A GOLUB–KAHAN-TYPE REDUCTION METHOD FOR MATRIX PAIRS[*]

MICHIEL E. HOCHSTENBACH[†], LOTHAR REICHEL[‡], AND XUEBO YU[‡]

Abstract. We describe a novel method for reducing a pair of large matrices $\{A, B\}$ to a pair of small matrices $\{H, K\}$. The method is an extension of Golub–Kahan bidiagonalization to matrix pairs, and simplifies to the latter method when B is the identity matrix. Applications to Tikhonov regularization of large linear discrete ill-posed problems are described. In these problems the matrix A represents a discretization of a compact integral operator and B is a regularization matrix.

Key words. Generalized Golub–Kahan bidiagonalization, generalized Lanczos bidiagonalization, generalized Krylov method, matrix pair decomposition, ill-posed problem, Tikhonov regularization, multi-parameter regularization.

1. Introduction.

This paper presents a new method for reducing a pair of large matrices $\{A, B\}$, with $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$, to a pair of small matrices $\{H_{\ell+1,\ell}, K_{\ell,\ell}\}$, with $H_{\ell+1,\ell} \in \mathbb{R}^{(\ell+1)\times\ell}$, $K_{\ell,\ell} \in \mathbb{R}^{\ell\times\ell}$, and $\ell \ll \min\{m, n, p\}$, by a generalization of Golub–Kahan bidiagonalization (also known as Lanczos bidiagonalization). The method reduces to standard Golub–Kahan bidiagonalization when B is the identity matrix and breakdown is handled appropriately. After ℓ steps of our reduction method, we obtain decompositions of the form

$$\begin{aligned} AV_\ell &= U_{\ell+1}\, H_{\ell+1,\ell},\\ BV_\ell &= W_\ell\, K_{\ell,\ell}, \end{aligned} \tag{1.1}$$

where $U_{\ell+1} \in \mathbb{R}^{m\times(\ell+1)}$, $V_\ell \in \mathbb{R}^{n\times\ell}$, and $W_\ell \in \mathbb{R}^{p\times\ell}$ have orthonormal columns with the first column of $U_{\ell+1}$ specified, and the matrices $H_{\ell+1,\ell}$ and $K_{\ell,\ell}$ are of upper Hessenberg and upper triangular form, respectively. Moreover, the matrices $H_{\ell+1,\ell}$ and $K_{\ell,\ell}$ are sparse in the sense that they contain many zero entries above the diagonal; see Section 2 for details. We assume for the moment that $\ell \le \min\{m, n, p\}$ is small enough so that breakdown in the recursion formulas does not occur. This restriction will be removed below.

Our reduction technique is a generalized Krylov method. It is inspired by the generalized Arnoldi method proposed by Li and Ye [17] for a pair of square matrices $\{A, B\}$, its application to the solution of linear discrete ill-posed problems described in [24], and by the two-sided Lanczos method for pairs of square matrices discussed by Hoffnung et al. [14].

A reason for our interest in the decompositions (1.1) is their applicability to the solution of large linear discrete ill-posed problems. Consider the linear least-squares problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \|A\boldsymbol{x} - \boldsymbol{b}\|, \qquad A \in \mathbb{R}^{m\times n}, \qquad \boldsymbol{b} \in \mathbb{R}^m, \tag{1.2}$$

[*]February 2, 2015.

[†]Department of Mathematics and Computer Science, TU Eindhoven, PO Box 513, 5600 MB Eindhoven, The Netherlands. `www.win.tue.nl/∼hochsten`. Supported by an NWO Vidi research grant.

[‡]Department of Mathematical Sciences, Kent State University, Kent, OH 44242, USA. E-mail: `reichel@math.kent.edu`, `xyu@math.kent.edu`. Research supported in part by NSF Grant DMS-1115385.

where A is a large matrix with many singular values near zero. In particular, the matrix A has a large condition number and may even be singular. Linear least-squares problems with a matrix of this kind commonly are called linear discrete ill-posed problems. One field or origin is the discretization of Fredholm integral equations of the first kind. Throughout this paper, $\|\cdot\|$ denotes the Euclidean vector norm or the associated induced matrix norm.

The vector $\boldsymbol{b}$ in the right-hand side of (1.2) usually represents available data and is often contaminated by a noise vector $\boldsymbol{e} \in \mathbb{R}^m$, which represents measurement errors. Let $\breve{\boldsymbol{b}}$ denote the unknown noise-free vector associated with $\boldsymbol{b}$, i.e.,

$$\boldsymbol{b} = \breve{\boldsymbol{b}} + \boldsymbol{e}. \tag{1.3}$$

We are interested in computing the solution $\breve{\boldsymbol{x}}$ of minimal norm of the unavailable error-free least-squares problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|A\boldsymbol{x} - \breve{\boldsymbol{b}}\|.$$

Since the vector $\breve{\boldsymbol{b}}$ is not known, we seek to find an approximation of $\breve{\boldsymbol{x}}$ by computing a suitable approximate solution to (1.2).

Straightforward solution of (1.2) usually does not result in a sensible approximation of $\breve{\boldsymbol{x}}$ since A is ill conditioned and $\boldsymbol{b}$ contains noise. For this reason, one often applies regularization: one replaces the minimization problem (1.2) by a nearby problem that is less sensitive to the noise. Tikhonov regularization replaces (1.2) by the least-squares problem with penalty

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \{\, \|A\boldsymbol{x} - \boldsymbol{b}\|^2 + \mu \, \|B\boldsymbol{x}\|^2 \,\}, \tag{1.4}$$

where $B \in \mathbb{R}^{p \times n}$ is a regularization matrix and $\mu > 0$ is a regularization parameter. We say that (1.4) is in standard form if B is the identity. We are interested in the situation when B is a fairly general matrix. Many commonly applied regularization matrices are rectangular. Both the cases $p \le n$ and $p > n$ arise in applications.

Let $\mathcal{N}(M)$ denote the null space of M. We require the regularization matrix B to satisfy

$$\mathcal{N}(A) \cap \mathcal{N}(B) = \{\boldsymbol{0}\}.$$

Then the minimization problem (1.4) has a unique solution for any $\mu > 0$. We denote this solution by $\boldsymbol{x}_\mu$. The value of μ influences the sensitivity of $\boldsymbol{x}_\mu$ to the noise $\boldsymbol{e}$ and to round-off errors introduced during the computations, and determines how close $\boldsymbol{x}_\mu$ is to $\breve{\boldsymbol{x}}$; see, e.g., Groetsch [10], Engl et al. [8], and Hansen [11] for discussions on and analyses of Tikhonov regularization.

