6)	[2]
KWMRNSD	k-weighted MRNSD	(6)	[2]
ReSt NNCG	Restarted CGLS with nonnegativity at each restart	(1)	[8]

approach merges the ability of delivering high-quality approximations typical for the MRNSD methods, with a fast convergence typical for Krylov methods for unconstrained problems. The example proposed in Section 1.1 suggests that, to succeed in this task, one should modify the usual space $\mathcal{K}_m(A^TA, A^Tr_0)$, and its basis vectors collected in V_m.

The starting point for the new method is the nonlinear system of equations in (7). Similarly to the MRNSD method, at the mth step of an iterative solver, the approximation $X \simeq \mathrm{diag}(x_{m-1}) =: X^{(m)}$ is chosen, so that the iteration-dependent linear system

$$X^{(m)}A^T(b - Ax) = 0 \tag{18}$$

should be solved. The condition $x_m \geq 0$ is imposed, while the condition $A^T(Ax_m - b) \geq 0$ is discarded during the iterations. The reason behind this choice lies in the "semi-convergence" phenomenon [11, 14], which is typical of regularizing iterative methods applied to solve least squares problems (13), where the matrix A is of ill-determined rank, and the data vector b is affected by noise. "Semi-convergence" means that the solution x_m approximates the solution x^{ex} of (2) at the beginning of the iterations, while it approaches the un-regularized and noise-dominated solution of (13) during the following iterations. "Semi-convergence" can usually be limited if some additional regularization is imposed during the iterative process. When problem (1) is solved (or, equivalently, when conditions (7) are satisfied), "semi-convergence" could still occur, as a nonnegative solution minimizing $\Phi(x)$ could be heavily corrupted by noise. Therefore, the goal of the new method is to efficiently deliver a nonnegative solution x_m at each iteration, and approximately satisfy the nonlinear system in (7). More precisely, the new idea is to consider a CGLS-like method for the normal equations (14), devised in such a way that the variable left "preconditioner" $X^{(m)}$ in (18) can be handled within the iterations, i.e., a so-called Flexible CGLS (FCGLS) method. The word "preconditioner" has been quoted, since the goal when solving (18) is to impose regularization within the iterations: therefore, in this setting, preconditioning is not intended in a classical sense, i.e., with the aim of accelerating the convergence of an iterative method. Flexible Krylov subspace methods were originally introduced a couple of decades ago to allow an increasingly improved preconditioner at each iteration of a standard Krylov subspace method: a typical instance is when the system defining the preconditioner is solved iteratively (possibly by another Krylov subspace method) with variable tolerance on the stopping criterion (cf. [1, Chapter 12], [22, §9.4], [23], and the references therein). The new FCGLS method derived in this paper is related to the Flexible CG (FCG) method described in [21]: indeed, FCGLS can be regarded as FCG applied to a normal equation system.

The specific use of flexible Krylov subspaces for regularizing linear ill-posed problems is quite recent (cf. [10, 18]). Both the approaches in [10, 18], and the one proposed in this paper, aim

at regularizing the original problem by including new information about the solution as soon as a new iteration is computed. For this reason, flexible methods are inherently very efficient. Indeed, when adopting a restarting strategy with a nonnegatively projected initial guess or an updated "preconditioner" for the new iterations [8, 16], a new Krylov subspace is generated at each outer iteration. On the contrary, the nonnegative FCGLS (NN-FCGLS) method presented in this paper (i.e., FCGLS devised to specifically deal with system (18) and secure $x_m \geq 0$) relies on both flexibility and suitable restarts, in such a way that only one (flexible) Krylov subspace is generated during the iterations.

NN-FCGLS can be also extended to handle problems affected by both Gaussian and Poisson noise, so that an approximate solution of (6) is computed. If the covariance matrix C_η is fixed, i.e., when approximation (11) is used, then the NN-FCGLS scheme can incorporate an additional preconditioner. If the covariance matrix C_η is adaptive, i.e., when approximation (12) is used, then the NN-FCGLS scheme can incorporate an updated $C_\eta^{(k)}$ at the kth restart. In this way, by exploiting the potentialities of flexible Krylov subspaces, the newly-proposed strategies embrace and improve the class of the MRNSD methods [2].

The remaining part of this paper is organized as follows: in Section 2, the FCGLS method is derived in a general framework. In Section 3, the NN-FCGLS for the approximate solution of (1) is introduced, and some its properties are discussed. In Section 4, two extensions of NN-FCGLS for the approximate solution of (6) are presented. Section 5 displays many numerical experiments performed with the new methods, and comparisons with the strategies reviewed in Section 1.1. Finally, Section 6 draws some conclusions and presents possible extensions.

2. Flexible Krylov subspace methods. Krylov subspace methods can be formulated as in Section 1.1 or, alternatively, in a mathematically equivalent way that consists in explicitly updating the current solution along a set of search directions: a new search direction is added at each iteration [1, Chapter 12]. More precisely, in the most general case, at the mth iteration of a Krylov subspace method for solving (13), one requires

$$x_m = x_{m-1} + \sum_{j=0}^{m-1} \alpha_j^{(m-1)} d_j \tag{19}$$

and computes the new search directions as

$$d_m = \bar{z}_m + \sum_{j=0}^{m-1} \beta_j^{(m-1)} d_j \,, \tag{20}$$

where $\bar{z}_m$ depends on the approximation space chosen in (16). This way of defining Krylov subspace methods is natural when imposing nonnegativity constraints, since one has the direct expression (23) for x_m. For CGLS-like methods applied to solve (14), $\bar{z}_m$ is a vector related to the normal equation residual $A^T r_m$, and the coefficients $\alpha_j^{(m-1)}$, $\beta_j^{(m-1)}$, $j = 0, \ldots, m-1$, are determined at each iteration by imposing the optimality condition (16) and the orthogonality of Ad_i, $i = 0, \ldots, m-1$. To enforce (16), one takes $\partial \Phi(x_m)/\partial \alpha_i^{(m-1)} = 0$ for $i = 0, \ldots, m-1$. This amounts to

$$(b - Ax_m, Ad_i) = 0 \,, \quad i = 0, \ldots, m-1 \,, \tag{21}$$

or, equivalently, by exploiting (19) and the orthogonality of Ad_i,

$$(r_{m-1}, Ad_i) = \sum_{j=0}^{m-1} \alpha_j^{(m-1)} (Ad_j, Ad_i) = \alpha_i^{(m-1)} (Ad_i, Ad_i)\,.$$

Here and in the following, $(\cdot,\cdot)$ denotes the standard inner product on $\mathbb{R}^N$ or $\mathbb{R}^M$. By considering (21) at step $m-1$, one gets that the only nonvanishing quantity on the left side of the previous equality is (r_{m-1}, Ad_{m-1}), and, therefore,

$$\alpha_{m-1}^{(m-1)} = \frac{(r_{m-1}, Ad_{m-1})}{(Ad_{m-1}, Ad_{m-1})} =: \alpha_{m-1}, \quad \alpha_j^{(m-1)} = 0\,, \quad j = 0, \dots, m-2\,. \tag{22}$$

The update formula (19) simplifies as

$$x_m = x_{m-1} + \alpha_{m-1} d_{m-1}\,, \tag{23}$$

and the corresponding residual r_m can be analogously updated as

$$r_m = r_{m-1} - \alpha_{m-1} Ad_{m-1}\,. \tag{24}$$

By enforcing the orthogonality of Ad_j in (20) it is immediate that

$$\beta_j^{(m-1)} = -\frac{(A\bar{z}_m, Ad_j)}{(Ad_j, Ad_j)}\,, \quad j = 0, \dots, m-1\,. \tag{25}$$

In the particular case of the unpreconditioned CGLS method, one takes $\bar{z}_j = A^T r_j =: z_j$. By exploiting (20) and (21), α_{m-1} in (22) can be alternatively redefined as

$$\alpha_{m-1} = \frac{(z_{m-1}, d_{m-1})}{(Ad_{m-1}, Ad_{m-1})} = \frac{(z_{m-1}, \bar{z}_{m-1})}{(Ad_{m-1}, Ad_{m-1})} = \frac{(z_{m-1}, z_{m-1})}{(Ad_{m-1}, Ad_{m-1})}\,; \tag{26}$$

moreover, after some straightforward algebraic manipulations that mainly involve (21) and (24), relation (20) reduces to

$$d_m = \bar{z}_m + \beta_{m-1} d_{m-1} = z_m + \beta_{m-1} d_{m-1}\,, \tag{27}$$

where, to keep the notation light,

$$\beta_{m-1}^{(m-1)} = -\frac{(A\bar{z}_m, Ad_{m-1})}{(Ad_{m-1}, Ad_{m-1})} = \frac{(\bar{z}_m, z_m - z_{m-1})}{(Ad_{m-1}, Ad_{m-1})\alpha_{m-1}} = \frac{(z_m, z_m)}{(z_{m-1}, z_{m-1})} =: \beta_{m-1}\,.$$

The simple two-term update formula (27) is linked to the fact that the matrix T_m in (17) is tridiagonal. A fixed symmetric positive definite left preconditioner $L \in \mathbb{R}^{N\times N}$ for the system (14) can be efficiently incorporated into CGLS by taking $\bar{z}_j = Lz_j = LA^T r_j$ (see [22, Chapter 9]).

