KRYLOV METHODS FOR LOW-RANK REGULARIZATION *

SILVIA GAZZOLA[†], CHANG MENG[‡], AND JAMES G. NAGY[‡]

Abstract. This paper introduces new solvers for the computation of low-rank approximate solutions to large-scale linear problems, with a particular focus on the regularization of linear inverse problems. Although Krylov methods incorporating explicit projections onto low-rank subspaces are already used for well-posed systems that arise from discretizing stochastic or time-dependent PDEs, we are mainly concerned with algorithms that solve the so-called nuclear norm regularized problem, where a suitable nuclear norm penalization on the solution is imposed alongside a fit-to-data term expressed in the 2-norm: this has the effect of implicitly enforcing low-rank solutions. By adopting an iteratively reweighted norm approach, the nuclear norm regularized problem is reformulated as a sequence of quadratic problems, which can then be efficiently solved using Krylov methods, giving rise to an inner-outer iteration scheme. Our approach differs from the other solvers available in the literature in that: (a) Kronecker product properties are exploited to define the reweighted 2-norm penalization terms; (b) efficient preconditioned Krylov methods replace gradient (projection) methods; (c) the regularization parameter can be efficiently and adaptively set along the iterations. Furthermore, we reformulate within the framework of flexible Krylov methods both the new inner-outer methods for nuclear norm regularization and some of the existing Krylov methods incorporating low-rank projections. This results in an even more computationally efficient (but heuristic) strategy, that does not rely on an inner-outer iteration scheme. Numerical experiments including image deblurring, computed tomography and inpainting show that our new solvers are competitive with other state-of-the-art solvers for low-rank problems, and deliver reconstructions of increased quality with respect to other classical Krylov methods.

Key words. low-rank solver, nuclear norm regularization, Krylov methods, flexible Krylov methods, Kronecker product, imaging problems

AMS subject classifications. 65F20, 65F30

1. Introduction.

Consider the following linear system

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \quad \text{where} \quad \boldsymbol{A} \in \mathbb{R}^{M\times N},\ \boldsymbol{x} \in \mathbb{R}^N,\ \boldsymbol{b} = \boldsymbol{b}^{\mathrm{ex}} + \boldsymbol{\eta} \in \mathbb{R}^M\,. \tag{1.1}$$

We are mainly interested in large-scale linear systems (1.1) arising from inverse problems, where $\boldsymbol{A}$ is a discretization of the linear forward operator, $\boldsymbol{x}$ is a quantity of interest, and $\boldsymbol{b}$ is the observed perturbed data ($\boldsymbol{b}^{\mathrm{ex}} = \boldsymbol{A}\boldsymbol{x}^{\mathrm{ex}}$ being the ideally exact data, and $\boldsymbol{\eta}$ being unknown Gaussian white noise). Our focus is on two-dimensional imaging problems, where the unknown vector $\boldsymbol{x} \in \mathbb{R}^N$ is obtained by stacking the columns of an unknown true image $\boldsymbol{X}$ of size $n \times n$, with $n = \sqrt{N}$ (this operation and its inverse are denoted by $\boldsymbol{x} = \mathrm{vec}(\boldsymbol{X})$ and $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$, respectively).

Discrete inverse problems are ill-posed in nature [13] and, because of the presence of noise in (1.1), regularization needs to be applied so that the solution of (1.1) is a meaningful approximation to $\boldsymbol{x}^{\mathrm{ex}}$. One typically achieves regularization by replacing the original problem (1.1) with a closely related one that is less sensitive to perturbations: effective regularization methods do so by incorporating known or desired properties of $\boldsymbol{x}$ into the solution process. In imaging applications, Tikhonov (ℓ_2) regularization, ℓ_1 regularization and total variation are typical techniques to be exploited, see, for example, [4, 7, 8, 9, 17]. In the field of geophysics, ℓ_0 regularization (also called compact regularization) is sometimes considered (e.g., [21, 34]).

[*] Submitted to the editors DATE.

Funding: This work was funded by the National Science Foundation under grant no. DMS-1819042. and partially supported by the EPSRC under grant no. EP/T001593/1.

[†] Department of Mathematical Sciences, University of Bath, United Kingdom. (s.gazzola@bath.ac.uk).

[‡] Department of Mathematics, Emory University, Atlanta, GA, USA. (chang.meng@emory.edu, jnagy@emory.edu)

In this paper we consider regularization methods that compute a low-rank approximate solution $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$ of (1.1): this is generally meaningful when the unknown $\boldsymbol{x}$ encodes a high-dimensional quantity and, in particular, in the case of a two-dimensional image. Indeed, two-dimensional images are often assumed to have low-rank or to be well-approximated by low-rank two-dimensional arrays (see [27] and the references therein).

Numerical linear algebra solvers for the estimation of low-rank solutions to linear systems have been developed in the literature, mainly targeting well-posed linear discrete problems, such as those arising when considering the numerical solution of stochastic PDEs (see [22] and the references therein). In particular, the authors of [22] devise a restarted GMRES-like method (RS-LR-GMRES) that involves low-rank projections of the basis vectors of the solution subspace, as well as a low-rank projection of the current solution at the end of each cycle. Since, in general, the basic operations involved in standard GMRES (such as matrix-vector products and vector sums) increase the ranks of the computed quantities, low-rank projections are needed to assure that the computed solution is low-rank. In the framework of compressive sensing, the authors of [2] consider a modified version of the conjugate gradient method that incorporates appropriate rank-truncation operations. All the methods mentioned so far employ, often in a heuristic way, Krylov subspace methods together with rank-reduction operations (e.g., projections onto a chosen set of low-rank matrices). Since many Krylov subspace methods are iterative regularization methods for (1.1), this brings us to the question of how incorporating rank-reduction operations would affect the solution of the discrete inverse problem (1.1), with a particular focus on imaging applications.

Low-rank matrix estimation can be naturally formulated as a nonconvex optimization problem having either: (i) a least-squares data fitting term as objective function and a rank constraint; (ii) the rank of $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$ as objective function and a constraint on the least-squares data fitting term. The last instance is commonly referred to as *affine rank minimization problem*, and both formulations are in general NP-hard [27]. In this paper we consider the unconstrained and convex optimization problem

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \|\mathrm{vec}^{-1}(\boldsymbol{x})\|_* \,, \tag{1.2}$$

where $\lambda > 0$ is a regularization parameter and $\|\cdot\|_*$ denotes the nuclear norm of $\mathrm{vec}^{-1}(\boldsymbol{x}) = \boldsymbol{X}$, defined as the sum of the singular values of $\boldsymbol{X}$. Indeed, if the singular value decomposition (SVD) of $\boldsymbol{X}$ is given by $\boldsymbol{X} = \boldsymbol{U_X}\boldsymbol{\Sigma_X}\boldsymbol{V_X}^T$, where $\boldsymbol{U_X}, \boldsymbol{V_X} \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\boldsymbol{\Sigma_X} \in \mathbb{R}^{n\times n}$ is the diagonal matrix whose diagonal entries are $\sigma_1(\boldsymbol{X}) \geqslant \cdots \geqslant \sigma_n(\boldsymbol{X}) \geqslant 0$, then

$$\|\boldsymbol{X}\|_* = \sum_{i=1}^{n} \sigma_i(\boldsymbol{X}) \,.$$

Problem (1.2) is refered to as a *nuclear norm regularized (NNR) problem.* In particular, the nuclear norm is a convex function that has been proven to be the best convex lower approximation of the rank function over the set of matrices $\boldsymbol{X}$ such that $\|\boldsymbol{X}\|_2 \leqslant 1$ (see [27] and the references therein). The nuclear norm has been used in many applications, such as low-rank matrix completion and compressed sensing; see, e.g., [3, 10, 16, 24, 27], where the constrained formulation of problem (1.2) has also been considered (note that, for a proper choice of $\lambda > 0$, constrained and unconstrained formulations are equivalent; see, e.g., [29]). In the framework of compressive sensing, under the assumption that the matrix $\boldsymbol{A}$ satisfies a certain null-space property, recovery guarantees for the affine rank minimization problem are proven in [5, 25]. We also consider the following formulation

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\lambda\|\mathrm{vec}^{-1}(\boldsymbol{x})\|_{*,p}\,, \quad \text{where} \quad \|\boldsymbol{X}\|_{*,p}=\sum_{i=1}^{n}(\sigma_i(\boldsymbol{X}))^p, \quad 0<p\leqslant 1\,. \tag{1.3}$$

Problem (1.3) is refereed to as *NNRp problem*, and it generalizes problem (1.2) (which is obtained taking $p = 1$ in (1.3)). The constrained version of (1.3) is already considered in [25], where the authors empirically show an improved recovery performance of the constrained formulation of problem (1.3) with $p < 1$ with respect to $p = 1$. Note, however, that the choice $p < 1$ in (1.3) results in a nonconvex minimization problem.

Many different optimization methods, such as singular value thresholding (i.e., projected gradient descent) and continuation methods [10], have been proposed for the solution of problem (1.2) or its constrained counterpart. In particular, the so-called IRLS(-p) (i.e., iteratively reweighted least squares) family of methods has recently attracted a lot of attention [5, 25, 27]. IRLS(-p) solves the affine rank minimization problem by solving a sequence of problems whose objective function only involves an iteratively updated weighted 2-norm term. The authors of [23] apply the IRLS(-p) framework to the unconstrained problem (1.3), requiring the solution of a sequence of sub-problems

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\lambda\|\boldsymbol{W}_k\mathrm{vec}^{-1}(\boldsymbol{x})\|_F^2\,, \tag{1.4}$$

where $\boldsymbol{W}_k$ is an appropriate weight matrix to be employed to solve the kth sub-problem, and $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. A gradient (projection) algorithm is typically used to solve each sub-problem (1.4). Since an IRLS(-p) approach is also commonly applied to objective functions involving a quadratic fit-to-data term and a general p-norm penalization on $\boldsymbol{x}$, and since efficient strategies based on Krylov methods have been devised to solve each quadratic sub-problem in the IRLS(-p) sequence [28, 30], this brings us to the question of how Krylov methods can be best employed to solve each problem (1.4) (recall that $\|\mathrm{vec}^{-1}(\boldsymbol{x})\|_{*,p}$ can be regarded as a p-norm of the vector whose entries are the singular values of $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$).

The goal of this paper is to propose new efficient Krylov methods for the estimation of low-rank solutions to (1.1). We will mainly consider an IRLS(-p) approach to problem (1.3) (rather than incorporating low-rank projections into a linear solver for (1.1)), the upside being that low-rank is implicitly enforced into the solution by penalizing the p-norm of the singular values for a suitable choice of λ. Our main contributions are the new IRN-GMRES-NNRp and IRN-LSQR-NNRp methods for (1.3), where automatic strategies for choosing a suitable λ are naturally incorporated. Here and in the following, the IRN acronym indicates an iteratively reweighted norm (rather than an iteratively reweighted least squares problem, [30]). One of the key points in deriving the new methods is expressing in matrix form the invertible linear operator mapping $\boldsymbol{x}$ to the reweighted 2-norm of the singular values of $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$: this can be achieved in a computationally affordable way by exploiting Kronecker product properties. Each iteratively reweighted quadratic sub-problem of the form (1.4) can then be expressed as a Tikhonov regularization problem in general form, which can be straightforwardly transformed into standard form. In this way, the inverse of the linear operator mapping $\boldsymbol{x}$ into the reweighted 2-norm of the singular values of $\boldsymbol{X} = \mathrm{vec}^{-1}(\boldsymbol{x})$ formally acts as a preconditioner for $\boldsymbol{A}$, and the so-called hybrid methods [26] based on the preconditioned Arnoldi (if $\boldsymbol{A}$ is square) or Golub-Kahan bidiagonalization algorithms can be used to efficiently approximate the solution of each problem of the form (1.4). Once a hybrid method is adopted, many automatic, adaptive, and efficient parameter choice strategies can be employed to choose a suitable λ; see [18] for an overview. Therefore, contrarily to many existing methods for (1.3), IRN-GMRES-NNRp and

IRN-LSQR-NNRp have the advantage of not requiring a regularization parameter (either λ or the desired rank of the solution) to be available in advance of the iterations, nor the repeated solution of (1.3) for different regularization parameters.

Although inherently efficient, both the IRN-GMRES-NNRp and IRN-LSQR-NNRp methods are inner-outer iteration schemes, where each outer iteration requires running a "preconditioned" Krylov subspace method until convergence (inner iteration) before updating the weights (and therefore the "preconditioner") in the next outer iteration. In order to avoid inner-outer iterations and with the aim of generating only one approximation subspace for the solution of (1.3), where a new "preconditioner" is incorporated as soon as a new approximate solution becomes available (i.e., at each iteration), we propose to solve (1.3) using flexible Krylov subspace methods, such as those based on the flexible Arnoldi [31] and Golub-Kahan [4] algorithms. The use of flexible Krylov methods for p-norm regularization of inverse problems was already proposed in [4, 7]; however, differently from the available solvers, our new approach involves iteratively defining both weights and transform matrices (i.e., the linear operator mapping $\mathrm{vec}^{-1}(\boldsymbol{x})$ into its singular values). Switching from IRN-GMRES-NNRp and IRN-LSQR-NNRp to their flexible counterparts (dubbed FGMRES-NNRp and FLSQR-NNRp, respectively) allows for savings in computations and, although FGMRES-NNRp and FLSQR-NNRp are purely heuristic, it leads to approximate solutions whose accuracy on many test problems is comparable to the ones of other well established solvers for (1.2). Motivated by the same idea of avoiding inner-outer iteration cycles while adaptively incorporating (low-rank) information into the approximation subspace for the solution, we also propose a flexible version of the projected and restarted Krylov subspace methods (such as RS-LR-GMRES, [22]) that were originally devised for square matrices, considering also extensions to rectangular matrices $\boldsymbol{A}$.

This paper is organized as follows. In Section 2 we review the available low-rank Krylov methods for square linear systems and, after surveying the available flexible Krylov solvers, we formulate new low-rank flexible Krylov solvers for both square and rectangular problems, where the basis vectors for the approximation subspace are truncated to low-rank. In Section 3 we derive the new iteratively reweighted methods for (1.3) as fixed-point methods, and we describe how to efficiently solve each reweighted problem of the form (1.4) using preconditioned Krylov methods: this leads to the IRN-GMRES-NNRp and IRN-LSQR-NNRp methods; their flexible counterparts (FGMRES-NNRp and FLSQR-NNRp, respectively) are also derived. Some implementation details, such as stopping criteria and regularization parameter choice strategies for the new methods, are unfolded in Section 4. Numerical results on image deblurring, computed tomography and inpainting are presented in Section 5, including comparisons between the proposed methods, low-rank projection methods, projected gradient methods, and standard Krylov subspace methods. Conclusions are drawn in Section 6.

Definitions and notations. Matching lower and upper case letters are used to denote the "vectorized" and "matricized" versions of a given quantity, respectively; e.g., $\boldsymbol{c} = \mathrm{vec}(\boldsymbol{C})$ and $\boldsymbol{C} = \mathrm{vec}^{-1}(\boldsymbol{c})$. We denote the ith entry of a vector $\boldsymbol{c}$ by $[\boldsymbol{c}]_i$, and the (i, j)th entry of a matrix $\boldsymbol{C}$ by $[\boldsymbol{C}]_{ij}$ or, using MATLAB-like notations, $[\boldsymbol{c}]_i = \boldsymbol{c}(i)$, $[\boldsymbol{C}]_{ij} = \boldsymbol{C}(i, j)$. Using again MATLAB-like notations, $\boldsymbol{d} = \mathrm{diag}(\boldsymbol{C})$ defines a vector $\boldsymbol{d}$ whose entries are the diagonal elements of a matrix $\boldsymbol{C}$. $\mathrm{Tr}(\boldsymbol{C})$ denotes the trace of a matrix $\boldsymbol{C}$. $\mathcal{R}(\boldsymbol{C})$ denotes the range (or column space) of a matrix $\boldsymbol{C}$, and $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b})$ denotes the m-dimensional Krylov subspace defined by $\boldsymbol{A}$ and $\boldsymbol{b}$, i.e., $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b}) = \mathrm{span}\left\{\boldsymbol{b}, \boldsymbol{A}\boldsymbol{b}, \boldsymbol{A}^2\boldsymbol{b}, \ldots, \boldsymbol{A}^{m-1}\boldsymbol{b}\right\}$. We denote by $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ the identity matrix of order d, and by $\boldsymbol{e}_i$ the ith canonical basis vector of $\mathbb{R}^d$, where d should be clear from the context. Note that, in the following, we will quite often interchange $\boldsymbol{x}$ and $\boldsymbol{X}$ and, with a slight abuse of notations, we will denote the action of a linear operator on $\boldsymbol{x}$ or $\boldsymbol{X}$ by $\mathcal{A}(\boldsymbol{X}) = \mathcal{A}\boldsymbol{X} = A\boldsymbol{x}$, and the action of the adjoint operator by $\mathcal{A}^*(\boldsymbol{Y}) = \mathcal{A}^*\boldsymbol{Y} = A^T\text{vec}(\boldsymbol{Y})$.