We are interested in determining a suitable value of $\mu > 0$ and an approximation of the corresponding solution $\boldsymbol{x}_\mu$ of (1.4). This typically requires the computation of the solution of (1.4) for several values of $\mu > 0$. For instance, when μ is determined by the discrepancy principle, the L-curve criterion, or the generalized cross validation method, the minimization problem (1.4) has to be solved for several values of $\mu > 0$ to be able to determine a suitable μ-value; see [4, 8, 11, 16, 23] for discussions on these and other methods for determining a suitable value of μ. The repeated solution of (1.4) for different μ-values can be carried out fairly inexpensively when A and B are of small to medium size by first computing the generalized singular value decomposition

(GSVD) of the matrix pair $\{A, B\}$. However, the evaluation of the GSVD can be very expensive when A and B are large. We are interested in this situation and propose to first determine the decompositions (1.1) for a fairly small value of ℓ. This yields small matrices $H_{\ell+1,\ell}$ and $K_{\ell,\ell}$ in (1.1). We then can inexpensively compute the GSVD of $\{H_{\ell+1,\ell}, K_{\ell,\ell}\}$. Substituting the computed decompositions (1.1) and the GSVD of $\{H_{\ell+1,\ell}, K_{\ell,\ell}\}$ into (1.4) allows us to compute approximations of $\boldsymbol{x}_\mu$ for many values of μ quite inexpensively; see, e.g., [11] for a discussion on the application of the GSVD to the solution of (1.4).

Another approach that for certain regularization matrices B may reduce the cost of repeated solution of (1.4) is to first transform the problem to standard form. This is achieved by substituting $\boldsymbol{y} = B\boldsymbol{x}$ into (1.4). The matrix A then is replaced by $AB_A^\dagger$ in (1.4), where

$$B_A^\dagger = \left(I - (A(I - B^\dagger B))^\dagger A\right) B^\dagger \in \mathbb{R}^{n \times p}$$

is known as the A-weighted generalized inverse of B. Here $M^\dagger$ denotes the Moore–Penrose pseudoinverse of the matrix M, and I is the identity matrix; see e.g., Eldén [7] or Hansen [11, Section 2.3] for details. This transformation is attractive to use when B is a banded matrix with small bandwidth or an orthogonal projection; see [19] for the latter. For general large matrices B, the evaluation of matrix-vector products with the matrices $AB_A^\dagger$ and $B_A^\dagger$ is prohibitively expensive.

We are interested in the use of regularization matrices B that do not have a structure that permits efficient evaluation of matrix-vector products with the matrices $AB_A^\dagger$ and $B_A^\dagger$. Such regularization matrices arise, e.g., when A is the discretization of a Fredholm integral operator in two or more space-dimensions. The use of the decompositions (1.1) allows us to determine approximate solutions of (1.4) with a general matrix B for several μ-values fairly inexpensively.

Kilmer et al. [15] describe an inner-outer method to solve of (1.4) with a general matrix B. A disadvantage is that this method may be expensive, since a fairly large number of inner iterations may be required. In [13], the regularization matrix is used to select a candidate solution from a standard Krylov subspace, which is generated by A alone. This approach works well for a variety of discrete ill-posed problems; see illustrations in [13]. However, for some discrete ill-posed problems more accurate approximations of $\breve{\boldsymbol{x}}$ can be determined by letting the solution subspace depend on both the matrices A and B; see Example 4.1 of Section 4. A generalized Krylov method for Tikhonov regularization using both A and B is proposed in [24]. This method is based on the generalized Arnoldi decomposition described by Li and Ye [17]. It requires that both matrices A and B be square. Also the method described in [25], which is based on the flexible Arnoldi process, requires A to be square. The method of this paper allows both A and B to be rectangular. Many commonly used regularization matrices B are rectangular; see, e.g., [11, 15] for examples. Linear discrete ill-posed problems with a rectangular matrix A arise, e.g., in computerized X-ray tomography; see [5, Example 4.4]. Another illustration can be found in Example 4.2 below.

This paper focuses on applications of the decompositions (1.1) to the solution of discrete ill-posed problems. However, the method developed also can be used to solve other problems that require simultaneous reduction of two or more matrices. The application to computing an approximation of a partial GSVD of a pair of large matrices already has been mentioned. A recent discussion of solution methods for partial differential equations that may require the reduction of pairs of large matrices can be found in [1].

The organization of this paper is as follows. Section 2 describes our reduction method for the matrix pair $\{A, B\}$ as well as its generalization to the reduction of matrix $(q+1)$-tuplets. Application to Tikhonov regularization (1.4) and to multi-parameter Tikhonov regularization of these reduction methods is discussed in Section 3. Several numerical experiments can be found in Section 4. We end with conclusions in Section 5.

2. New reduction methods for matrix pairs and $(q+1)$-tuplets.

We first review the standard Golub–Kahan bidiagonalization method, then describe our generalization to the reduction of matrix pairs, and finally discuss how the latter method can be generalized to be applicable to the reduction of matrix $(q+1)$-tuplets.

2.1. Standard Golub–Kahan bidiagonalization.

We recall some properties of partial Golub–Kahan bidiagonalization of a matrix $A \in \mathbb{R}^{m\times n}$; pseudocode is given in Algorithm 2.1 below. The choice of initial vector $\boldsymbol{u}_1 \in \mathbb{R}^m$ determines the partial bidiagonalization. The parameter ℓ specifies the number of steps. For now, we assume ℓ to be small enough so that breakdown is avoided. Breakdown is discussed below. It occurs when certain vectors $\widehat{\boldsymbol{u}}$ or $\widehat{\boldsymbol{v}}$ in the algorithm vanish. The superscript * denotes transposition.

Algorithm 2.1. Partial Golub–Kahan bidiagonalization of A.

```
1.  Input: matrix $A \in \mathbb{R}^{m\times n}$, initial unit vector $\boldsymbol{u}_1$, and number of steps $\ell$
2.  $\boldsymbol{v}_0 := \boldsymbol{0}$
3.  for $j = 1, 2, \dots, \ell$ do
4.      $\widehat{\boldsymbol{v}} := A^*\boldsymbol{u}_j - h_{j,j-1}\boldsymbol{v}_{j-1}$
5.      $h_{j,j} := \|\widehat{\boldsymbol{v}}\|$
6.      $\boldsymbol{v}_j := \widehat{\boldsymbol{v}}/h_{j,j}$            if $h_{j,j} = 0$ : see text
7.      $\widehat{\boldsymbol{u}} := A\boldsymbol{v}_j - h_{j,j}\boldsymbol{u}_j$
8.      $h_{j+1,j} := \|\widehat{\boldsymbol{u}}\|$
9.      $\boldsymbol{u}_{j+1} := \widehat{\boldsymbol{v}}/h_{j+1,j}$        if $h_{j+1,j} = 0$ : see text
10. end for
```