When considering an iteration-dependent left preconditioner for the system (14), i.e., when solving a system of the form

$$L^{(m)} A^T A x = L^{(m)} A^T b\,, \tag{28}$$

where the matrix $L^{(m)} \in \mathbb{R}^{N\times N}$ may vary at each iteration, the short recurrence formula (27) does not hold anymore. In theory, in these situations, the full recurrence (20) should be implemented. In the following, the main steps to derive a new flexible version of CGLS, dubbed FCGLS, are outlined. The starting points for FCGLS are still the update formulas (19) and (20), where

$$\bar{z}_m = L^{(m)} z_m = L^{(m)} A^T r_m \tag{29}$$

in the latter. By enforcing condition (16), and requiring the vectors Ad_i, $i = 0, \dots, m$, to be orthogonal, the solution can be expressed as in (23), with α_{m-1} given by (22), while the new descent direction is given by (20), with the scalars $\beta_j^{(m-1)}$ given by (25). Note that, because of multiplication by the iteration-dependent matrix $L^{(m)}$, the full recurrence (20) should be implemented. Therefore, all the vectors d_j and Ad_j, $j = 0, \dots, m-1$, must be stored in order to compute d_m as in (20): this may result in a high storage cost as the number of iterations increases. The computational cost of each iteration of the FCGLS method is dominated by one matrix-vector product with A^T, one matrix-vector product with A, the update of $L^{(m)}$, and one matrix-vector product with $L^{(m)}$. More precisely, at the mth iteration, the residual r_m must be multiplied by A^T in order to compute the normal equation residual z_m, and the preconditioned normal equation residual $\bar{z}_m = L^{(m)} z_m$ must be multiplied by A in order to compute the coefficients $\beta_j^{(m-1)}$ and the new vectors d_m and Ad_m (relation (20) premultiplied by A is used for the latter).

To avoid high storage costs, sometimes it is appropriate to truncate the recursion (20) and retain at most $\hat{m} > 0$ terms. If a maximum number of iterations $m_{\max}$ is assigned, the choice $\hat{m} = m_{\max}$ corresponds to the full (untruncated) recursion (20), while the choice $\hat{m} = 1$ corresponds to the CGLS-like recurrence (27). In [21], a cyclic approach for defining the truncation parameter is outlined, which basically gives rise to a "mixed truncation-restart" strategy. When truncation happens, the orthogonality of all the directions Ad_j, $j = 0, \dots, m-1$ does not hold anymore; as a consequence, also the optimality property (16) is not guaranteed anymore (see (21) - (23)). However, in the FCG case, the author of [21] claims that no deterioration of the convergence might happen, and that the biggest difference in the behavior of the truncated and untruncated versions should be expected when the extremal eigenvalues are well separated (see also [23] and the references therein). In the case of matrices with ill-determined rank, the largest singular values are typically separated, while the smallest ones are clustered, so nothing can be concluded in principle. The above derivations are summarized in Algorithm 1.

Algorithm 1 breaks down at the mth iteration if $\alpha_{m-1} = 0$. This means that, at the $(m-1)$th iteration, a descent direction d_{m-1} has been computed, which is orthogonal to the current normal equation residual z_{m-1}. Although a formal convergence proof for FCGLS would require additional assumptions on A and $L^{(m)}$, one can claim that, if no breakdown happens in the untruncated version of Algorithm 1, then x_m converges monotonically to a solution of (13), i.e., $\|r_m\|_2 \leq \|r_{m-1}\|_2$. This is immediate from the fact that the approximation subspaces of the solution are nested, and the optimality condition (16) on the residual is imposed at each iteration.

3. Incorporating nonnegativity constraints. In order to approximate a solution of (1), the KKT conditions are enforced, i.e., at the mth iteration of an iterative solver, the system (18) is considered and the constraint $x_m \geq 0$ is imposed. The normal equation system (18) can be regarded as a left-preconditioned system (28), where the variable "preconditioner" is defined at the mth iteration as $L^{(m)} := X^{(m)} = \mathrm{diag}(x_{m-1})$. Therefore, the FCGLS method can be in principle used to solve (18). However, some modifications of the generic framework outlined in Algorithm 1 should be considered, which will lead to the NonNegative Flexible CGLS (NN-FCGLS) method

Algorithm 1 Flexible CGLS (FCGLS) method

Input: A, b, $L^{(0)}$, x_0, $\hat{m}$, $m_{\max}$.
Initialize: $r_0 = b - Ax_0$, $z_0 = A^T r_0$, $\bar{z}_0 = L^{(0)} z_0$.
Take $d_0 = \bar{z}_0$, and compute $w_0 := Ad_0 = A\bar{z}_0$.
For $m = 1, \dots,$ till a stopping criterion is satisfied OR $m = m_{\max}$:

1. Set $\alpha_{m-1} = (r_{m-1}, w_{m-1})/(w_{m-1}, w_{m-1})$.
2. Update $x_m = x_{m-1} + \alpha_{m-1} d_{m-1}$.
3. Update $L^{(m)}$.
4. Update $r_m = r_{m-1} - \alpha_{m-1} w_{m-1}$.
5. Compute $z_m = A^T r_m$ and $\bar{z}_m = L^{(m)} z_m$.
6. Compute $A\bar{z}_m$.
7. For $j = \max\{0, m - \hat{m}\}, \dots, m-1$, set $\beta_j^{(m-1)} = -(A\bar{z}_m, w_j)/(w_j, w_j)$.
8. Compute $d_m = \bar{z}_m + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} d_j$.
9. Update $w_m := Ad_m = A\bar{z}_m + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} w_j$.

described in Algorithm 2.

First of all, a nonnegative initial guess x_0 should be set; typical choices for x_0 are the projections of b or $A^T b$ onto the nonnegative orthant. Moreover, since a solution update of the form (23) is considered, one should bound the step-length along the search directions to guarantee nonnegativity at each iteration (this remedy is already suggested in [2, 16, 20]). It is immediate to prove that, in the FCGLS case, the bounded step-length $\bar{\alpha}_{m-1}$ is computed as follows:

$$\bar{\alpha}_{m-1} = \min\left(\alpha_{m-1}, \min\left(-x_{m-1}(d_{m-1} < 0)/d_{m-1}(d_{m-1} < 0) \right) \right), \tag{30}$$

where α_{m-1} is defined as in (26), and the MATLAB-like notations $x_{m-1}(d_{m-1} < 0)$ and $d_{m-1}(d_{m-1} < 0)$ mean that only the components of the vectors x_{m-1} and d_{m-1} corresponding to negative values of d_{m-1} are evaluated, respectively. In this way, the new step-length $\bar{\alpha}_{m-1}$ is automatically determined by simultaneously imposing $x_m \geq 0$, the optimality condition (16), and the orthogonality of Ad_j, $j = 0, \dots, m-1$. Therefore, if the full recurrence (20) is considered (i.e., if $\hat{m} = m_{\max}$ in Algorithm 1), the NN-FCGLS residuals decrease monotonically, i.e., $\|r_m\|_2 \leq \|r_{m-1}\|_2$. Moreover, because of the additional constraint $x_m \geq 0$, the NN-FCGLS residuals may decrease slower than the FCGLS and the standard CGLS ones (applied to solve the unconstrained least squares problem (13)). The following holds:

PROPOSITION 1. *The scalar $\bar{\alpha}_{m-1}$ defined in (30) is nonnegative.*

Proof. When $\bar{\alpha}_{m-1} = \alpha_{m-1}$, directly from (22) it suffices to prove that (r_{m-1}, Ad_{m-1}) is nonnegative. This follows from

$$\begin{aligned}
(r_{m-1}, Ad_{m-1}) &= \left(r_{m-1}, A \left(\bar{z}_{m-1} + \sum_{j=\max\{0,m-\hat{m}\}}^{m-2} \beta_j^{(m-2)} d_j \right) \right) \\
&= (r_{m-1}, A\bar{z}_{m-1}) + \sum_{j=\max\{0,m-\hat{m}\}}^{m-2} \beta_j^{(m-2)} (r_{m-1}, Ad_j) = (r_{m-1}, A\bar{z}_{m-1}) \\
&= (A^T r_{m-1}, \bar{z}_{m-1}) = (z_{m-1}, X^{(m-1)} z_{m-1}) \geq 0\,,
\end{aligned}$$

where the following are exploited: the property (21), the truncated update at step 8 of Algorithm 1, and the fact that the entries of the diagonal matrix $X^{(m-1)}$ are nonnegative. When $\bar{\alpha}_{m-1} \neq \alpha_{m-1}$, $\bar{\alpha}_{m-1}$ is nonnegative by definition. ∎

It must be remarked that the FCGLS method with α_{m-1} in (26) replaced by $\bar{\alpha}_{m-1}$ is very prone to stagnation. Indeed, $\bar{\alpha}_{m-1} = 0$ as soon as $[x_{m-1}]_i = 0$ for some $i = 1, \dots, N$ such that $[d_{m-1}]_i < 0$. At this point, $\bar{\alpha}_n = 0$ for all $n \geq m$, so that no updates of x_n happen, and Algorithm 1 with the choice (30) breaks down. The same is not true for the class of the MRNSD methods since, if $[x_{m-1}]_i = 0$, then $[d_{m-1}]_i = 0$, cf. (9). In order to overcome stagnation, NN-FCGLS relies on suitable restarts of FCGLS: a restart happens as soon as a maximum number of inner iterations $m_{\max}^{\text{in}}$ is performed, or $\bar{\alpha}_m = 0$. The iterations are terminated as soon as a stopping criterion is satisfied or, alternatively, when a maximum number of outer iterations $k_{\max}^{\text{out}}$ is performed. If a good estimate of the norm of the noise $\|\eta\|_2$ is known, a typical stopping criterion is the discrepancy principle (see [14, Chapter 5] and [8]), which prescribes to terminate the iterations when $\|r_m\|_2$ drops below $\|\eta\|_2$. If $\|\eta\|_2$ is not available, then one can monitor the relative change in the residual norms between two successive iterations, and stop when it drops below a prescribed tolerance. In addition to preventing stagnation, the restarting strategy of NN-FCGLS is beneficial to reduce the storage requirements of FCGLS, assuming that the untruncated update (20) of d_m is considered, and that $m_{\max}^{\text{in}}$ is low. Notation-wise, when considering NN-FCGLS, it is appropriate to denote some of the vectors appearing in Algorithm 1 by double indices: the lower index counts the number of inner iterations, while the upper index counts the number of outer iterations. Some properties

Algorithm 2 NonNegative FCGLS (NN-FCGLS) method

Input: A, b, $x_0^0 \geq 0$, $\hat{m}$, $m_{\max}^{\text{in}}$, $k_{\max}^{\text{out}}$.