2. Low-rank projection methods: classical and new approaches.

As recalled in Section 1, when solving square well-posed linear systems coming from the discretization of some instances of stochastic or time-dependent PDEs, a suitable rearrangement of the solution is expected to be low-rank: for this reason, schemes that incorporate low-rank projections within the basis vectors and the approximate solution obtained by a Krylov method have been proposed in the literature. In the following we summarize the working ideas underlying the so-called restarted low-rank-projected GMRES (RS-LR-GMRES) method proposed in [22].

The starting points for the derivation of RS-LR-GMRES are the basic properties and relations underlying GMRES. Indeed, one can define GMRES for the solution of (1.1) with a square $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ and initial guess $\boldsymbol{x}_0 = \boldsymbol{0}$ by generating a matrix $\boldsymbol{V}_m = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_m] \in \mathbb{R}^{N\times m}$ with orthonormal columns, such that $\mathcal{R}(\boldsymbol{V}_m) = \mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b})$, and imposing that the residual $\boldsymbol{r}_m = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m$ is orthogonal to $\boldsymbol{U}_m = \boldsymbol{A}\boldsymbol{V}_m$. In practice, at the kth iteration of GMRES, one computes

$$\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{v}_{k-1} \quad \text{and} \quad \|\boldsymbol{v}_k\|_2\boldsymbol{v}_k = (\boldsymbol{I} - \boldsymbol{V}_{k-1}(\underbrace{\boldsymbol{V}_{k-1}^T\boldsymbol{V}_{k-1}}_{=\boldsymbol{I}})^{-1}\boldsymbol{V}_{k-1}^T)\boldsymbol{u}_k\,, \tag{2.1}$$

and the approximate solution is computed as

$$\boldsymbol{x}_k = \boldsymbol{V}_k\boldsymbol{y}_k\,, \quad \text{where} \quad (\boldsymbol{U}_k^T\boldsymbol{A}\boldsymbol{V}_k)\boldsymbol{y}_k = \boldsymbol{U}_k^T\boldsymbol{b}\,. \tag{2.2}$$

This procedure is mathematically equivalent to the somewhat more standard procedure that, at the kth iteration of GMRES, updates the partial Arnoldi factorization and computes the approximate solution as follows:

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_{k+1}\boldsymbol{H}_k\,, \quad \boldsymbol{x}_k = \boldsymbol{V}_k\boldsymbol{y}_k\,, \text{ where } \boldsymbol{y}_k = \arg\min_{\boldsymbol{y}\in\mathbb{R}^k} \|\boldsymbol{H}_k\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\|_2\,. \tag{2.3}$$

Note that, in particular, the matrix $\boldsymbol{V}_k$ appearing in (2.2) coincides with the matrix $\boldsymbol{V}_k$ appearing in (2.3). However, since matrix-vector products and vector sums of low-rank vectorized matrices increase the rank of the latter, relations (2.1) and (2.2) obviously do not guarantee that the new basis vectors $\boldsymbol{v}_k$ for the solution nor the new solution $\boldsymbol{x}_k$ are low-rank. To force the basis vector for the solution and the approximate solution to be low-rank, a truncation operator should be incorporated into the GMRES algorithm. Given a vectorized matrix $\boldsymbol{c} = \text{vec}(\boldsymbol{C})$, and given a desired low-rank κ for $\boldsymbol{C}$, one can define a truncation operator $\tau_\kappa(\boldsymbol{c})$ by the following standard operations:

$$\left[\begin{array}{ll} 1. & \text{Take } \boldsymbol{C} = \text{vec}^{-1}(\boldsymbol{c}); \\ 2. & \text{Compute the SVD of } \boldsymbol{C},\ \boldsymbol{C} = \boldsymbol{U_C}\boldsymbol{\Sigma_C}\boldsymbol{V_C}^T; \\ 3. & \text{Compute } \boldsymbol{C}_\kappa = \boldsymbol{U_C}(:, 1:\kappa)\boldsymbol{\Sigma_C}(1:\kappa, 1:\kappa)\boldsymbol{V_C}(:, 1:\kappa)^T; \\ 4. & \text{Take } \tau_\kappa(\boldsymbol{c}) = \text{vec}(\boldsymbol{C}_\kappa). \end{array}\right. \tag{2.4}$$

RS-LR-GMRES is a restarted version of the standard GMRES method where the basis vectors for the solution are truncated at each inner iteration, and the solution itself is truncated at the beginning of each outer iteration. Note that truncating the basis vectors does not guarantee that the solution has low rank (which is the reason we still need to truncate the approximate solution). The reason for truncating the basis vectors is to keep the original solution rank from increasing drastically, since it is computed as a linear combination of basis vectors. More precisely, at the ℓth outer iteration of RS-LR-GMRES, one takes $\boldsymbol{v}_1 = \boldsymbol{r}_{\ell-1}/\|\boldsymbol{r}_{\ell-1}\|_2$, where $\boldsymbol{r}_{\ell-1} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{\ell-1}$, and, at the kth inner iteration, one computes

$$\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{v}_{k-1} \quad \text{and} \quad \|\boldsymbol{v}_k\|_2\boldsymbol{v}_k = \tau_\kappa\left((\boldsymbol{I} - \boldsymbol{V}_{k-1}(\boldsymbol{V}_{k-1}^T\boldsymbol{V}_{k-1})^{-1}\boldsymbol{V}_{k-1}^T)\boldsymbol{u}_k\right) . \tag{2.5}$$

Once m inner iterations are performed, the approximate solution at the ℓth outer iteration is computed as

$$\boldsymbol{x}_\ell = \tau_\kappa\left(\boldsymbol{x}_{\ell-1} + \boldsymbol{V}_m\boldsymbol{y}_m\right) , \quad \text{where} \quad (\boldsymbol{U}_m^T\boldsymbol{A}\boldsymbol{V}_m)\boldsymbol{y}_m = \boldsymbol{U}_m^T\boldsymbol{r}_{\ell-1} . \tag{2.6}$$

The operations in (2.5) and (2.6) heavily depend on the value κ of the truncated rank, which eventually coincides with the rank of the approximate solution. In the framework of stochastic PDEs, a suitable estimate for κ can be obtained by first performing coarse-grid computations (see [22] for details, and [19, 33] for similar approaches). Comparing (2.5) and (2.1) one can see that, as in standard GMRES, RS-LR-GMRES computes a new basis vector for the solution by applying the linear operator $\boldsymbol{A}$ to the previous basis vector $\boldsymbol{v}_{k-1}$ and orthogonalizing it against the previous basis vectors $\boldsymbol{v}_i$, $i = 1, \dots, k-1$. However, since the basis vectors are truncated to low rank, the matrix $\boldsymbol{V}_k$ does not have orthonormal columns anymore, and $\mathcal{R}(\boldsymbol{V}_m)$ is not a Krylov subspace anymore. This remark leads us to the derivation of alternative low-rank projection solvers, which can be (re)casted into the framework of flexible Krylov methods and can work with both square and rectangular systems (1.1).

Low-rank flexible GMRES (LR-FGMRES) and low-rank flexible LSQR (LR-FLSQR). Flexible Krylov methods are a class of linear solvers that can handle iteration-dependent preconditioners: they were originally introduced in [31] for FGMRES, where a preconditioner for GMRES was allowed to change from one iteration to the next (either because at each iteration the preconditioner is implicitly defined by applying an iterative linear solver, or because the preconditioner can be updated with newly-computed information; see [32] for an overview). In the framework of regularizing linear solvers, flexible Krylov methods were proposed in [4, 7, 9], where the iteration-dependent "preconditioner" was associated to an iteratively reweighted norm approach to Tikhonov-like regularized problems involving penalization terms expressed in some p-norm, $0 < p \leqslant 1$ (and, indeed, these "preconditioners" have the effect of enforcing specific regularity into the approximation subspace for the solution, rather than accelerating the convergence of the iterative solvers). Leveraging flexible Krylov subspaces in this setting comes with the upside of avoiding restarts of the iterative solver, which is the approach commonly used when adopting an iteratively reweighted norm method. When considering low-rank projections of the basis vectors within RS-LR-GMRES, we enforce the basis vectors to have low-rank, so to better reproduce available information about the solution of (1.1) (i.e., the solution should be low-rank). It is therefore natural to consider flexible Krylov methods that involve truncation of the basis vectors at each iteration, as a computationally cheaper alternative to RS-LR-GMRES that does not involve restarts.

Considering first the case of a square $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, we can use the flexible Arnoldi algorithm [31] to naturally incorporate low-rank basis vectors for the solution of (1.1). In general, starting with $\boldsymbol{x}_0 = \boldsymbol{0}$, at the kth iteration, FGMRES updates a partial flexible Arnoldi factorization and computes the kth approximate solution as follows:

$$\boldsymbol{A}\boldsymbol{Z}_k = \boldsymbol{V}_{k+1}\boldsymbol{H}_k\,, \quad \boldsymbol{x}_k = \boldsymbol{Z}_k\boldsymbol{y}_k\,, \text{ where } \boldsymbol{y}_k = \arg\min_{\boldsymbol{y}\in\mathbb{R}^k}\|\boldsymbol{H}_k\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\|_2\,, \tag{2.7}$$

where $\boldsymbol{V}_{k+1} = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_{k+1}] \in \mathbb{R}^{N\times(k+1)}$ has orthonormal columns, $\boldsymbol{H}_k \in \mathbb{R}^{(k+1)\times k}$ is upper Hessenberg, and $\boldsymbol{Z}_k = [\boldsymbol{P}_1\boldsymbol{v}_1, \dots, \boldsymbol{P}_k\boldsymbol{v}_k] \in \mathbb{R}^{N\times k}$ has columns that span the approximation subspace for the solution ($\boldsymbol{P}_i$ is an iteration-dependent preconditioner that is applied to $\boldsymbol{v}_i$ and, in the particular case of low-rank truncation, $\boldsymbol{P}_i\boldsymbol{v}_i = \tau_{\kappa_B}(\boldsymbol{v}_i)$, is the truncation operator defined in (2.4), so that $\text{rank}(\text{vec}^{-1}(\boldsymbol{Z}_k\boldsymbol{e}_i)) = \kappa_B,\ i = 1,\dots,k$). The subscript B for the truncation rank κ_B suggests that the truncation is done on the original basis vectors $\boldsymbol{v}_i$'s. The resulting algorithm is dubbed "LR-FGMRES", and it is summarized in Algorithm 2.1. Note that the approximate solution computed as in (2.7) is also truncated to guarantee rank κ (in general, we assume $\kappa_B \neq \kappa$). Also LR-FGMRES is started with $\boldsymbol{x}_0 = \boldsymbol{0}$, to guarantee that the basis vectors for the solution (rather than a correction thereof) are low-rank.

Algorithm 2.1 LR-FGMRES

```
Inputs: $\boldsymbol{A}$, $\boldsymbol{b}$, $\tau_{\kappa_B}$, $\tau_\kappa$
Take $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$
for $i = 1, 2, \dots$ until a stopping criterion is satisfied do
    Compute $\boldsymbol{z}_i = \tau_{\kappa_B}(\boldsymbol{v}_i)$ and $\boldsymbol{w} = A\boldsymbol{z}_i$
    Compute $h_{ji} = \boldsymbol{w}^T\boldsymbol{v}_j$ for $j = 1,\dots,i$ and set $\boldsymbol{w} = \boldsymbol{w} - \sum_{j=1}^{i} h_{ji}\boldsymbol{v}_j$
    Compute $h_{i+1,i} = \|\boldsymbol{w}\|_2$, and if $h_{j+1,j} \neq 0$, take $\boldsymbol{v}_{i+1} = \boldsymbol{w}/h_{i+1,i}$
end for
Compute $\boldsymbol{y}_k = \arg\min_{\boldsymbol{y}} \|\boldsymbol{H}_k\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\|_2^2$ and take $\boldsymbol{x}_k = \tau_\kappa(\boldsymbol{Z}_k\boldsymbol{y}_k)$
```

A few remarks are in order. Differently from the kth iteration in the inner cycle of the RS-LR-GMRES method (2.5), the kth iteration of LR-FGMRES expands the approximation subspace by modifying (i.e., truncating) the previous orthonormal basis vector for the space $\mathcal{R}([\boldsymbol{b}, \boldsymbol{A}\boldsymbol{Z}_k])$. Analogously to RS-LR-GMRES, the basis vectors for the approximate LR-FGMRES solution are all of rank κ_B, are not orthogonal, and do not span a Krylov subspace. Differently from RS-LR-GMRES, the basis vector for the space $\mathcal{R}([\boldsymbol{b}, \boldsymbol{A}\boldsymbol{Z}_k])$ are orthogonal. Also, the kth LR-FGMRES approximate solution is obtained by solving an order-k projected least squares problem that is formally analogous to the GMRES one (see (2.3) and (2.7)).

With LR-FGMRES in place, the extension to more general matrices $\boldsymbol{A} \in \mathbb{R}^{M\times N}$, with M not necessarily equal to N, can be naturally devised considering the flexible Golub-Kahan (FGK) process [4]. Taking $\boldsymbol{x}_0 = \boldsymbol{0}$ as initial guess, the kth FGK iteration updates partial factorizations of the form

$$\boldsymbol{A}\boldsymbol{Z}_k = \boldsymbol{U}_{k+1}\boldsymbol{M}_k \quad \text{and} \quad \boldsymbol{A}^T\boldsymbol{U}_{k+1} = \boldsymbol{V}_{k+1}\boldsymbol{T}_{k+1}, \tag{2.8}$$

where the columns of $\boldsymbol{U}_{k+1} \in \mathbb{R}^{M\times(k+1)}$, $\boldsymbol{V}_{k+1} \in \mathbb{R}^{N\times(k+1)}$ are orthonormal, $\boldsymbol{M}_k \in \mathbb{R}^{(k+1)\times k}$ is upper Hessenberg, $\boldsymbol{T}_{k+1} \in \mathbb{R}^{(k+1)\times(k+1)}$ is upper triangular, and $\boldsymbol{Z}_k = [\boldsymbol{P}_1\boldsymbol{v}_1, \dots, \boldsymbol{P}_k\boldsymbol{v}_k] \in \mathbb{R}^{N\times k}$ has columns that span the approximation subspace for the solution ($\boldsymbol{P}_i$ is an iteration-dependent preconditioner that is applied to $\boldsymbol{v}_i$ and, in the particular case of low-rank truncation, $\boldsymbol{P}_i\boldsymbol{v}_i = \tau_{\kappa_B}(\boldsymbol{v}_i)$, as defined in (2.4), so that $\text{rank}(\text{vec}^{-1}(\boldsymbol{Z}_k\boldsymbol{e}_i)) = \kappa_B,\ i = 1,\dots,k$). The flexible LSQR method (FLSQR) uses the FGK process (2.8) to generate iterates of the form $\boldsymbol{x}_k = Z_k\boldsymbol{y}_k$, where the vector $\boldsymbol{y}_k$ is computed as $\boldsymbol{y}_k = \arg\min_{\boldsymbol{y}} \left\|\boldsymbol{M}_k\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\right\|_2^2$. When rank-truncation of the basis vectors takes place at each iteration, and the final approximate solution is rank-truncated as well, the resulting algorithm is dubbed "LR-FLSQR", and it is summarized in Algorithm 2.2.