The following aspects of the above algorithm are essential. An initial unit vector $\boldsymbol{u}_1$ is provided and the algorithm alternatingly multiplies a vector $\boldsymbol{u}_j$ by A^* followed by orthogonalization to the most recently generated vector $\boldsymbol{v}_{j-1}$, and alternatingly multiplies a vector $\boldsymbol{v}_j$ by A followed by orthogonalization to the most recently generated vector $\boldsymbol{u}_j$. This secures that, in exact arithmetic, all computed vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_\ell$ are orthonormal and all computed vectors $\boldsymbol{u}_1, \dots, \boldsymbol{u}_{\ell+1}$ are orthonormal. After $\ell \le \min\{m, n\}$ steps, Algorithm 2.1 has determined the partial Golub–Kahan decompositions

$$AV_\ell = U_{\ell+1}H_{\ell+1,\ell}, \qquad A^*U_\ell = V_\ell H_{\ell,\ell}^*,$$

where the matrices $U_{\ell+1} = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_{\ell+1}] \in \mathbb{R}^{m\times(\ell+1)}$ and $V_\ell = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_\ell] \in \mathbb{R}^{n\times\ell}$ have orthonormal columns, $U_\ell \in \mathbb{R}^{m\times\ell}$ is made up of the first ℓ columns of $U_{\ell+1}$. The columns of $U_{\ell+1}$ form an orthonormal basis for the Krylov subspace

$$\mathcal{K}_{\ell+1}(AA^*, \boldsymbol{u}_1) = \mathrm{span}\{\boldsymbol{u}_1, AA^*\boldsymbol{u}_1, \dots, (AA^*)^\ell\boldsymbol{u}_1\} \tag{2.1}$$

and the columns of V_ℓ form an orthonormal basis for the Krylov subspace

$$\mathcal{K}_\ell(A^*A, A^*\boldsymbol{u}_1) = \mathrm{span}\{A^*\boldsymbol{u}_1, (A^*A)A^*\boldsymbol{u}_1, \dots, (A^*A)^{\ell-1}A^*\boldsymbol{u}_1\}. \tag{2.2}$$

The matrix $H_{\ell+1,\ell} \in \mathbb{R}^{(\ell+1)\times\ell}$ is lower bidiagonal; its nontrivial entries are the scalars $h_{j,k}$ determined by the algorithm. Finally, $H_{\ell,\ell}$ is the leading principal $\ell\times\ell$ submatrix of $H_{\ell+1,\ell}$.

Breakdown of the computations with Algorithm 2.1 occurs when either $h_{j,j} = 0$ in line 6 or $h_{j+1,j} = 0$ in line 9. In some applications it is then appropriate to terminate the computations, while in others we may choose to continue the method. For instance, if breakdown occurs in line 6, then we can continue the computations by letting $\boldsymbol{v}_j$ be an arbitrary unit vector that is orthogonal to the already generated vectors $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{j-1}\}$. Breakdown in line 9 can be handled similarly. By carrying out $\ell = \min\{m, n\}$ steps with Algorithm 2.1 in this manner, we see that any matrix $A \in \mathbb{R}^{m \times n}$ can be bidiagonalized.

Proposition 2.1. *(a) For an arbitrary unit vector $\boldsymbol{u}_1 \in \mathbb{R}^m$, there are orthogonal matrices $U = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m] \in \mathbb{R}^{m \times m}$ and $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{n \times n}$ such that*

$$H = U^* A V$$

is a lower bidiagonal matrix with nonnegative entries.
(b) The matrix H is uniquely determined if all diagonal and subdiagonal elements are nonzero (and hence positive).
(c) Given $\boldsymbol{u}_1$, the matrices U and V are uniquely determined if A is square and all diagonal and subdiagonal elements of H are nonzero.

We conclude this subsection with two remarks about Proposition 2.1: First, if $m > n$ or $n < m$, then the last vectors $\boldsymbol{u}_{n+1}, \ldots, \boldsymbol{u}_m$ or $\boldsymbol{v}_{m+1}, \ldots, \boldsymbol{v}_n$ are not determined uniquely. Moreover, if some nontrivial element of the bidiagonal matrix H vanishes, then the next $\boldsymbol{u}$-vector or $\boldsymbol{v}$-vector is not determined uniquely. Second, we note that the Golub–Kahan bidiagonalization process is a particularly efficient way of determining the leading entries of H when the matrix A is large and sparse, because then the required matrix-vector products with A and A^* are fairly inexpensive to evaluate.

2.2. A Golub–Kahan-type reduction method for matrix pairs. We describe a generalization of partial Golub–Kahan bidiagonalization for the reduction of two large matrices A and B to small ones; cf. (1.1). Algorithm 2.2 below presents pseudocode for the method. In the main loop of the algorithm, matrix-vector products with the matrices A, B, A^*, and B^* are evaluated in a cyclic fashion. Details of the algorithm are discussed below.

Algorithm 2.2. Partial Golub–Kahan-type reduction of matrix pair $\{A, B\}$.

```
1.  Input: matrices $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{p\times n}$, unit vector $\boldsymbol{u}_1$, and number or steps $\ell$
2.  $\widehat{\boldsymbol{v}} := A^*\boldsymbol{u}_1$;  $h_{1,1} := \|\widehat{\boldsymbol{v}}\|$;  $\boldsymbol{v}_1 := \widehat{\boldsymbol{v}}/h_{1,1}$;
3.  $N(\boldsymbol{u}) := 1$;  $N(\boldsymbol{v}) := 1$;  $N(\boldsymbol{w}) := 0$
4.  for $j = 1, 2, \dots, \ell$ do                                  $\boldsymbol{v}_j$-vector under consideration
5.      $\widehat{\boldsymbol{u}} := A\boldsymbol{v}_j$                                        new $\boldsymbol{u}$-vector
6.      for $i = 1, 2, \dots, N(\boldsymbol{u})$ do
7.          $h_{i,j} := \boldsymbol{u}_i^*\widehat{\boldsymbol{u}}$;  $\widehat{\boldsymbol{u}} := \widehat{\boldsymbol{u}} - \boldsymbol{u}_i h_{i,j}$
8.      end for
9.      $h_{N(\boldsymbol{u})+1,j} := \|\widehat{\boldsymbol{u}}\|$
10.     $N(\boldsymbol{u}) := N(\boldsymbol{u}) + 1$;  $\boldsymbol{u}_{N(\boldsymbol{u})} := \widehat{\boldsymbol{u}}/h_{N(\boldsymbol{u}),j}$          if $h_{N(\boldsymbol{u}),j} = 0$ : see text
11.     $\widehat{\boldsymbol{w}} := B\boldsymbol{v}_j$                                        new $\boldsymbol{w}$-vector
12.     for $i = 1, 2, \dots, N(\boldsymbol{w})$ do
13.         $k_{i,j} := \boldsymbol{w}_i^*\widehat{\boldsymbol{w}}$;  $\widehat{\boldsymbol{w}} := \widehat{\boldsymbol{w}} - \boldsymbol{w}_i k_{i,j}$
14.     end for
15.     $k_{N(\boldsymbol{w})+1,j} := \|\widehat{\boldsymbol{w}}\|$
16.     $N(\boldsymbol{w}) := N(\boldsymbol{w}) + 1$;  $\boldsymbol{w}_{N(\boldsymbol{w})} := \widehat{\boldsymbol{w}}/k_{N(\boldsymbol{w}),j}$          if $k_{N(\boldsymbol{w}),j} = 0$ : see text
17.     $\widehat{\boldsymbol{v}} := A^*\boldsymbol{u}_{N(\boldsymbol{u})}$                                  new $\boldsymbol{v}$-vector from $\boldsymbol{u}$-vector
18.     for $i = 1, 2, \dots, N(\boldsymbol{v})$ do
19.         $h_{N(\boldsymbol{u}),i} := \boldsymbol{v}_i^*\widehat{\boldsymbol{v}}$;  $\widehat{\boldsymbol{v}} := \widehat{\boldsymbol{v}} - \boldsymbol{v}_i h_{N(\boldsymbol{u}),i}$
20.     end for
21.     $h_{N(\boldsymbol{u}),N(\boldsymbol{v})+1} := \|\widehat{\boldsymbol{v}}\|$
22.     $N(\boldsymbol{v}) := N(\boldsymbol{v}) + 1$;  $\boldsymbol{v}_{N(\boldsymbol{v})} := \widehat{\boldsymbol{v}}/h_{N(\boldsymbol{u}),N(\boldsymbol{v})}$     if $h_{N(\boldsymbol{u}),N(\boldsymbol{v})} = 0$ : see text
23.     $\widehat{\boldsymbol{v}} := B^*\boldsymbol{w}_{N(\boldsymbol{w})}$                                  new $\boldsymbol{v}$-vector from $\boldsymbol{w}$-vector
24.     for $i = 1, 2, \dots, N(\boldsymbol{v})$ do
25.         $k_{N(\boldsymbol{w}),i} := \boldsymbol{v}_i^*\widehat{\boldsymbol{v}}$;  $\widehat{\boldsymbol{v}} := \widehat{\boldsymbol{v}} - \boldsymbol{v}_i k_{N(\boldsymbol{w}),i}$
26.     end for
27.     $k_{N(\boldsymbol{w}),N(\boldsymbol{v})+1} := \|\widehat{\boldsymbol{v}}\|$
28.     $N(\boldsymbol{v}) := N(\boldsymbol{v}) + 1$;  $\boldsymbol{v}_{N(\boldsymbol{v})} := \widehat{\boldsymbol{v}}/k_{N(\boldsymbol{w}),N(\boldsymbol{v})}$     if $k_{N(\boldsymbol{w}),N(\boldsymbol{v})} = 0$ : see text
29. end for
```

