

Accelerating the iterative solution of convection–diffusion problems using singular value decomposition

G. Pitton | L. Heltai 

SISSA—International School for
Advanced Studies, Via Bonomea 265,
34136 Trieste, Italy

Correspondence

Luca Heltai, SISSA—International School
for Advanced Studies, Via Bonomea 265,
34136 Trieste, Italy.
Email: luca.heltai@sisa.it

Summary

The discretization of convection–diffusion equations by implicit or semi-implicit methods leads to a sequence of linear systems usually solved by iterative linear solvers such as the generalized minimal residual method. Many techniques bearing the name of *recycling Krylov space methods* have been proposed to speed up the convergence rate after restarting, usually based on the selection and retention of some Arnoldi vectors. After providing a unified framework for the description of a broad class of recycling methods and preconditioners, we propose an alternative recycling strategy based on a singular value decomposition selection of previous solutions and exploit this information in classical and new augmentation and deflation methods. The numerical tests in scalar nonlinear convection–diffusion problems are promising for high-order methods.

KEYWORDS

augmentation methods, deflation methods, GMRES, singular value decomposition

1 | INTRODUCTION

The approximation of nonlinear time-dependent partial differential equations (PDEs), independently on the actual numerical scheme used to discretize the equations and the domain, results in sequences of large and possibly nonsymmetric linear systems, deriving from the linearization of time-varying nonlinear terms. The most popular iterative solvers used for these problems are Krylov subspace methods, like the conjugate gradient (CG) method for symmetric positive definite systems,¹ the generalized conjugate residual (GCR) method,² and the generalized minimal residual (GMRES) method³ for general nonsymmetric systems.

These are direct methods in exact arithmetic, since they find the exact solution of the system in, at most, n iterations by computing an orthogonal basis for the entire vector space, and are approximate methods when the iteration process is terminated before convergence. In the CG method, the full Krylov subspace is explored through a three-term recursion, which guarantees a constant work and storage cost per iteration.⁴ This is in contrast with the GMRES method, where the work and storage per iteration grow linearly with the iteration number. Despite the fact that finite-precision arithmetic makes the construction of Krylov subspaces a more and more ill-conditioned process as the iteration count increases, it has been shown that the resulting loss of orthogonality among the Arnoldi vectors does not prevent the method from converging, as long as the vectors are linearly independent. The effect of finite precision on the stability of Krylov space methods was extensively studied in the works of Greenbaum et al.,⁵ Paige and Strakoš,⁶ and Paige et al.⁷

In real-life applications, the increasing storage cost of unrestarted GMRES makes it unfeasible for large nonsymmetric systems. The most effective strategy to improve the rate of convergence in iterative solvers comes from the use of properly

chosen preconditioners. While several preconditioners are available, which perform very well for symmetric and positive definite linear systems, this is not the case for general nonsymmetric systems, where the construction of good preconditioners is generally much more difficult and problem dependent. When preconditioning alone fails to reduce the number of iterations below a reasonable threshold, it is common to use a restarted or truncated version of GMRES or GMRES(m).

The most important side effect of the restart procedure is that the information of the vectors thrown away is lost at each restart, compromising or delaying the superlinear convergence often observed in the final GMRES iterations.⁸ The analysis of the GMRES(m) method⁹ is nontrivial and leads to some surprising results.^{10,11} Some strategies vary heuristically the restart parameter in order to improve the convergence rate,¹² but the most effective solutions try to keep orthogonality with respect to some previously spanned subspaces, for example, by carefully retaining part of the previously constructed Krylov subspace, *augmenting* the search space of the following cycle, or *deflating* the linear operator's spectrum.^{13,14}

Even though these acceleration techniques are often presented as *preconditioning* techniques, they are conceptually different. Deflation methods, first introduced in the work of Nicolaides¹⁵ for the CG method, consist in the left multiplication of the linear system by a suitably chosen projector, with the aim of removing the components responsible for slow convergence (e.g., small-magnitude eigenvalues), resulting in a *singular* algebraic system to be solved. In augmentation methods, a selected subspace responsible for a slowdown of the iterative method is removed from the matrix range, resulting in a “tamed” Krylov subspace.

We group these methods under the general name of *acceleration methods*, as they aim at accelerating the computation of an approximate solution by using wisely at least two vector spaces: a Krylov space and some recycling space. The first one is usually obtained through Arnoldi or two-sided Lanczos algorithms, whereas the generation and the exploitation of the second vector space are largely algorithm dependent. An application of these ideas to sequences of linear systems is presented in the work of Parks et al.,¹⁶ whereas computational fluid dynamics applications are presented in the work of Amritkar et al.¹⁷ Recycling methods for sequences of symmetric linear systems arising in topology optimization are studied in the work of Wang et al.,¹⁸ whereas in the works of Mello et al.¹⁹ and Kilmer and de Sturler,²⁰ these are applied as efficient solvers for inverse problems in electrical impedance tomography.

However, when solving time-dependent problems, one may expect that it should be possible to extract useful spectral information also from a number of solutions to previous time steps, and not only from previous GMRES cycles, even though the solutions are obtained from different matrices and right-hand sides.

We provide a unified framework for six different recycling methods, some of which are equivalent to deflation preconditioners known in the literature. These recycling methods are applied on the basis of subspaces that exploit the information contained in a number of previous solution steps (using singular value decomposition [SVD], as done first in the work of Ali and Sadkane²¹) to accelerate the solution of the current time step.

In particular, we explore recycling methods in two different variations: (a) by forming good initial guesses for the solution of the current linear systems and (b) by constructing subspaces for augmented and deflated methods. The two approaches are known to be related to each other and are often interpreted similarly.

The first procedure is inspired by extrapolation methods commonly used in time-dependent problems to extrapolate the solution at the current time to compute nonlinear terms of the system. Extrapolated solutions can be used as a seed for Krylov subspaces, and in this work we show that a much more efficient result is obtained by replacing extrapolation with projection, as in the work of Ali and Sadkane,²¹ and combining it with proper recycling Krylov methods. This removes the stability issues associated to equispaced extrapolation, allowing one to keep a larger number of previous solutions and to obtain best approximation-type estimates.

Moreover, by replacing extrapolation with projection, additional freedom is available in the choice of the projection subspace. In this work, we explore and then compare several existing possibilities, starting from orthonormalization of some previous solutions, and propose a new approach based on the retainment of the principal components of a set of previous solutions using SVD.

A similar strategy is used to construct efficient augmented and deflative methods that exploit the SVD of a collection of previous time step solutions to construct the recycling space. We use this space to construct several preconditioners and show that in many test cases, applying SVD for the selection of the recycling spaces works very well, at least for scalar convection–diffusion problems and for moderate values of the Péclet number.

In Sections 2 and 3, we introduce the setting of our time-dependent problem in terms of general Petrov–Galerkin approximations and review the solution of time-dependent linear problems in terms of Krylov subspace methods, along with some of the most popular acceleration methods. We introduce a new framework for the unified discussion of some recycling spaces, which allows the systematic construction of three augmentation and three deflation methods, some of which are equivalent to known methods and some of which are new. In Section 4, we consider procedures for the construction of

a good initial guess based on extrapolation or restriction on a space of previous solutions, and we propose a new technique based on a multiple subspace correction principle, whereas in Section 5, we review some techniques for the generation of a recycling subspace, and we consider two cases where the subspace is obtained from a fixed set of previous solutions, or by means of an SVD performed on the space spanned by some previous solution vectors. In Sections 6 and 7, we present a selection of numerical experiments and draw some conclusions.

2 | PROBLEM DEFINITION

Let X and Y be normed spaces, Y' be the dual of Y , and $L(u) : X \rightarrow Y'$ be a time-dependent, linear, and bounded map, possibly depending on the current solution $u \in X$, which represents the differential operator of a PDE on the space X . We are interested in finding approximate solutions of nonlinear time-dependent PDEs of the form

$$\partial_t u + L(u)u = f, \quad (1)$$

where $\partial_t u$ is a shorthand for the time derivative of u , and $f \in Y'$ is given.

In the following, we will consider principally a weak form of Equation (1):

$$\langle v, \partial_t u \rangle + \langle v, L(u)u \rangle = \langle v, f \rangle \quad \forall v \in Y. \quad (2)$$

From the weak Equation (2), we can derive a Petrov–Galerkin approximation²² by introducing some finite-dimensional subspaces $X_h \subset X$ and $Y_h \subset Y$ of dimension n , so that the approximation $u_h \in X_h$ solves

$$\langle v, \partial_t u_h \rangle + \langle v, L(u_h)u_h \rangle = \langle v, f \rangle \quad \forall v \in Y_h. \quad (3)$$

It is understood that, as the discretization parameter h tends to 0, X_h tends to span all of X . Letting $\{\varphi_j\}_{j=1}^n$ and $\{\psi_i\}_{i=1}^n$ be two basis sets for X_h and Y_h , respectively, the approximate solution can be written as $u_h = \sum_{j=1}^n u_j(t)\varphi_j$. Applying a finite-difference–like discretization of the time derivative, the coefficients $u_j(t)$ are evaluated only at certain fixed points in time $u_j(t_k)$. We obtain a sequence of nonlinear problems

$$\beta_0 \langle \psi_i, \varphi_j \rangle u_j(t_k) + \sum_{j=1}^n \langle \psi_i, L(u_h)\varphi_j \rangle u_j(t_k) = \langle \psi_i, f \rangle - \sum_{\alpha=1}^p \beta_\alpha \langle \psi_i, \varphi_j \rangle u_j(t_{k-\alpha}) \quad i = 1, \dots, N, \quad (4)$$

where β_α are the coefficients coming from the chosen finite difference scheme, and p is the number of steps of the time-advancing scheme.

The nonlinear term can be linearized around a (known) guess solution \tilde{u} by considering, for example, a Newton-like approximation

$$L(u_h) \sim L(\tilde{u}_h) + D_u L(\tilde{u}_h)(u_h - \tilde{u}_h), \quad (5)$$

where $D_u L$ is the Fréchet derivative of L .

This leads to the time-dependent linear systems

$$A(t_k)\mathbf{x} = \mathbf{b}(t_k), \quad (6)$$

where

$$\begin{aligned} (A(t_k))_{ij} &= \beta_0 \langle \psi_i, \varphi_j \rangle + \langle \psi_i, L(\tilde{u}_h)\varphi_j \rangle + \langle \psi_i, D_u L(\tilde{u}_h)\varphi_j \rangle \\ (\mathbf{x})_i &= u_i(t_k) \\ (\mathbf{b})_i &= \langle \psi_i, f \rangle + \sum_{\alpha=1}^p \beta_\alpha \langle \psi_i, \varphi_j \rangle u_j(t_{k-\alpha}) + \langle \psi_i, D_u L(\tilde{u}_h)\tilde{u}_h \rangle. \end{aligned} \quad (7)$$

In the examples presented in this work, we neglect the term $D_u L(\tilde{u}_h)(u_h - \tilde{u}_h)$ and consider simple extrapolations for \tilde{u}_h . For the sake of simplicity, we assume that $A \in \mathbb{R}^{n \times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$, but in principle, it is possible to recast the problem for complex-valued functions by replacing the transpose operation with the Hermitian and by using duality pairings in normed spaces over \mathbb{C} .