Note that, similarly to RS-LR-GMRES, both LR-FGMRES and LR-FLSQR are quite heuristic. Although the low-rank projection idea can be formulated in the flexible framework, we lack a formal

Algorithm 2.2 LR-FLSQR

```
Inputs: $\boldsymbol{A}$, $\boldsymbol{b}$, $\tau_{\kappa_B}$, $\tau_\kappa$
Take $\boldsymbol{u}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$
for $i = 1, 2, \dots,$ until a stopping criterion is satisfied do
    Compute $\boldsymbol{w} = \boldsymbol{A}^T\boldsymbol{u}_i$, $t_{ji} = \boldsymbol{w}^T\boldsymbol{v}_j$ for $j = 1, \dots, i-1$
    Set $\boldsymbol{w} = \boldsymbol{w} - \sum_{j=1}^{i-1} t_{ji}\boldsymbol{v}_j$, compute $t_{ii} = \|\boldsymbol{w}\|$ and take $\boldsymbol{v}_i = \boldsymbol{w}/t_{ii}$
    Compute $\boldsymbol{z}_i = \tau_{\kappa_B}(\boldsymbol{v}_i)$ and $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{z}_i$
    Compute $m_{ji} = \boldsymbol{w}^T\boldsymbol{u}_j$ for $j = 1, \dots, i$ and set $\boldsymbol{w} = \boldsymbol{w} - \sum_{j=1}^{i} m_{ji}\boldsymbol{u}_j$
    Compute $m_{i+1,i} = \|\boldsymbol{w}\|$ and take $\boldsymbol{u}_{i+1} = \boldsymbol{w}/m_{i+1,i}$
end for
Compute $\boldsymbol{y}_k = \arg\min_{\boldsymbol{y}} \|\boldsymbol{M}_k\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\|_2^2$ and take $\boldsymbol{x}_k = \tau_\kappa(\boldsymbol{Z}_k\boldsymbol{y}_k)$
```

formulation of the problem that is being solved, and also a justification of why they work. Strategies for selecting κ_B and κ are not so clear either. To stabilize the behavior of LR-FGMRES as the iterations proceed, one may consider imposing additional Tikhonov regularization on the projected least-squares problem in (2.7), in a hybrid fashion; the same holds for LR-FLSQR (see Sections 3.3 and 5 for more details).

3. Proposed Method.

In this section, we first derive the IRN method for the solution of the NNRp problem (1.3). The starting point for our derivations is the approximation of the nondifferentiable nuclear norm regularizer by a smooth Schatten function (similarly to what is proposed in [25] for the affine rank minimization problem). The optimality conditions associated to the smoothed problem give rise to a nonlinear system of equations in $\boldsymbol{X}$, which is handled by a fixed-point iteration scheme. We show that each iteration amounts to the solution of a Tikhonov-regularized problem involving an iteratively reweighted 2-norm regularization term, which can be efficiently solved employing "preconditioned" Krylov methods. Flexible Krylov methods are introduced to approximate the solution of the IRN problem within only one adaptively defined approximation subspace for the solution, bypassing the inner-outer iteration scheme required by standard Krylov methods.

3.1. Derivation.

Define the smooth Schatten-p function as

$$\mathcal{S}_p^\gamma(\boldsymbol{X}) = \mathrm{Tr}((\boldsymbol{X}^T\boldsymbol{X} + \gamma\boldsymbol{I})^{p/2}), \quad \text{with } \gamma > 0\,.$$

Note that $\mathcal{S}_p^\gamma(\boldsymbol{X})$ is differentiable for $p > 0$ and convex for $p \geqslant 1$. In particular, for $p = 1$ and $\gamma = 0$ (i.e., no smoothing),

$$\mathcal{S}_1^0(\boldsymbol{X}) = \mathrm{Tr}((\boldsymbol{X}^T\boldsymbol{X})^{1/2}) = \|\boldsymbol{X}\|_*\,.$$

We start by considering the following smooth approximation to (1.3):

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \|\mathcal{A}(\boldsymbol{X}) - \boldsymbol{B}\|_F^2 + \lambda\mathcal{S}_p^\gamma(\boldsymbol{X})\,. \tag{3.1}$$

The following derivations are valid for $p > 0$ (and we keep them generic, being aware that $p = 1$ approximates (1.2)). The optimality conditions associated to (3.1) read

$$\begin{aligned} 0 &= \nabla_{\boldsymbol{X}}\left(\|\mathcal{A}(\boldsymbol{X}) - \boldsymbol{B}\|_F^2 + \lambda\mathcal{S}_p^\gamma(\boldsymbol{X})\right) \\ &= 2\mathcal{A}^*(\mathcal{A}(\boldsymbol{X}) - \boldsymbol{B}) + \lambda\, p(\boldsymbol{X}\boldsymbol{X}^T + \gamma\boldsymbol{I})^{p/2-1}\boldsymbol{X}\,, \end{aligned} \tag{3.2}$$

where we have used that

$$\nabla_{\boldsymbol{X}} \operatorname{Tr}((\boldsymbol{X}^T\boldsymbol{X}+\gamma\boldsymbol{I})^{p/2}) = p\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X}+\gamma\boldsymbol{I})^{p/2-1} = p(\boldsymbol{X}\boldsymbol{X}^T+\gamma\boldsymbol{I})^{p/2-1}\boldsymbol{X}\,.$$

Equivalently, the nonlinear system of equations (3.2) with respect to $\boldsymbol{X}$ can be expressed as

$$\begin{aligned}\boldsymbol{X} &= \left(\mathcal{A}^*\mathcal{A}+\widehat{\lambda}(\boldsymbol{X}\boldsymbol{X}^T+\gamma\boldsymbol{I})^{p/2-1}\right)^{-1}\mathcal{A}^*\boldsymbol{B}\\ &= \left(\mathcal{A}^*\mathcal{A}+\widehat{\lambda}((\boldsymbol{X}\boldsymbol{X}^T+\gamma\boldsymbol{I})^{p/4-1/2})^T(\boldsymbol{X}\boldsymbol{X}^T+\gamma\boldsymbol{I})^{p/4-1/2}\right)^{-1}\mathcal{A}^*\boldsymbol{B}\,,\quad \text{with } \widehat{\lambda}=\lambda\, p/2\,,\end{aligned}$$

which is naturally associated to the following fixed-point iteration scheme

$$\boldsymbol{X}_{k+1} = \left(\mathcal{A}^*\mathcal{A}+\widehat{\lambda}((\boldsymbol{X}_k\boldsymbol{X}_k^T+\gamma\boldsymbol{I})^{p/4-1/2})^T(\boldsymbol{X}_k\boldsymbol{X}_k^T+\gamma\boldsymbol{I})^{p/4-1/2}\right)^{-1}\mathcal{A}^*\boldsymbol{B}\,, \tag{3.3}$$

which leads to the solution of (3.1). Equivalently,

$$\boldsymbol{X}_{k+1} = \arg\min_{\boldsymbol{X}}\left\|\begin{bmatrix}\mathcal{A}\\ \sqrt{\widehat{\lambda}}(\boldsymbol{X}_k\boldsymbol{X}_k^T+\gamma\boldsymbol{I})^{p/4-1/2}\end{bmatrix}\boldsymbol{X}-\begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{0}\end{bmatrix}\right\|_F^2,$$

i.e., (3.3) are the normal equations associated to the penalized least squares problem written above or, equivalently,

$$\boldsymbol{X}_{k+1} = \arg\min_{\boldsymbol{X}}\|\mathcal{A}\boldsymbol{X}-\boldsymbol{B}\|_F^2+\widehat{\lambda}\left\|(\boldsymbol{X}_k\boldsymbol{X}_k^T+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{X}\right\|_F^2\,. \tag{3.4}$$

We now reformulate problem (3.4) in vectorial form.

Let $\boldsymbol{U}_{\boldsymbol{X}_k}\boldsymbol{\Sigma}_{\boldsymbol{X}_k}\boldsymbol{V}_{\boldsymbol{X}_k}^T = \boldsymbol{X}_k$ be the SVD of $\boldsymbol{X}_k$; thanks to the invariance of the Frobenius norm under orthogonal transformations, the regularization term in the above problem can be rewritten as

$$\left\|(\boldsymbol{X}_k\boldsymbol{X}_k^T+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{X}\right\|_F^2=\left\|\boldsymbol{U}_{\boldsymbol{X}_k}(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{U}_{\boldsymbol{X}_k}^T\boldsymbol{X}\right\|_F^2=\left\|(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{U}_{\boldsymbol{X}_k}^T\boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}_k}\right\|_F^2.$$

Using well-known Kronecker product properties

$$\begin{aligned}\left\|(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{U}_{\boldsymbol{X}_k}^T\boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}_k}\right\|_F^2 &= \left\|\operatorname{vec}\left((\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{U}_{\boldsymbol{X}_k}^T\boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}_k}\right)\right\|_2^2\\ = \left\|\left(\boldsymbol{V}_{\boldsymbol{X}_k}^T\otimes\left((\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\boldsymbol{U}_{\boldsymbol{X}_k}^T\right)\right)\boldsymbol{x}\right\|_2^2 &= \left\|\left(\boldsymbol{I}\otimes(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\right)\left(\boldsymbol{V}_{\boldsymbol{X}_k}^T\otimes\boldsymbol{U}_{\boldsymbol{X}_k}^T\right)\boldsymbol{x}\right\|_2^2.\end{aligned}$$

Problem (3.4) is therefore equivalent to

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\widehat{\lambda}\|\underbrace{\left(\boldsymbol{I}\otimes(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2+\gamma\boldsymbol{I})^{p/4-1/2}\right)}_{=:(\boldsymbol{W}_p^\gamma)_k}\overbrace{\left(\boldsymbol{V}_{\boldsymbol{X}_k}^T\otimes\boldsymbol{U}_{\boldsymbol{X}_k}^T\right)}^{=:\boldsymbol{S}_k}\boldsymbol{x}\|_2^2\,. \tag{3.5}$$

In the above formulation, $(\boldsymbol{W}_p^\gamma)_k$ is a diagonal weighting matrix and $\boldsymbol{S}_k$ is an orthogonal matrix; both $(\boldsymbol{W}_p^\gamma)_k$ and $\boldsymbol{S}_k$ depend on the current approximation $\boldsymbol{x}_k$ of the solution $\boldsymbol{x}$. Intuitively, the matrix $\boldsymbol{S}_k$ maps $\boldsymbol{x}$ into the "singular value domain" of $\boldsymbol{X}_k$ (and acts as an iteration-dependent sparsity transform), and the matrix $(\boldsymbol{W}_p^\gamma)_k$ assigns suitable weights that allow to approximate a p-norm of the singular values. Therefore, the penalization term in (3.5) can be interpreted as a reweighted vectorial 2-norm, with respect to a transformation of the solution $\boldsymbol{x}$. For this reason, the proposed approach is dubbed "IRN-NNRp" and is summarized in Algorithm 3.1.

Algorithm 3.1 IRN-NNRp

```
Inputs: $\boldsymbol{A}$, $\boldsymbol{b}$, $(\boldsymbol{W}_p^\gamma)_0 = \boldsymbol{I}$, $\boldsymbol{S}_0 = \boldsymbol{I}$
for $k = 0, 1, \dots$ until a stopping criterion is satisfied do
    Solve problem (3.5)
    "Decrease" $\gamma$
    Update $(\boldsymbol{W}_p^\gamma)_{k+1}$ and $\boldsymbol{S}_{k+1}$
end for
```

The next subsection derives new strategies for the efficient solution of the sequence of subproblems (3.5) appearing in Algorithm 3.1.

3.2. Solution of problem (3.5) via Krylov methods.

First rewrite problem (3.5) using an appropriate change of variable as

$$\widehat{\boldsymbol{x}}_{k+1} = \arg\min_{\widehat{\boldsymbol{x}}} \|\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\widehat{\boldsymbol{x}} - \boldsymbol{b}\|_2^2 + \widehat{\lambda}\|\widehat{\boldsymbol{x}}\|_2^2, \text{ with } \widehat{\boldsymbol{x}} = (\boldsymbol{W}_p^\gamma)_k\boldsymbol{S}_k\boldsymbol{x}\,. \tag{3.6}$$

Note that

$$\boldsymbol{S}_k^T = \boldsymbol{S}_k^{-1} = \boldsymbol{V}_{\boldsymbol{X}_k} \otimes \boldsymbol{U}_{\boldsymbol{X}_k} \quad \text{and} \quad (\boldsymbol{W}_p^\gamma)_k^{-1} = \boldsymbol{I} \otimes (\boldsymbol{\Sigma}_{\boldsymbol{X}_k}^2 + \gamma\boldsymbol{I})^{1/2-p/4}\,, \tag{3.7}$$

so that the above transformations (inversion of an orthogonal and a diagonal matrix) are numerically affordable by exploiting properties of Kronecker products. The Tikhonov-regularized problem (3.6) in standard form is equivalent to the Tikhonov-regularized problem (3.5) in general form. Many Krylov subspace methods based on the Golub-Kahan Bidiagonalization (GKB) or Arnoldi algorithms can be employed to approximate the solution of (3.6). Moreover, if the regularization parameter $\widehat{\lambda}$ is not known a priori, many efficient strategies to set its value adaptively within the sequence of projected problems can be used (i.e., in the framework of hybrid methods; see [18, 8]). The matrices $\boldsymbol{S}_k$ and $(\boldsymbol{W}_p^\gamma)_k^{-1}$ can be formally thought of as preconditioners for the original problem (1.1), whose purpose is to enforce additional regularization into the solution subspace, rather than speeding-up the convergence of linear solvers applied to (1.1).

Methods based on the GKB algorithm. The mth step of the GKB algorithm applied to the matrix $\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}$ with starting vector $\boldsymbol{b}$ (i.e., taking $\boldsymbol{x}_0 = \boldsymbol{0}$) can be expressed by the following partial matrix factorizations

$$(\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1})\boldsymbol{V}_m = \boldsymbol{U}_{m+1}\bar{\boldsymbol{B}}_m \quad \text{and} \quad ((\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{S}_k\boldsymbol{A}^T)\boldsymbol{U}_{m+1} = \boldsymbol{V}_{m+1}\boldsymbol{B}_{m+1}^T, \tag{3.8}$$

where $\boldsymbol{U}_j \in \mathbb{R}^{M\times j}$ and $\boldsymbol{V}_j \in \mathbb{R}^{N\times j}$ (with $j = m, m+1$ and $\boldsymbol{U}_j\boldsymbol{e}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$) have orthonormal columns, and $\boldsymbol{B}_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$ is lower bidiagonal (with $\bar{\boldsymbol{B}}_m$ obtained by removing the last column of $\boldsymbol{B}_{m+1}$). The orthonormal columns of $\boldsymbol{V}_m$ are such that

$$\mathcal{R}(\boldsymbol{V}_m) = \mathcal{K}_m\left(((\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{S}_k\boldsymbol{A}^T)(\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}), ((\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{S}_k\boldsymbol{A}^T)\boldsymbol{b}\right)\,.$$

We find an approximate solution of (3.6) by imposing $\widehat{\boldsymbol{x}} \in \mathcal{R}(\boldsymbol{V}_m)$, i.e., $\widehat{\boldsymbol{x}}_m = \boldsymbol{V}_m \boldsymbol{y}_m$, where, by exploiting the first decomposition in (3.8) and the properties of the matrices appearing therein, $\boldsymbol{y}_m \in \mathbb{R}^m$ is such that

$$\boldsymbol{y}_m = \arg\min_{\boldsymbol{y}\in\mathbb{R}^m} \|\bar{\boldsymbol{B}}_m \boldsymbol{y} - \|\boldsymbol{b}\|_2 \boldsymbol{e}_1\|_2^2 + \widehat{\lambda}_m \|\boldsymbol{y}\|_2^2 \,. \tag{3.9}$$

We used the notation $\widehat{\lambda}_m$ for the regularization parameter to highlight that its value can be adaptively set within the iterations. The approximate solution to problem (3.5) is such that

$$\boldsymbol{x} = \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \widehat{\boldsymbol{x}} \in \mathcal{K}_m \left((\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-2} \boldsymbol{S}_k) \boldsymbol{A}^T \boldsymbol{A}, (\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-2} \boldsymbol{S}_k) \boldsymbol{A}^T \boldsymbol{b} \right) \,. \tag{3.10}$$

Looking at the above approximation subspace for the solution $\boldsymbol{x}$, it is evident that the "preconditioner" acts by first mapping into the "singular value domain" (by applying $\boldsymbol{S}_k$), enforcing sparsity in the singular values (by reweighting with $(\boldsymbol{W}_p^\gamma)_k^{-2}$), and eventually transforming back into the "solution domain" (by applying $\boldsymbol{S}_k^T$).