The following observations about Algorithm 2.2 are pertinent. Some elements $h_{i,j}$ are computed both in lines 7 and 19, and similarly certain entries $k_{i,j}$ are computed twice. It suffices, of course, to compute each entry once in an actual implementation of the algorithm. However, the details required to achieve this would clutter the algorithm and therefore are omitted. The initial vector $\boldsymbol{u}_1$ is the first column of the matrix $U_{\ell+1}$ in the decompositions (2.3) below. We choose this vector, rather than the first column in one of the matrices V_ℓ or W_ℓ in the decompositions (2.3), because of the application to Tikhonov regularization described in Section 3. For now, assume that the vector $\boldsymbol{u}_1$ is not orthogonal to the range of A, so that the vector $\widehat{\boldsymbol{v}}$ in line 2 does not vanish. The integers $N(\boldsymbol{u})$, $N(\boldsymbol{v})$, and $N(\boldsymbol{w})$ in the algorithm track the number of vectors $\boldsymbol{u}_j$, $\boldsymbol{v}_j$, and $\boldsymbol{w}_j$ generated so far during the computations. Next, we discuss the situation when a denominator in lines 10, 16, 22, or 28 vanishes. This is a rare event in actual computations. We consider the case when $h_{N(\boldsymbol{u}),j} = 0$ in line 10; the other cases can be handled similarly. There are two main options. First, we can skip line 10, i.e., we neither increase $N(\boldsymbol{u})$ nor add a new $\boldsymbol{u}$-vector. Subsequently, we also omit lines 17–22. Differently from the situation for (standard) Golub–Kahan bidiagonalization (Algorithm 2.1), the method does not necessarily break down when $h_{N(\boldsymbol{u}),j} = 0$, because the vector $\boldsymbol{w}_{N(\boldsymbol{w})}$ may give a new $\boldsymbol{v}$-vector; see lines 23–28. Finally, we note that if the vector $\widehat{\boldsymbol{v}}$ vanishes in line 2, then we may choose $\widehat{\boldsymbol{v}}$ to be any unit vector. With this approach of handling vanishing denominators, we obtain the following result.

Proposition 2.2. *Suppose that Algorithm 2.2 is applied to the matrix pair $\{A, B\}$, where $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times n}$ is the identity matrix I. Omit the steps in lines 16 and 28 when a zero element is encountered. Then the generated $\boldsymbol{u}$-vectors and $\boldsymbol{v}$-vectors are the same as those determined by Algorithm 2.1, $\boldsymbol{w}_i = \boldsymbol{v}_i$ for all i, and the matrix $K_{\ell,\ell}$ is the identity.*

An alternative approach to handling the situation when $h_{N(\boldsymbol{u}),j} = 0$ is to add a random unit vector $\boldsymbol{u}_{N(\boldsymbol{u})}$ that is orthogonal to the already generated vectors $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_{N(\boldsymbol{u})-1}\}$, and then continue the computations with this vector. In particular, the lines 23–28 are executed, and we are guaranteed to have a specific number of vectors.

Proposition 2.3. *If we add extra vectors in case zero denominators are encountered in lines 2, 10, 16, 22, or 28, then after line 28 (with counter j), we have*

$$N(\boldsymbol{u}) = j + 1, \quad N(\boldsymbol{v}) = 2j + 1, \quad N(\boldsymbol{w}) = j,$$

as long as $j + 1 \le m$, $2j + 1 \le n$, and $j \le p$.

This latter approach is preferable for the sake of presentation. We will therefore in the remainder of the paper assume that random orthogonal unit vectors are added when some denominator in Algorithm 2.2 vanishes. We stress that the occurrence of zero denominators is rare.

Execution of $\ell > 1$ steps of Algorithm 2.2 yields the decompositions

$$\begin{array}{lcllcl} AV_\ell & = & U_{\ell+1} H_{\ell+1,\ell}, & BV_\ell & = & W_\ell K_{\ell,\ell}, \\ A^* U_\ell & = & V_{2\ell-2}\,(H_{\ell,2\ell-2})^*, & B^* W_\ell & = & V_{2\ell+1}\,(K_{\ell,2\ell+1})^*, \end{array} \tag{2.3}$$

where the vectors $\boldsymbol{u}_i$, $\boldsymbol{v}_i$, and $\boldsymbol{w}_i$ generated by Algorithm 2.2 are columns of the matrices U_j, V_j, and W_j, respectively, and the coefficients $h_{i,j}$ and $k_{i,j}$ determined by the algorithm are the nontrivial entries of the matrices H and K. Here and below, we sometimes omit the subscripts of the matrices in (2.3) for notational simplicity. The expressions (2.3), except for the one involving A^*U_ℓ, are valid also for $\ell = 1$. The matrices H and K are sparse. The nontrivial entries of the first five and four rows of H and K, respectively, are marked by $\times$ in the following depictions:

$$H : \begin{bmatrix} \times & & & & & & & \\ \times & \times & & & & & & \\ & \times & \times & \times & & & & \\ & & \times & \times & \times & \times & & \\ & & & \times & \times & \times & \times & \times \end{bmatrix}, \quad K : \begin{bmatrix} \times & \times & \times & & & & & & \\ & \times & \times & \times & \times & & & & \\ & & \times & \times & \times & \times & \times & & \\ & & & \times & \times & \times & \times & \times & \times \end{bmatrix}$$