In the rest of the current section, we consider mainly the linear system

$$A\mathbf{x} = \mathbf{b}, \quad (8)$$

where, to simplify the notation, the time dependence of A is left implicit. In general, the dimension n of the linear system can depend on t_k , but this is not emphasized here to avoid the proliferation of indices. We assume that the sequence of matrices and right-hand sides depend on the previous solution, $A(t_i) = A(t_i, \mathbf{x}(t_{i-1}))$, $\mathbf{b}^i = \mathbf{b}^i(t_i, \mathbf{x}(t_{i-1}))$, and they are not entirely available until the very last element of the sequence.

3 | REVIEW OF KRYLOV SUBSPACE METHODS

Given an initial guess \mathbf{x}_0 for each time step, Krylov subspace methods for linear systems (see, for instance, the works of Golub and Van Loan,⁴ Greenbaum,²³ Hackbusch,²⁴ Liesen and Strakoš,²⁵ and Saad²⁶) provide a sequence of approximate solutions $\{\mathbf{x}_k\}$ in the form (following the notation of Liesen and Strakoš²⁵)

$$\begin{aligned}\mathbf{x}_k &= \mathbf{x}_0 + \mathbf{z}_k \\ \mathbf{r}_k &= \mathbf{r}_0 - A\mathbf{z}_k\end{aligned}\quad \mathbf{z}_k \in S_k, \quad (9)$$

where S_k is a k -dimensional *search space*, and the approximation vector \mathbf{z}_k is usually determined by imposing some constraints on the residual $\mathbf{r}_k := \mathbf{b} - A\mathbf{x}_k$.

Two common choices to select \mathbf{z}_k are as follows:

minimal residual : $\min_{\mathbf{z} \in S_k} \|\mathbf{r}_k\|$;

orthogonal residual : $\mathbf{r}_k \perp C_k$ for some *constraint space* C_k .

We denote with $\{\mathbf{c}_i\}_{i=1}^k$ and $\{\mathbf{s}_i\}_{i=1}^k$ two basis sets for C_k and S_k , which will be identified with the column vectors of the orthogonal matrices $C_k \in \mathbb{R}^{n \times k}$ and $S_k \in \mathbb{R}^{n \times k}$.

In practice, if $S_k, C_k \in \mathbb{R}^{n \times k}$ are constructed such that their column space spans the search space and the constraint space, respectively, that is, $\text{Im } S_k = S_k$ and $\text{Im } C_k = C_k$, then the orthogonal residual solutions can be characterized by

$$C_k^T \mathbf{r}_k = 0 \quad \implies \quad C_k^T A S_k \mathbf{t}_k = C_k^T \mathbf{r}_0, \quad (10)$$

where $\mathbf{z}_k = S_k \mathbf{t}_k$, $\mathbf{t}_k \in \mathbb{R}^k$, and $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ is the initial residual. In this case, the approximate solution at the k th iteration is given by

$$\begin{aligned}\mathbf{x}_k &= \mathbf{x}_0 + S_k (C_k^T A S_k)^{-1} C_k^T \mathbf{r}_0 \\ \mathbf{r}_k &= \mathbf{r}_0 - A S_k (C_k^T A S_k)^{-1} C_k^T \mathbf{r}_0.\end{aligned} \quad (11)$$

While minimal residual solutions exist and are unique for any choice of S_k , this is not the case for the orthogonal residual. Indeed, the existence of orthogonal residual solutions can be ensured provided that some compatibility conditions hold between the search and constraint spaces. However, for some choices of S_k , the minimal residual and orthogonal residual solutions coincide. We refer to the work of Liesen and Strakoš²⁵ for an in-depth discussion on this topic.

Equation (11) can be expressed in terms of a projection operator as

$$\begin{aligned}\mathbf{x}_k &= \mathbf{x}_0 + A^{-1} P_{AS_k, C_k} \mathbf{r}_0, \\ \mathbf{r}_k &= (I - P_{AS_k, C_k}) \mathbf{r}_0,\end{aligned} \quad (12)$$

where the notation $P_{\mathcal{A}, \mathcal{B}} \mathbf{x}$ denotes the projection of \mathbf{x} onto the space \mathcal{A} orthogonal to \mathcal{B} , that is, $P_{AS_k, C_k} := AS_k (C_k^T AS_k)^{-1} C_k^T$ is a projector onto AS_k orthogonal to C_k (see, e.g., the work of Liesen and Strakoš²⁵).

We remark that in all practical applications, the iteration \mathbf{x}_k is not computed naively by solving the reduced linear system (10) but in more computationally efficient ways.²⁶

Now, we are left with the choice of the spaces S_k and C_k . The most popular methods construct their iterations on Krylov subspaces, that is, successive powers of A times the initial residual,

$$S_k = \mathcal{K}_k(A, \mathbf{r}_0) := \text{span} \{ \mathbf{r}_0, A\mathbf{r}_0, A^2\mathbf{r}_0, \dots, A^{k-1}\mathbf{r}_0 \}, \quad (13)$$

and in the nonsymmetric case, the most frequent choices are $C_k = S_k$ for the full orthogonalization method, $C_k = AS_k$ for the GMRES method,³ and $C_k = \mathcal{K}_k(A^T, \mathbf{r}_0)$ for the methods based on the nonsymmetric Lanczos procedure such as the biconjugate gradient stabilized (BiCGStab) method and the quasi-minimal residual (QMR) family.

In Krylov space methods, two directions can be explored to speed up the convergence rate, as follows:

1. select a good initial guess, so that some norm of \mathbf{r}_0 is as small as possible;

2. manipulate the matrix (and possibly the right-hand side) in order to obtain an equivalent linear system that is easier to solve.

These two strategies are not always so sharply distinguished, as is the case with multiple subspace correction methods.

Manipulation of the system matrix is usually referred to as *preconditioning* of the system (see the work of Benzi²⁷ for a thorough introduction): Both sides of the equation could be multiplied on the left by a matrix M (called, in this case, the *left preconditioner*) such that the new linear system

$$MAx = Mb \quad (14)$$

can be solved more effectively by the iterative method. An alternative is *right preconditioning*, that is,

$$AMy = b \quad x = M^{-1}y, \quad (15)$$

which, in some circumstances, allows much more freedom in the choice of M .²⁸

When both approaches are used at the same time, it is desirable to make sure that the resulting Krylov subspaces do not contain components in the directions that have already been explored when constructing the initial guess, resulting in multilevel preconditioners.

Over the last 20 years, many techniques have been proposed to exploit this principle in restarted GMRES, most of which are under the name of *recycling Krylov spaces*, *augmentation methods*, or *deflation methods*. A common feature shared by these *acceleration methods* is to compute approximate solutions to large linear systems using wisely at least two low-dimensional vector spaces: a Krylov space and some recycling space. The first one is usually obtained through Arnoldi or two-sided Lanczos algorithms, whereas the generation and the exploitation of the second vector space are largely algorithm dependent.

3.1 | Augmented Krylov methods

The discussion starting in this section aims at providing a common framework for some augmentation and deflation methods, described in Section 3.3. The theoretical treatment of recycling methods is kept to a minimum, referring to the works of Saad²⁹ and Chapman and Saad³⁰ for a general analysis of deflated and augmented methods.

Augmentation methods enrich the search space S_k of the Krylov method with an “augmentation subspace” \mathcal{V}_s , so that the approximate solution will lie in the new search space, that is,

$$S_k = \mathcal{K}_k(A, r_0) + \mathcal{V}_s. \quad (16)$$

As a consequence, in augmentation methods applied to the GMRES solver, the residual at the k th step is orthogonal to the following “augmented” constraint space:

$$C_k = \mathcal{K}_k(A, Ar_0) + A\mathcal{V}_s. \quad (17)$$

This method is effective in reducing the iteration count when the space \mathcal{V}_s contains a good approximation of the solution and when successive powers $A^k r_0$ fail to cover the elements of \mathcal{V}_s within a reasonable number of iterations. Due to the additional orthogonalization operations required by larger search and constraint spaces, the dimension s of \mathcal{V}_s should be chosen with a grain of salt to achieve a good compromise in the computational cost, as will be explained in Section 6.2.

In the case of augmentation methods, the k th iteration does not formally differ from the one in Equation (11), the only difference being the specific choices (16) and (17) for the matrices S_k and C_k .

3.2 | Deflated Krylov methods

If the recycling subspace \mathcal{V}_s does not contain a good approximation of the solution, that is, the solution projection onto this space has a small magnitude, but, at the same time, \mathcal{V}_s has components along the eigenvectors of A with small eigenvalues, it may be convenient to *deflate* the search space of the entire subspace \mathcal{V}_s (after solving for the components of the solution along \mathcal{V}_s).

In the following, we denote by V_s any orthogonal matrix whose columns span the space \mathcal{V}_s and by $E_s := V_s^T A V_s$ the restriction of A onto \mathcal{V}_s , and we consider only left preconditioning (the right-preconditioning counterparts are constructed similarly).

The linear system is multiplied on the left by the projector $M_D = I - AV_s E_s^{-1} V_s^T$, and the resulting problem, that is,

$$M_D A \mathbf{x} = M_D \mathbf{b}, \quad (18)$$

is solved with a Krylov space method.

The “preconditioner” qualifier for the deflation projection M_D should be taken with care, since the resulting operator is no longer invertible, and many authors make a distinction in this case, avoiding the term “preconditioner.” The effect of M_D on A is to set to zero any element of \mathcal{V}_s , that is, by construction $M_D A V_s = 0$.

More generally, deflation preconditioners can be used to move some eigenvalues of matrix A to any target value λ^* or to inject artificial eigenvalues into the resulting system by adding a term to the preconditioner, that is,

$$M_D = I - AV_s E_s^{-1} V_s^T + \lambda^* V_s E_s^{-1} V_s^T.$$

Standard deflation preconditioners construct V_s using (an approximation of) the eigenvectors corresponding to the smallest eigenvalues of A .

The works of Nabben and Vuik³¹ and Tang et al.³² present a comparison of many deflation preconditioner instances with standard multigrid and domain decomposition preconditioners. In the work of Nabben and Vuik,³¹ the authors prove that the deflative preconditioners perform better than coarse grid correction preconditioners for any choice of the deflation space V_s . This is contrasted by the class of adaptive algebraic multigrid³³ and adaptive smoothed aggregation multigrid^{34,35} methods, where a coarse grid correction method in the form of an algebraic multigrid solver/preconditioner is enriched with the information obtained from relevant algebraic subspaces to enhance the robustness and to extend the range of applicability of algebraic multigrid methods. The founding idea of adaptive smoothed aggregation multigrid is that of retrieving information on the subspace where the smoother is less effective by solving the homogeneous equation associated to the original problem. The solution of the homogeneous equation provides an approximation for the “coarse” components of the error, which are then addressed by selecting the coarser correction spaces precisely with the goal of reducing those error components on which the smoother has proven to be not very effective. An additional advantage of adaptive multigrid methods is the possibility of incorporating some known information coming, for example, from some recycling strategy in the definition of the hierarchy of coarse spaces and of the related prolongation/restriction operators. This feature could be the starting point for an application of adaptive multigrid methods to the solution of sequences of linear systems, coupling these methods with some recycling technique to retain the relevant information coming from the solution of the previous linear systems. Such possibility, however, is conceptually quite different from the approaches used in standard augmentation/deflation preconditioners for Krylov space methods and is left for future investigation.

Finally, in the work of Tang et al.,³² it is shown that for symmetric positive definite systems, deflation, abstract balancing, and two-level multigrid preconditioners are comparable in terms of work per iteration and convergence properties. In the following, we consider only standard deflation preconditioners.