For $k = 1, \dots,$ till a stopping criterion is satisfied OR $k = k_{\max}^{\text{out}}$:

- Take $L^{(0)} = X^{(0)} = \text{diag}(x_0^{k-1})$,
  $r_0^{k-1} = b - Ax_0^{k-1}$, $\bar{z}_0^{k-1} = L^{(0)} A^T r_0^{k-1}$, $d_0^{k-1} = \bar{z}_0^{k-1}$, and $w_0^{k-1} = A\bar{z}_0^{k-1}$.
- For $m = 1, \dots$ till $\bar{\alpha}_{m-1} = 0$, OR $m = m_{\max}^{\text{in}}$, OR a stopping criterion is satisfied:

  Run steps 1 – 9 of Algorithm 1. In particular:
  - at step 2 compute x_m^{k-1}, with α_{m-1} replaced by $\bar{\alpha}_{m-1}$, computed as in (30);
  - at step 3 take $L^{(m)} = X^{(m)} = \text{diag}(x_m^{k-1})$;
  - at steps 4–9 compute r_m^{k-1}, $\bar{z}_m^{k-1}$, d_m^{k-1}, and w_m^{k-1}.
- Let n_k be the stopping iteration. Take $x_0^k = x_{n_k}^{k-1}$.

of Algorithm 2 are derived in the following two results.

PROPOSITION 2. *If, at the beginning of the kth outer iteration, the ith component of d_0^{k-1} is not zero, then the ith component of x_0^{k-1} is not zero.*

Proof. One equivalently proves that $[x_0^{k-1}]_i = 0$ implies $[d_0^{k-1}]_i = 0$. This follows immediately from the definition of d_0^{k-1} given in Algorithm 2:

$$\left[d_0^{k-1}\right]_i = \left[x_0^{k-1}\right]_i \left[A^T(b - Ax_0^{k-1})\right]_i . \qquad \blacksquare$$

Note that the above proposition might not be true for d_m^{k-1} and x_m^{k-1}, with $m > 0$. Indeed, it can

happen that $[x_m^{k-1}]_i = 0$ and

$$\left[d_m^{k-1}\right]_i = \left[x_m^{k-1}\right]_i \left[A^T(b - Ax_m^{k-1})\right]_i + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} \left[d_j^{k-1}\right]_i$$
$$= 0 + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} \left[d_j^{k-1}\right]_i \neq 0\,.$$

If, in particular, $[d_m^{k-1}]_i < 0$, then the quantity $\bar{\alpha}_m$ in (30) would be zero, and a restart would happen in the NN-FCGLS method. Proposition 2 is important to assure that Algorithm 2 does not stagnate, i.e., at least one inner iteration can be computed during each outer iteration cycle, unless α_0 in (22) is zero. If $\alpha_0 = 0$, then $X^{(0)}A^T(b - Ax_0^k) = 0$, and x_0^k is a solution of the nonlinear system in (7). The following property also holds:

PROPOSITION 3. *Assume that $[x_0^{k-1}]_i = 0$ for some $i = 1, \ldots, N$, and that n_k iterations of FCGLS are performed at the kth outer cycle of Algorithm 2. Then $[x_m^{k-1}]_i = 0$ for all $m = 1, \ldots, n_k$.*

Proof. By induction. If $[x_0^{k-1}]_i = 0$ for some $i = 1, \ldots, N$, then

$$[x_1^{k-1}]_i = [x_0^{k-1}]_i + \bar{\alpha}_0[d_0^{k-1}]_i = [x_0^{k-1}]_i + \bar{\alpha}_0[x_0^{k-1}]_i[z_0^{k-1}]_i = 0\,.$$

In particular, $[d_0^{k-1}]_i = 0$. Now assume that $[x_2^{k-1}]_i = 0, \ldots, [x_\ell^{k-1}]_i = 0$ for $\ell < n_k$. This implies that $[d_2^{k-1}]_i = 0, \ldots, [d_{\ell-1}^{k-1}]_i = 0$, since

$$[d_j^{k-1}]_i = \bar{\alpha}_j^{-1}\left([x_{j+1}^{k-1}]_i - [x_j^{k-1}]_i\right) = 0 \quad \text{and} \quad \bar{\alpha}_j \neq 0 \quad \text{for} \quad j = 0, \ldots, \ell - 1\,.$$

Proving that $[x_{\ell+1}^{k-1}]_i = 0$ is immediate, since

$$[x_{\ell+1}^{k-1}]_i = [x_\ell^{k-1}]_i + \bar{\alpha}_\ell[d_\ell^{k-1}]_i = 0 + \bar{\alpha}_\ell \left([x_\ell^{k-1}]_i[z_\ell^{k-1}]_i + \sum_{j=\max\{0,\ell-\hat{m}\}}^{\ell-1} \beta_j^{(\ell-1)}[d_j^{k-1}]_i \right) = 0\,.$$

∎

The above result can be easily extended across the outer iterations, since $x_0^k = x_{n_k}^{k-1}$. Moreover, a similar property holds for the class of the MRNSD methods (this follows immediately from the update formula (9)), and it is important to guarantee that no oscillations occur around the newly-recovered zero components.

Similarly to Algorithm 1, the computational cost of each iteration of Algorithm 2 is dominated by a matrix-vector product with A, and a matrix-vector product with A^T; indeed, in the NN-FCGLS case, the cost of updating $L^{(m)} = X^{(m)}$, and computing a matrix-vector product with $X^{(m)}$ is negligible. Therefore, the cost of one iteration of NN-FCGLS is comparable to the cost of one iteration of CGLS, gradient projection (FISTA), and MRNSD.

4. Incorporating a covariance preconditioner. As explained in Section 1.1, when considering problems affected by both Gaussian and Poisson noise one has to deal with formulation (6). The KKT conditions associated with the constrained problem (6) are still leveraged; similarly to (7), they can be expressed as:

$$X(C_\eta^{-1/2}A)^T C_\eta^{-1/2}(Ax - b_\beta) = 0\,,\ \text{where } X = \operatorname{diag}(x)\,,\ x \geq 0\,,\ A^T C_\eta^{-1}(Ax - b_\beta) \geq 0\,. \tag{31}$$

Analogously to the Gaussian noise case, the last condition is discarded, while at the mth step of an iterative solver for the nonlinear system in (31), the approximation $X \simeq X^{(m)} =: \mathrm{diag}(x_{m-1})$ is chosen, so that the iteration-dependent linear system

$$X^{(m)} A^T C_\eta^{-1}(Ax - b_\beta) = 0\,, \quad \text{with} \quad x_m \geq 0 \tag{32}$$

should be approximately solved. The linear system in (32) has two preconditioners: while $X^{(m)}$ is updated at each iteration, at this stage C_η is assumed to be fixed (and, therefore, approximation (11) is used). Because of the presence of $X^{(m)}$ and the constraint $x_m \geq 0$, an iterative scheme such as NN-FCGLS must be used to approximate the solution of (32). However, the NN-FCGLS as described in Algorithm 2 should be reviewed in order to account for C_η, and the following derivations will lead to the Covariance-Preconditioned NN-FCGLS (CP-NN-FCGLS) method. The starting points for CP-NN-FCGLS are still the generic update formulas (19) and (20), where the coefficients $\alpha_j^{(m-1)}$ and $\beta_j^{(m-1)}$, $j = 0, \ldots, m-1$ are set by imposing an optimality condition similar to (16), i.e., $\partial \Phi_C(x_m)/\partial \alpha_i^{(m-1)} = 0$, and by imposing the $C_\eta^{-1/2} A d_i$ to be orthogonal, $i = 0, \ldots, m-1$. In particular, relation

$$(C_\eta^{-1/2} r_m, C_\eta^{-1/2} A d_i) = 0, \quad i = 0, \ldots, m-1$$

holds and, with the same reasoning used in Section 2,

$$\alpha_{m-1}^{(m-1)} = \frac{(C_\eta^{-1/2} r_{m-1}, C_\eta^{-1/2} A d_{m-1})}{(C_\eta^{-1/2} A d_{m-1}, C_\eta^{-1/2} A d_{m-1})} =: \alpha_{m-1}, \quad \alpha_j^{(m-1)} = 0, \quad j = 0, \ldots, m-2\,,$$

so that a short formula like (23) can be used to update the solution. Concerning the vector d_m, by exploiting the orthogonality of $C_\eta^{-1/2} A d_i$, $i = 0, \ldots, m-1$, the coefficients in (20) are computed as follows

$$\beta_j^{(m-1)} = -\frac{(C_\eta^{-1/2} A \bar{z}_m, C_\eta^{-1/2} A d_j)}{(C_\eta^{-1/2} A d_j, C_\eta^{-1/2} A d_j)}\,, \quad j = 0, \ldots, m-1\,,$$

where $\bar{z}_m = X^{(m)} A^T C_\eta^{-1} r_m = X^{(m)} A^T C_\eta^{-1}(b_\beta - A x_m)$. In order to guarantee $x_m \geq 0$ at the mth iteration, a bound analogous to (30) should be imposed and, similarly to NN-FCGLS, this could lead to stagnation. To overcome stagnation, an inner-outer iteration strategy is devised, which is based on restarts of the underlying FCGLS method. With the goal of giving an alternative approximation for the covariance matrix C_η defined in (4), when the kth restart happens one can choose to update the covariance in the following way:

$$C_\eta^{(k)} = \mathrm{diag}(A x_{n_k}^{k-1} + \beta \mathbf{1} + \sigma^2 \mathbf{1})\,, \tag{33}$$

where $x_{n_k}^{k-1}$ is the last approximation computed at the $(k-1)$th iteration cycle. In this way, a restart-dependent covariance matrix is defined, which is fixed during each inner iteration cycle. The previous derivations are summarized in Algorithm 3. Like Algorithm 2, the computational cost of one iteration of Algorithm 3 is dominated by one matrix-vector products with A and one matrix-vector product with A^T; the cost of updating $X^{(m)}$, $C_\eta^{(k)}$, and $(C_\eta^{(k)})^{-1/2}$, and performing matrix-vector multiplications with them, is negligible.