The sparsity patterns of the matrices H and K can be justified as follows. Since $H_{\ell,\ell} = U_\ell^* A V_\ell$, we have that $H_{\ell,\ell}$ is an upper Hessenberg matrix. Also, since $H_{\ell,\ell} = (V_\ell^* A^* U_\ell)^*$, its elements $h_{i,j}$ are zero if $i > 1$ and $j > 2i - 2$, because $A^*\boldsymbol{u}_i$ is orthogonalized against $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_{2i-3}\}$ and yields $\boldsymbol{v}_{2i-2}$. Similarly, it follows from $K_{\ell,\ell} = W_\ell^* B V_\ell$ that $K_{\ell,\ell}$ is an upper triangular matrix. Additionally, $k_{i,j} = 0$ if $j > 2i + 1$, because $B^*\boldsymbol{w}_i$ is orthogonalized against $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_{2i}\}$ to produce $\boldsymbol{v}_{2i+1}$. Thus, both matrices $H_{\ell+1,\ell}$ and $K_{\ell,\ell}$ are upper banded with their upper bandwidths growing linearly with ℓ. We therefore may consider H and K as generalized bidiagonal matrices.

It follows from the sparsity patterns of H and K that the "full" orthogonalization in lines 6–8, 12–14, 18–20, and 24–26 is not necessary, since some of the inner products are guaranteed to vanish in exact arithmetic. However, some reorthogonalization might be beneficial in finite precision arithmetic. For maximal stability, we recommend to orthogonalize against the necessary vectors only (corresponding to the elements marked by "$\times$" in the sparsity patterns of H and K), followed by reorthogonalization against all previous vectors. Carrying out this procedure the full $\min\{m,n,p\}$ steps, we obtain the following result.

THEOREM 2.4. *Let $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$.*
(a) For an arbitrary unit vector $\boldsymbol{u}_1 \in \mathbb{R}^m$, there are orthogonal matrices $U = [\boldsymbol{u}_1,\ldots,\boldsymbol{u}_m] \in \mathbb{R}^{m\times m}$, $V = [\boldsymbol{v}_1,\ldots,\boldsymbol{v}_n] \in \mathbb{R}^{n\times n}$, and $W = [\boldsymbol{w}_1,\ldots,\boldsymbol{w}_p] \in \mathbb{R}^{p\times p}$ such that the matrices

$$H = U^*AV, \qquad K = W^*BV$$

have the following properties:

- *H is upper Hessenberg and in addition $h_{i,j} = 0$ if* $\begin{cases} j > 1, & \text{if } i = 1, \\ j > 2i-2, & \text{if } i > 1. \end{cases}$
- *K is upper triangular and in addition $k_{i,j} = 0$ if $j > 2i+1$.*

Moreover, the elements

$$\begin{aligned} &h_{1,1}, \quad h_{i+1,i} \quad \text{for all } \; i, \quad h_{i,2i-2} \quad \text{for } \; i > 1, \\ &k_{i,i}, \quad k_{i,2i+1} \quad \text{for all } \; i \end{aligned} \tag{2.4}$$

are nonnegative.
(b) Given $\boldsymbol{u}_1$, the matrices H and K are uniquely determined provided that the elements (2.4) are nonzero (and hence positive).
(c) Given $\boldsymbol{u}_1$, the matrices U, V, and W are uniquely determined if the matrices A and B are square and the elements (2.4) are nonzero.

An inspection of Algorithm 2.2 shows that $\text{span}\{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_\ell\}$ is generated from the vectors obtained by multiplying $\boldsymbol{v}_1$ by A^*A and B^*B in a periodic fashion,

$$\begin{array}{ll} \text{Group 0:} & \boldsymbol{v}_1, \\ \text{Group 1:} & A^*A\boldsymbol{v}_1,\; B^*B\boldsymbol{v}_1, \\ \text{Group 2:} & (A^*A)^2\boldsymbol{v}_1,\; (B^*B)(A^*A)\boldsymbol{v}_1,\; (A^*A)(B^*B)\boldsymbol{v}_1,\; (B^*B)^2\boldsymbol{v}_1, \\ \quad\vdots & \vdots \end{array} \tag{2.5}$$

ordered from top to bottom and from left to right; recall that $\boldsymbol{v}_1 = A^*\boldsymbol{u}_1/\|A^*\boldsymbol{u}_1\|$. Let $\mathrm{g}\mathcal{K}_\ell(\{A^*A, B^*B\}, \boldsymbol{v}_1)$ denote the subspace spanned by the first ℓ vectors in the sequence (2.5). This definition is compatible with those in [17, 24]. We refer to $\mathrm{g}\mathcal{K}_\ell(\{A^*A, B^*B\}, \boldsymbol{v}_1)$ as a generalized Krylov subspace.

The subspace $\text{span}\{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{\ell+1}\}$ is generated from the vectors $\boldsymbol{u}_1$ and $A\boldsymbol{v}_i$ for $i = 1,2,\ldots,\ell$. The latter vectors can be expressed in terms of $\boldsymbol{u}_1$. This yields

$$\begin{array}{ll} \text{Group 0:} & \boldsymbol{u}_1, \\ \text{Group 1:} & AA^*\boldsymbol{u}_1, \\ \text{Group 2:} & (AA^*)^2\boldsymbol{u}_1,\; A(B^*B)A^*\boldsymbol{u}_1, \\ \text{Group 3:} & (AA^*)^3\boldsymbol{u}_1,\; A(B^*B)(A^*A)A^*\boldsymbol{u}_1,\; A(A^*A)(B^*B)A^*\boldsymbol{u}_1,\; A(B^*B)^2A^*\boldsymbol{u}_1, \\ \quad\vdots & \vdots \end{array}$$

Finally, $\text{span}\{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_\ell\}$ is generated from the vectors $B\boldsymbol{v}_i$ for $i = 1,2,\ldots,\ell$.

Algorithm 2.2 evaluates matrix-vector products with the matrices A, B, A^*, and B^* periodically in the order indicated. It may readily be verified that interchanging the matrices A and B (in lines 5–10 and lines 11–16) is irrelevant. Interchanging A^* and B^* (in lines 17–22 and lines 23–28) changes the matrices H and K as well as the order of the $\boldsymbol{v}$-vectors, but does not change the span of the columns of the matrices U, V, and W.

The following result provides the relation of Algorithm 2.2 to Golub–Kahan bidiagonalization.