We rewrite the k th iteration as

$$\mathbf{x}_k = \mathbf{x}_0 + S_k (C_k^T (I - AV_s E_s^{-1} V_s^T) A S_k)^{-1} C_k^T (I - AV_s E_s^{-1} V_s^T) \mathbf{r}_0. \quad (19)$$

If the initial guess is built as for the multiple subspace correction case, namely, $\mathbf{x}_0 = V_s E_s^{-1} V_s^T \mathbf{b}$, then Equation (19) becomes

$$\mathbf{x}_k = V_s E_s^{-1} V_s^T \mathbf{b} + S_k (C_k^T (I - AV_s E_s^{-1} V_s^T) A S_k)^{-1} C_k^T (I - AV_s E_s^{-1} V_s^T) \mathbf{b}, \quad (20)$$

where it can be immediately seen that the difference between the deflative approach and the multiple subspace correction method consists in the form of the projector acting on the left of the system matrix. In this case, we see that the solution is again obtained through a projection of the initial residual, except that, now, it cannot be seen as an instance of the multiple subspace correction method. Indeed, Equation (19) clearly shows the projector $P_{\mathcal{V}_s, AV_s} = AV_s E_s V_s^T$ onto \mathcal{V}_s orthogonal to AV_s , but this time, the projection onto S_k is modified so that in the new operator, that is,

$$P_{AS_k, C_k, D} = AS_k (C_k^T (I - AV_s E_s^{-1} V_s^T) A S_k)^{-1} C_k^T,$$

the deflation space is explicitly removed from the range of A by means of the projector $(I - P_{\mathcal{V}_s, AV_s})A$, so that the “troublesome” eigenvalues are removed from the spectrum of the modified operator.

Regarding the convergence properties of the deflation preconditioner compared to the augmentation strategies, in the work of Eiermann et al.,³⁶ it is shown that for minimal residual approximations, if the augmentation/deflation space \mathcal{V}_s

spans an invariant subspace* of A , then

$$0 = \|P_{\mathcal{V}_s} \mathbf{r}_{\text{aug}}\| \leq \|P_{\mathcal{V}_s} \mathbf{r}_{\text{def}}\| \quad \text{and} \quad \|(I - P_{\mathcal{V}_s}) \mathbf{r}_{\text{aug}}\| \leq \|(I - P_{\mathcal{V}_s}) \mathbf{r}_{\text{def}}\|. \quad (21)$$

3.3 | A unified view of augmented and deflated methods

The augmented Krylov methods described in Section 4.3 and the deflation preconditioners of Section 3.2 can be interpreted as Krylov space methods with a special choice of the initial guess, followed by suitable modifications of the Krylov space, usually achieved by projections. We suppose that given a recycling space V_s , the standard initial guess is obtained as in Equation (32), namely, $\mathbf{x}_0 = V_s E_s^{-1} V_s^T \mathbf{b}$. Writing the linear system's solution as the sum of the initial guess \mathbf{x}_0 and a correction \mathbf{x}_c , $\mathbf{x} = \mathbf{x}_0 + \mathbf{x}_c$, we get the equation for the correction as follows:

$$A \mathbf{x}_c = \mathbf{r}_0 = (I - P_{\mathcal{V}_s, AV_s}) \mathbf{b}. \quad (22)$$

Solving Equation (22) with a Krylov method leads to the search space

$$S_k = \mathcal{K}_k(A, (I - P_{\mathcal{V}_s, AV_s}) \mathbf{b}). \quad (23)$$

However, the spectral information contained in \mathbf{x}_0 can be removed from the correction Equation (22), suggesting the use of a different Krylov space, that is,

$$S_k = \mathcal{K}_k(P_i A, P_j (I - P_{\mathcal{V}_s, AV_s}) \mathbf{b}), \quad (24)$$

instead of the one of Equation (23). In Equation (24), the operators P_i and P_j are projectors whose purpose is that of removing the information contained in the recycling space V_s from the correction equation (22).

Some possible choices for P_i are

- $P_{\mathcal{V}_s} := I - V_s V_s^T$: an orthogonal projector onto \mathcal{V}_s ;
- $P_{\mathcal{V}_s, AV_s} := I - AV_s E_s^{-1} V_s^T$: an oblique projector onto V_s , orthogonal to AV_s ; and
- $P_{AV_s, AV_s} := I - AV_s (V_s^T A^T AV_s)^{-1} V_s^T A^T$: an orthogonal projector onto AV_s ,

whereas for P_j , it is possible to choose $P_j = I$, which corresponds to augmentation methods, or $P_j = P_i$ for deflation methods.

An advantage of the projector $P_{\mathcal{V}_s}$ is that its application to a vector requires fewer operations than $P_{\mathcal{V}_s, AV_s}$. However, $P_{\mathcal{V}_s}$ has maximum rank $n - s$ only if $V_s^T V_s = I_s$, whereas $P_{\mathcal{V}_s, AV_s}$ has maximum rank whenever the columns of V_s are linearly independent. A more general variant of $P_{\mathcal{V}_s}$ is

$$P_{\mathcal{V}_s} := I - V_s (V_s^T V_s)^{-1} V_s^T, \quad (25)$$

which works even if V_s is not orthogonal.

Regarding the third choice, P_{AV_s, AV_s} , we remark that the product $A^T A$ may increase the condition number of the problem, as is well known for methods such as the Conjugate Gradient on the Normal Equations (CGNE) or the Conjugate Gradient for Least Squares (CGLS).²⁶

In the following, we will consider the six recycling methods in Table 1. All of these methods can be implemented by constructing a (left) preconditioner M and by considering a specific starting vector \mathbf{x}_0 . For example, the *deflated oblique* method is obtained by considering $\mathbf{0}$ as the starting vector and $M = (I - AV_s E_s^{-1} V_s^T)$ as the (left) preconditioner, whereas the *deflated orthogonal* method is obtained by using the (left) preconditioner $M = (I - V_s V_s^T)$ and $\mathbf{x}_0 = AV_s E_s^{-1} V_s^T \mathbf{b}$.

Lastly, we remark that the *augmented oblique* and *deflated oblique* methods are equivalent in the nonpreconditioned case, since the initial vector for the latter method would be

$$(I - AV_s E_s^{-1} V_s^T)^2 \mathbf{b} = (I - AV_s E_s^{-1} V_s^T) \mathbf{b} = \mathbf{r}_0. \quad (26)$$

However, since in general, the presence of a preconditioner would prevent the first equality in Equation (26) from holding true, we consider the two cases as independent.

Regarding the methods in Table 1, we remark that the *deflated oblique* method coincides with the class of preconditioners studied in the work of Nabben and Vuik³⁷ if V contains an approximation of some eigenvectors of A . The *deflated*

*That is to say, $AV_s \subseteq \mathcal{V}_s$.

TABLE 1 Krylov space used by the recycling methods considered in this work

Acceleration method	Krylov operator	Krylov initial vector
Augmented orthogonal	$(I - V_s V_s^T) A$	\mathbf{r}_0
Augmented oblique	$(I - AV_s E_s^{-1} V_s^T) A$	\mathbf{r}_0
Augmented LS	$(I - AV_s (V_s^T A^T AV_s)^{-1} V_s^T A^T) A$	\mathbf{r}_0
Deflated orthogonal	$(I - V_s V_s^T) A$	$(I - V_s V_s^T) \mathbf{r}_0$
Deflated oblique	$(I - AV_s E_s^{-1} V_s^T) A$	$(I - AV_s E_s^{-1} V_s^T) \mathbf{r}_0$
Deflated LS	$(I - AV_s (V_s^T A^T AV_s)^{-1} V_s^T A^T) A$	$(I - AV_s (V_s^T A^T AV_s)^{-1} V_s^T A^T) \mathbf{r}_0$

Note. LS = least squares.

least squares (LS) method is equivalent to the inner iteration of the Generalized Conjugate-Residual with inner Orthogonalization (GCRO) method introduced in the work of de Sturler.³⁸ The other methods in Table 1 are, to the best of our knowledge, new. Finally, we mention that alternative, but not equivalent, classifications of augmented and deflated methods are available in the works of Gaul et al.³⁹ and Gutknecht.⁴⁰

4 | ACCELERATION BY INITIAL GUESS SELECTION

The discussion so far has been quite general, but in the context of sequences of linear systems such as those of Equation (8), arising, for example, from the discretization of a time-dependent PDE, it is reasonable to expect that some information could be extracted from the solutions of previous linear systems and successfully used to accelerate the solution of successive linear systems.

For time-dependent problems, it is often standard to use an extrapolation based on a few preceding steps as the initial guess for the current linear system, on the basis that if the time step is sufficiently small, the solutions will not differ too much from each other.

Following this basic idea, we go one step further and explore a strategy based on *projection* rather than extrapolation. These methods can be interpreted as recycling methods, if the projection step is done coherently with the Krylov solver used in the solution of the linear system.

Recycling of Krylov subspaces for sequences of linear systems with changing matrices and right-hand sides was proposed in the work of Parks et al.,¹⁶ where a generalization of the Truncated GCRO (GCROT)¹³ and GMRES with Deflated Restarting (GMRES-DR)⁴¹ methods, as well as a new GCRO with Deflated Restart (GCRO-DR) method, is applied to Hermitian and non-Hermitian problems. The effectiveness of the recycling methods mentioned above was proven in practical applications in the works of Wang et al.,¹⁸ Mello et al.,¹⁹ and Kilmer and de Sturler.²⁰ Recycling biconjugate gradient (BiCG) and BiCGStab methods was developed in the works of Ahuja et al.,^{42,43} respectively, where the recycling techniques play a key role in the setup of very efficient model order reduction methods. It is also possible to combine deflation with augmentation, as shown in the work of Gaul et al.,³⁹ but this combined approach is not considered here.

The selection of a good initial guess for a Krylov space method has been studied especially for symmetric positive definite matrices not depending on time, with multiple right-hand sides.^{44,45}

For the model problem of Equation (8), we are interested in providing good initial guesses for sequences of nonsymmetric linear systems where both the matrix and the right-hand side change at each time step (but we suppose that the dimension of the linear systems remains constant). Furthermore, in later sections, we develop on this idea by selecting only a subset of previous solutions.

4.1 | Extrapolation

Extrapolation methods are used to construct an initial guess for the iterative solver by Lagrangian interpolation from previous solutions. An initial guess $\mathbf{x}_0^i = \mathbf{x}_0(t_i)$ for the current linear system $A_i \mathbf{x}^i = \mathbf{b}^i$ can be derived imposing that the \mathbf{x}_0^i lie on the family of planes passing through \mathbf{x}^{i-1} and \mathbf{x}^{i-2} . The expression for \mathbf{x}_0^i can be obtained formally by introducing a pseudo time step h and requiring that the function

$$\mathbf{f}(\tau) = c_1 \mathbf{x}^{i-2} \tau + c_2 \mathbf{x}^{i-1} \tau + c_3$$

satisfies $\mathbf{f}(-2h) = \mathbf{x}^{i-2}$, $\mathbf{f}(-h) = \mathbf{x}^{i-1}$, and then, we have

$$\mathbf{x}_0^i = \mathbf{f}(0) = 2\mathbf{x}^{i-1} - \mathbf{x}^{i-2}. \quad (27)$$

Similar expressions can be obtained using quadratic interpolation formulas

$$\mathbf{x}_0^i = 3\mathbf{x}^{i-1} - 3\mathbf{x}^{i-2} + \mathbf{x}^{i-3} \quad (28)$$

and cubic interpolation formulas

$$\mathbf{x}_0^i = 4\mathbf{x}^{i-1} - 6\mathbf{x}^{i-2} + 4\mathbf{x}^{i-3} - \mathbf{x}^{i-4}. \quad (29)$$

Usually, there is no advantage in further increasing the order of the interpolation polynomial, due to instabilities of equispaced interpolation for high-order polynomials. Furthermore, the desired initial guess lies outside the interpolation interval, hence the name *extrapolation*.