Algorithm 3 Covariance-Preconditioned NN-FCGLS (CP-NN-FCGLS) method

Input: A, b, $x_0^0 \geq 0$, $\hat{m}$, $m_{\max}^{\text{in}}$, $k_{\max}^{\text{out}}$.

For $k = 1, \dots,$ till a stopping criterion is satisfied OR $k = k_{\max}^{\text{out}}$:

- Take $X^{(0)} = \text{diag}(x_0^{k-1})$, and $\bar{C}_\eta = C_\eta$ as in (11) OR $\bar{C}_\eta = C_\eta^{(k)}$ as in (33);
  $r_0^{k-1} = b - Ax_0^{k-1}$, $\bar{r}_0^{k-1} = \bar{C}_\eta^{\ -1/2} r_0^{k-1}$, $\bar{z}_0^{k-1} = X^{(0)} A^T \bar{C}_\eta^{\ -1/2} \bar{r}_0^{k-1}$,
  $d_0^{k-1} = \bar{z}_0^{k-1}$, $w_0^{k-1} = A\bar{z}_0^{k-1}$, and $\bar{w}_0^{k-1} = \bar{C}_\eta^{\ -1/2} w_0^{k-1}$.
- For $m = 1, \dots$ till $\bar{\alpha}_{m-1} = 0$, OR $m = m_{\max}^{\text{in}}$, OR a stopping criterion is satisfied:
  - Set $\alpha_{m-1} = (\bar{r}_{m-1}^{k-1}, \bar{w}_{m-1}^{k-1}) / (\bar{w}_{m-1}^{k-1}, \bar{w}_{m-1}^{k-1})$ and take $\bar{\alpha}_{m-1}$ as in (30).
  - Update $x_m^{k-1} = x_{m-1}^{k-1} + \bar{\alpha}_{m-1} d_{m-1}^{k-1}$.
  - Update $X^{(m)}$.
  - Update $r_m^{k-1} = r_{m-1}^{k-1} - \bar{\alpha}_{m-1} w_{m-1}^{k-1}$ and $\bar{r}_m^{k-1} = \bar{C}_\eta^{\ -1/2} r_m^{k-1}$.
  - Compute $\bar{z}_m^{k-1} = X^{(m)} A^T \bar{C}_\eta^{\ -1/2} \bar{r}_m^{k-1}$.
  - Compute $A\bar{z}_m^{k-1}$.
  - For $j = \max\{0, m - \hat{m}\}, \dots, m-1$,
    set $\beta_j^{(m-1)} = -(\bar{C}_\eta^{\ -1/2} A\bar{z}_m^{k-1}, \bar{w}_j^{k-1}) / (\bar{w}_j^{k-1}, \bar{w}_j^{k-1})$.
  - Compute $d_m^{k-1} = \bar{z}_m^{k-1} + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} d_j^{k-1}$.
  - Update $w_m^{k-1} = Ad_m^{k-1} = A\bar{z}_m^{k-1} + \sum_{j=\max\{0,m-\hat{m}\}}^{m-1} \beta_j^{(m-1)} w_j^{k-1}$,
    and $\bar{w}_m^{k-1} = \bar{C}_\eta^{\ -1/2} w_m^{k-1}$
- Let n_k be the stopping iteration. Take $x_0^k = x_{n_k}^{k-1}$.

5. Numerical experiments. This section proposes three examples concerned with imaging problems. The first and the second ones are realistic astronomical image restoration test problems. The third one is an image reconstruction test problem that simulates an acquisition by computerized tomography with parallel beams. All the tests are performed running MATLAB R2015a on a single processor 2.2 GHz Intel Core i7. The image restoration test data are available within the toolbox [19]. In both Examples 1 and 2 the exact and perturbed images are of size 256×256 pixels, and the blurring matrix A is of size 65536×65536. A is not available explicitly but, using the software in [19], matrix-vector products with A and A^T are computed recurring to the point spread function, which defines the blur. The tomography test problem is available in [15], as `paralleltomo`. In Example 3 the Shepp-Logan phantom of size 256×256 pixels is used as exact image, and the parameters modeling the acquisition process are the angles where the sources are located, the number of rays for each angle, and the distance between the first and the last ray. The sparse sensing matrix A is of size $M \times 65536$, where M is varied by considering different acquisition parameters.

The new NN-FCGLS method is compared with other state-of-the-art solvers for the nonnegatively constrained linear least squares problems (1) or (6): almost all of them are among the ones surveyed in Section 1.1, whose acronyms are reported in Table 1. In particular, two versions of FISTA are considered: the basic one introduced in [4], without backtracking (for the choice of the stepsize), and the monotonic version of FISTA (MFISTA), with backtracking, described in [3]. Indeed, the performance of (M)FISTA is very much dependent on the stepsize t. According to the theory, one should take $t = 1/(2\sigma_1^2)$, where σ_1 is the largest singular value of A; in practice, t might be extremely expensive to compute: this is the case for all the test problems considered in

this section. When testing FISTA, an approximation of σ_1 is obtained by running a few iterations of the Golub-Kahan bidiagonalization algorithm [12]: in the following examples, 5 iterations are considered, so that the computational cost of this process is dominated by 5 matrix-vector products with A, and 5 matrix-vector products with A^T. When testing MFISTA, sometimes the notation MFISTA($1/t$) is used to emphasize the chosen stepsize; MFISTA stands for MFISTA($2\sigma_1^2$).

For each of the following examples, a table reporting the behavior of the different solvers is displayed: the second column (labeled "rel.error") reports the minimum relative error

$$\frac{\|x^{ex} - x_m\|_2}{\|x^{ex}\|_2} \tag{34}$$

attained by each method, where the iteration number m is displayed in the third column (labeled "iterations"). The fourth column (labeled "tot.time") reports the total time to perform m iterations, while the last column (labeled "av.time") shows the average time per iteration. The time is measured in seconds. All the values are averages over 10 runs of each test problem, with different noise realizations. For each test problem, the graphs show the error history, i.e., the values of the relative errors (34) at each iteration versus the number of iterations. Concerning NN-FCGLS, the two stopping criteria

$$\frac{|\|r_{m-1}\|_2 - \|r_m\|_2|}{\|r_{m-1}\|_2} < \tau\,, \quad \text{or} \quad \frac{\|r_{m-1}\|_2}{\|b\|_2} < \theta\widetilde{\varepsilon} \tag{35}$$

are taken into account. The first one monitors the stabilization of the residual norms, i.e., the iterative process terminates as soon as the relative difference between the residual norm of two consecutive iterates drops below a prescribed tolerance $\tau > 0$; note that the absolute value must be considered when the recurrences (20) are truncated, as the property $\|r_m\|_2 \leq \|r_{m-1}\|_2$ is not guaranteed anymore. The second one is the well-known discrepancy principle that, in the Gaussian noise case, can be applied only if a good estimate of the norm of the noise $\|\eta\|_2$ is known; the scalar θ is a safety factor ($\theta = 1.01$ is set in the following examples), while the scalar $\widetilde{\varepsilon}$ is the noise level, defined as $\widetilde{\varepsilon} = \|\eta\|_2/\|b^{ex}\|_2$. In order to produce the graphs of the error history, NN-FCGLS runs once for each stopping criterion, and special markers are used to emphasize the iterations satisfying each stopping criterion. Note that, in the following experiments, some additional iterations are typically performed after the stoping criterion is satisfied, in order to assess the stability of the method. Special markers also highlight the restarting iterations (recall that a restart happens as soon as a maximum number of inner iterations $m^{\text{in}}_{\max}$ is performed, or $\bar{\alpha}_m$ in (30) is zero).