PROPOSITION 2.5. *Let the initial vector $\boldsymbol{u}_1 \in \mathbb{R}^m$ for Algorithms 2.1 and 2.2 be such that breakdown does not occur in these algorithms. Let the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{N(\boldsymbol{v})}$ be determined by Algorithm 2.2 with $\boldsymbol{v}_1 = A^*\boldsymbol{u}_1/\|A^*\boldsymbol{u}_1\|$. Then for k sufficiently small,*

$$\begin{aligned}
&\mathcal{K}_k(A^*A, A^*\boldsymbol{u}_1) \subseteq \mathrm{g}\mathcal{K}_{N(\boldsymbol{v})}(\{A^*A, B^*B\}, A^*\boldsymbol{u}_1) = \mathrm{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{N(\boldsymbol{v})}\},\\
&\mathcal{K}_k(B^*B, A^*\boldsymbol{u}_1) \subseteq \mathrm{g}\mathcal{K}_{N(\boldsymbol{v})}(\{A^*A, B^*B\}, A^*\boldsymbol{u}_1),
\end{aligned}$$

where the Krylov subspace $\mathcal{K}_k(A^*A, A^*\boldsymbol{u}_1)$ is defined by (2.1) and $\mathcal{K}_k(B^*B, A^*\boldsymbol{u}_1)$ is defined analogously. Thus, the solution subspace $\mathcal{K}_k(A^*A, A^*\boldsymbol{u}_1)$ determined by k steps of Golub–Kahan bidiagonalization is a subset of the solution subspace computed by Algorithm 2.2. The dimension k of the former subspace can be chosen to be an increasing function of $N(\boldsymbol{v})$.

Proof. The result follows by comparing the subspaces (2.1) and (2.5). □

The solution subspace determined by Golub–Kahan bidiagonalization is used by the LSQR iterative method; see Section 4 for comments on this method.

2.3. Reduction of matrix tuplets. The reduction method described in the above subsection can be generalized to a reduction method for matrix $(q+1)$-tuplets $\{A, B_1, B_2, \ldots, B_q\}$, where $A \in \mathbb{R}^{m\times n}$ and $B_r \in \mathbb{R}^{p_r\times n}$ for $1 \le r \le q$. Recall that Algorithm 2.2 generates the decompositions (2.3). An extension of this algorithm can be devised for determining the decompositions

$$\begin{aligned}
AV_\ell &= U_{\ell+1}H_{\ell+1,\ell},\\
A^*U_\ell &= V_{(\ell-2)(q+1)+2}\,(H_{\ell,(\ell-2)(q+1)+2})^*,\\
B_rV_\ell &= W_\ell^{(r)}K_{\ell,\ell}^{(r)},\\
B_r^*W_\ell^{(r)} &= V_{(\ell-1)(q+1)+r+2}\,(K_{\ell,(\ell-1)(q+1)+r+2}^{(r)})^*,
\end{aligned} \tag{2.6}$$

for $r = 1, 2, \ldots, q$. The matrices U, V, and $W^{(r)}$ have orthonormal columns with the first column of U a specified unit vector. The matrix H is of upper Hessenberg form and the matrices $K^{(r)}$ are upper triangular. Some entries above the diagonal of these matrices vanish, but the number of vanishing entries typically decreases when q increases. The following result is analogous to Theorem 2.4. Throughout this paper $\boldsymbol{e}_1$ denotes the first axis vector.

THEOREM 2.6. *Let $A \in \mathbb{R}^{m\times n}$, and $B_r \in \mathbb{R}^{p_r\times n}$, $1 \le r \le q$. (a) Given $\boldsymbol{u}_1 \in \mathbb{R}^n$ of unit length, there are orthogonal matrices $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{n\times n}$, and $W^{(r)} \in \mathbb{R}^{p_r\times p_r}$ for $1 \le r \le q$, with $U\boldsymbol{e}_1 = \boldsymbol{u}_1$, such that the matrices*

$$\begin{aligned}
&U^*AV = H \equiv [h_{i,j}],\\
&(W^{(r)})^*B_rV = K_\ell^{(r)} \equiv [k_{i,j}^{(r)}] \qquad \textit{for}\ \ 1 \le r \le q,
\end{aligned}$$

have the following properties:

- *The matrix H is upper Hessenberg and in addition $h_{i,j} = 0$ if* $\begin{cases} j > 1, & \text{if } i = 1, \\ j > (i-2)(q+1)+2, & \text{if } i > 1. \end{cases}$
- *The matrices $K^{(r)}$, for $1 \leq r \leq q$, are upper triangular and in addition $k^{(r)}_{i,j} = 0$ if $j > (i-1)(q+1)+r+2$.*

Moreover, the elements

$$\begin{aligned} &h_{1,1}, \quad h_{i+1,i} \quad \textit{for all } i, \quad h_{i,(i-2)(q+1)+2} \quad \textit{for } i > 1, \\ &k^{(r)}_{i,i} \quad \textit{for all } i, \quad k^{(r)}_{i,(i-1)(q+1)+r+2} \quad \textit{for all } i, \quad \textit{for } 1 \leq r \leq q, \end{aligned} \tag{2.7}$$

are nonnegative.
(b) Given $\boldsymbol{u}_1$, the matrices H and K are uniquely determined if the elements (2.7) *are nonzero (and hence positive).*
(c) Given $\boldsymbol{u}_1$, the matrices U, V, and W are uniquely determined if the matrices A and B_r are square for $1 \leq r \leq q$, and the elements (2.7) *are nonzero.*

Theorem 2.6 and the associated reduction algorithm are of interest for multi-parameter Tikhonov regularization. This application is considered in Section 3.2 below.

3. Applications to Tikhonov regularization.

We first look at the application of the decomposition (2.3) to the solution of large-scale Tikhonov problems (1.4) and subsequently discuss how the reduction (2.6) can be applied to multi-parameter Tikhonov regularization.

3.1. One-parameter Tikhonov regularization.

Let the starting vector for Algorithm 2.2 be

$$\boldsymbol{u}_1 = \boldsymbol{b} \,/\, \|\boldsymbol{b}\|,$$

and choose the number of steps ℓ of the algorithm small enough so that exit at line 4 of Algorithm 2.2 is avoided. Then the algorithm determines the decompositions (2.3). Let $\mathcal{R}(M)$ denote the range of of the matrix M. Since $\boldsymbol{v}_1 = A^*\boldsymbol{u}_1/h_{1,1}$, equation (2.5) implies that

$$\mathrm{g}\mathcal{K}_\ell(\{A^*A, B^*B\}, \boldsymbol{b}) \subset \mathcal{R}(A^*) \cup \mathcal{R}(B^*BA^*).$$