Methods based on the Arnoldi algorithm. If $\boldsymbol{A}$ is square, the mth step of the Arnoldi algorithm applied to the matrix $\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}$ with starting vector $\boldsymbol{b}$ (i.e., taking $\boldsymbol{x}_0 = \boldsymbol{0}$) can be expressed by the following partial matrix factorization

$$(\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}) \boldsymbol{V}_m = \boldsymbol{V}_{m+1} \boldsymbol{H}_m \,, \tag{3.11}$$

where $\boldsymbol{V}_j \in \mathbb{R}^{N\times j}$ (with $j = m, m+1$ and $\boldsymbol{V}_j \boldsymbol{e}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$) have orthonormal columns such that

$$\mathcal{R}(\boldsymbol{V}_m) = \mathcal{K}_m \left(\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}, \boldsymbol{b} \right) \,,$$

and $\boldsymbol{H}_m \in \mathbb{R}^{(m+1)\times m}$ is upper Hessenberg. Similarly to the GKB case, we find an approximate solution of (3.6) by imposing $\widehat{\boldsymbol{x}} \in \mathcal{R}(\boldsymbol{V}_m)$ and by solving a projected Tikhonov problem of order m. The approximate solution to problem (3.5) is such that

$$\boldsymbol{x} = \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \widehat{\boldsymbol{x}} \in \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \mathcal{K}_m \left(\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}, \boldsymbol{b} \right) \,,$$

where

$$\begin{aligned} \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \mathcal{K}_m \left(\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}, \boldsymbol{b} \right) &= \operatorname{span}\{\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \boldsymbol{b}, \dots, \left(\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \boldsymbol{A} \right)^{m-1} \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \boldsymbol{b}\} \\ &= \mathcal{K}_m \left(\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \boldsymbol{A}, \boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \boldsymbol{b} \right) . \end{aligned}$$

Contrarily to the GKB case, we immediately notice that, in this context, $\boldsymbol{x}$ does not belong to a meaningful approximation subspace. Indeed, just by looking at the first vector: $\boldsymbol{b}$ is in the image space and $(\boldsymbol{W}_p^\gamma)_k^{-1}$ is supposed to act on the singular value space of $\boldsymbol{X}_k$, so $(\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{b}$ is hard to interpret; furthermore, $\boldsymbol{S}_k^T$ is supposed to link the singular value space of $\boldsymbol{X}_k$ to the image space, so $\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{b}$ is also hard for us to interpret. Although the generated solution subspace is not meaningful for our applications, it may still have the potential to be a good subspace in other contexts. Similarly to what is proposed in [1, 4], where the Arnoldi algorithm is applied to a regularized problem that enforces sparsity in the wavelet domain, we propose to fix this issue by incorporating $\boldsymbol{S}_k$ also as an orthogonal left "preconditioner" for the original system (1.1) so that, by exploiting the invariance of the vectorial 2-norm under orthogonal transformations, problem (3.6) can be equivalently reformulated as

$$\widehat{\boldsymbol{x}}_{k+1} = \arg\min_{\widehat{\boldsymbol{x}}} \|\boldsymbol{S}_k (\boldsymbol{A}\boldsymbol{S}_k^T (\boldsymbol{W}_p^\gamma)_k^{-1} \widehat{\boldsymbol{x}} - \boldsymbol{b})\|_2^2 + \widehat{\lambda} \|\widehat{\boldsymbol{x}}\|_2^2, \text{ with } \widehat{\boldsymbol{x}} = (\boldsymbol{W}_p^\gamma)_k \boldsymbol{S}_k \boldsymbol{x} \,. \tag{3.12}$$

The (right and left) preconditioned Arnoldi algorithm applied to problem (3.12) can now be expressed by the following partial matrix factorization

$$(\boldsymbol{S}_k\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1})\boldsymbol{V}_m = \boldsymbol{V}_{m+1}\boldsymbol{H}_m\,. \tag{3.13}$$

We find an approximate solution of (3.12) by imposing $\widehat{\boldsymbol{x}} \in \mathcal{R}(\boldsymbol{V}_m) = \mathcal{K}_m(\boldsymbol{S}_k\boldsymbol{A}\boldsymbol{S}_k^T\boldsymbol{W}^{-1}, \boldsymbol{S}_k\boldsymbol{b})$, i.e., $\widehat{\boldsymbol{x}}_m = \boldsymbol{V}_m\boldsymbol{y}_m$, where, by exploiting (3.13) and the properties of the matrices appearing therein, $\boldsymbol{y}_m \in \mathbb{R}^m$ is such that

$$\boldsymbol{y}_m = \arg\min_{\boldsymbol{y}\in\mathbb{R}^m} \|\boldsymbol{H}_m\boldsymbol{y} - \|\boldsymbol{b}\|_2\boldsymbol{e}_1\|_2^2 + \widehat{\lambda}_m\|\boldsymbol{y}\|_2^2\,. \tag{3.14}$$

Hence

$$\boldsymbol{x} \in \boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\mathcal{K}_m(\boldsymbol{S}_k\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}, \boldsymbol{S}_k\boldsymbol{b}) = \mathcal{K}_m(\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{S}_k\boldsymbol{A}, \boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{S}_k\boldsymbol{b})\ , \tag{3.15}$$

which is suitable for approximating the solution. The new methods based on the GKB algorithm (for generic matrices) and Arnoldi algorithm (only if $\boldsymbol{A} \in \mathbb{R}^{N\times N}$) are dubbed "IRN-LSQR-NNRp" and "IRN-GMRES-NNRp", respectively, and are summarized in Algorithm 3.2.

Algorithm 3.2 IRN-LSQR-NNRp and IRN-GMRES-NNRp

```
Inputs: $\boldsymbol{A}$, $\boldsymbol{b}$, $(\boldsymbol{W}_p^\gamma)_0 = \boldsymbol{I}$, $\boldsymbol{S}_0 = \boldsymbol{I}$
for $k = 0, 1, \dots$ until a stopping criterion is satisfied do
    for $m = 1, 2, \dots$ until a stopping criterion is satisfied do
        Update the factorizations (3.8) and (3.13), respectively
        Solve the projected problem (3.9) and (3.14), respectively, tuning $\widehat{\lambda}_m$ if necessary
    end for
    "Decrease" $\gamma$
    Update the new $(\boldsymbol{W}_p^\gamma)_{k+1}$ and $\boldsymbol{S}_{k+1}$
end for
```

3.3. Solution through flexible Krylov subspaces. Problem (1.3) reformulated as (3.6) allows us to naturally apply the flexible Golub-Kahan (FGK) and flexible Arnoldi algorithms. Indeed, instead of updating the "preconditioners" $\boldsymbol{S}_k$ and $(\boldsymbol{W}_p^\gamma)_k$ at the kth outer iteration of the nested iteration schemes of Algorithm 3.2, we propose to consider new "preconditioners" as soon as a new approximation of the solution is available, i.e., at each iteration of a Krylov subspace solver. Therefore, at the $(i+1)$th iteration of the new solvers, the "preconditioners" $(\boldsymbol{W}_p^\gamma)_i$ and $\boldsymbol{S}_i$ are computed as in (3.7), but using the SVD of the ith approximate solution

$$\boldsymbol{X}_i = \text{vec}^{-1}(\boldsymbol{x}_i) = \boldsymbol{U}_{\boldsymbol{X}_i}\boldsymbol{\Sigma}_{\boldsymbol{X}_i}\boldsymbol{V}_{\boldsymbol{X}_i}^T, \quad \text{for } i = 1, \dots, k-1\,,$$

with $(\boldsymbol{W}_p^\gamma)_0 = \boldsymbol{I}$ and $\boldsymbol{S}_0 = \boldsymbol{I}$. In order to incorporate iteration-dependent preconditioning, the flexible versions of the Golub-Kahan and Arnoldi factorizations have to be used.

Namely, at the ith iteration, the new instance of the FGK algorithm updates partial factorizations of the form (2.8), i.e., $\boldsymbol{A}\boldsymbol{Z}_i = \boldsymbol{U}_{i+1}\boldsymbol{M}_i$ and $\boldsymbol{A}^T\boldsymbol{U}_{i+1} = \boldsymbol{V}_{i+1}\boldsymbol{T}_{i+1}$, where

$$\boldsymbol{Z}_i = [\boldsymbol{S}_0^T(\boldsymbol{W}_p^\gamma)_0^{-2}\boldsymbol{S}_0\boldsymbol{v}_1, \dots, \boldsymbol{S}_{i-1}^T(\boldsymbol{W}_p^\gamma)_{i-1}^{-2}\boldsymbol{S}_{i-1}\boldsymbol{v}_i]\,, \quad \boldsymbol{v}_1 = \boldsymbol{A}^T\boldsymbol{b}/\|\boldsymbol{A}^T\boldsymbol{b}\|_2\,.$$

Taking $\boldsymbol{x}_0 = \mathbf{0}$, the ith approximate solution is such that $\boldsymbol{x}_i = \boldsymbol{Z}_i \boldsymbol{y}_i$, where

$$\boldsymbol{y}_i = \arg\min_{\boldsymbol{y}\in\mathbb{R}^i} \|\boldsymbol{M}_i \boldsymbol{y} - \|\boldsymbol{b}\|_2 \boldsymbol{e}_1\|_2^2 + \widehat{\lambda}_i \|\boldsymbol{y}\|_2^2 \,. \tag{3.16}$$

Note that the subspace for the solution $\mathcal{R}(\boldsymbol{Z}_i)$ can be regarded as a generalization of the subspace (3.10) computed when considering preconditioned GKB within the IRN-LSQR-NNRp method. The new method is dubbed "FLSQR-NNRp", and is summarized in Algorithm 3.3.

For $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{x}_0 = \mathbf{0}$, at the ith iteration, the new instance of the flexible Arnoldi algorithm updates a partial factorization of the form (2.7), with $k = i$, and generates

$$\boldsymbol{Z}_i = [\boldsymbol{S}_0^T (\boldsymbol{W}_p^\gamma)_0^{-1} \boldsymbol{S}_0 \boldsymbol{v}_1, \dots, \boldsymbol{S}_{i-1}^T (\boldsymbol{W}_p^\gamma)_{i-1}^{-1} \boldsymbol{S}_{i-1} \boldsymbol{v}_i] \,, \quad \boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2 \,,$$

where both right and left preconditioners are used analogously to IRN-GMRES-NNRp. The ith approximate solution is such that $\boldsymbol{x}_i = \boldsymbol{Z}_i \boldsymbol{y}_i$, where

$$\boldsymbol{y}_i = \arg\min_{\boldsymbol{y}\in\mathbb{R}^i} \|\boldsymbol{H}_i \boldsymbol{y} - \|\boldsymbol{b}\|_2 \boldsymbol{e}_1\|_2^2 + \widehat{\lambda}_i \|\boldsymbol{y}\|_2^2 \,. \tag{3.17}$$

Note that the subspace for the solution $\mathcal{R}(\boldsymbol{Z}_i)$ can be regarded as a generalization of the subspace (3.15) computed when considering the preconditioned Arnoldi algorithm within the IRN-GMRES-NNRp method. The new method is dubbed "FGMRES-NNRp", and is summarized in Algorithm 3.3.

Algorithm 3.3 FLSQR-NNRp and FGMRES-NNRp

```
Inputs: $\boldsymbol{A}$, $\boldsymbol{b}$, $(\boldsymbol{W}_p^\gamma)_0 = \boldsymbol{I}$, $\boldsymbol{S}_0 = \boldsymbol{I}$
for $i = 1, 2, \dots$ until a stopping criterion is satisfied do
    Update a factorization of the form (2.8) and (2.7), respectively, to expand the space $\mathcal{R}(\boldsymbol{Z}_i)$
    Solve the projected problem (3.16) and (3.17), respectively, tuning $\widehat{\lambda}_i$ if necessary
    "Decrease" $\gamma$
    Update the new $(\boldsymbol{W}_p^\gamma)_i$ and $\boldsymbol{S}_i$, using the SVD $\boldsymbol{X}_i = \text{vec}^{-1}(\boldsymbol{x}_i) = \boldsymbol{U}_{\boldsymbol{X}_i} \boldsymbol{\Sigma}_{\boldsymbol{X}_i} \boldsymbol{V}_{\boldsymbol{X}_i}^T$.
end for
```

Note that, although the approach of Algorithm 3.3 is quite heuristic, it avoids nested iteration cycles and computes only one approximation subspace for the solution of (1.3), where low-rank penalization is adaptively incorporated. Because of this, in many situations, Algorithm 3.3 computes solutions of quality comparable to the ones computed by Algorithm 3.2, with a significant reduction in the number of iterations. We should also mention that, in the framework of affine rank minimization problems, [25] outlines an algorithm that avoids inner projected gradient iterations for the solution of each quadratic subproblem in the sequence generated within the IRN strategy.

Finally, we underline that, within the framework of flexible Krylov subspaces, the approximation subspaces $\mathcal{R}(\boldsymbol{Z}_i)$ for the ith approximate solution can be further modified, with some insight into the desired properties of the solution. Indeed, since the ith basis vector for the solution is of the form $\boldsymbol{z}_i = \boldsymbol{S}_{i-1}^T (\boldsymbol{W}_p^\gamma)_{i-1}^{-2} \boldsymbol{S}_{i-1} \boldsymbol{v}_i$ for FLSQR-NNRp, and $\boldsymbol{z}_i = \boldsymbol{S}_{i-1}^T (\boldsymbol{W}_p^\gamma)_{i-1}^{-1} \boldsymbol{S}_{i-1} \boldsymbol{v}_i$ for FGMRES-NNRp, one can consider alternative "preconditioners" $\boldsymbol{S}_{i-1}$ and $(\boldsymbol{W}_p^\gamma)_{i-1}$ that are still effective in delivering low-rank solutions. For instance, focusing on FGMRES, and given $\boldsymbol{v}_i = \boldsymbol{V}_i \boldsymbol{e}_i$, where $\boldsymbol{V}_i$ is the matrix appearing on the right-hand side of the factorization (2.8), and given the SVD of $\text{vec}^{-1}(\boldsymbol{v}_i) = \boldsymbol{U}_{\boldsymbol{V}_i} \boldsymbol{\Sigma}_{\boldsymbol{V}_i} \boldsymbol{V}_{\boldsymbol{V}_i}^T$, one can take

$$\boldsymbol{S}_{i-1} = \boldsymbol{V}_{\boldsymbol{V}_i}^T \otimes \boldsymbol{U}_{\boldsymbol{V}_i}^T \quad \text{and} \quad (\boldsymbol{W}_p^\gamma)_{i-1}^{-1} = \boldsymbol{I} \otimes (\boldsymbol{\Sigma}_{\boldsymbol{V}_i})^{1-p/2} \,, \tag{3.18}$$

and as a result,

$$\begin{aligned}
\boldsymbol{S}_{i-1}\boldsymbol{v}_i &= \mathrm{vec}(\boldsymbol{U}_{\boldsymbol{V}_i}^T \mathrm{vec}^{-1}(\boldsymbol{v}_i)\boldsymbol{V}_{\boldsymbol{V}_i}) = \mathrm{vec}(\boldsymbol{\Sigma}_{V_i}),\\
(\boldsymbol{W}_p^\gamma)_{i-1}^{-1}\boldsymbol{S}_{i-1}\boldsymbol{v}_i &= \mathrm{vec}((\boldsymbol{\Sigma}_{\boldsymbol{V}_i})^{1-p/2}\boldsymbol{\Sigma}_{\boldsymbol{V}_i}) = \mathrm{vec}((\boldsymbol{\Sigma}_{\boldsymbol{V}_i})^{2-p/2}),\\
\boldsymbol{S}_{i-1}^T(\boldsymbol{W}_p^\gamma)_{i-1}^{-1}\boldsymbol{S}_{i-1}\boldsymbol{v}_i &= \mathrm{vec}(\boldsymbol{U}_{\boldsymbol{V}_i}((\boldsymbol{\Sigma}_{\boldsymbol{V}_i})^{2-p/2})\boldsymbol{V}_{\boldsymbol{V}_i}^T) = \boldsymbol{z}_i,
\end{aligned}$$

so that the singular values of $\mathrm{vec}^{-1}(\boldsymbol{v}_i)$ are rescaled: taking $0 < p \leqslant 1$, the power of $\boldsymbol{\Sigma}_{\boldsymbol{V}_i}$, $2 - p/2$, is always larger than 1, which means that large singular values get magnified and small singular values become even smaller. In this way, the gaps between singular values are emphasized and to some extent contribute to the low rank properties of the basis vectors. Similar derivations hold for FLSQR. Hence, methods analogous to LR-FLSQR and LR-FGMRES are obtained, and are dubbed FGMRES-NNRp(v) and FLSQR-NNRp(v), respectively.