Example 1. The first experiment is concerned with the deblurring and denoising of the so-called `star_cluster` test image of size 256×256 pixels [19]. The matrix A models a spatially variant blur, consisting of 25 locally spatially invariant point spread functions. Gaussian noise η of level $\widetilde{\varepsilon} = 10^{-2}$ is added to the blurred image b^{ex}. The exact image x^{ex} and the perturbed data b are among the ones displayed in Figure 3. The parameter $m^{\text{in}}_{\max}$ of Algorithm 2 is set to 20, and the initial guess x_0^0 is the projection of b onto the nonnegative orthant, while the parameter τ appearing in (35) is set to 10^{-4}; the untruncated version of NN-FCGLS ($\hat{m} = m^{\text{in}}_{\max}$) is implemented. For this problem, also a "naive" version of a "nonnegative CGLS" method (dubbed "naive NNCG") is considered. Naive NNCG is obtained by simply projecting the approximation (23) onto the nonnegative orthant at each iteration: recalling the remarks in Section 1.1, the success of this strategy is not guaranteed. Table 2 summarizes the performance of the different solvers, which are stopped after 400 (total) iterations. The initial guess for all the solvers is the projection of b onto the nonnegative orthant. Figure 2 displays the error history for a single run of various methods. The

TABLE 2
`star_cluster` *test problem, with* $\widetilde{\varepsilon} = 10^{-2}$. *Average values over* 10 *runs of the test problem, with different noise realizations.*

	rel.error	iterations	tot.time	av.time
NN-FCGLS	2.8132e-03	248.67	62.56	0.25
ReSt NNCG	5.3699e-03	261.00	113.51	0.43
FISTA	9.1283e-02	72.00	42.06	0.58
MFISTA	3.2803e-03	400.00	216.11	0.54
MFISTA(0.2)	3.2445e-03	400.00	194.78	0.49
MFISTA(5)	4.2834e-03	400.00	185.22	0.46
MRNSD	1.9889e-02	400.00	91.11	0.23
NNSD	8.3206e-02	400.00	91.59	0.23
naive NNCG	1.4028e-01	400.00	105.02	0.26

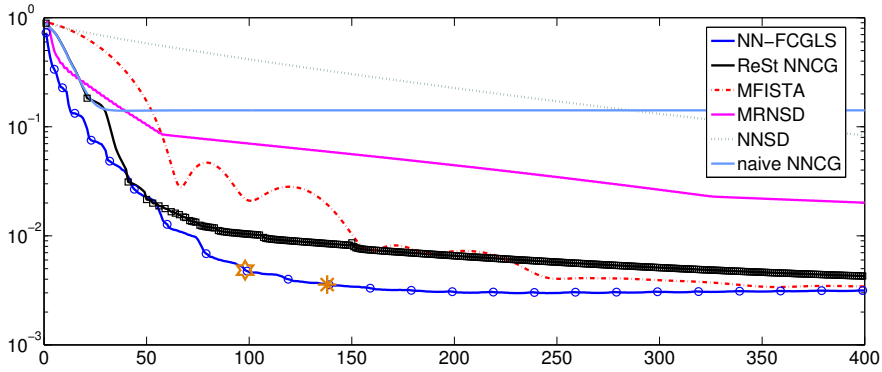

FIG. 2. `star_cluster` *test problem, with* $\widetilde{\varepsilon} = 10^{-2}$. *History of the relative errors. For the NN-FCGLS and ReSt CG methods, a small marker is used to emphasize the iterations where restarts happen. The big markers for NN-FCGLS highlight the stopping iterations.*

NN-FCGLS iteration satisfying the first criterion in (35) is marked by an asterisk, while the NN-FCGLS iteration satisfying the second criterion in (35) is marked by an hexagram. The stopping criterion provided in [8] for the ReSt NNCG method is based on the discrepancy principle, and it would prescribe to stop at the 78th total iteration: one can clearly see that immediate restarts happen after this iteration. The qualitative reason behind this phenomenon is that the CGLS residual (i.e., the discrepancy) stabilizes after a few outer iterations, even when incorporating the nonnegative initial guess: at this point, after just one inner iteration, the method is restarted and the decrease in the relative error is slower (with a rate comparable to the NNSD method). The opposite phenomenon occurs for the restarts of the NN-FCGLS method: indeed, restarts are more frequent during the early iterations, i.e., when there are more changes in the preconditioning matrix $X^{(m)}$ in (18). One can also clearly see that the behavior of the relative errors of MFISTA is not monotonic at all, as only the residuals $\|r_m\|_2$ are required to decrease monotonically. Moreover, a slight "semi-convergence" can be detected in the later iterations of NN-FCGLS: a similar phenomenon would appear also for the other solvers, provided that some more iterations are performed. Looking at Table 2, it should be also remarked that applying MFISTA($1/t$) with an initial large stepsize t results in a faster convergence, i.e., a lower relative error can be attained with the same number of

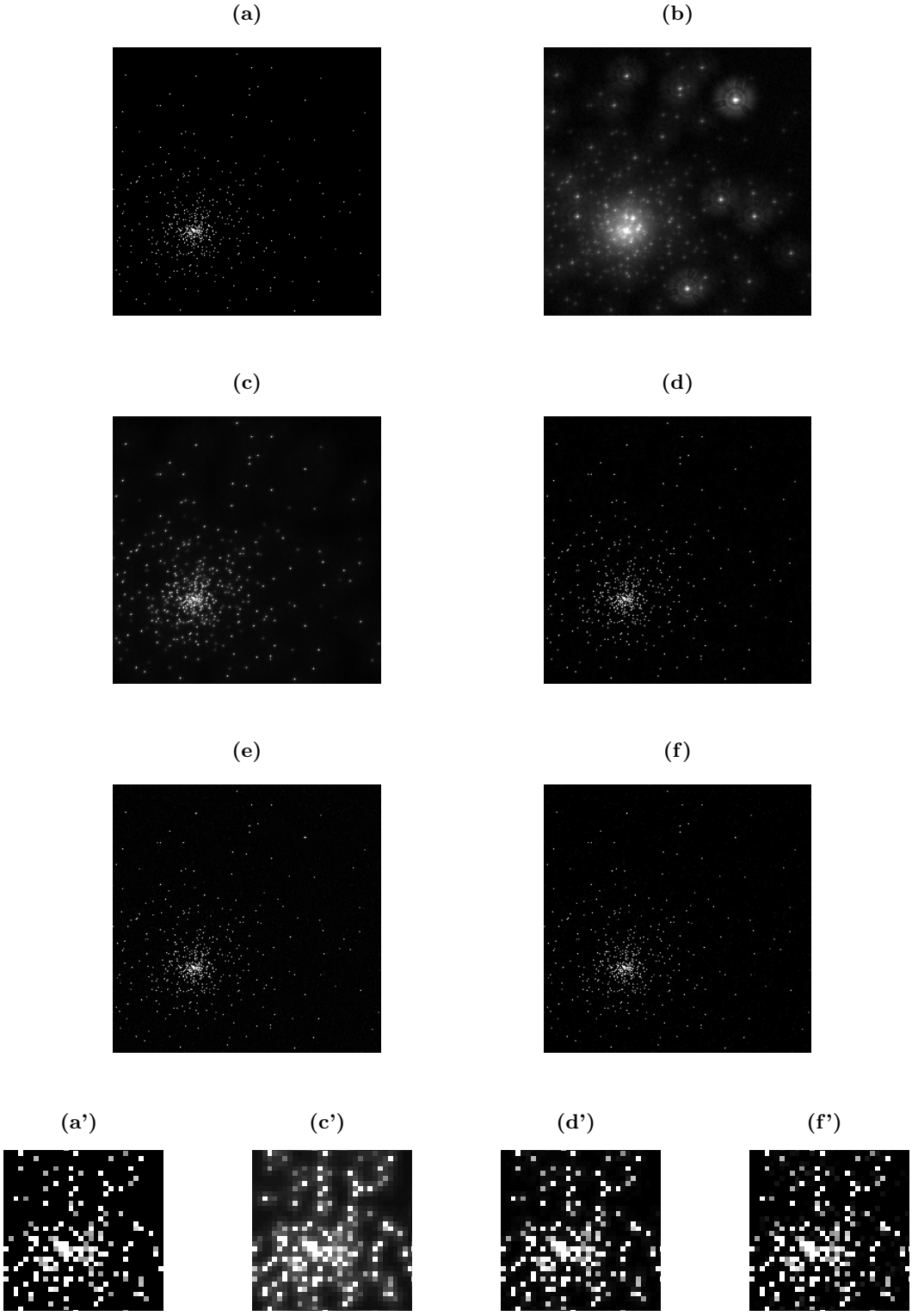

FIG. 3. *Images related to the* `star_cluster` *test problem, with* $\widetilde{\varepsilon} = 10^{-2}$. *Relative errors are reported within parentheses.* ***(a)*** *Exact image.* ***(b)*** *Blurred and noisy image. Restorations obtained at the* 200*th iteration of the:* ***(c)*** *MRNSD method* (0.0489), ***(d)*** *NN-FCGLS method* (0.0029), ***(e)*** *ReSt NNCG method* (0.0067), *and* ***(f)*** *MFISTA method* (0.0070). ***(a')***, ***(c')***, ***(d')***, ***(f')***: *blow-up of a portion of the images displayed in* ***(a)***, ***(c)***, ***(d)***, ***(f)***, *respectively.*

TABLE 3
`satellite` *test problem, with Gaussian noise of level* $\widetilde{\varepsilon} = 10^{-1}$. *Minimum relative error achieved by running NN-FCGLS with different combinations of* x_0^0 *and* $\hat{m}$.

rel.error	**iterations**	x_0^0	$\hat{m}$
3.4363e-01	63	b_0	$m_{\max}^{\text{in}}$
3.4131e-01	68	$\mathbf{0}$	$m_{\max}^{\text{in}}$
3.4121e-01	72	$(A^T b)_0$	$m_{\max}^{\text{in}}$
3.4363e-01	65	b_0	5
3.4347e-01	80	b_0	1

iterations. However, most likely, additional (inner) steps should be performed for backtracking, so that the time per iteration and the overall computational time might be higher than MFISTA($1/t$) with a small t. Moreover, though the relative errors delivered by NN-FCGLS and MFISTA are comparable, the latter requires a larger number of iterations, and each iteration is on average slower than any NN-FCGLS iteration (this is clear looking at Figure 2). Figure 3 shows the exact and acquired images, and some restorations obtained at the 200th iteration of various methods. The bottom images in Figure 3 are blow-ups of some of the upper images, and it is evident that the image computed by the NN-FCGLS method has less ringing artifacts around the white dots (stars), though the restoration computed by MFISTA has similar quality. In this experiment, as well as some of the following ones, the reconstructed images are displayed after performing a fixed number of iterations, so that the different progress of each method can be visually compared.