Let $\boldsymbol{x} \in \mathrm{g}\mathcal{K}_\ell(\{A, B\}, \boldsymbol{b})$ and let $\boldsymbol{y} \in \mathbb{R}^\ell$ be such that $\boldsymbol{x} = V_\ell \boldsymbol{y}$. Then

$$\begin{aligned} &\|A\boldsymbol{x} - \boldsymbol{b}\|^2 + \mu\,\|B\boldsymbol{x}\|^2 \\ &\quad = \|AV_\ell\,\boldsymbol{y} - \boldsymbol{b}\|^2 + \mu\,\|BV_\ell\,\boldsymbol{y}\|^2 \\ &\quad = \|U_{\ell+1}H_{\ell+1,\ell}\,\boldsymbol{y} - \boldsymbol{b}\|^2 + \mu\,\|W_\ell K_{\ell,\ell}\boldsymbol{y}\|^2 \\ &\quad = \|H_{\ell+1,\ell}\,\boldsymbol{y} - \|\boldsymbol{b}\|\,\boldsymbol{e}_1\|^2 + \mu\,\|K_{\ell,\ell}\boldsymbol{y}\|^2. \end{aligned}$$

Therefore, solving (1.4) over the generalized Krylov subspace $\mathrm{g}\mathcal{K}_\ell(\{A, B\}, \boldsymbol{b})$ results in the solution of the projected small-sized problem

$$\min_{\boldsymbol{y}\in\mathbb{R}^\ell} \{\, \|H_{\ell+1,\ell}\,\boldsymbol{y} - \|\boldsymbol{b}\|\,\boldsymbol{e}_1\|^2 + \mu\,\|K_{\ell,\ell}\boldsymbol{y}\|^2 \,\}. \tag{3.1}$$

We may solve this problem with the aid of the GSVD of the pair $\{H_{\ell+1,\ell}, K_{\ell,\ell}\}$. Since $K_{\ell,\ell}$ is usually of full rank and not ill-conditioned, we also can compute the solution avoiding the GSVD by using the QR factorization

$$K_{\ell,\ell} = QR. \tag{3.2}$$

Here $Q \in \mathbb{R}^{\ell\times\ell}$ is orthogonal, and $R \in \mathbb{R}^{\ell\times\ell}$ is upper triangular and nonsingular. When we substitute (3.2) and $\boldsymbol{z} = R\boldsymbol{y}$ into (3.1) we get a Tikhonov minimization problem in standard form,

$$\min_{\boldsymbol{z}\in\mathbb{R}^\ell} \{ \|\widetilde{H}\boldsymbol{z} - \|\boldsymbol{b}\|\, \boldsymbol{e}_1\|^2 + \mu\, \|\boldsymbol{z}\|^2 \}, \tag{3.3}$$

where $\widetilde{H}$ is defined by

$$\widetilde{H} = H_{\ell+1,\ell}R^{-1}. \tag{3.4}$$

The entries of $\widetilde{H}$, or equivalently of $\widetilde{H}^*$, can conveniently be computed by forward substitution with the matrix R^*. Since the size of $\widetilde{H}$ is small, we can use its singular value decomposition (SVD) to solve (3.3).

Let $\boldsymbol{z}_\mu^{(\ell)}$ denote the solution of (3.3) for a fixed $\mu > 0$. The corresponding approximate solution of (1.4) is

$$\boldsymbol{x}_\mu^{(\ell)} = V_\ell R^{-1}\boldsymbol{z}_\mu^{(\ell)}. \tag{3.5}$$

The value of ℓ is generally modest, but sufficiently large such that the approximations $\boldsymbol{x}_\mu^{(\ell)}$ for μ-values of interest yield sensible approximations to the associated solutions $\boldsymbol{x}_\mu$ of (1.4). The residual corresponding to $\boldsymbol{x}_\mu^{(\ell)}$,

$$\boldsymbol{r}_\mu^{(\ell)} = \boldsymbol{b} - A\boldsymbol{x}_\mu^{(\ell)},$$

is often called the discrepancy.

A variety of approaches can be used to determine a suitable value of the regularization parameter μ, including the L-curve, generalized cross validation, and the discrepancy principle; see, e.g., [4, 8, 11, 16, 23] for discussions and references. We will use the discrepancy principle in the computed examples of Section 4. It requires that a bound ε for the norm of the error $\boldsymbol{e}$ in $\boldsymbol{b}$ is available, c.f. (1.3), and determines $\mu > 0$ so that the solution $\boldsymbol{z}_\mu^{(\ell)}$ of (3.3) satisfies

$$\|\widetilde{H}\boldsymbol{z}_\mu^{(\ell)} - \|\boldsymbol{b}\|\, \boldsymbol{e}_1\| = \eta\, \varepsilon, \tag{3.6}$$

where $\eta \geq 1$ is a user-specified constant independent of ε. Let $\mu_\ell > 0$ be such that the associated solution $\boldsymbol{z}_{\mu_\ell}^{(\ell)}$ of (3.3) solves (3.6) with $\mu = \mu_\ell$. The value μ_ℓ of the regularization parameter may be computed by using the SVD of $\widetilde{H}$. The following proposition gives properties of the left-hand side of (3.6) as a function of $\nu = \mu^{-1}$.

Proposition 3.1. *Let the matrix $\widetilde{H}$ be defined by (3.4) and assume that $\widetilde{H}^*\boldsymbol{e}_1 \neq \boldsymbol{0}$. Let $P_{\mathcal{N}(\widetilde{H}^*)}$ denote the orthogonal projector onto $\mathcal{N}(\widetilde{H}^*)$. Then the function*

$$\varphi(\nu) = \|\boldsymbol{b}\|^2\boldsymbol{e}_1^*(\nu\widetilde{H}\widetilde{H}^* + I)^{-2}\boldsymbol{e}_1, \qquad \nu \geq 0,$$

is strictly decreasing, convex, and

$$\varphi(0) = \|\boldsymbol{b}\|^2, \qquad \lim_{\nu\to\infty}\varphi(\nu) = \|P_{\mathcal{N}(\widetilde{H}^*)}\boldsymbol{e}_1\|^2\|\boldsymbol{b}\|^2.$$

In addition,

$$\varphi(\mu^{-1}) = \|\widetilde{H}\boldsymbol{z}_\mu^{(\ell)} - \|\boldsymbol{b}\|\, \boldsymbol{e}_1\|^2.$$

Proof. The result can be shown with the aid of the SVD of $\widetilde{H}$; see [6, Theorem 2.1] for details. □

Generally, $\|P_{\mathcal{N}(\widetilde{H}^*)}\boldsymbol{e}_1\|\|\boldsymbol{b}\| \ll \eta\,\varepsilon < \|\boldsymbol{b}\|$. Then, by Proposition 3.1, the equation

$$\varphi(\nu) = \eta^2\varepsilon^2 \tag{3.7}$$

has a unique solution $\widehat{\nu}$. We determine the value $\mu_\ell = \widehat{\nu}^{-1}$ of the regularization parameter in (3.3) and compute the approximate solution $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ of (1.4) with the aid of (3.5).

We may apply Newton's method to solve (3.7) with a starting value $\nu_0 < \widetilde{\nu}$, such as $\nu_0 = 0$. This method converges monotonically and quadratically for the function $\varphi(\nu)$ since this function is monotonically decreasing and convex.