4. Implementation details. All the methods considered in this paper are iterative, and therefore at least one suitable stopping criterion should be set for the iterations. When considering hybrid formulations (like the ones in Algorithms 3.2 and 3.3), one could simultaneously set a good value for the regularization parameter $\widehat{\lambda}_j$ at the jth iteration, as well as properly stop the iterations. Strategies for achieving this are already available in the literature (see [6, 8]).

Assuming that a good estimate for the norm of the noise $\boldsymbol{\eta}$ affecting the right-hand-side of (1.1) is available, i.e., $\varepsilon \simeq \|\boldsymbol{\eta}\|_2$, one can consider the discrepancy principle and stop the iterative scheme at the first iteration j such that

$$\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_j\|_2 \leqslant \theta\varepsilon\,, \quad \text{where } \theta > 1,\ \theta \simeq 1 \text{ is a safety threshold.} \tag{4.1}$$

Applying the discrepancy principle to LR-FGMRES (Algorithm 2.1) and LR-FLSQR (Algorithm 2.2) is particularly convenient, as the norm of the residual on the left-hand side of (4.1) can be monitored using projected quantities, i.e.,

$$\big\|\|\boldsymbol{b}\|_2\boldsymbol{e}_1 - \boldsymbol{H}_j\boldsymbol{y}_j\big\|_2 \quad \text{for LR-FGMRES} \quad \text{and} \quad \big\|\|\boldsymbol{b}\|_2\boldsymbol{e}_1 - \boldsymbol{M}_j\boldsymbol{y}_j\big\|_2 \quad \text{for LR-FLSQR,}$$

where decompositions (2.7) and (2.8), respectively, and the properties of the matrices appearing therein, have been exploited. When running hybrid methods (see Algorithms 3.2 and 3.3), we employ the so-called “secant method”, which updates the regularization parameter for the projected problem in such a way that stopping by the discrepancy principle is ensured. We highlight again that the quantities needed to implement the “secant method” (namely, the norm of the residual and the discrepancy associated to (3.6) at each iteration) can be conveniently monitored using projected quantities: this is obvious for IRN-LSQR-NNRp and FLSQR-NNRp, as only right-“preconditioning” is employed; it is less obvious for IRN-GMRES-NNRp and FGMRES-NNRp, but since the left-“preconditioner” is orthogonal, one can still write

$$\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_j\|_2 = \|\boldsymbol{S}_k\boldsymbol{b} - \boldsymbol{S}_k\boldsymbol{A}\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\widehat{\boldsymbol{x}}_j\|_2 = \|\|\boldsymbol{b}\|_2\boldsymbol{e}_1 - \boldsymbol{H}_j\boldsymbol{y}_j\|_2\,.$$

Note that all the methods in Algorithm 3.2 and 3.3 can also run with $\widehat{\lambda} = 0$, and still achieve low-rank approximate solutions: this is because the approximation subspace for the solution incorporates regularizing “preconditioning” (see [12, 14] for details on this approach in the case of smoothing “preconditioning” with finite-difference approximations of derivatives operators). Finally, when dealing with the inner-outer iteration scheme of Algorithm 3.2, in addition to a parameter choice strategy and stopping criterion for the hybrid projected problems (3.9) and (3.14), one should also consider a stopping criterion for the outer iterations. We propose to do this by monitoring the norm of the difference of the singular values (normalized by the largest singular value so that $\sigma_1(\boldsymbol{\Sigma}_{\boldsymbol{X}_{k+1}}) = \sigma_1(\boldsymbol{\Sigma}_{\boldsymbol{X}_k}) = 1$) of two approximations of the solution of (1.3) obtained at two consecutive outer iterations of Algorithm 3.2, i.e., we stop as soon as

$$\|\mathrm{diag}(\boldsymbol{\Sigma}_{\boldsymbol{X}_{k+1}}) - \mathrm{diag}(\boldsymbol{\Sigma}_{\boldsymbol{X}_k})\|_2 < \tau_\sigma, \quad k = 1, 2, \ldots, \tag{4.2}$$

where $\mathrm{vec}^{-1}(\boldsymbol{x}_i) = \boldsymbol{X}_i = \boldsymbol{U}_{\boldsymbol{X}_i}\boldsymbol{\Sigma}_{\boldsymbol{X}_i}\boldsymbol{V}_{\boldsymbol{X}_i}^T$ $(i = k, k+1)$, and $\tau_\sigma > 0$ is a user-specified threshold. If no significant changes happen in the rank and singular values of two consecutive approximations of the solution, then (4.2) is satisfied.

We conclude this section with a few remarks about the computational cost of the proposed methods. Note that, if $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, IRN-GMRES-NNR$p$ is intrinsically cheaper than IRN-LSQR-NNRp (since, at each iteration, the former requires only one matrix-vector product with $\boldsymbol{A}$, while the latter requires one matrix-vector product with $\boldsymbol{A}$ and one with $\boldsymbol{A}^T$). However, methods based on the Arnoldi algorithm are typically less successful than methods based on the GKB algorithm for regularization; see [15]. Other key operations for implementing our proposed methods are the computation of the SVDs of relevant quantities, and/or the application of the "preconditioners" in (3.18). Namely, each iteration of LR-FGMRES, LR-FLSQR, FLSQR-NNRp, and FGMRES-NNRp requires the computation of the SVD of an $n \times n$ matrix, which amounts to $\mathcal{O}(n^3) = \mathcal{O}(N^{3/2})$ floating point operations. When considering IRN-LSQR-NNRp and IRN-GMRES-NNRp, only the SVD of the approximate solution should be computed once at each outer iteration. However, each inner iteration of IRN-LSQR-NNRp and IRN-GMRES-NNRp, as well as each iteration of FLSQR-NNRp and FGMRES-NNRp, requires the computation of matrix-vector products of the form $\boldsymbol{S}_k^T(\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{v}_i$: this can be achieved within a two-step process, where first the rescaling $\widetilde{\boldsymbol{v}}_i = (\boldsymbol{W}_p^\gamma)_k^{-1}\boldsymbol{v}_i$ is applied with $\mathcal{O}(N) = \mathcal{O}(n^2)$ floating-point operations, and then $\boldsymbol{S}_k^T\widetilde{\boldsymbol{v}}_i = (\boldsymbol{V}_{\boldsymbol{X}_k} \otimes \boldsymbol{U}_{\boldsymbol{X}_k})\widetilde{\boldsymbol{v}}_i$ is computed. While a straightforward implementation of the latter would require $\mathcal{O}(N^2) = \mathcal{O}(n^4)$ floating-point operations, exploiting Kronecker product properties can bring down the cost of this operation to $\mathcal{O}(n^3) = \mathcal{O}(N^{3/2})$, by computing $\boldsymbol{S}_k^T\widetilde{\boldsymbol{v}}_i = \mathrm{vec}(\boldsymbol{U}_{\boldsymbol{X}_k}^T\mathrm{vec}^{-1}(\boldsymbol{v}_i)\boldsymbol{V}_{\boldsymbol{X}_k})$. We emphasize that the incorporation of the flexible "preconditioners" does not increase the order of computational complexity and is very practical, since operations are done on matrices of size $n \times n$ (n is the dimension of the image). In particular, the full SVD's of $n \times n$ matrices can be computed easily with MATLAB's built-in `svd` function (this is what we used in our numerical experiments); one can also use Lanczos bidiagonalization [20] or randomized SVD [11] to compute the approximate leading singular values and vectors.

5. Experimental Results.

In this section, we present results of numerical experiments on several image processing problems to demonstrate the performance of the new IRN-GMRES-NNRp, IRN-LSQR-NNRp, FGMRES-NNRp, and FLSQR-NNRp methods. Variants of FGMRES-NNRp and FLSQR-NNRp (marked with "(v)") are also tested. To shorten the acronyms, we omit p when $p = 1$, which means IRN-GMRES-NNR denotes IRN-GMRES-NNRp when $p = 1$, etc. Examples are generated using *IR Tools* [6].

In general, we compare the performances of the proposed methods to standard Krylov subspace methods GMRES and LSQR, also used in a hybrid fashion. We also test against the low-rank projection methods described in Section 2 and the singular value thresholding (SVT) algorithm [3], which was originally proposed for low-rank matrix completion problems, and can be extended to problems with linear constraints of the form

$$\min_{\boldsymbol{x}} \tau\|\mathrm{vec}^{-1}(\boldsymbol{x})\|_* + \frac{1}{2}\|\mathrm{vec}^{-1}(\boldsymbol{x})\|_F^2 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \text{ where } \mathrm{vec}^{-1}(\boldsymbol{x}) = \boldsymbol{X}. \tag{5.1}$$

The kth iteration of the SVT algorithm for (5.1) reads

$$\begin{cases} \boldsymbol{X}_k = \mathcal{D}_\tau(\boldsymbol{A}^T \boldsymbol{y}_{k-1}) \\ \boldsymbol{y}_k = \boldsymbol{y}_{k-1} + \delta_k(\boldsymbol{b} - A\boldsymbol{x}_k) \end{cases}, \tag{5.2}$$

where δ_k is a step size and $\mathcal{D}_\tau$ is the *singular value shrinkage operator*, defined as

$$\mathcal{D}_\tau(\boldsymbol{X}) = \boldsymbol{U}_{\boldsymbol{X}} \mathcal{D}_\tau(\boldsymbol{\Sigma}_{\boldsymbol{X}}) \boldsymbol{V}_{\boldsymbol{X}}^T, \;\; \mathcal{D}_\tau(\boldsymbol{\Sigma}_{\boldsymbol{X}}) = \max\{\boldsymbol{\Sigma}_{\boldsymbol{X}} - \tau \boldsymbol{I}, \boldsymbol{0}\},$$

where $\boldsymbol{X} = \boldsymbol{U}_{\boldsymbol{X}} \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{V}_{\boldsymbol{X}}^T$ is the SVD of $\boldsymbol{X}$, $\boldsymbol{0}$ is a matrix of zeros, and the maximum is taken component-wise. Although (5.1) is not the same problem as (1.2), they are similar in that both penalize the nuclear norm of $\mathrm{vec}^{-1}(\boldsymbol{x})$ and they respect the constraint $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.

The Schatten-p function is introduced in Section 3.1 as a smooth approximation for $\|\cdot\|_{*,p}$. The smooth approximation allows for further derivations including computation of optimality conditions, where the "smoothing coefficient" γ is crucial. However, γ is not so crucial numerically, and we can set it to 0 without affecting the results (compared to using a very small γ). Howevrer, to be consistent with Algorithms 3.2 and 3.3, in our experiments, we have set the initial value of γ to 10^{-10}, and every time we need to decrease γ, we divide the current γ value by 2.

Regarding the comparisons with the low-rank projection methods presented in Section 2, there are no universal and theoretically informed ways of choosing the truncation ranks for the solutions and for the basis vectors of the solution subspace. Hence, for all test problems, we experiment on a reasonable number of trials, each with different truncation rank choices, and select the best performing rank out of all ranks tested. For simplicity, we consider the same truncation rank for basis vectors and solutions ($\tau_{\kappa_B} = \tau_\kappa$). We follow the same process to choose the number of restarts and the number of iterations for each restart for RS-LR-GMRES, as well as the shrinkage threshold τ in SVT; strategies to select the step size for SVT are described in [3].

Example 1: Binary Star. We consider an image deblurring problem involving a binary star test image of size 256 × 256: this test image has rank 2. The true image is displayed in the leftmost frame of Figure 2. A standard Gaussian blur is applied to the test image, and Gaussian white noise of level $\|\boldsymbol{\eta}\|_2/\|\boldsymbol{b}^{\mathrm{ex}}\|_2 = 10^{-3}$ is added. The blurred and noisy image is shown in Figure 2, second frame from the left. Due to the presence of noise, the blurred image has full rank. For this example, the blurring operator $\boldsymbol{A}$ is square of size 65536 × 65536, hence GMRES-related methods are used for comparison, namely: GMRES, IRN-GMRES-NNR, FGMRES-NNR, LR-FGMRES and RS-LR-GMRES (i.e., we only consider the case $p = 1$ here). SVT is also taken into consideration. The truncation rank for LR-FGMRES and RS-LR-GMRES is set to 30 for both basis vectors and approximate solutions (i.e., $\tau_{\kappa_B} = \tau_\kappa = 30$). RS-LR-GMRES is restarted every 40 iterations. The step size for SVT is set to be $\delta_k = \delta = 2$ and the singular value shrinkage threshold τ is 1. Note that, although the true solution has only rank 2, setting truncation rank to 2 for low rank methods produces solutions of worse quality (compared to setting the rank to 30). This might be because of the inherent ill-posedness of the problem, which makes it harder to obtain solutions with desired properties (e.g., with rank 2): indeed, if we do truncate to rank 2, a lot of information about the solution might be lost.

Figure 1 displays the histories of relative errors $\|\boldsymbol{x}^{\mathrm{ex}} - \boldsymbol{x}_m\|_2/\|\boldsymbol{x}^{\mathrm{ex}}\|_2$ for the first 200 iterations (i.e., $m = 1, \dots, 200$) of these methods. For IRN-GMRES-NNR, 4 outer cycles were run, each with a maximum of 50 iterations: a new outer cycle is initiated as soon as the discrepancy principle is satisfied in the inner cycle. No additional regularization is used (i.e., $\widehat{\lambda} = 0$ for all methods).

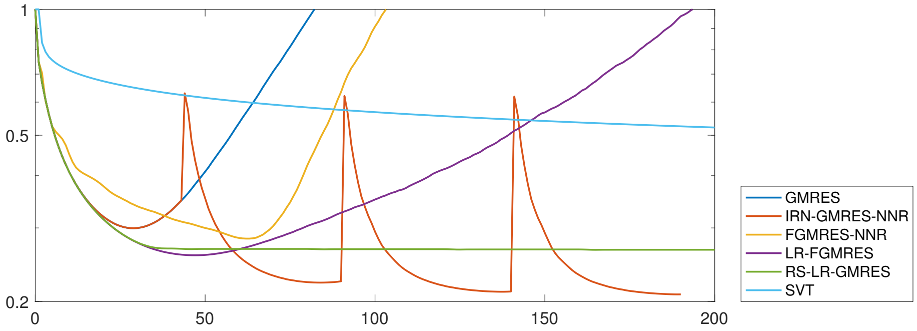


Fig. 1: *Example 1.* Relative errors vs. number of iterations for GMRES-based methods and SVT.