Example 2. The second set of experiments deals with the deblurring and denoising of the well-known `satellite` test image of size 256×256 pixels [19]. Two different matrices A are taken into account, which model spatially invariant atmospheric blur. In the first case, only Gaussian noise of level $\widetilde{\varepsilon} = 10^{-1}$ is added, while in the second case both Gaussian and Poisson noise of total level around $\widetilde{\varepsilon} = 1.5 \cdot 10^{-2}$ is added. In both cases, the parameter $m_{\max}^{\text{in}}$ of Algorithm 2 is set to 20, while the parameter τ appearing in (35) is set to 10^{-4}. The first tests consider only Gaussian noise. The left frame of Figure 4 displays the effect of different initial guesses x_0^0 on the relative errors computed by the NN-FCGLS method, with full recurrences for the updates of d_m^{k-1}. The vectors x_0^0 used for this experiment are the projection of the data vector b onto the nonnegative orthant (denoted by b_0), and the projection of the vector $A^T b$ onto the nonnegative orthant (denoted by $(A^T b)_0$); also the option $x_0^0 = \mathbf{0}$ is considered and, to avoid immediate stagnation of NN-FCGLS, the identity matrix of order N is set as first "preconditioner" $X^{(0)}$ in (18). The right frame of Figure 4 considers the effect of different values of the truncation parameter $\hat{m}$ on the relative errors computed by the NN-FCGLS method, with $x_0^0 = b_0$. When updating the descent direction d_m^{k-1} in Algorithm 2 (see also steps 7 and 8 of Algorithm 1), the following values are used: $\hat{m} = m_{\max}^{\text{in}}$ (corresponding to a full recurrence), $\hat{m} = 1$ (corresponding to a CGLS-like recurrence), and $\hat{m} = 5$. Table 3 summarizes the results obtained running these first experiments: more precisely, the values of the minimum relative error (34), and the corresponding m are reported. Looking at Table 3, one can conclude that NN-FCGLS is very robust with respect to both the choices of x_0^0 and $\hat{m}$. The effect of different choices of x_0^0 is mostly evident during the early iterations, when $x_0^0 = b_0$ seems to outperform the other options (this is quite common when considering image deblurring problems). Moreover, though $\hat{m} = m_{\max}^{\text{in}}$ should be chosen according to the theory of FCGLS, also lower values of $\hat{m}$ can deliver results of the same quality, except for the early iterations. In the following tests performed with the `satellite` test image, the choices $x_0^0 = b_0$ and $\hat{m} = m_{\max}^{\text{in}}$ will be considered. The vector b_0 is taken as initial guess for all the solvers.

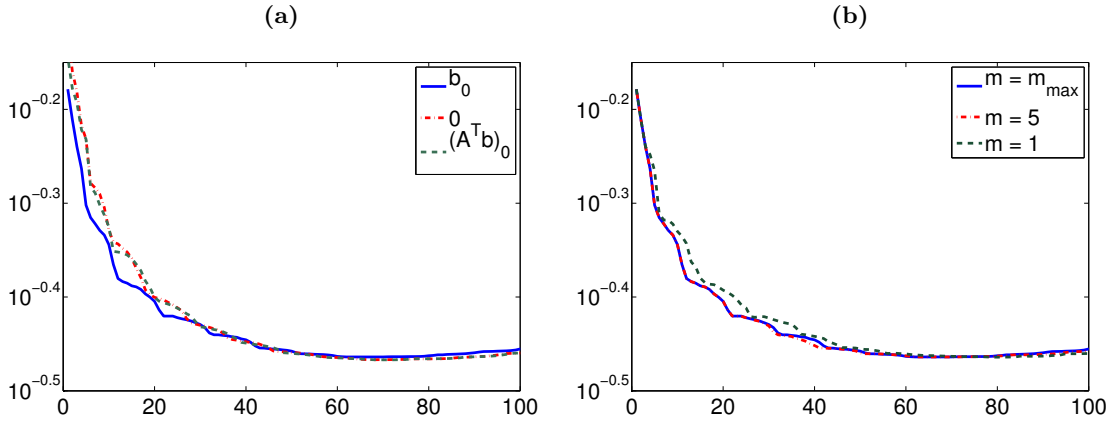

FIG. 4. `satellite` *test problem, with Gaussian noise of level* $\widetilde{\varepsilon} = 10^{-1}$. *History of the relative errors of the NN-FCGLS method, varying the initial guess* x_0^0 *(frame* ***(a)****), and varying the truncation parameter* $\hat{m}$ *for the update of* d_m^{k-1} *(frame* ***(b)****).*

TABLE 4
`satellite` *test problem, with Gaussian noise of level* $\widetilde{\varepsilon} = 10^{-1}$. *Average values over* 10 *runs of the test problem, with different noise realizations.*

	rel.error	**iterations**	**tot.time**	**av.time**
NN-FCGLS	3.5098e-01	70.33	5.49	0.08
ReSt NNCG	4.0957e-01	106.67	9.38	0.08
FISTA	3.2969e-01	164.33	21.22	0.12
MFISTA	3.2583e-01	177.00	23.10	0.13
MFISTA(0.2)	3.3318e-01	137.00	20.58	0.15
MFISTA(5)	3.3397e-01	200.00	26.86	0.13
MRNSD	3.7720e-01	200.00	12.55	0.06
PMRNSD	4.0032e-01	37.33	2.62	0.07
NNSD	4.3095e-01	200.00	13.82	0.07

The next tests compare NN-FCGLS with the ReSt NNCG, (M)FISTA($1/t$), MRNSD, and NNSD methods. Following the suggestions in [5, 20], also a preconditioned version of MRNSD (dubbed "PMRNSD") is taken into account. A special (fixed) preconditioner for PMRNSD is computed at the beginning of the iterations by exploiting an approximation of the singular value decomposition of A by means of the fast Fourier transform. The smaller approximate singular values are set to one, so that they are not inverted when applying preconditioning. Table 4 summarizes the average results over 10 runs of the `satellite` test problem, with different realizations of the of the random noise. All the methods are stopped after 200 (total) iterations. Figure 5 displays the history of the relative errors for some of the methods considered in Table 4; for this test, the NN-FCGLS iterations satisfying the first and the second stopping criteria in (35) coincide, and they are highlighted by big markers. The "semi-convergence" phenomenon is evident for both the PMRNSD and NN-FCGLS methods, which are the fastest solvers for this test problem. Figure 6 displays the restored images of best quality obtained by different methods. One can see MFISTA to deliver slightly better results than NN-FCGLS for this particular example. However, looking at

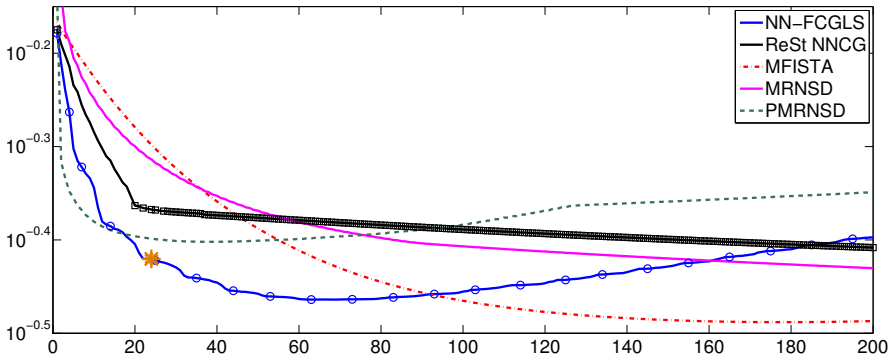

FIG. 5. `satellite` *test problem, with Gaussian noise of level* $\widetilde{\varepsilon} = 10^{-1}$. *History of the relative errors. For the NN-FCGLS and ReSt CG methods, a small marker is used to emphasize the iterations where restarts happen. The big markers for NN-FCGLS highlight the stopping iterations (which coincide, in this case).*

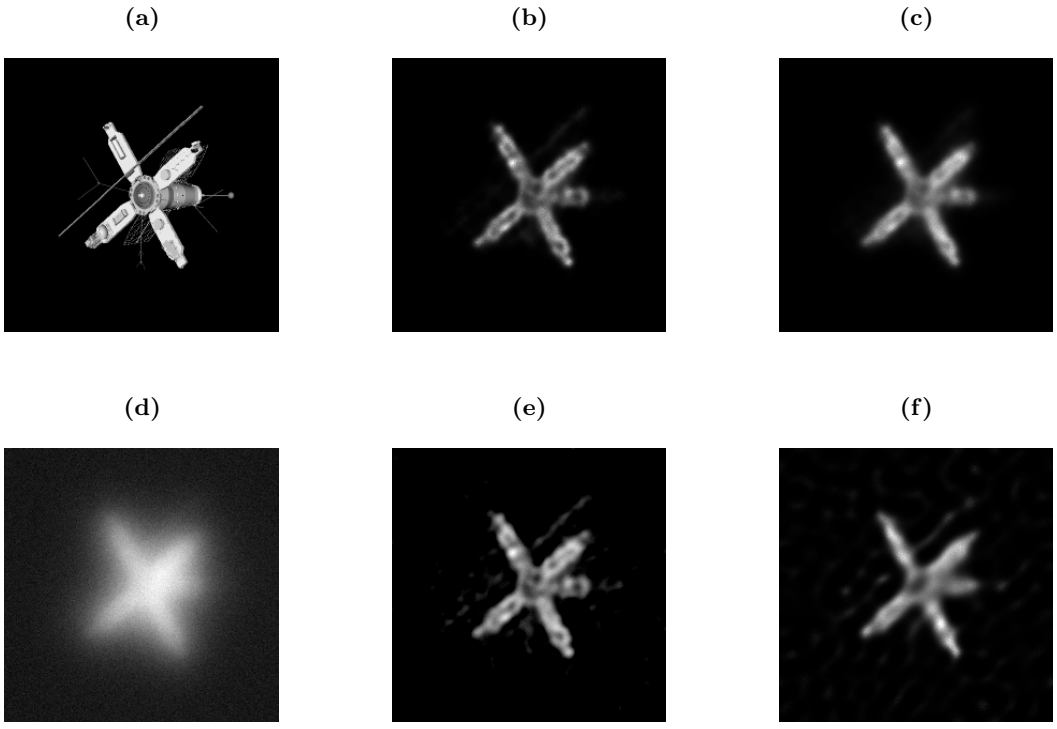

FIG. 6. *Images related to the* `satellite` *test problem, with Gaussian noise of level* $\widetilde{\varepsilon} = 10^{-1}$. *Relative errors are reported within parentheses.* ***(a)*** *Exact image.* ***(b)*** *Restoration by NN-FCGLS, iteration* # 63 (0.3436). ***(c)*** *Restoration by MRNSD, iteration* # 200 (0.3711). ***(d)*** *Blurred and noisy image.* ***(e)*** *Restoration by MFISTA, iteration* # 200 (0.3259). ***(f)*** *Restoration by PMRNSD, iteration* # 38 (0.3962).