The full expression of the k th iterate for the linear extrapolation method becomes

$$\mathbf{x}_k^i = 2\mathbf{x}^{i-1} - \mathbf{x}^{i-2} + \mathbf{S}_k (\mathbf{C}_k^T \mathbf{A}_i \mathbf{S}_k)^{-1} \mathbf{C}_k^T (\mathbf{b}^i - 2\mathbf{A}_i \mathbf{x}^{i-1} + \mathbf{A}_i \mathbf{x}^{i-2}), \quad (30)$$

showing that the projector $\mathbf{P}_{\mathbf{A}\mathbf{S}_k, \mathbf{C}_k}$ is not modified by the extrapolation method, whose only effect is to perturb the right-hand side so that the special form of the initial guess is taken into account. In practice, nothing changes from the standard case of Equation (11).

The main advantages of this method are its simplicity and low storage and work requirements. To the best of our knowledge, there exist no theoretical estimates on the effectiveness of extrapolation methods in reducing the initial residual.

4.2 | Simple projection

The idea of constructing a good initial guess through the extrapolation method of Section 4.1 can be greatly improved if the interpolation on a set of previous solutions is replaced by projection. This approach was first introduced in the work of Ali and Sadkane²¹ for time-dependent linear systems with a fixed matrix. The analysis we carry over differs from the original work,²¹ leading us to introduce a new initial guess selection based on an instance of the multiple subspace correction method.⁴⁶

Suppose that the s previous solutions $\{\mathbf{x}^{i-j}\}_{j=1}^s$ are orthonormalized, for example, through a modified Gram–Schmidt algorithm and stored columnwise in the matrix \mathbf{V}_s . Then, we can look for an initial guess in the range of \mathbf{V}_s , which we name $\mathcal{V}_s = \text{Im}\mathbf{V}_s$, $\mathbf{x}_0^k \in \mathcal{V}_s$, by requiring that the residual is orthogonal to the column space of \mathbf{V}_s , that is,

$$\mathbf{x}_0 = \mathbf{V}_s \mathbf{t}_s, \quad \mathbf{V}_s^T \mathbf{r}_0 = 0 \quad \implies \quad \mathbf{V}_s^T \mathbf{A} \mathbf{V}_s \mathbf{t}_s = \mathbf{V}_s^T \mathbf{b}, \quad (31)$$

so that

$$\mathbf{x}_0 = \mathbf{V}_s \mathbf{t}_s = \mathbf{V}_s (\mathbf{V}_s^T \mathbf{A} \mathbf{V}_s)^{-1} \mathbf{V}_s^T \mathbf{b} = \mathbf{V}_s \mathbf{E}_s^{-1} \mathbf{V}_s^T \mathbf{b}. \quad (32)$$

A standard iterative method of choice can then be applied to the full linear system using the initial guess computed as above. In this case, the iterations satisfy

$$\mathbf{x}_k = \mathbf{V}_s \mathbf{E}_s^{-1} \mathbf{V}_s^T \mathbf{b} + \mathbf{S}_k (\mathbf{C}_k^T \mathbf{A} \mathbf{S}_k)^{-1} \mathbf{C}_k^T (\mathbf{I} - \mathbf{A} \mathbf{V}_s \mathbf{E}_s^{-1} \mathbf{V}_s^T) \mathbf{b}. \quad (33)$$

From Equation (33), we see that the solution at the k th iterate is obtained as a projection of the right-hand side on a suitable sequence of spaces.

Introducing the projector on $\mathcal{V}_s = \text{Im}\mathbf{V}_s$, orthogonal to $\mathcal{A}\mathcal{V}_s$, that is, $\mathbf{P}_{\mathcal{V}_s, \mathcal{A}\mathcal{V}_s} := \mathbf{A} \mathbf{V}_s \mathbf{E}_s^{-1} \mathbf{V}_s^T$, iterate (33) can be rewritten as

$$\mathbf{x}_k = \mathbf{A}^{-1} (\mathbf{P}_{\mathcal{V}_s, \mathcal{A}\mathcal{V}_s} + \mathbf{P}_{\mathbf{A}\mathbf{S}_k, \mathbf{C}_k} (\mathbf{I} - \mathbf{P}_{\mathcal{V}_s, \mathcal{A}\mathcal{V}_s})) \mathbf{b}, \quad (34)$$

where it is clear that the correction of this method is based on a sequence of two projectors. However, we cannot conclude that this method is an instance of the multiple subspace correction method because, in general, \mathcal{V}_s is not orthogonal to the Krylov space $\mathbf{A}\mathbf{S}_k$, that is, $\mathcal{V}_s \cap \mathbf{A}\mathbf{S}_k \neq \{\mathbf{0}\}$.

4.3 | Multiple subspace correction

To exploit the best approximation properties of the multiple subspace correction method, the spaces \mathcal{V}_s and S_k should be orthogonal, and this can be achieved by replacing matrix A in the linear systems (8) with $\tilde{A} := (I - P_{\mathcal{V}_s})A$, where $P_{\mathcal{V}_s} = V_s V_s^T$ is the orthogonal projector onto V_s .

In this case, the iteration satisfies

$$\mathbf{x}_k^i = V_s E_s^{-1} V_s^T \mathbf{b} + S_k (C_k^T (I - V_s V_s^T) A S_k)^{-1} C_k^T (I - A V_s E_s^{-1} V_s^T) \mathbf{b}, \quad (35)$$

a slight modification of the operator in Equation (33). Also, the projector expression (34) holds with the difference that in this case, $\tilde{A} S_k$ and \mathcal{V}_s are orthogonal. The higher computational costs due to the extra matrix–vector products required by the application of $I - P_{\mathcal{V}_s}$ are hopefully recovered by the best approximation estimate on the residual (following the lines of Eiermann et al.³⁶), that is,

$$\left\| \mathbf{y} - (P_{\mathcal{V}_s, A \mathcal{V}_s} \mathbf{y} + P_{A S_k, C_k} (I - P_{\mathcal{V}_s}) \mathbf{y}) \right\| = \min_{\mathbf{t} \in \mathcal{V}_s + S_k} \|\mathbf{y} - \mathbf{t}\|. \quad (36)$$

This property can be proved by noting that

$$P_{\mathcal{V}_s, A \mathcal{V}_s} \mathbf{x} + P_{A S_k, C_k} (I - P_{\mathcal{V}_s}) \mathbf{x} = P_{\mathcal{V}_s, A \mathcal{V}_s} \mathbf{x} + P_{A S_k, C_k} \mathbf{x} = P_{\mathcal{V}_s + S_k} \mathbf{x} \quad \forall \mathbf{x} \in \mathbb{R}^n, \quad (37)$$

where $P_{S_k} P_{\mathcal{V}_s} = 0$ was used since \mathcal{V}_s and S_k are orthogonal.

However, in many cases, the spaces \mathcal{V}_s and S_k can become relatively large, and the advantages of having larger approximation spaces may be invalidated by excessive computational and storage costs. To mitigate this effect, restarting and truncation methods have been introduced, which means that the approximation spaces are not allowed to exceed a fixed dimension.

To conclude, we summarize the main advantages of using multiple projections over extrapolation methods as follows.

- The stability issues of equispaced interpolation are removed; hence, it can be worthwhile to keep a larger number of previous solutions.
- For the projection method (Galerkin), best approximation-type estimates hold.
- The vector space for the projection step can be obtained in any way, for instance, the previous solutions can be orthonormalized and augmented with any other kind of basis functions, regardless of their origin.

This last point is worth noting, since, in general, we are not interested in the previous solutions *per se*, but only in how representative they are as solutions to the current step. In the next section, we claim that a good way to keep as much information as possible from previous solutions and, at the same time, maintain the size of the projection space to be small is to use the SVD of a fixed number of preceding solutions and only retain those modes that contain information related to the singular values of largest or smallest magnitude. This strategy is also very effective when combined with deflation preconditioners, as shown in the next section.

5 | CONSTRUCTION OF THE RECYCLING SPACES

In the previous sections, we implied more or less explicitly that the augmentation spaces should be built by performing some orthonormalization procedure on a fixed number of previous solutions. In fact, there is complete freedom in the choice of the augmentation spaces, and the only obstructions are considerations on the approximation, on the algebraic stability, and on the effectiveness of the resulting algorithm. In Section 5.1, we shortly review some common methods for the selection of the augmentation spaces and the restarting techniques used when their dimension becomes too large, whereas in Section 5.2, we propose an alternative strategy based on the construction of the enriching space using SVD. This idea has been traditionally used for the construction of efficient model order reduction methods,⁴⁷ but its application to the construction of recycling spaces could be convenient over the computation of Ritz vectors, at least in the case with time-dependent matrices.

5.1 | Ritz and harmonic Ritz pairs

The discussion in Section 3.2 suggests constructing the recycling spaces by computing the s eigenvalues of A with the smallest magnitude and storing the respective eigenvectors columnwise in matrix V_s . However, when A is too large to

resort to algorithms such as QR subspace iterations for the approximation of the eigenvalues of A , a few interesting eigenpairs can be approximated using the information generated by a GMRES cycle.

This approach was introduced by Morgan⁴⁸; see also another work of Morgan⁴¹ where it is shown that keeping approximate eigenvectors corresponding to small eigenvalues can greatly improve the convergence rate of the GMRES method after restart. Deflative preconditioners based on approximate eigenvalues of the linear operators are also discussed in the work of Kharchenko and Yu. Yeremin⁴⁹ for the restarted GMRES method. One further step in this direction was done in the work of Giraud et al.,⁵⁰ with the introduction of a flexible GMRES with a deflation space built on some approximate eigenvectors, so that a different preconditioner could be used for each GMRES iteration.

Specifically, if $V_k \in \mathbb{R}^{n \times k}$ is a rectangular matrix spanning the subspace $\mathcal{V}_s \subset \mathbb{R}^n$, the couple $(\vartheta, \mathbf{y}) \in \mathbb{C} \times V_s$ is called a *Ritz pair*^{51,52} of A with respect to the space \mathcal{V}_s if

$$\mathbf{v}^H(A\mathbf{y} - \vartheta\mathbf{y}) = 0 \quad \forall \mathbf{v} \in \mathcal{V}_s. \quad (38)$$

Equivalently, (ϑ, \mathbf{y}) is a Ritz pair for A with respect to \mathcal{V}_s if $(A - \vartheta I)\mathbf{y} \in \mathcal{V}_s^\perp$. In this case, ϑ and \mathbf{y} are respectively called the *Ritz value* and the *Ritz vector* for A . The Ritz pairs can be computed directly by restricting A to \mathcal{V}_s and then computing the eigenvalues of the resulting Hessenberg matrix $H_s = V_s^T A V_s$, that is,

$$H_s \mathbf{z}_i - \vartheta_i V_s^H V_s \mathbf{z}_i = \mathbf{0} \quad \mathbf{y}_i = V_s \mathbf{z}_i. \quad (39)$$

As the dimension of \mathcal{V}_s is increased, the Ritz values converge (see, e.g., chapter 34 in the work of Trefethen and Bau⁵³ and chapter 7 in the work of Demmel⁵⁴) to the external eigenvalues of A .