We can observe from Figure 1 that when the truncation ranks are chosen reasonably, LR-FGMRES and RS-LR-GMRES both produce a less pronounced semi-convergence behavior than GMRES, with LR-FGMRES attaining a smaller relative error than RS-LR-GMRES. FGMRES-NNR, on the other hand, shows slower semi-convergence than GMRES, but it also converges to a slightly better relative error. IRN-GMRES-NNR behaves especially well in this case, with significantly reduced relative errors even at the end of the second outer cycle. The "jumps" at the beginning of each outer IRN-GMRES-NNR iteration are due to the strategy used for restarts (the older basis vectors are cleared at each restart).

Figure 2 displays the exact and the corrupted images, as well as the best reconstructions computed by LR-FGMRES and IRN-GMRES-NNR: these are obtained at the 47th and the 189th (total) iteration of LR-FGMRES and IRN-GMRES-NNR, respectively. By looking at relative errors in Figure 1, we see that LR-FGMRES is the second best out of all methods, and yet the quality of the solution is inferior compared to IRN-GMRES-NNR. Compared to the LR-FGMRES solution, the IRN-GMRES-NNR one is a more truthful reconstruction of the exact image: it not only has less artifacts immediately around the stars, but also has less background noise, in the sense that the pixel intensities in the background are closer to the true ones (as it can be seen by looking at the background color). More details can be spotted if we zoom into the central part (51 × 51

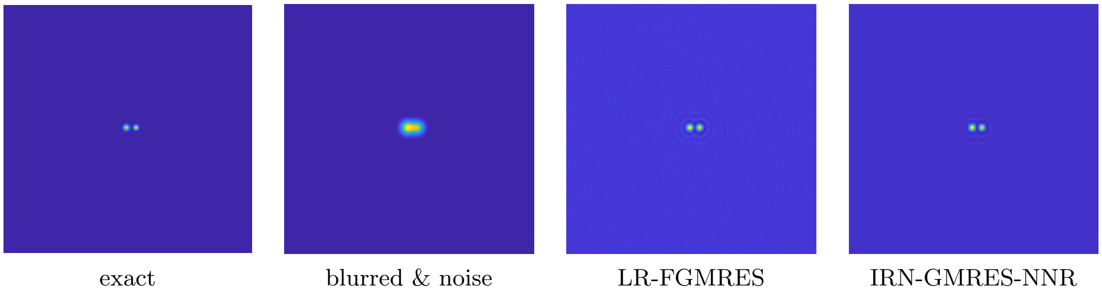

exact blurred & noise LR-FGMRES IRN-GMRES-NNR

Fig. 2: *Example 1.* Exact and corrupted test images, together with the best reconstructions obtained by the LR-FGMRES and the IRN-GMRES-NNR methods.

pixels) of the computed images, as shown in Figure 3: here the best LR-FGMRES reconstruction, as well as the IRN-GMRES-NNR reconstructions at the end of the 2nd, 3rd and 4th inner cycles are displayed. It is clear that the IRN reconstructions are improving over each outer cycle, and that even the solution at the end of the 2nd cycle is significantly better than the LR-FGMRES solution, which means that not all four outer iterations need to be run to achieve solutions of superior qualities (even if more outer iterations allow further improvement in the solution). Figure 4 displays surfaces plots of the central part (51 × 51 pixels) of the test problem data, as well as the best reconstructed images (for RS-LR-GMRES and FGMRES-NNR these are obtained at the 165th and the 63th (total) iterations, respectively). It can be seen that for all the solutions shown here, the reconstructed central two stars approximately have the same intensity, although they are somewhat less intense than in the exact image. These surface plots also confirm our earlier observation that IRN-GMRES-NNR does an exceptional job removing background noise. In addition, FGMRES-NNR also gives a good background reconstruction. Finally, Figure 5 displays the singular values of the best solutions obtained adopting different GMRES-based solvers, as well as the evolution of the singular values of the solution at the end of each inner IRN-GMRES-NNR cycle (matching the reconstructions displayed in Figure 3). The singular values are "normalized" (i.e., divided by the largest one), and the graphs are cropped to focus on the relevant values. Looking at the displayed values, we can conclude that the solutions computed by all the low-rank solvers have indeed some low-rank properties, with very quickly-decaying large singular values followed by slowly-decaying smaller singular values. Compared to GMRES, the new FGMRES-NNR and IRN-GMRES-NNR methods give solutions that have a more pronounced low rank, as shown by the large gaps between the smaller singular values of the solutions computed by these methods. Regarding IRN-GMRES-NNR, the evolution of the singular values stabilizes as we move toward later outer iterations, which validates the stopping criterion proposed in Section 4.

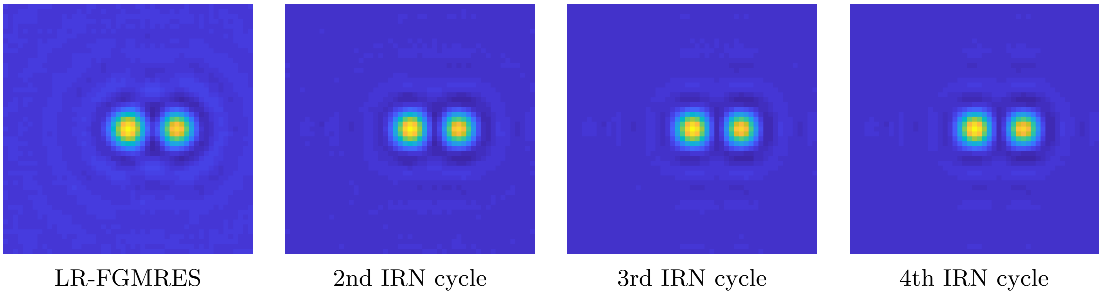
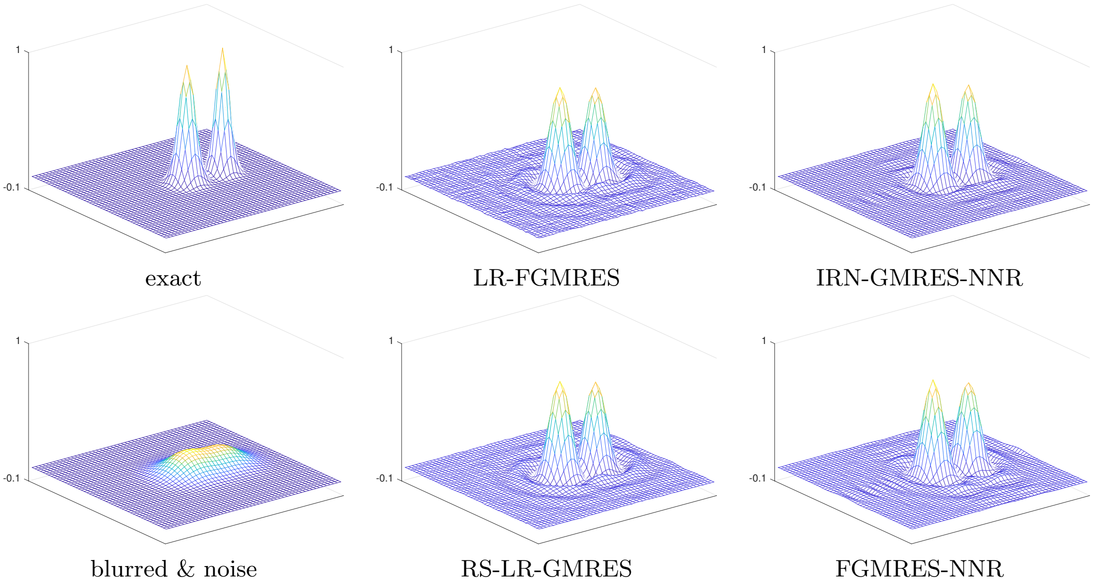


Fig. 3: *Example 1.* Zoom-ins of the LR-FGMRES best solution, and the IRN-GMRES-NNR solutions at the end of each inner cycle.

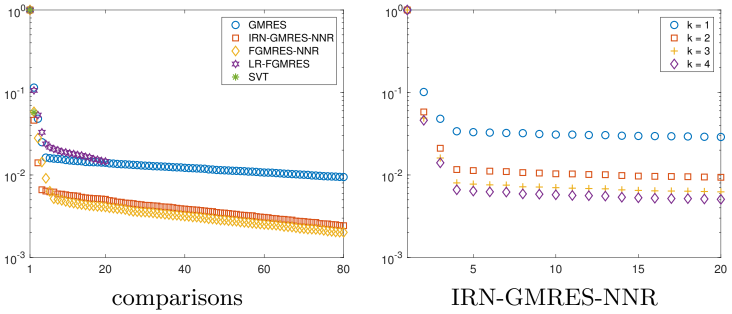
Example 2: Limited angle parallel-ray tomography. We consider a computed tomography (CT) test problem, modeling an undersampled X-ray scan with parallel beam geometry. This is a so called "limited angle" CT reconstruction problem, where the viewing angles for the object span less than 180 degrees. A smooth and rank-4 phantom is considered, as shown in the leftmost frame of Figure 7 (note that the yellow straight lines in the northwestern corner do not belong to the phantom; they are shown for later purposes). Gaussian white noise of level 10^{-2} is added to the data. The coefficient matrix $\boldsymbol{A}$ has size 32942 × 65536. Because of this, among the new solvers, only LR-FLSQR, FLSQR-NNRp, FLSQR-NNRp(v), and IRN-LSQR-NNRp will be tested, against



exact LR-FGMRES IRN-GMRES-NNR

blurred & noise RS-LR-GMRES FGMRES-NNR

Fig. 4: *Example 1.* Zoomed-in surfaces of the exact solution and the available data, as well as the best reconstructions obtained by the new GMRES-based methods.



comparisons IRN-GMRES-NNR

Fig. 5: *Example 1.* Left frame: normalized singular values of the best solutions computed by each GMRES-based method. Right frame: evolution of the singular values of the solutions computed by IRN-GMRES-NNR at each outer iteration. Singular values less than 10^{-3} are omitted.

their standard counterpart LSQR. Recall that FLSQR-NNRp(v) is the FLSQR-NNRp variant that defines the preconditioners using the basis vectors of the solution subspace. The hybrid strategy is not used here, meaning that we set $\widehat{\lambda} = 0$ for all methods. For this test problem, we consider both the values $p = 1$ and $p = 0.75$ (recall that, when $p = 1$, we omit p from the notation). The results obtained running the available low-rank solvers SVT and RS-LR-GMRES are shown, too. Note that RS-LR-GMRES only works for square matrices $\boldsymbol{A}$, hence this solver is tested on the normal equations $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{b}$, which is not the problem solved by the other methods (therefore this comparison may not be completely fair). Parameters for SVT are chosen to be: step size $\delta_k = \delta = 8 \times 10^{-5}$ and threshold $\tau = 100$. RS-LR-GMRES is set to restart every 20 iterations. The truncation rank is 10 for both basis vectors and solutions, and for both the LR-FLSQR and the RS-LR-GMRES methods. The maximum number of iterations is 100 for all methods.

Figure 6 displays the history of the relative errors for LSQR, LR-LSQR, FLSQR-NNRp, FLSQR-NNRp(v), and IRN-LSQR-NNRp, for $p = 1$ and $p = 0.75$. Figure 7 displays the exact phantom together with the best reconstructions obtained by LSQR, FLSQR-NNRp(v), and IRN-LSQR-NNR. Figure 8 displays surface plots of the northwestern corner of the exact and reconstructed phantoms (64×64 pixels, as highlighted in the leftmost frame of Figure 7).

Looking at relative errors in Figure 6, it is obvious that the winners are the FLSQR-NNRp(v) methods, with both $p = 1$ and $p = 0.75$: they give the lowest relative errors, and the fastest semi-convergences. For this test problem, using a value of $p < 1$ lowers the relative error of FLSQR-NNRp(v); however, the same does not hold for IRN-LSQR-NNRp. Therefore we can conclude that the the choice of p is problem and solver dependent, and using $p < 1$ does not necessarily improve the quality of the solution. We regard $p = 1$ as a safe choice for this parameter. Although both the FLSQR-NNRp(v) methods with $p = 1$ and $p = 0.75$ perform well, the latter is able to further reduce the noise in the reconstructed solution, especially on the boundary.

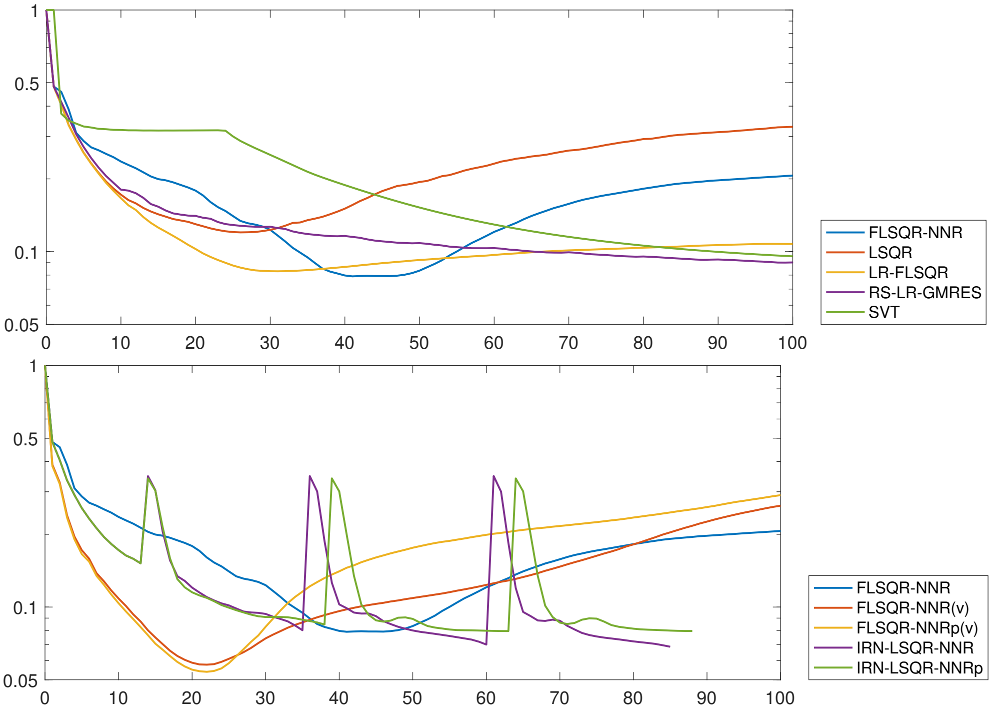
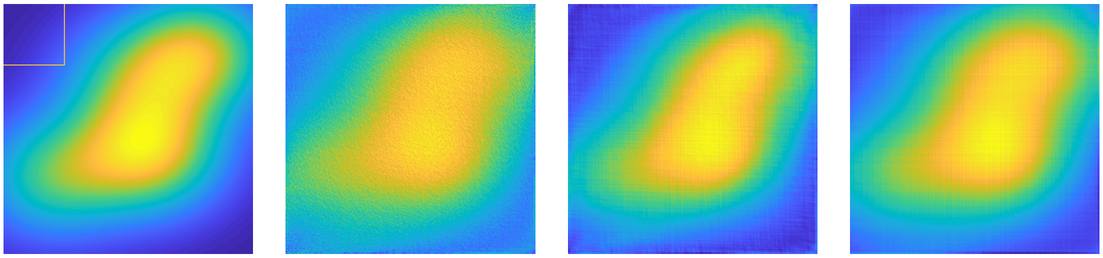
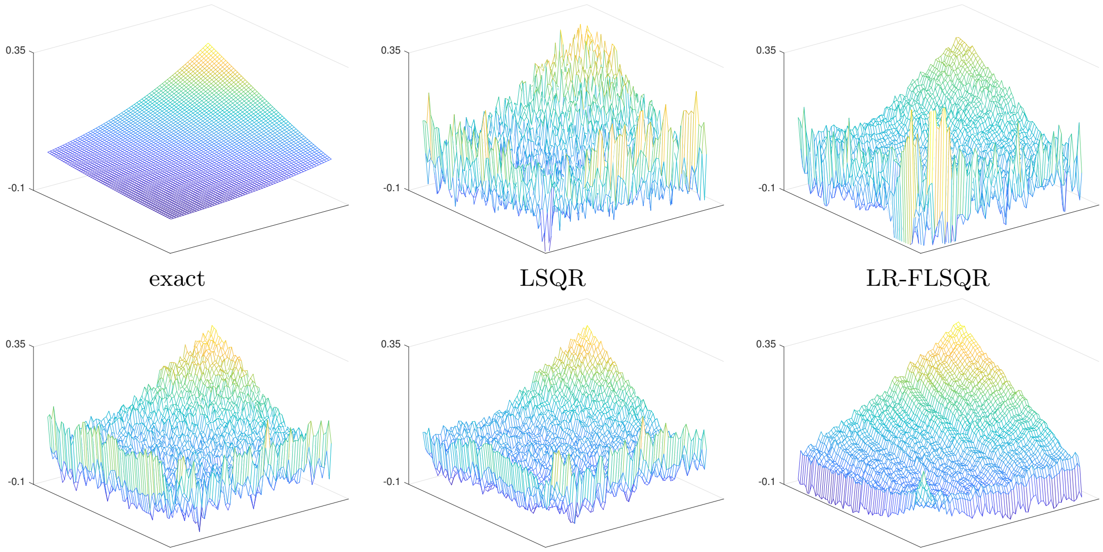