TABLE 5
`satellite` *test problem, with both Gaussian and Poisson noise of level around* $\widetilde{\varepsilon} = 1.5 \cdot 10^{-2}$. *Average values over* 10 *runs of the test problem, with different noise realizations.*

	rel.error	iterations	tot.time	av.time
CP-NN-FCGLS	1.2785e-01	300.00	31.65	0.08
CP-NN-FCGLS(k)	1.2778e-01	300.00	32.17	0.08
WMRNSD	1.8201e-01	300.00	28.34	0.09
KWMRNSD	1.3590e-01	300.00	37.19	0.12

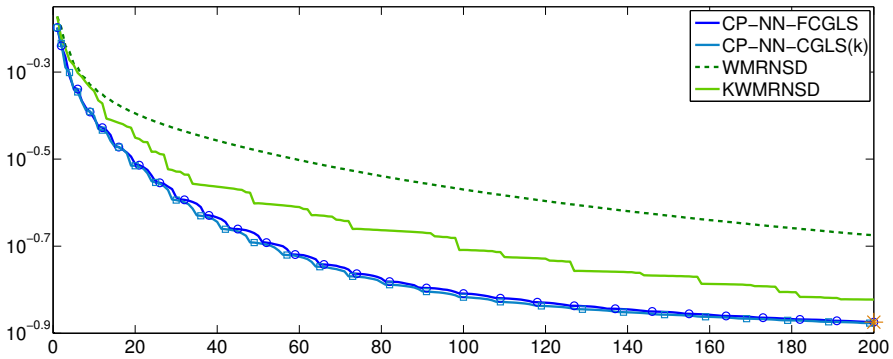

FIG. 7. `satellite` *test problem, with Gaussian and Poisson noise of level around* $\widetilde{\varepsilon} = 1.5 \cdot 10^{-2}$. *History of the relative errors. For the CP-NN-FCGLS and CP-NN-FCGLS(k) methods, a small marker is used to emphasize the iterations where restarts happen. The big mark highlights the iteration satisfying the first stopping criterion for CP-NN-FCGLS in (35): in this case, the maximum number of total iterations is reached.*

the displayed images, the differences are not huge, and NN-FCGLS is much faster. The remaining tests are concerned with the deblurring and denoising of the `satellite` image, perturbed by both Gaussian and Poisson noise. The Gaussian standard deviation parameter is $\sigma = 20$, while the Poisson parameter is $\beta = 60$. Table 5 compares different methods for solving the nonnegatively constrained covariance-preconditioned problem (6). In particular, the WMRNSD and KWMRNSD methods [2] are compared with the new CP-NN-FCGLS method. Regarding the latter, the notation CP-NN-FCGLS(k) is used to emphasize that the restart-dependent covariance matrix $C_\eta^{(k)}$ is used in Algorithm 3. Figure 7 displays the history of the relative errors, while Figure 8 displays the restorations obtained at the 100th iteration of the KWMRNSD and CP-NN-FCGLS(k) methods. Looking at the results for the Gaussian and Poisson noise case, one can see that the new methods are inherently faster than the MRNSD-like methods. However, while KWMRNSD has a better performance than WMRNSD for this test problem, CP-NN-FCGLS behaves very similarly to CP-NN-FCGLS(k).

Example 3. The last set of experiments uses the `parallelomo` test problem [15]. By varying the acquisition parameters, two sparse sensing matrices A are obtained: one of size 81088×65536 (overdetermined case), and one of size 32580×65536 (underdetermined case). Gaussian noise of level $\widetilde{\varepsilon} = 5 \cdot 10^{-2}$ is added. The parameter $m_{\max}^{\text{in}}$ of Algorithm 3 is set to 10, the untruncated version of FCGLS is considered, and the parameter τ appearing in (35) is set to 10^{-2}. Table 6 reports the average results obtained running 10 times the overdetermined and underdetermined

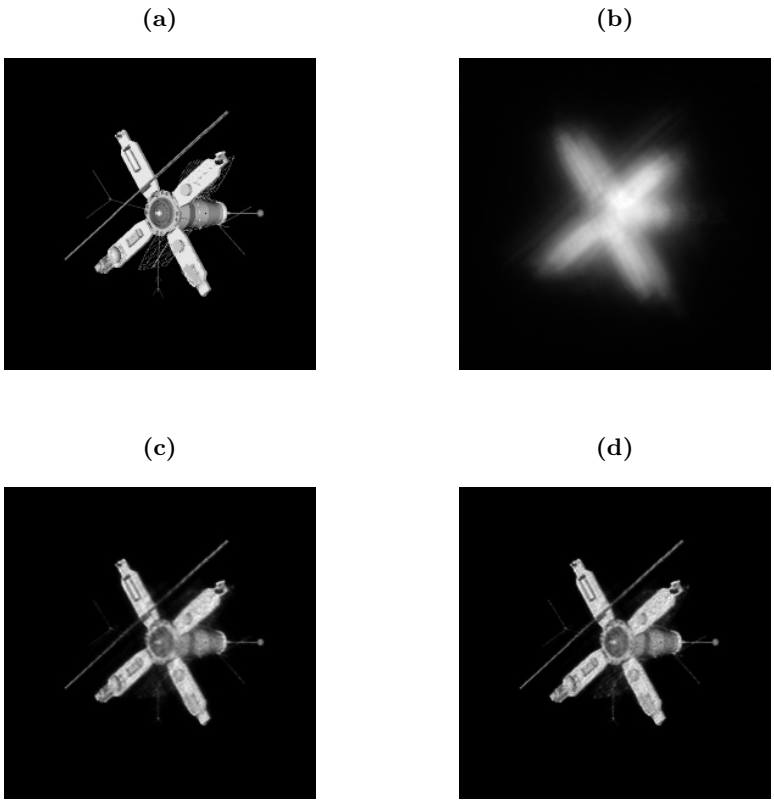

FIG. 8. *Images related to the* `satellite` *test problem, with both Gaussian and Poisson noise of level around* $\widetilde{\varepsilon} = 1.5 \cdot 10^{-2}$. *Relative errors are reported within parentheses.* ***(a)*** *Exact image.* ***(b)*** *Blurred and noisy image. Restorations obtained at the* 100*th iteration of the:* ***(c)*** *KWMRNSD method* (0.1844), *and* ***(d)*** *CP-NN-FCGLS(k) method* (0.1563).

TABLE 6
`paralleltomo` *test problem, with* $\widetilde{\varepsilon} = 5 \cdot 10^{-2}$. *Average values over* 10 *runs of the test problem, with different noise realizations.*

	rel.error	**iterations**	**tot.time**	**av.time**
overdetermined (size 81088×65536)				
NN-FCGLS	1.8268e-01	17.33	0.92	0.09
ReSt NNCG	2.0133e-01	56.00	16.31	0.11
MFISTA	2.0029e-01	37.00	53.13	1.44
MRNSD	1.8506e-01	45.00	4.10	0.09
Cimmino	1.9982e-01	100.00	33.47	0.33
underdetermined (size 32580×65536)				
NN-FCGLS	2.3145e-01	13.00	0.15	0.07
ReSt NNCG	2.4572e-01	51.00	0.59	0.05
MFISTA	2.4634e-01	32.00	12.41	0.39
MRNSD	2.3485e-01	35.00	3.43	0.09
Cimmino	2.4715e-01	94.33	8.84	0.09