Alternatively, Ritz pairs can be replaced or enhanced through the computation of *harmonic Ritz pairs*,^{51,55} defined as the solutions to the Petrov–Galerkin projection, that is,

$$V_s^H A^H A V_s \mathbf{z}_i - \vartheta_i V_s^H A^H V_s \mathbf{z}_i = \mathbf{0} \quad \mathbf{y}_i = V_s \mathbf{z}_i. \quad (40)$$

The harmonic Ritz values converge to the inner eigenvalues of A with minimum distance from the origin of the complex plane. Indeed, introducing the matrix $W_s = A V_s$, problem (40) can be rewritten as

$$W_s^H W_s \mathbf{z}_i - \vartheta_i W_s^H A^{-1} W_s \mathbf{z}_i = \mathbf{0} \quad \mathbf{y}_i = W_s \mathbf{z}_i, \quad (41)$$

showing that the harmonic Ritz values of A with respect to \mathcal{V}_s coincide with the Ritz values of A^{-1} with respect to $\mathcal{W}_s = A \mathcal{V}_s$.

5.2 | Simplified SVD truncation

In this work, we explore an alternative enriching strategy that aims at retaining the relevant information contained in the previous solution steps. In particular, we collect a number of previous solution steps, perform SVD, and keep a fixed number of singular vectors associated with the higher singular values that are then used as basis for the enriching subspace. A similar strategy was used in a different context, namely, model order reduction of parameterized matrix-valued linear systems, in the work of Kressner and Tobler.⁴⁷

The m previous solutions \mathbf{x}_m are stored in the matrix $X_m \in \mathbb{R}^{n \times m}$. By computing an SVD of X_m , we keep the s left-singular vectors with the largest (or smallest) singular values.

If we let $X_m = \Psi_m \Sigma_m \Phi_m^T$ be the SVD of X_m and Ψ_s the matrix whose columns are the s largest left-singular vectors, then the Schmidt–Eckart–Young theorem guarantees the following optimality property of the SVD: If Y is a given matrix, the best approximation of Y by a matrix of rank s is given by

$$Y_s := \sum_{i=1}^s \sigma_i \psi_i \phi_i^T, \quad (42)$$

where $Y = \Psi \Sigma \Phi^T$ is the SVD of Y , σ_i is the entry in the i th row of Σ , and ψ_i and ϕ_i are the columns of Ψ and Φ , respectively. The optimality property given by the Schmidt–Eckart–Young theorem can be written explicitly as

$$Y_s = \arg \min_{\substack{\Xi \in \mathbb{R}^{n \times s} \\ \text{rank } \Xi = s}} \|Y - \Xi\|_F. \quad (43)$$

To clarify how this alternative truncation strategy could be implemented in a solver for a sequence of related linear systems, we refer to Algorithm 1.

Algorithm 1. The SVD-based truncation algorithm described in Section 5.2

Input : Three nonnegative integers m, s, ℓ ; m is the number of retained previous solutions, $s \leq m$ is the dimension of the recycling space, $\ell \geq 1$ is the interval measured in time steps between two consecutive SVDs.

```

1 for  $i = 1, \dots$  do
2   solve  $A_i \mathbf{x}_i = \mathbf{b}_i$  with any recycling scheme of Section 3.3;
3   store the solution  $\mathbf{x}_i$  in the matrix  $X_m \in \mathbb{R}^{n \times m}$ :
                                     
$$X_m = [\mathbf{x}_{i-m}, \dots, \mathbf{x}_i].$$

   If necessary, remove the  $i - m - 1$ th row from  $X_m$ ;
4   if  $(i \bmod \ell) = 0$  then
5     compute the SVD decomposition of  $X_m$ :  $X_m = \Psi_m \Sigma_m \Phi_m^T$ ;
6     update the recycling space  $V_s \in \mathbb{R}^{n \times s}$  by copying the first  $s$  largest left-singular modes of  $X_m$ :
                                     
$$V_s = \Psi_m[1 : s];$$

7   end
8 end

```

Note that some of the recycling algorithms presented in Section 3.3 require the solution of a linear system involving the matrix $E_s = V_s^T A V_s$ as in Equation (31). The reader should be aware that, in principle, the matrix E_s may be singular when S_k and C_k are the same space.

However, for the choices of V_s used in this work, we never had to face such difficulties; hence, we do not take any preventive measure against this breakdown possibility.

6 | NUMERICAL EXPERIMENTS

In this section, we compare the accelerating methods proposed in Sections 3.3 and 4 on a standard test case for nonsymmetric matrices: a parabolic scalar convection–diffusion equation discretized with finite elements.

6.1 | Scalar convection–diffusion

This benchmark consists of a scalar parabolic convection–diffusion equation on a time interval $[0, T]$ and a domain $\Omega \subset \mathbb{R}^2$, which, in weak form, reads⁵⁶: Find $u \in L^2([0, T], H_0^1(\Omega)) \cap C^0((0, T), L^2(\Omega))$ such that

$$\begin{cases} (v, \partial_t u) + \nu(\nabla v, \nabla u) - (v, \mathbf{u} \mathbf{b} \cdot \nabla u) = (v, f) & \forall v \in H_0^1(\Omega) \\ u(t = 0) = u_0, & u_0 \in L^2(\Omega), \end{cases} \quad (44)$$

where ν is the diffusion coefficient, $\mathbf{b} : \Omega \rightarrow \mathbb{R}^2$ is a given solenoidal vector field, and $f : (0, T) \times \Omega \rightarrow \mathbb{R}$ is a possibly time-dependent given forcing term. Here, we choose Ω to be the unit square with coordinates $(x, y) \in [0, 1]^2$ and $\mathbf{b} = (-\sin(\pi x) \cos(\pi y), \cos(\pi x) \sin(\pi y))$.

Following the work of LaBryer et al.,⁵⁷ we model f as acting randomly on the low-frequency modes of the Laplace operator, namely,

$$f = \frac{C}{2} \sum_{j=1}^{16} c_j \exp\left(-\frac{j^2}{20}\right) \sin(2j\pi x) \sin(2j\pi y), \quad (45)$$

where $C = 0.1$, $c_1 = 1$ and $\{c_j\}_{j=2}^{16} \subset [-1, 1]$ are random coefficients sampled at every time step. The rationale of this choice is to avoid the solution settling to a stationary configuration, so that the turbulent behavior of the one-dimensional Burgers equation can be simulated in this more complex two-dimensional setting.

TABLE 2 Average number of iterations (av) and standard deviation of the iteration count (dev) for the extrapolation (ext) and projection (proj) av initial guess, as a function of the recycling space dimension (s). The results are for the case $\nu = 10^{-1}$

s	ext av	ext stddev	proj av	proj stddev
2	65.31	3.86	62.38	3.56
3	67.57	4.23	61.17	3.39
4	70.17	4.20	60.72	3.06

Treating Equation (44) with a finite difference method in time (in this case, an implicit Euler method) and with a first-order finite element method, we get the sequence of discrete problems: For $n \in [1, N_T]$, find $u_h^n \in V_h$ such that

$$\begin{cases} \frac{1}{\Delta t} (v, u_h^n - u_h^{n-1}) + \nu (\nabla v, \nabla u_h^n) + (v, u_h^{n-1} P_{V_h} \mathbf{b} \cdot \nabla u_h^n) = (v, P_{V_h} f) & \forall v \in V_h \\ u_h^0 = P_{V_h} u_0, \end{cases} \quad (46)$$

where Δt is the time step for the finite difference discretization, and $N_T = T/\Delta t$ is the number of time steps. For the finite element space V_h , we adopt first-order Lagrange elements on a grid formed by 64×64 uniform square elements, as provided by the `deal.II` library.⁵⁸

We design three test cases, with $\nu = 10^{-1}$, $\nu = 10^{-2}$, and $\nu = 10^{-3}$. Values of ν lower than 10^{-3} are not stable in the grid described above. For each test case, we run a total of 1,000 time steps of width $\Delta t = 0.5$, with a zero initial value and with a symmetric successive over-relaxation (SSOR) preconditioner. The iterative solver is halted when the 2-norm of the residual relative to the 2-norm of the right-hand side is below a tolerance set to 10^{-8} .

In the first test, we compare the extrapolation initial guess described in Section 4.1 with the projection initial guess of Section 4.2. This test is performed with $\nu = 10^{-1}$, and the results are reported in Table 2, where it is possible to see how the projection-based initial guess reduces the number of GMRES iterations to converge. Furthermore, increasing the dimension of the recycling space does not increase the iteration count for the projection method, whereas it clearly worsens the convergence rate of the extrapolation method.

After verifying the expected behavior for the two prediction methods, we compare the four recycling solvers in Table 1.

To get some insight on the influence of the recycling parameters on the convergence properties of the iterative methods, we set up a campaign of simulations as follows. We consider the same problem described above for the case $\nu = 10^{-2}$ and run the various recycling algorithms, letting the dimension of the Krylov space[†] and the number of saved solutions sweep between 10 and 60 (in steps of 2). The dimension of the recycling space is chosen equal to the GMRES restart. Each campaign of simulations requires $26^2 = 676$ runs.

At first, we run a standard solver without recycling, using only an SSOR preconditioner. From the results, shown in Figure 1, it can be seen that increasing the dimension of the Krylov space (higher values of the restart parameter) yields convergence in fewer iterations, although there are counterexamples to this behavior as shown in the work of Embree.¹⁰ In this case, there is no variation of the iteration number in the horizontal direction, since the recycling method is not set up.

Then, we run the same campaign of simulations for the *augmented orthogonal* and *augmented oblique* recycling methods, corresponding to the first two rows of Table 1. As in the previous case, the SSOR preconditioner is used within the recycling method, and the SVD selection is performed at every time step. As reported in Figure 2, in most of the runs, the augmentation methods are reducing the iteration count with respect to the unrecycled case of Figure 1. In both cases, it is difficult to figure out a clear pattern; however, in average, we observe that increasing the dimension of the recycling space reduces the iteration count.

The second set of runs involves the *deflated orthogonal* and *deflated oblique* methods, corresponding to the fourth and fifth rows of Table 1. As for the previous two cases, the numerical experiments take advantage of the SSOR preconditioner, and SVD selection is performed at the end of each time step. The results of this test are reported in Figure 3. It can be seen that the *augmented orthogonal* and *deflated orthogonal* methods lead to visually similar convergence patterns, corresponding to Figures 2 (left) and 3 (left). However, the augmentation method appears to be superior, sparing a few iterations for almost all the cases. By comparing Figures 2 (right) and 3 (right), we see that the oblique augmentation

[†]Equal to the GMRES restart value.