Fig. 6: *Example 2.* Relative errors vs. number of iterations for different solvers. Upper frame: some of the new solvers are compared to the already available solvers. Lower frame: comparisons of different instances of the new solvers (here $p = 0.75$).



exact LSQR FLSQR-NNRp(v) IRN-LSQR-NNR

Fig. 7: *Example 2.* Exact phantom and best reconstructions obtained by different solvers.



exact LSQR LR-FLSQR

FLSQR-NNR(v) FLSQR-NNRp(v) IRN-LSQR-NNR

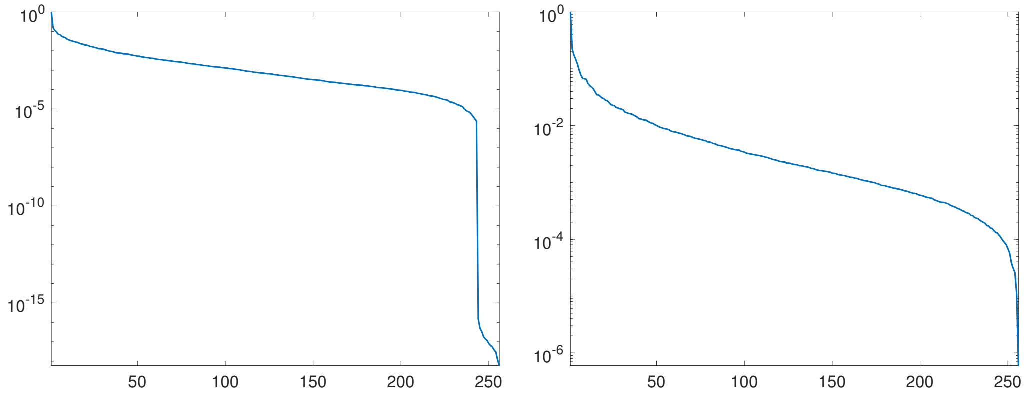
Fig. 8: *Example 2.* Surface plots of the northwestern corner of the exact phantom (highlighted in Figure 7) and the best reconstructed phantoms computed by different solvers.

Looking at all the displayed results, the advantages of our new FLSQR-NNRp(v) and IRN-LSQR-NNR methods are evident. Namely, they produce smooth solutions that preserve the original concave shape of the exact phantom, and they retain similar intensities of pixels at the same locations of the exact phantom (although the LR-FLSQR solution is smooth within the boundary, it fails to reconstruct intensity at the high point). Differences between FLSQR-NNRp(v) and IRN-LSQR-NNR reconstructions are clear, too: while both are smooth, the IRN-LSQR-NNR reconstruction has a less concave shape compared to that of FLSQR-NNRp(v), but a smoother boundary.

Example 3: Inpainting. We consider two different inpainting test problems. Inpainting is the process of restoring images that have missing or deteriorated parts. These images are likely to have quite a few lost pixels, either in the form of salt and pepper noise, or missing patches with regular or irregular shapes. The two examples considered here are of different nature: the first one has less structured and more randomly distributed missing patches, while the second one has more structured and regularly shaped missing parts. The corrupted images (shown in top-middle frames of Figures 11 and 13) are constructed by first applying a blur operator, and then superimposing the undersampling pattern to the ideally exact images (shown in the top-left frames of Figures 11 and 13). We follow this particular order of first blurring and then taking out pixels to simulate the real process of photo-taking. For both these test problems, white noise of level 10^{-2} is added to the data, and we consider purely iterative methods (i.e., $\widehat{\lambda} = 0$). We always take $p = 1$, and we run 100 iterations of all the methods.

Firstly, we consider a test problem where 58.2% of the pixels are missing (following some random and not very regular patterns). The exact image is commonly known as the `house` test image, whose rank is 243 and has a total number of 65536 (256×256) pixels; the corrupted image has the same size and number of pixels, but out of which only 27395 are non-zero. The singular values of the exact image is shown in Figure 9(a). Correspondingly, the forward operator $\boldsymbol{A}$ is of size 27395×65536, so we have an underdetermined linear system: $\boldsymbol{A}$ is obtained by first applying a shaking blur, and by then undersampling the blurred image. This can be easily coded within the *IR Tools* framework.

Figure 10 displays the history of the relative errors for LSQR, LR-FLSQR (with truncation of the basis vectors for the solution, as well as the solution, to rank 20), FLSQR-NNR, FLSQR-NNR(v) and IRN-LSQR-NNR. Figure 11 displays the exact and corrupted images, together with the best reconstructions obtained by the methods listed above: these correspond to the 16th, 32nd, 67th, 30th and 62nd iterations of LSQR, LR-FLSQR, FLSQR-NNR, FLSQR-NNR(v) and IRN-LSQR-NNR, respectively (i.e., these are the iterations where the minimum relative error is attained over the total 100 iterations).



(a). `house` (b). `peppers`

Fig. 9: *Example 3.* Singular values of exact test images `house` and `peppers` scaled by the largest singular values respectively.

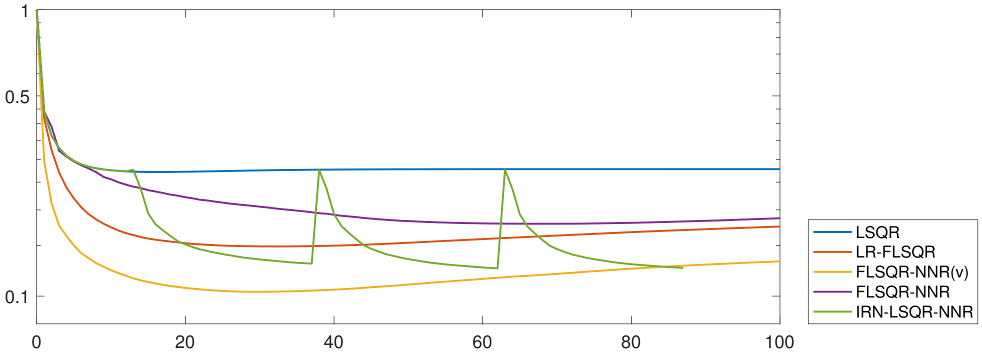


Fig. 10: *Example 3* (`house`). Relative errors vs. number of iterations for different solvers.

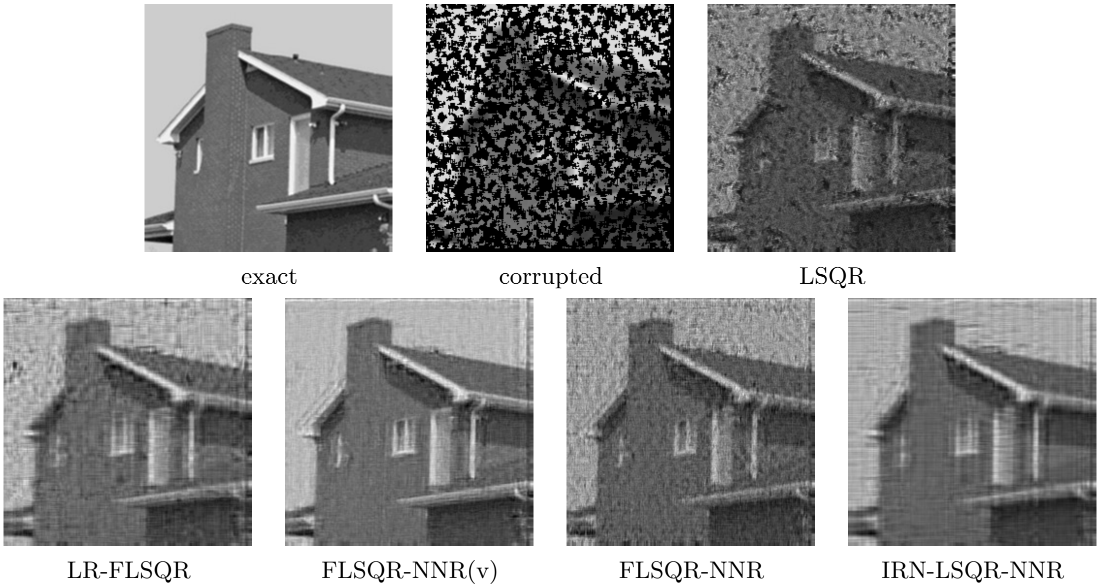


Fig. 11: *Example 3* (`house`). Exact and corrupted images; best reconstructions obtained by standard and new solvers.

Secondly, we consider a test problem similar to the previous one, i.e., we take an exact image commonly known as the `peppers` test image, which has full rank (its singular values are shown in Figure 9(b)), and we obtain the forward operator $\boldsymbol{A}$ by first applying a shaking blur, and by then undersampling the blurred image. Here the exact image has a total number of 65536 (256×256) pixels, and only around 1.3% of pixels are missing and should be inpainted: differently from the previous problem, the missing pixels follow particular patterns (e.g., circles, squares, and rectangles), and this makes the inpainting task somewhat more challenging. Figure 12 displays the history of the relative errors for LSQR, LR-FLSQR (with truncation of the basis vectors for the solution, as well as the solution, to rank 50), FLSQR-NNR, FLSQR-NNR(v) and IRN-LSQR-NNR. Figure 13 displays the exact and corrupted images, together with the best reconstructions obtained by the methods listed above: these correspond to the 11th, 18th, 60th, 33rd and 34th iterations of LSQR, LR-FLSQR, FLSQR-NNR, FLSQR-NNR(v) and IRN-LSQR-NNR, respectively (i.e., these are the iterations where the minimum relative error is attained over the total 100 iterations).

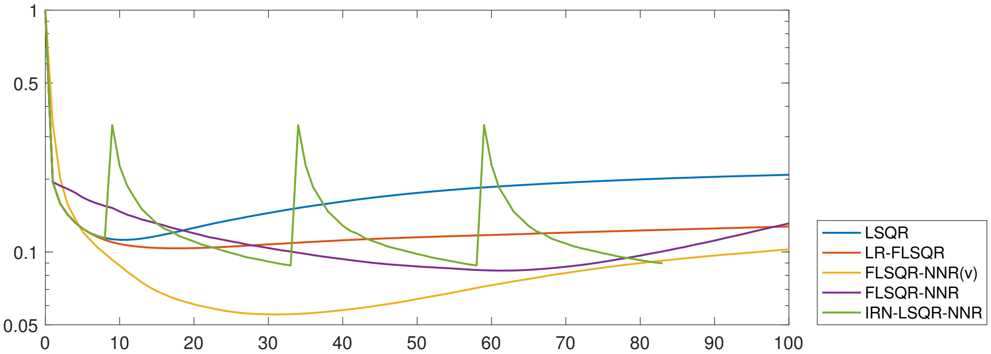
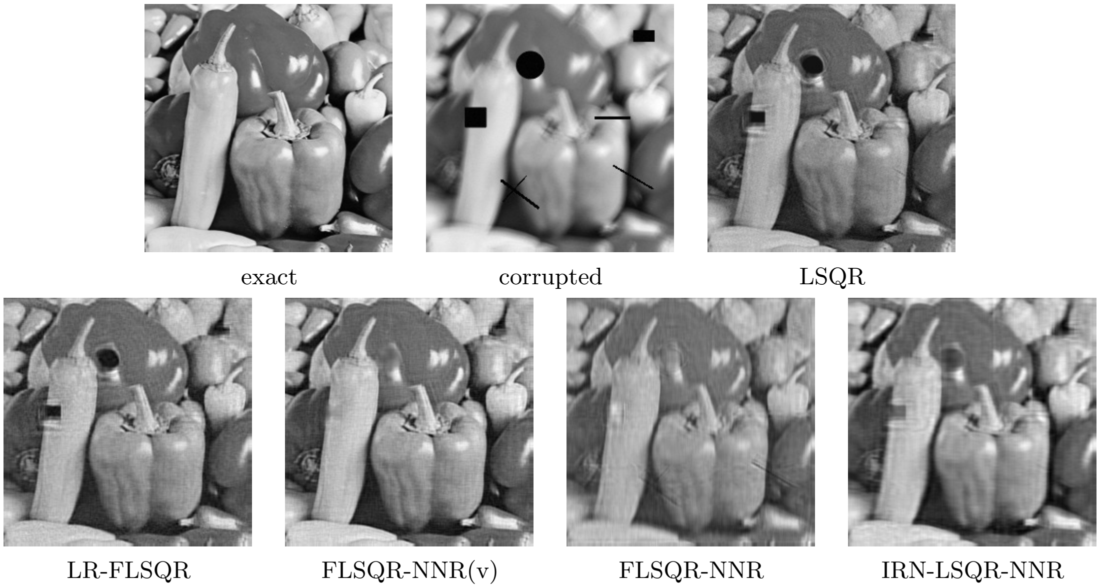


Fig. 12: *Example 3* (`peppers`). Relative errors vs. number of iterations for different solvers.



exact corrupted LSQR

LR-FLSQR FLSQR-NNR(v) FLSQR-NNR IRN-LSQR-NNR

Fig. 13: *Example 3* (`peppers`). Exact and corrupted images; best reconstructions obtained by standard and new solvers.

It is evident that FLSQR-NNR(v) achieves reconstructions of superior quality, including clarity, brightness, and smoothness. Its ability to fill-in missing spots with pixels that are of similar intensity to their surroundings is the best among all methods. The best reconstructions are computed by IRN-LSQR-NNR for the `house` test image, and by FLSQR-NNR for the `pepper` test image: in both cases, these methods are also good at removing noise and restoring missing pixels. However, for both test images, the reconstructions obtained by IRN-LSQR-NNR lack clarity compared to ones obtained by both FLSQR-NNR and FLSQR-NNR(v) methods; compared to the reconstructions obtained by LSQR and LR-FLSQR, they are anyway more desirable in terms of recovered brightness and fill-in of the missing pixels. Moreover, we have seen in these two examples that our newly proposed methods perform very well not only for low rank, but also for full or nearly full rank image reconstruction, thanks to the regularizing properties of our newly derived "preconditioners" $(\boldsymbol{W}_p^\gamma)_k$ and $\boldsymbol{S}_k$. Our methods can also be extensively tested for higher noise levels (for example, 10^{-1}) and yield similar results. However, for space considerations we are not able to show all of them here.