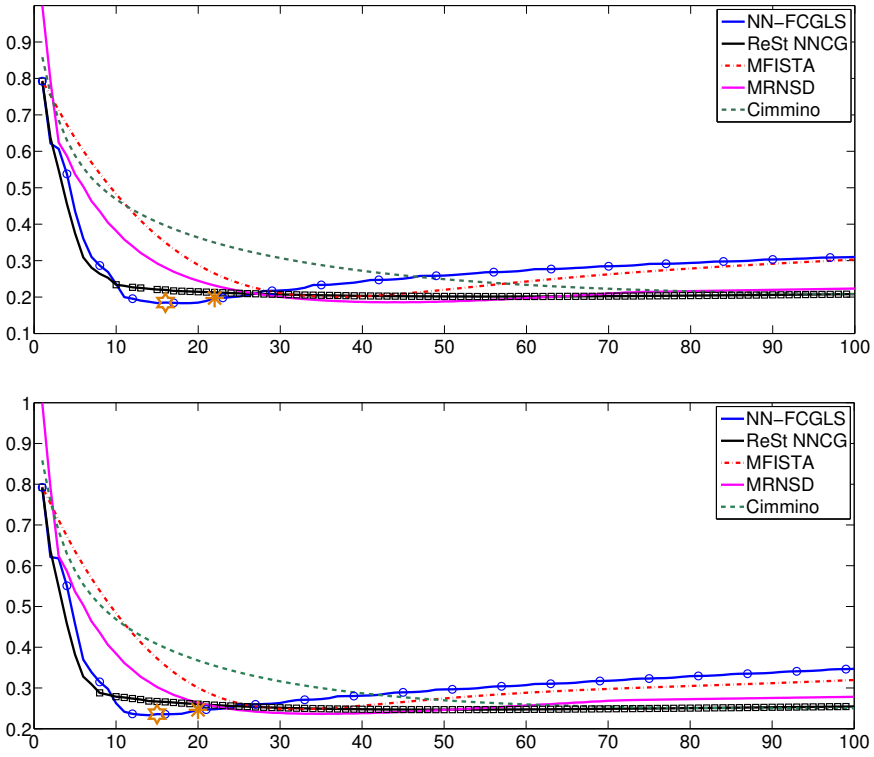

FIG. 9. `paralleltomo` *test problem, with* $\widetilde{\varepsilon} = 5 \cdot 10^{-2}$. *Upper frame: comparisons of the relative errors obtained when solving the overdetermined problem, with coefficient matrix of size* 81088×65536. *Lower frame: comparisons of the relative errors obtained when solving the underdetermined problem, with coefficient matrix of size* 32580×65536. *For the NN-FCGLS and ReSt NNCG methods, a small marker is used to emphasize the iterations where restarts happen. Two big markers highlight the iterations satisfying the first stopping criterion in (35) (asterisk) and the second stopping criterion in (35) (hexagram) for NN-FCGLS.*

problems, with different noise realizations. In addition to the methods considered in the previous examples, also the nonnegative Cimmino method as implemented in [15] is tested. All the methods are stopped after 100 (total) iterations, and a zero initial guess is used (so that, for NN-FCGLS, the identity matrix of order N is taken as $X^{(0)}$). Figure 9 displays the history of the relative errors for the overdetermined and underdetermined cases. Quite interestingly, one can see the ReSt NNCG method to perform slightly better than the NN-FCGLS method during the first iterations. Indeed, when performing the ReSt NNCG method, nonnegativity is imposed only at each restart, and no preconditioning is considered. For this reason, ReSt CGNN is initially faster than NN-FCGLS, but then it rapidly slows down. The NN-FCGLS, MFISTA and MRNSD methods are clearly Figure 10 shows the reconstructions obtained at the 15th iteration of the NN-FCGLS and MFISTA method applied to the underdetermined problem. For this test problem, NN-FCGLS is extremely efficient. Indeed, in both the overdetermined and the underdetermined cases, NN-FCGLS can deliver the best reconstructions in the least number of iterations.

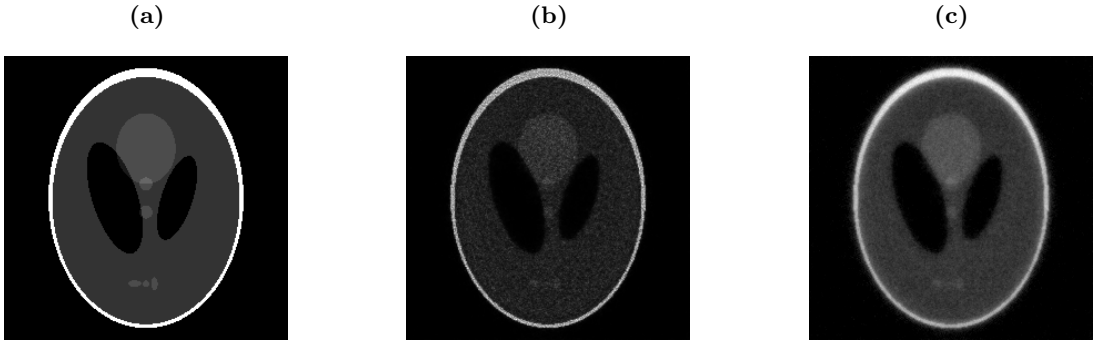

FIG. 10. *Images related to the underdetermined* `paralleltomo` *test problem, with* $\widetilde{\varepsilon} = 5 \cdot 10^{-2}$. *Relative errors are reported within parentheses.* ***(a)*** *Exact image. Restorations obtained at the* 15*th iteration of the:* ***(b)*** *NN-FCGLS method* (0.2358)*, and* ***(c)*** *MFISTA method* (0.3726).

6. Final remarks and future work. An original, efficient and promising strategy was presented to solve nonnegative linear least squares problems. The new approach is called NN-FCGLS, and it exploits flexible Krylov subspaces methods. To the best of our knowledge, NN-FCGLS is the first systematic attempt to enforce nonnegative approximations within the framework of Krylov subspace methods. The extensive numerical tests displayed in the paper show that the new method is very competitive with other state-of-the-art approaches. Indeed, NN-FCGLS provides faster and better reconstructions with respect to many methods already available in the literature. Moreover, it can be used to solve problems that are affected by both Gaussian and Poisson noise.

Although a preliminary theoretical analysis of NN-FCGLS is already provided in this paper, future work should concentrate on a deeper theoretical understanding of the regularizing properties of iterative solvers based on flexible Krylov subspaces. Handling additional constraints (e.g., box constraints, sparsity), considering Krylov methods other than CGLS, and developing parallel implementations, are interesting and challenging extensions of the present method, which could significantly impact large-scale imaging applications, in particular radio-interferometric imaging in astronomy and magnetic resonance imaging in medicine.

Acknowledgements. We wish to thank the anonymous Referees for providing many insightful remarks that considerably helped us to improve the paper. Silvia Gazzola is very grateful to James Nagy, for engaging in many stimulating discussions about nonnegatively constrained least squares problems over the last years.

REFERENCES

[1] O. AXELSSON, *Iterative Solution Methods*, Cambridge University Press, Cambridge, UK, 1994.
[2] J. M. BARDSLEY AND J. G. NAGY, *Covariance-preconditioned iterative methods for nonnegatively constrained astronomical imaging*, SIAM Journal on Matrix Analysis and Applications, 27 (2006), pp. 1184–1197.
[3] A. BECK AND M. TEBOULLE, *Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems*, IEEE Transactions on Image Processing, 18 (2009), pp. 2419–2434.
[4] A. BECK AND M. TEBOULLE, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.
[5] S. BERISHA AND J. G. NAGY, *Iterative image restoration*, in Academic Press Library in Signal Processing,

R. Chellappa and S. Theodoridis, eds., vol. 4, Elsevier, 2014, ch. 7, pp. 193–243.
[6] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, PA, 1996.
[7] A. M. Bruckstein, M. Elad, and M. Zibulevsky, *On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations*, IEEE Trans. Inform. Theory, 54 (2008), pp. 4813 – 4820.
[8] D. Calvetti, G. Landi, L. Reichel, and F. Sgallari, *Non-negativity and iterative methods for ill-posed problems*, Inverse Problems, 20 (2004), pp. 1747–1758.
[9] R. E. Carrillo, J. D. McEwen, and Y. Wiaux, *Sparsity averaging reweighted analysis (sara): a novel algorithm for radio-interferometric imaging*, Mon. Not. Roy. Astron. Soc., 426 (2012), pp. 1223–1234.
[10] S. Gazzola and J. G. Nagy, *Generalized arnoldi–tikhonov method for sparse reconstruction*, SIAM Journal on Scientific Computing, 36 (2014), pp. B225–B247.
[11] S. Gazzola, P. Novati, and M. R. Russo, *On Krylov projection methods and Tikhonov regularization*, Electron. Trans. Numer. Anal., 44 (2015), pp. 83–123.
[12] G. H. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224.
[13] M. Hanke, J. G. Nagy, and C. Vogel, *Quasi-newton approach to nonnegative image restorations*, Linear Algebra and its applications, 316 (2000), pp. 223–236.
[14] P. C. Hansen, *Discrete Inverse Problems: Insight and Algorithms*, SIAM, Philadelphia, PA, 2010.
[15] P. C. Hansen and M. Saxild-Hansen, *AIR tools — a MATLAB package of algebraic iterative reconstruction methods*, Journal of Computational and Applied Mathematics, 236 (2012), pp. 2167 – 2178.
See also: http://www2.compute.dtu.dk/~pcha/AIRtools/.
[16] L. Kaufman, *Maximum likelihood, least squares, and penalized least squares for PET*, IEEE Trans. Med. Imag., 12 (1993), pp. 200–214.
[17] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, *Compressed sensing mri*, Signal Processing Magazine, IEEE, 25 (2008), pp. 72–82.
[18] K. Morikuni, L. Reichel, and K. Hayami, *FGMRES for linear discrete ill-posed problems*, Appl. Numer. Math., 75 (2014), pp. 175–187.
[19] J. G. Nagy, K. M. Palmer, and L. Perrone, *Iterative methods for image deblurring: a Matlab object oriented approach*, Numer. Algorithms, 36 (2004), pp. 73–93.
See also: http://www.mathcs.emory.edu/~nagy/RestoreTools.
[20] J. G. Nagy and Z. Strakos, *Enforcing nonnegativity in image reconstruction algorithms*, in International Symposium on Optical Science and Technology, International Society for Optics and Photonics, 2000, pp. 182–190.
[21] Y. Notay, *Flexible conjugate gradients*, SIAM Journal on Scientific Computing, 22 (2000), pp. 1444–1460.
[22] Y. Saad, *Iterative Methods for Sparse Linear Systems, 2nd Ed.*, SIAM, Philadelphia, PA, 2003.
[23] V. Simoncini and D. B. Szyld, *Recent computational developments in Krylov subspace methods for linear systems*, Numerical Linear Algebra with Applications, 14 (2007), pp. 1–59.