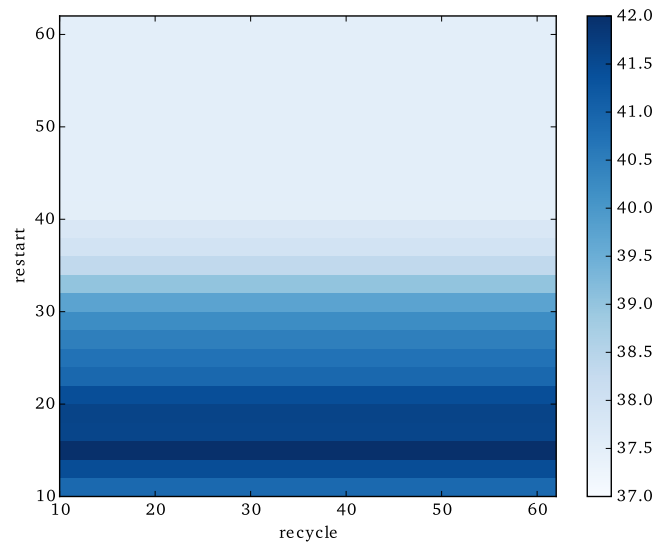


FIGURE 1 Average number of iterations to convergence without applying the recycling methods. Note that, in this case, the horizontal axis does not denote a true variable, since the recycling is not present. However, we keep the two-dimensional map style for consistency with the next Figures

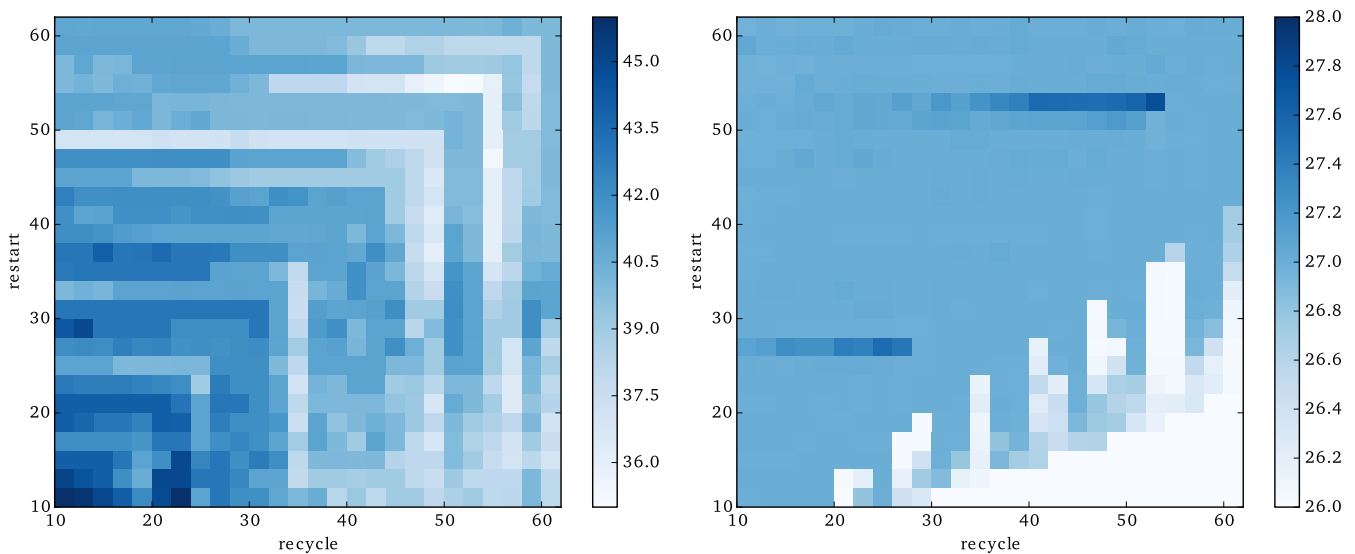


FIGURE 2 Average number of iterations to convergence for the *augmented orthogonal* (left) and *augmented oblique* (right) methods of Table 1. In both Figures, singular value decomposition selection is performed at the end of each time step, and the symmetric successive over-relaxation preconditioner is used

method performs much better than its deflated counterpart, since, in this case, the improvement amounts to almost halving the iteration count. In the map on the right of Figure 3, it is possible to recognize an interesting pattern relating faster convergence to higher values of the restart parameter.

Next, we compare the performance of the two *orthogonal LS* acceleration methods (rows 3 and 6 of Table 1). We have not been able to make the *augmented LS* method converge; hence, in Figure 4, we show only the results for the *deflated LS* method. Overall, it seems that despite the higher computational cost due to the higher number of matrix–vector multiplications, this strategy is not paying back with an increased convergence rate.

As a last test, we compare the two deflative preconditioners with different choices of the recycling subspace. The first choice consists of the SVD modes associated with higher singular values, and the second choice conversely consists of the SVD modes with smaller singular values. The comparison for the two deflative preconditioners is available in Figure 5. For both the deflative preconditioners, we see that the second choice for the recycling subspace introduces a clear diagonal pattern in the diagrams of Figure 5. However, while for the orthogonal deflation preconditioner, the effect of the

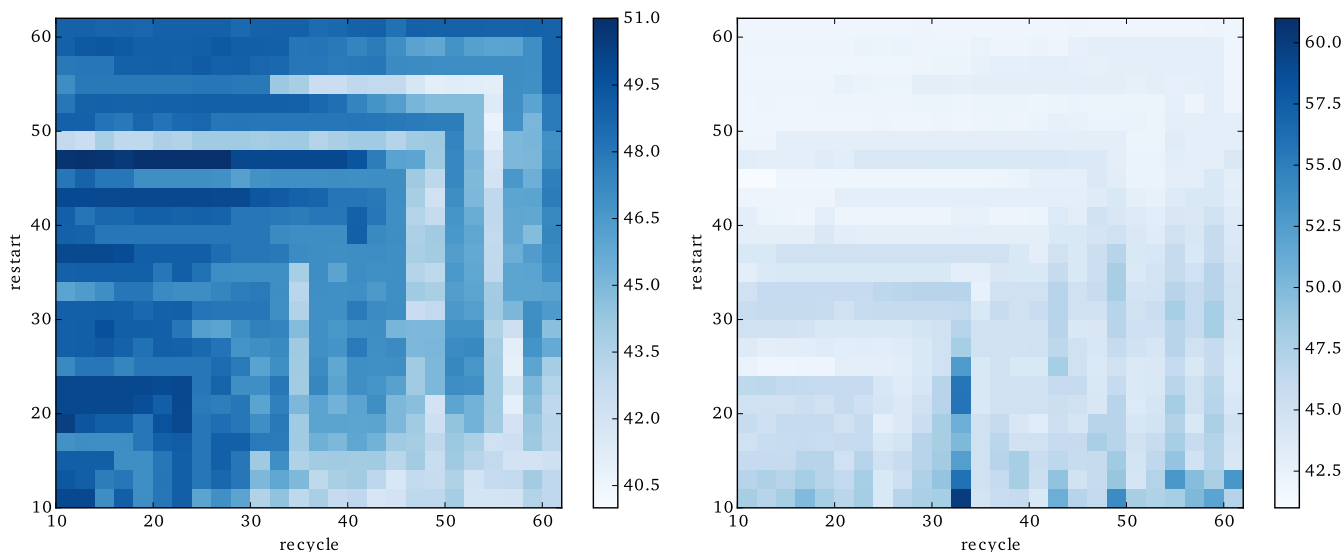


FIGURE 3 Average number of iterations to convergence for the *deflated orthogonal* (left) and *deflated oblique* (right) methods of Table 1. In both Figures, singular value decomposition selection is performed at each time step, and the symmetric successive over-relaxation preconditioner is used

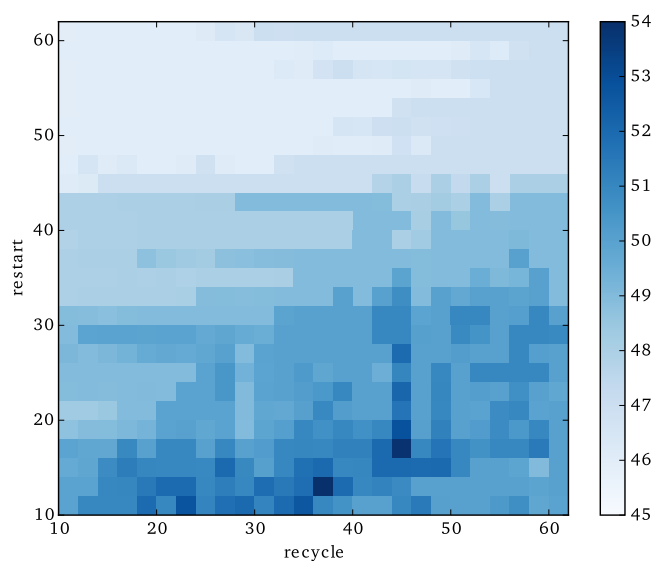


FIGURE 4 Average number of iterations to convergence for the *deflated least squares* (right) method in Table 1. Singular value decomposition selection is performed at each time step, and the symmetric successive over-relaxation preconditioner is used

“low-energy modes” is to clearly reduce the average iteration number for a large portion of the recycle–restart plane, for the oblique deflation preconditioner, it seems that there is little change in the lower-right half of the diagram, whereas on the upper-left half, the iteration count is clearly increasing. Consequently, we cannot draw clear conclusions regarding the optimal choice of SVD vectors.

In all the numerical experiments carried out so far, the methods achieving the lowest iteration count have taken advantage of the SVD for the construction of the recycling spaces. However, the SVD is an expensive computation, whose cost more than balances the time saved by the fewer iterations needed to converge. For this reason, we test the performance of the first two augmentation methods in Table 1, except that in this case, SVD is not performed at each time step, but only after a fixed number of time steps, and the GMRES restart is fixed to 30. The results of this set of simulations are available in Figure 6. The maps in Figure 6, although not directly comparable with the maps of the previous Figures, show clearly that having a larger recycling space in this case can help mitigate the effect of the fewer SVD selections carried out (and hence of the lower update frequency of the recycling spaces).