A study of regularization parameters. In the previous examples we have seen that the IRN-NNR methods and the flexible Krylov NNR methods perform exceptionally well on image deblurring, tomography, and inpainting problems, producing superior reconstructions compared to existing methods including SVT, RS-LR-GMRES and the low-rank flexible Krylov methods inspired by RS-LR-GMRES, even without the use of additional regularization. In this section, we explore the effect of additional regularization (i.e., we set $\widehat{\lambda} \neq 0$) on the reconstructed images and the corresponding relative errors. In particular, additional regularization allows the new methods to be used in a hybrid fashion. We are going to observe that there is only little to negligible room for the methods to improve when they are used in a hybrid fashion (as their performance is already very good with $\widehat{\lambda} = 0$).

We consider three different ways of choosing the regularization parameter $\widehat{\lambda}$. (i) We take the "secant method" mentioned in Section 4, which updates the regularization parameter at each iteration using the discrepancy. (ii) We select the optimal regularization parameter which minimizes the 2-norm of the difference between the exact solution and the regularized solution at each iteration. Namely, when using standard GMRES and LSQR, at the mth iteration we seek to minimize with respect to $\widehat{\lambda}$

$$\|\boldsymbol{x}^{\mathrm{ex}} - \boldsymbol{x}_{m,\widehat{\lambda}}\| = \|\boldsymbol{V}_m^T \boldsymbol{x}^{\mathrm{ex}} - \boldsymbol{V}_m^T \boldsymbol{x}_{m,\widehat{\lambda}}\| = \|\boldsymbol{V}_m^T \boldsymbol{x}^{\mathrm{ex}} - \boldsymbol{y}_{m,\widehat{\lambda}}\|\,;$$

when using the IRN methods we should incorporate the appropriate preconditioners $(\boldsymbol{W}_p^\gamma)_k$ and $\boldsymbol{S}_k$ and, for all the iterations in the inner iteration cycle corresponding to the kth outer iteration, we seek to minimize with respect to $\widehat{\lambda}$

$$\|\widehat{\boldsymbol{x}}^{\mathrm{ex}} - \boldsymbol{V}_m \boldsymbol{y}_{m,\widehat{\lambda}}\| = \|\boldsymbol{V}_m^T \widehat{\boldsymbol{x}}^{\mathrm{ex}} - \boldsymbol{V}_m^T \boldsymbol{V}_m \boldsymbol{y}_{m,\widehat{\lambda}}\| = \|\boldsymbol{V}_m^T \widehat{\boldsymbol{x}}^{\mathrm{ex}} - \boldsymbol{y}_{m,\widehat{\lambda}}\|, \text{ where } \widehat{\boldsymbol{x}}^{\mathrm{ex}} = (\boldsymbol{W}_p^\gamma)_k \boldsymbol{S}_k \boldsymbol{x}^{\mathrm{ex}}.$$

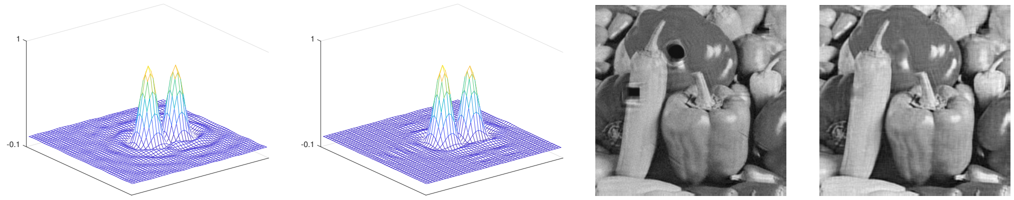
It is intrinsically difficult to implement this strategy for flexible Krylov subspace methods, because of the complexity of changing preconditioners at each iteration. (iii) We perform a manual exhaustive search. Namely, we first run the solvers multiple times using various regularization parameters $\widehat{\lambda}$, starting with a larger range and narrowing down to a smaller range containing the best parameter; we then record the minimum relative errors among all iterations for all values of $\widehat{\lambda}$, and select the corresponding $\widehat{\lambda}$. This approach is the most expensive one, and differs from the previous one in that the (optimal) regularization parameter $\widehat{\lambda}$ is fixed for all iterations. Of course, both the second and third approaches require the knowledge of the exact solution and we test them only to investigate the best possible performance of the hybrid approach.

Table 1 compares the performances (in terms of minimum relative error achieved by each method) of standard Krylov methods (GMRES and LSQR) and their IRN-NNR and flexible NNR

(F-NNR) counterparts, with and without using a hybrid approach. In this way we can understand how the use of additional regularization affects each solver differently. The three parameter choice methods described above are dubbed "Secant (i)", "Optimal (ii)" and "Fixed (iii)", respectively. All the previous examples are considered here. GMRES and its counterparts IRN-GMRES-NNR, FGMRES-NNR are used for Example 1, while LSQR and its counterparts IRN-LSQR-NNR, FLSQR-NNR(v) are used for Examples 2 and 3.

		$\widehat{\lambda} = 0$	$\widehat{\lambda} \neq 0$	$\widehat{\lambda} = 0$	$\widehat{\lambda} \neq 0$	$\widehat{\lambda} = 0$	$\widehat{\lambda} \neq 0$	$\widehat{\lambda} = 0$	$\widehat{\lambda} \neq 0$
		Example 1		*Example 2*		*Example 3* (house)		*Example 3* (peppers)	
Standard	Secant (i)	0.2995	0.2528	0.1201	0.1389	0.2712	0.2715	0.1141	0.1138
	Optimal (ii)	0.2995	0.2268	0.1201	0.1201	0.2712	0.2710	0.1141	0.1138
	Fixed (iii)	0.2995	0.2268	0.1201	0.1183	0.2712	0.2710	0.1141	0.1138
IRN-NNR	Secant (i)	0.2081	0.2096	0.0685	0.0696	0.1249	0.1250	0.0964	0.0967
	Optimal (ii)	0.2081	0.2292	0.0685	0.0685	0.1249	0.1249	0.0964	0.0964
	Fixed (iii)	0.2081	X	0.0685	0.0660	0.1249	X	0.0964	0.0960
F-NNR	Secant (i)	0.2829	0.2658	0.0577	0.0684	0.1035	0.1046	0.0625	0.0618
	Fixed (iii)	0.2829	0.2640	0.0577	0.0568	0.1035	X	0.0625	0.0618

Table 1: Minimum relative errors without ($\widehat{\lambda} = 0$) and with ($\widehat{\lambda} \neq 0$) a hybrid approach. The mark "X" means that the optimal regularization parameter found by the "Fixed (iii)" method is 10^{-16}, hence there is no need for additional regularization.



hybrid GMRES IRN-GMRES-NNR hybrid LSQR FLSQR-NNR(v)

Fig. 14: Reconstructions obtained by standard hybrid Krylov methods and by the new methods without using additional regularization. Left side: zoomed in surface plots of the reconstructions of *Example 1*; right side: reconstructions of *Example 3* (peppers).

It is easy to observe that the use of additional regularization is most effective for the standard GMRES solver, where the minimum relative error is reduced significantly. However, for the other solvers, the hybrid approach does not have a notable advantage over not using regularization. At times the "Fixed (iii)" parameter choice strategy delivers a regularization parameter of the order of 10^{-16}, which is numerically equivalent to not having regularization. This indicates that our new IRN-NNR and F-NNR methods are successful in computing good reconstructions and, even without additional regularization, they perform much better than standard Krylov methods used in a hybrid fashion (comparing IRN-GMRES-NNR to GMRES in Example 1, and FLSQR-NNR(v) to LSQR in the other examples). Figure 14 shows a couple of such comparisons.

6. Conclusions. This paper introduced new solvers, based on Krylov subspace methods, for the computation of approximate low-rank solutions to large-scale linear systems of equations. Our main goal was to apply the new methods to regularize inverse problems arising in imaging applications. The starting point of our derivations was an IRN approach to the NNRp problem (1.3). In this way, the original problem (1.3) is reduced to the solution of a sequence of quadratic problems, where an appropriate smoothed linear transformation is introduced to approximate the nondifferentiable nuclear norm regularization term. Our new methods make smart use of Kronecker product properties to reformulate each quadratic problem in the IRN sequence as a Tikhonov-regularized problem in standard form. We use both Krylov methods with fixed "preconditioners" within an inner-outer iteration scheme (namely, IRN-LSQR-NNRp and IRN-GMRES-NNRp), and Krylov methods with flexible iteration-dependent "preconditioners" within a single iteration scheme (namely, FLSQR-NNRp, FGMRES-NNRp, LR-FGMRES, and LR-FLSQR). Some of these methods (namely, IRN-LSQR-NNRp, IRN-GMRES-NNRp, FLSQR-NNRp, and FGMRES-NNRp) can be used in a hybrid framework, so that the Tikhonov regularization parameter can be efficiently, effectively, and adaptively chosen. These new solvers are shown to perform exceptionally well on the test problems described in Section 5, and they give reconstructions of significantly improved quality over existing methods.

Future work includes the extension of the present methods to handle cases where the solution of (1.1) is low-rank but rectangular, i.e., $\mathrm{vec}^{-1}(\boldsymbol{x}) = \boldsymbol{X} \in \mathbb{R}^{m\times n}$ with $m \neq n$. Also, while a solid theoretical justification is provided for IRN-LSQR-NNRp and IRN-GMRES-NNRp, the same is not true for FGMRES-NNRp and FLSQR-NNRp: further analysis will be needed to deeply understand the regularization properties of these flexible solvers. Finally, the new IRN-LSQR-NNRp and IRN-GMRES-NNRp methods can be reformulated to work with well-posed problems and in the framework of matrix equations, possibly providing a valid and principled alternative to the current popular methods based on low-rank-projected and restarted Krylov solvers.

REFERENCES

[1] M. Belge, M. E. Kilmer, and E. L. Miller, *Wavelet domain image restoration with adaptive edge-preserving regularization*, IEEE Trans. Image Process., 9 (2000), pp. 597–608, https://doi.org/10.1109/83.841937.

[2] J. D. Blanchard, J. Tanner, and K. Wei, *CGIHT: conjugate gradient iterative hard thresholding for compressed sensing and matrix completion*, Information and Inference: A Journal of the IMA, 4 (2015), pp. 289–327, https://doi.org/10.1093/imaiai/iav011.

[3] J. Cai, E. Candès, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM Journal on Optimization, 20 (2010), pp. 1956–1982, https://doi.org/10.1137/080738970.

[4] J. Chung and S. Gazzola, *Flexible Krylov Methods for ℓ_p Regularization*, ArXiv e-prints, (2018), https://arxiv.org/abs/1806.06502.

[5] M. Fornasier, H. Rauhut, and R. Ward, *Low-rank matrix recovery via iteratively reweighted least squares minimization*, SIAM Journal on Optimization, 21 (2011), pp. 1614–1640, https://doi.org/10.1137/100811404.

[6] S. Gazzola, P. C. Hansen, and J. G. Nagy, *IR Tools: A MATLAB Package of Iterative Regularization Methods and Large-Scale Test Problems*, 2017, https://arxiv.org/abs/arXiv:1712.05602.

[7] S. Gazzola and J. G. Nagy, *Generalized Arnoldi-Tikhonov Method for Sparse Reconstruction*, SIAM J. Sci. Comput., 36 (2014).

[8] S. Gazzola, P. Novati, and M. R. Russo, *On krylov projection methods and tikhonov regularization*, Electronic Transactions on Numerical Analysis, 44 (2014), pp. 82–123.

[9] S. Gazzola and M. Sabaté Landman, *Flexible gmres for total variation regularization*, BIT Numerical Mathematics, (2019), https://doi.org/10.1007/s10543-019-00750-x.

[10] D. Goldfarb and S. Ma, *Convergence of fixed-point continuation algorithms for matrix rank minimization*, Foundations of Computational Mathematics, 11 (2011), pp. 183–210, https://doi.org/10.1007/s10208-011-9084-6.

[11] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, (2011).

[12] M. Hanke and P. C. Hansen, *Regularization methods for large-scale problems*, Surv. Math. Ind., 3 (1993), pp. 253–315.

[13] P. C. Hansen, *Discrete Inverse Problems: Insight and Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2010.

[14] P. C. Hansen and T. K. Jensen, *Smoothing-norm preconditioning for regularizing minimum-residual methods*, SIAM J. Matrix Anal. Appl., 29 (2006), pp. 1–14.

[15] T. K. Jensen and P. C. Hansen, *Iterative regularization with minimum-residual methods*, BIT, 47 (2007), pp. 103–120.

[16] R. H. Keshavan, A. Montanari, and S. Oh, *Matrix completion from noisy entries*, J. Mach. Learn. Res., 11 (2010), pp. 2057–2078, http://dl.acm.org/citation.cfm?id=1756006.1859920.

[17] M. Kilmer and D. O'Leary., *Choosing regularization parameters in iterative methods for ill-posed problems*, SIAM Journal on Matrix Analysis and Applications, 22 (2001), pp. 1204–1221, https://doi.org/10.1137/S0895479899345960.

[18] M. E. Kilmer and D. O'Leary, *Choosing regularization parameters in iterative methods for ill-posed problems*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 1204–1221.

[19] D. Kressner and C. Tobler, *Low-rank tensor Krylov suspace methods for parameterized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1288–1316.

[20] R. M. Larsen, *Lanczos bidiagonalization with partial reorthogonalization*, Department of Computer Science, Aarhus University, Technical report, (1998).

[21] B. J. Last and K. Kubik, *Compact gravity inversion*, GEOPHYSICS, 48 (1983), pp. 713–721, https://doi.org/10.1190/1.1441501.

[22] K. Lee and H. C. Elman, *A Preconditioned Low-Rank Projection Method with a Rank-Reduction Scheme for Stochastic Partial Differential Equations*, SIAM J. Sci. Comput., 9 (2017).

[23] C. Lu, Z. Lin, and S. Yan, *Smoothed low rank and sparse matrix recovery by iteratively reweighted least squares minimization*, IEEE Transactions on Image Processing, 24 (2015), pp. 646–654, https://doi.org/10.1109/TIP.2014.2380155.

[24] S. Ma, D. Goldfarb, and L. Chen, *Fixed point and bregman iterative methods for matrix rank minimization*, Mathematical Programming, 128 (2011), pp. 321–353, https://doi.org/10.1007/s10107-009-0306-5.

[25] K. Mohan and M. Fazel, *Iterative reweighted algorithms for matrix rank minimization*, J. Mach. Learn. Res., 13 (2012), pp. 3441–3473, http://dl.acm.org/citation.cfm?id=2503308.2503351.

[26] D. P. O'Leary and J. A. Simmons, *A bidiagonalization-regularization procedure for large scale discretizations of ill-posed problems*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 474–489.

[27] B. Recht, M. Fazel, and P. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM Review, 52 (2010), pp. 471–501, https://doi.org/10.1137/070697835.

[28] R. A. Renaut, S. Vatankhah, and V. E. Ardestani, *Hybrid and iteratively reweighted regularization by unbiased predictive risk and weighted GCV for projected systems*, SIAM J. Scientific Computing, 39 (2017), https://doi.org/10.1137/15M1037925.

[29] R. T. Rockafellar, *Convex analysis*, vol. 28, Princeton university press, 1970.

[30] P. Rodriguez and B. Wohlberg, *An efficient algorithm for sparse representations with ℓ_p data fidelity term*, (2008).

[31] Y. Saad, *A flexible inner-outer preconditioned gmres algorithm*, SIAM Journal on Scientific Computing, 14 (1993), pp. 461–469, https://doi.org/10.1137/0914028.

[32] 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, https://doi.org/10.1002/nla.499.

[33] M. Stoll and T. Breiten, *A low-rank in time approach to PDE-constrained optimization*, SIAM J. Sci. Comput., 27 (2015), pp. B1–B29.

[34] S. Vatankhah, R. Renaut, and S. Liu, *Research note: A unifying framework for widely-used stabilization of potential field inverse problems*, Geophysical Prospecting, (2019), https://doi.org/10.1111/1365-2478.12926.