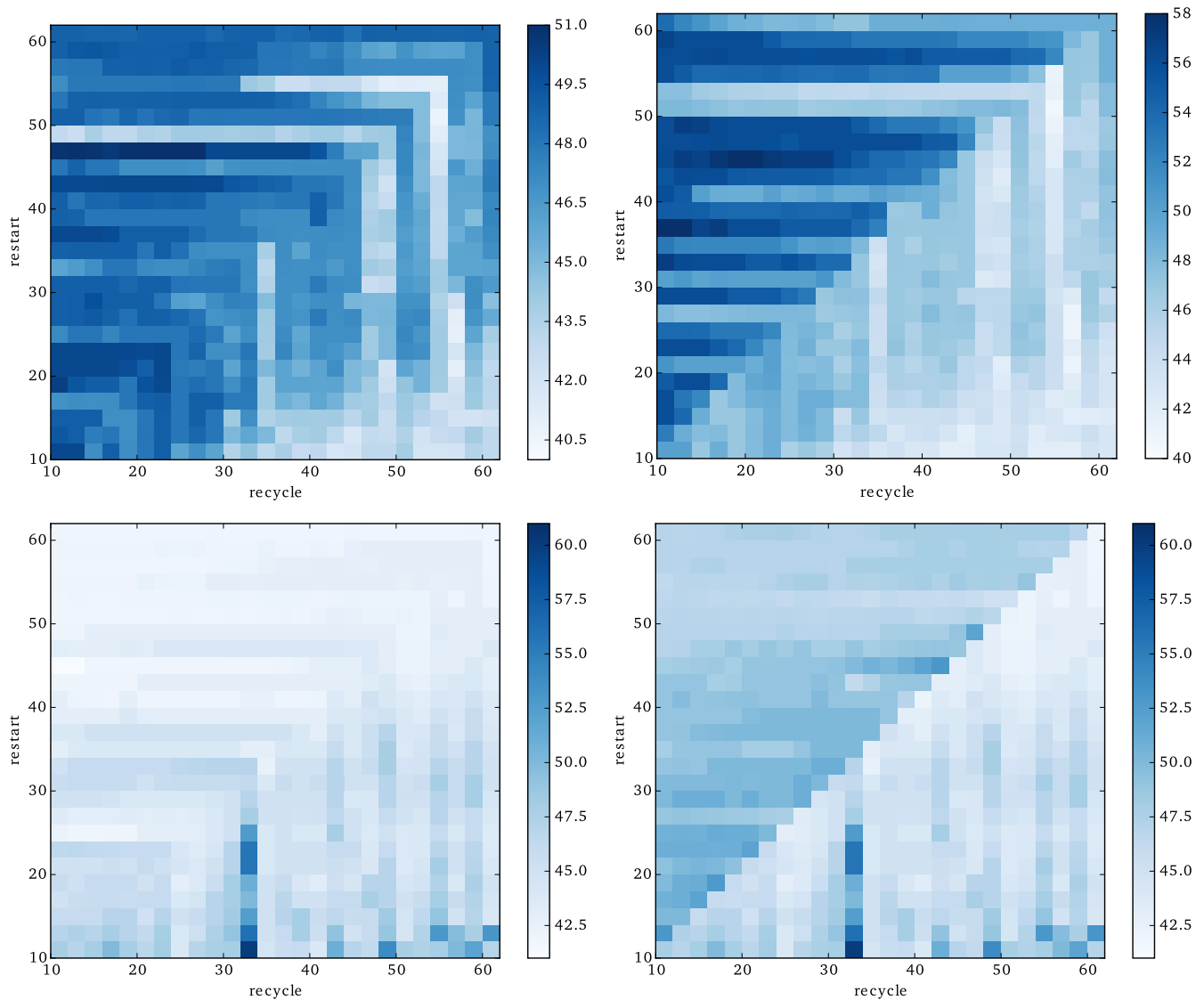


FIGURE 5 Average number of iterations to convergence for the *deflated orthogonal* (top row) and *deflated oblique* (bottom row) methods in Table 1. On the left column, the recycling subspace is spanned by the singular value decomposition (SVD) modes with higher singular values, whereas on the right column, the recycling subspace is formed by the proper orthogonal decomposition modes with smaller singular values. In every case, SVD selection is performed at each time step, and the symmetric successive over-relaxation preconditioner is used

6.2 | Efficiency of SVD-based preconditioners

Despite the beneficial effects of the aforementioned recycling strategies on the iteration count and on the stability of the iterative solver, it is unlikely that such methods will be of interest if they fail in reducing the time required by the full computation. For this reason, we give a back-of-the-envelope estimate on the opportunity to adopt recycling strategies.

Let m be the dimension of the original linear system, n be the number of GMRES iterations before restarting, b be the bandwidth of the full matrix, and k be the dimension of the recycling space. The operations required by the linear system solvers are then as follows.

Dense GMRES: The Arnoldi orthogonalization algorithm requires $O(mn^2)$ operations, and the matrix–vector multiplications require $O(m^2n)$ operations. However, in the typical cases that lead to dense matrices (e.g., single-domain spectral methods, boundary element method), fast matrix multiplication is often available, reducing the cost of a matrix–vector product to $O(m \log m)$ or even to $O(m)$.

Sparse GMRES: If the matrix is sparse, the cost of matrix–vector multiplication reduces to $O(mb)$, and then, the cost of each GMRES iteration reduces to $O(mb + mn)$.

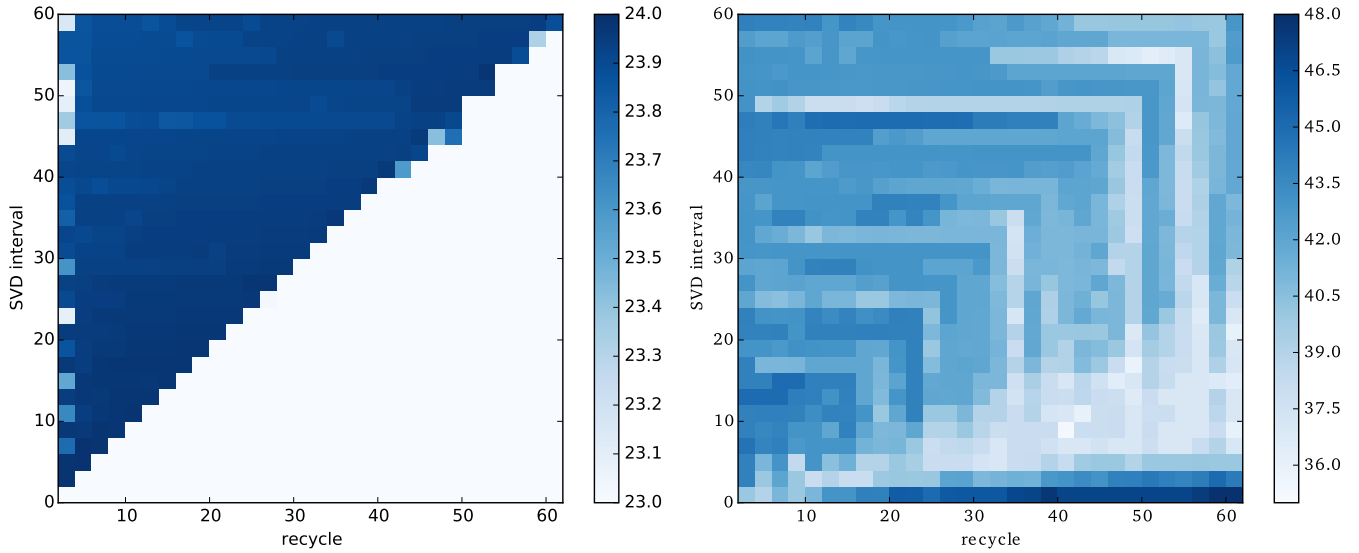


FIGURE 6 Average number of iterations to convergence for the *augmented orthogonal* (left) and *augmented oblique* (right) methods in Table 1. In this Figure, the generalized minimal residual restart is fixed to 30, and the dimension of the recycling space and the interval between two singular value decompositions (SVDs) are varied

Truncated SVD: If $Z \in \mathbb{R}^{m \times s}$ is the matrix where the s previous solutions are stored columnwise, then the computation of the first k singular vectors from Z requires $O(msk) + O(m^2k)$ operations⁴.

Initial guess: The computation of an initial guess in the recycling space requires the solution of a linear system of dimension k , which can be achieved in $O(k^3)$ operations, and two matrix–vector multiplications, requiring $O(mk)$ operations.

Augmented GMRES: In this case, at each iteration, two rectangular matrix–vector products are required in addition to the usual matrix–vector product, for an overall cost of $O(mn^2) + O(m^2n) + O(mkn)$. For a sparse matrix, the cost is $O(mn^2) + O(mbn) + O(mkn)$.

Suppose now that the iteration count of the original system is mn , with r number of GMRES restarts, and that for the recycling method is $\tilde{r}n$, with $\tilde{r} < r$. The total cost C and C_r for each iteration of the system without recycling and for the system with recycling are, respectively,

$$C = r \sum_{j=1}^n (mb + mj) \quad (47)$$

$$C_4 = \tilde{r} \sum_{j=1}^n \left(mb + (j + k)m + (k + j)^3 \right) + 2mk + k^3 + \frac{ksm + m^2k}{\ell},$$

where we make the hypothesis that two consecutive SVD operations are spaced by ℓ time steps. A reasonable way to proceed is to choose n, s, k , and b , make a guess on the ratio r/\tilde{r} , and derive ℓ so that $C_r = C$. Then, any value of ℓ larger than this will lead to time savings[‡] if the ratio r/\tilde{r} does not change much with ℓ . For example, in Figure 7, we show the minimum value of ℓ as a function of the recycling space dimension k and the ratio $\frac{r-\tilde{r}}{\tilde{r}}$ expressed in percentage. For the generation of Figure 7, we assumed $s = 60$, $n = 30$, $b = 300$, $r = 3$, and $m = 10^6$. Since the cost expressions (47) are not exact (i.e., the coefficients are missing), this plot is valuable only as a trend indicator, and not as an exact tool for choice making. However, the trend in Figure 7 suggests that if the recycling method is not much effective in reducing the iteration count (lower part of the image), it is convenient to space the SVDs as much as possible, and that the same consideration holds as the recycling subspace dimension increases. These considerations are highly dependent on the linear system dimension m : If the number of unknowns is very large, GMRES iterations will be very expensive, and even a small reduction in the iteration count can return significant time savings. Conversely, for small linear systems, usually,

[‡]In modern cpu architectures, the direct proportionality between operation count and computational time is not strictly true; hence, this statement must be received with care.

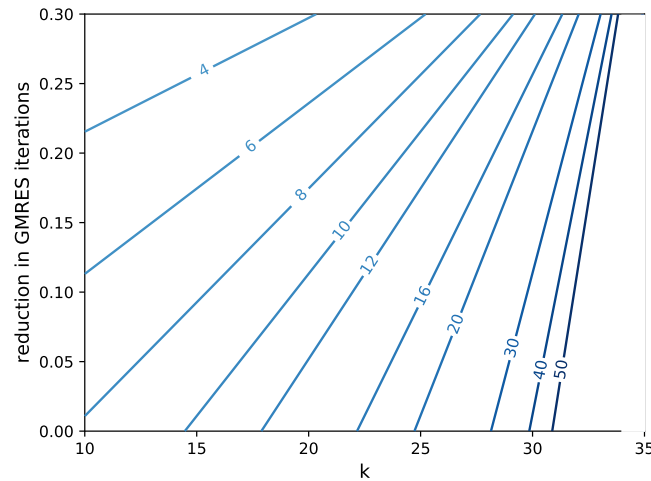


FIGURE 7 Minimum value of ℓ depending on the recycling space dimension k and the ratio $\frac{r-\tilde{r}}{r}$ in percentage. GMRES = generalized minimal residual

TABLE 3 Iterations and time required by the augmentation algorithm for first-order finite elements. In the first row, the code is run without a recycling preconditioner. In the remaining rows, the augmentation preconditioner is enabled using a recycling space of dimension 20 (for all rows), updated every 20, 40, 60, and 80 time steps, respectively

Method	Average iteration count	Solver wall time (s)
No aug	124.5	222.7
Aug 20	78.36	249.0
Aug 40	78.00	243.6
Aug 60	77.48	268.9
Aug 80	78.20	271.8

GMRES iterations are relatively inexpensive, and even a 30% reduction in the iteration count can be easily overcome by the additional SVD expenditure.

To verify the time requirements of the *augmentation orthogonal* method as opposed to the unrecycled case, we measure the time required to complete 1,000 time steps of width $\Delta t = 0.5$, with $\nu = 10^{-2}$. In all the recycling runs, take an augmentation subspace of dimension 20, and we consider the following four choices for the generation of the recycling subspace.

Aug 20: SVD is performed once every 20 time steps; SVD is performed among the previous 20 solutions.

Aug 40: SVD is performed once every 40 time steps; SVD is performed among the previous 40 solutions.

Aug 60: SVD is performed once every 60 time steps; SVD is performed among the previous 60 solutions.

Aug 80: SVD is performed once every 80 time steps; SVD is performed among the previous 80 solutions.

The results are reported in Table 3 for the case of first-order finite elements on a 64×64 grid. In terms of wall time, it seems that there is no advantage in choosing a recycling method over a traditional one. An obvious reason for this behavior is the fact that first-order finite elements lead to very sparse matrices, whose matrix-vector products are too inexpensive to produce any sensible time reduction, even with a 35% reduction of the iteration count.

To test this hypothesis, we repeat the experiment with a 2×2 grid of 23rd-degree Legendre spectral elements. The related results are available in Table 4.

In all the cases of Tables 3 and 4, the auxiliary time required by the storage of previous solutions and by the SVD is less than 1% of the solvers' time.

TABLE 4 Iterations and time required by the augmentation algorithm for Legendre spectral elements of order 23. In the first row, the code is run without a recycling preconditioner. In the last four rows, the augmentation preconditioner is enabled using a recycling space of dimension 20 (for all rows), updated every 20, 40, 60, and 80 time steps, respectively

Method	Average iteration count	Solver wall time (s)
No aug	53.73	190.7
Aug 20	48.07	179.9
Aug 40	44.77	170.2
Aug 60	45.89	176.6
Aug 80	46.38	175.0

7 | CONCLUSIONS

In this work, we have explored several strategies to accelerate the solution of slowly varying linear systems coming from the discretization of time-dependent nonlinear PDEs with convection terms and diffusion. We developed a common framework for recycling methods that allowed to systematically construct deflative and augmentation methods, some of which are new.

The two families of recycling methods considered here were then compared for a choice of recycling subspaces coming from the SVD of previous solutions. After introducing the new recycling subspaces, we studied their application in the construction of a good initial guess by means of Galerkin projection, as opposed to standard extrapolation.

Replacing extrapolation with projection introduces additional freedom in the choice of the projection subspace, and we studied the convergence properties of two different enriching subspaces: The first one was constructed by orthonormalizing only a few previous solution vectors, and the second one was constructed by means of an SVD of a larger set of previous solutions. This last approach is particularly convenient if one is interested anyway in constructing a POD approximation of some sort, and the POD basis is already available.

The recycling methods considered in this work were able to offer a reduction in the number of iterations in the GMRES solver in all the test cases. By weighting the overall cost of the algorithm, we conclude that SVD-accelerated methods may be an interesting strategy in all those cases where the matrices come from high-order finite element methods, or spectral element methods, at least for scalar convection–diffusion problems, and moderate values of the Péclet number.

ORCID

L. Heltai  <http://orcid.org/0000-0001-5514-4683>

REFERENCES

1. Hestenes MR, Stiefel E. Methods of conjugate gradients for solving linear systems. *J Res Nat Bur Stand.* 1952;49(6):409.
2. Eisenstat SC, Elman HC, Schultz MH. Variational iterative methods for nonsymmetric systems of linear equations. *SIAM J Numer Anal.* 1983;20(2):345–357.
3. Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J Sci Stat Comp.* 1986;7(3):856–869.
4. Golub GH, Van Loan CF. *Matrix computations.* Baltimore, MD: Johns Hopkins University Press; 2012.
5. Greenbaum A, Rozložník Z, Strakoš M. Numerical behaviour of the modified Gram-Schmidt GMRES implementation. *BIT Numer Math.* 1997;37(3):706–719.
6. Paige CC, Strakoš Z. Residual and backward error bounds in minimum residual Krylov subspace methods. *SIAM J Sci Comput.* 2002;23(6):1898–1923.
7. Paige CC, Rozložník M, Strakoš Z. Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES. *SIAM J Matrix Anal A.* 2006;28(1):264–284.
8. Van der Vorst HA, Vuik C. The superlinear convergence behaviour of GMRES. *J Comput Appl Math.* 1993;48(3):327–341.
9. Joubert W. On the convergence behavior of the restarted GMRES algorithm for solving nonsymmetric linear systems. *Numer Linear Algebr.* 1994;1(5):427–447.
10. Embree M. The tortoise and the hare restart GMRES. *SIAM Rev.* 2003;45(2):259–266. <https://doi.org/10.1137/S003614450139961>
11. Greenbaum A, Pták V, Strakoš Z. Any nonincreasing convergence curve is possible for GMRES. *SIAM J Matrix Anal A.* 1996;17(3):465–469.

12. Baker AH, Jessup ER, Kolev TzV. A simple strategy for varying the restart parameter in GMRES. *J Comput Appl Math*. 2009;230(2):751–761. <https://doi.org/10.1016/j.cam.2009.01.009>
13. De Sturler E. Truncation strategies for optimal Krylov subspace methods. *SIAM J Numer Anal*. 1999;36(3):864–889.
14. Saad Y, Yeung M, Erhel J, Guyomarc'h F. A deflated version of the conjugate gradient algorithm. *SIAM J Sci Comput*. 2000;21(5):1909–1926.
15. Nicolaides RA. Deflation of conjugate gradients with applications to boundary value problems. *SIAM J Numer Anal*. 1987;24(2):355–365.
16. Parks ML, de Sturler E, Mackey G, Johnson DD, Maiti S. Recycling Krylov subspaces for sequences of linear systems. *SIAM J Sci Comput*. 2006;28(5):1651–1674.
17. Amritkar A, de Sturler E, Świrydowicz K, Tafti D, Ahuja K. Recycling Krylov subspaces for CFD applications and a new hybrid recycling solver. *J Comput Phys*. 2015;303:222–237.
18. Wang S, de Sturler E, Paulino GH. Large-scale topology optimization using preconditioned Krylov subspace methods with recycling. *Int J Numer Methods Eng*. 2007;69(12):2441–2468.
19. Mello LAM, de Sturler E, Paulino GH, Silva ECN. Recycling Krylov subspaces for efficient large-scale electrical impedance tomography. *Comput Methods Appl Mech Eng*. 2010;199(49–52):3101–3110.
20. Kilmer ME, de Sturler E. Recycling subspace information for diffuse optical tomography. *SIAM J Sci Comput*. 2006;27(6):2140–2166.
21. Ali MAS, Sadkane M. Improved predictor schemes for large systems of linear ODEs. *Electron T Numer Ana*. 2012;39:253–270.
22. Quarteroni A. Numerical models for differential problems. Cham, Switzerland: Springer International Publishing AG; 2014.
23. Greenbaum A. Iterative methods for solving linear systems. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1997. <https://doi.org/10.1137/1.9781611970937>
24. Hackbusch W. Iterative solution of large sparse systems of equations. Cham, Switzerland: Springer International Publishing Switzerland; 2016. <https://doi.org/10.1007/978-3-319-28483-5>
25. Liesen J, Strakoš Z. Krylov subspace methods. Oxford, UK: Oxford University Press; 2012.
26. Saad Y. Iterative methods for sparse linear systems. Philadelphia, PA: Society for Industrial and Applied Mathematics; 2003.
27. Benzi M. Preconditioning techniques for large linear systems: a survey. *J Comput Phys*. 2002;182(2):418–477.
28. Saad Y. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J Sci Comput*. 1993;14(2):461–469.
29. Saad Y. Analysis of augmented Krylov subspace methods. *SIAM J Matrix Anal A*. 1997;18(2):435–449.
30. Chapman A, Saad Y. Deflated and augmented Krylov subspace techniques. *Numer Linear Algebra Appl*. 1997;4(1):43–66.
31. Nabben R, Vuik C. A comparison of deflation and coarse grid correction applied to porous media flow. *SIAM J Numer Anal*. 2004;42(4):1631–1647.
32. Tang JM, MacLachlan SP, Nabben R, Vuik C. A comparison of two level preconditioners based on multigrid and deflation. *SIAM J Matrix Anal A*. 2010;31(4):1715–1739.
33. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive algebraic multigrid. *SIAM J Sci Comput*. 2006;27:1261–1286.
34. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive smoothed aggregation (α sa). *SIAM J Sci Comput*. 2004;25:1896–1920.
35. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive smoothed aggregation (α sa) multigrid. *SIAM Rev*. 2005;47:317–346.
36. Eiermann M, Ernst OG, Schneider O. Analysis of acceleration strategies for restarted minimal residual methods. *J Comput Appl Math*. 2000;123:261–292.
37. Nabben R, Vuik C. A comparison of deflation and the balancing preconditioner. *SIAM J Sci Comput*. 2006;27(5):1742–1759.
38. de Sturler E. Nested Krylov methods based on GCR. *J Comput Appl Math*. 1996;67(1):15–41.
39. Gaul A, Gutknecht MH, Liesen J, Nabben R. A framework for deflated and augmented Krylov subspace methods. *SIAM J Matrix Anal A*. 2013;34(2):495–518.
40. Gutknecht MH. Deflated and augmented Krylov subspace methods: a framework for deflated BiCG and related solvers. *SIAM J Matrix Anal A*. 2014;35(4):1444–1466.
41. Morgan RB. GMRES with deflated restarting. *SIAM J Sci Comput*. 2002;24(1):20–37.
42. Ahuja K, de Sturler E, Gugercin S, Chang ER. Recycling BiCG with an application to model reduction. *SIAM J Sci Comput*. 2012;34(4):A1925–A1949.
43. Ahuja K, Benner P, de Sturler E, Feng L. Recycling BiCGSTAB with an application to parametric model order reduction. *SIAM J Sci Comput*. 2015;37(5):S429–S446.
44. Fischer PF. Projection techniques for iterative solution of $Ax = b$ with successive right-hand sides. *Comput Methods Appl Mech Eng*. 1998;163(1):193–204.
45. Erhel J, Guyomarc'h F. An augmented conjugate gradient method for solving consecutive symmetric positive definite linear systems. *SIAM J Matrix Anal A*. 2000;21(4):1279–1299.
46. Xu J, Zikatanov L. The method of alternating projections and the method of subspace corrections in Hilbert space. *J Am Math Soc*. 2002;15(3):573–597.
47. Kressner D, Tobler C. Low-rank tensor Krylov subspace methods for parametrized linear systems. *SIAM J Matrix Anal A*. 2011;32(4):1288–1316.

48. Morgan RB. A restarted GMRES method augmented with eigenvectors. *SIAM J Matrix Anal A*. 1995;16(4):1154–1171.
49. Kharchenko SA, Yeremin AYU. Eigenvalue translation based preconditioners for the GMRES(k) method. *Numer Linear Algebra Appl*. 1995;2(1):51–77.
50. Giraud L, Gratton S, Pinel X, Vasseur X. Flexible GMRES with deflated restarting. *SIAM J Sci Comput*. 2010;32(4):1858–1878.
51. Paige CC, Parlett BN, van der Vorst H. Approximate solutions and eigenvalue bounds from Krylov subspaces. *Numer Linear Algebra Appl*. 1995;2(2):115–133.
52. Saad Y. Numerical methods for large eigenvalue problems. Philadelphia, PA: Society for Industrial and Applied Mathematics; 2011.
53. Trefethen LN, Bau D III. Numerical linear algebra. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1997.
54. Demmel JW. Applied numerical linear algebra. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1997.
55. Sorensen DC. Numerical methods for large eigenvalue problems. *Acta Numer*. 2002;11:519–584.
56. Quarteroni A, Valli A. Numerical approximation of partial differential equations. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1994.
57. LaBryer A, Attar PJ, Vedula P. A framework for large eddy simulation of Burgers turbulence based upon spatial and temporal statistical information. *Phys Fluids*. 2015;27(3).
58. Bangerth W, Davydov D, Heister T, et al. The deal.II library, version 8.4. *J Numer Math*. 2016;24(3):135–141.

How to cite this article: Pitton G, Heltai L. Accelerating the iterative solution of convection–diffusion problems using singular value decomposition. *Numer Linear Algebra Appl*. 2019;26:e2211. <https://doi.org/10.1002/nla.2211>