We conclude this subsection by noting that when $B = I$, the Tikhonov regularization method described simplifies to the method discussed in [6], which is based on (standard) partial Golub–Kahan bidiagonalization of A.

3.2. Multi-parameter Tikhonov regularization. We briefly discuss multi-parameter Tikhonov regularization

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \Big\{\, \|A\boldsymbol{x} - \boldsymbol{b}\|^2 + \sum_{r=1}^{q} \mu_r\,\|B_r\boldsymbol{x}\|^2 \Big\}, \tag{3.8}$$

where the B_r are regularization matrices and the scalars $\mu_r \geq 0$ regularization parameters. Brezinski et al. [3] consider this problem for small to moderately sized problems. More recent treatments are provided by Gazzola and Novati [9] and Lu and Pereverzyev [18].

With $\boldsymbol{x} = V_\ell\boldsymbol{y}$ as before, and applying the decompositions (2.6), we get similarly as for (3.1) that (3.8) minimized over $\mathcal{R}(V_\ell)$ is equivalent to the reduced minimization problem

$$\min_{\boldsymbol{y}\in\mathbb{R}^\ell} \Big\{\, \|H_{\ell+1,\ell}\,\boldsymbol{y} - \|\boldsymbol{b}\|\,\boldsymbol{e}_1\|^2 + \sum_{r=1}^{q} \mu_r\,\|K_{\ell,\ell}^{(r)}\boldsymbol{y}\|^2 \Big\}.$$

Methods for determining suitable regularization parameters for this minimization problem are discussed in [2, 3, 9, 18].

4. Numerical examples. We present several numerical examples with the one-parameter Tikhonov regularization method of Section 3.1. To illustrate the importance of the use of a user-chosen regularization matrix, we also report results obtained with the LSQR iterative method, which determines an approximate solution in a Krylov subspace of the form (2.2) with $\boldsymbol{u}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$. The ℓth iterate, $\boldsymbol{x}_\ell$, determined by LSQR when applied to the approximate solution of (1.2), satisfies

$$\|A\boldsymbol{x}_\ell - \boldsymbol{b}\| = \min_{\boldsymbol{x}\in\mathbb{K}_\ell(A^*A,A^*\boldsymbol{b})} \|A\boldsymbol{x} - \boldsymbol{b}\|, \qquad \boldsymbol{x}_\ell \in \mathbb{K}_\ell(A^*A, A^*\boldsymbol{b});$$

see [11, 20] for details. In all iterative methods applied in this section, the initial iterate is chosen to be $\boldsymbol{x}_0 = \boldsymbol{0}$.

The components of the noise vector $\boldsymbol{e}$ are in all examples normally distributed with mean zero, and $\boldsymbol{e}$ is normalized to correspond to a chosen noise level

$$\delta = \|\boldsymbol{e}\| \,/\, \|\check{\boldsymbol{b}}\|.$$

Here $\breve{\boldsymbol{b}}$ denotes the noise-free vector associated with $\boldsymbol{b}$; cf. (1.3). We determine the regularization parameter μ by the discrepancy principle where η in (3.6) equals one.

The iterations are terminated as soon as the computed iterates $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ and the regularization terms $\mu_\ell\|B\boldsymbol{x}_{\mu_\ell}^{(\ell)}\|^2$ do not change much. Our reason for considering the regularization terms is that we would like to minimize (1.4) and the fidelity terms $\|A\boldsymbol{x}_{\mu_\ell}^{(\ell)} - \boldsymbol{b}\|^2$ are fixed since we impose the discrepancy principle. Specifically, we terminate the iterations at step ℓ when for the first time

$$\|\boldsymbol{x}_{\mu_\ell}^{(\ell)} - \boldsymbol{x}_{\mu_{\ell-1}}^{(\ell-1)}\|/\|\boldsymbol{x}_{\mu_\ell}^{(\ell)}\| < \gamma \tag{4.1}$$

and

$$|\mu_\ell\|B\boldsymbol{x}_{\mu_\ell}^{(\ell)}\|^2 - \mu_{\ell-1}\|B\boldsymbol{x}_{\mu_{\ell-1}}^{(\ell-1)}\|^2|/\|B\boldsymbol{x}_{\mu_\ell}^{(\ell)}\|^2 < \gamma. \tag{4.2}$$

We let $\gamma = 1\cdot 10^{-3}$ in the examples.

Examples 4.1 and 4.2 illustrate the performance when our Tikhonov regularization method is used for the solution of linear discrete ill-posed problems from Regularization Tools [12] with rectangular regularization matrices. These problems are small enough to allow the application of the GSVD for their solution. These examples illustrate that Algorithm 2.2 can produce approximate solutions of the same or higher quality than factorization of the matrix pair $\{A, B\}$ by GSVD.

Examples 4.3-4.5 consider the restoration of blurred and noisy 2D-images. These examples are too large to compute the GSVD of the matrix pair $\{A, B\}$. Moreover, they use a regularization matrix, for which it is not attractive to use the A-weighted pseudoinverse. The starting vector for Algorithm 2.2 is $\boldsymbol{u}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$ for all examples. All computations were carried out in MATLAB with about 15 significant decimal digits.

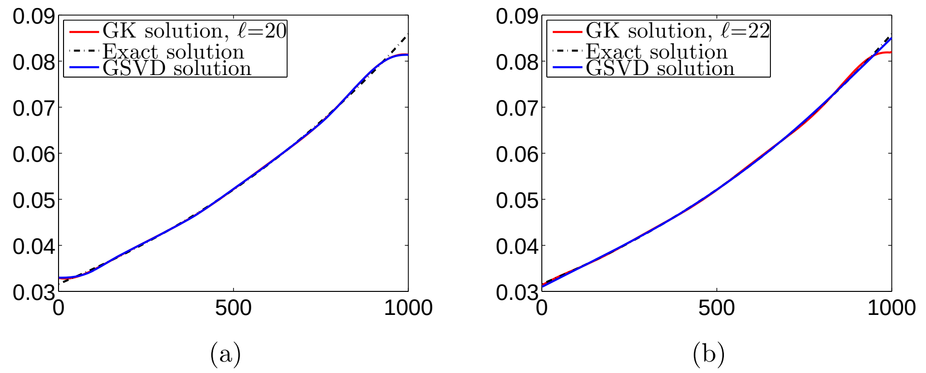

Fig. 4.1. *Example 4.1. Computed approximate solutions for the test problem* deriv2 *with noise level* 10^{-3}. *(a) B is the bidiagonal matrix (4.4). (b) B is the tridiagonal matrix (4.5). The curves determined with the GSVD are on top of the curves obtained with Algorithm 2.2.*

Example 4.1. Consider the Fredholm integral equation of the first kind

$$\int_0^1 k(s,t)\,x(t)\,dt = \exp(s) + (1-e)\,s - 1, \qquad 0 \le s \le 1, \tag{4.3}$$

B	Algorithm 2.2	GSVD	LSQR
Bidiagonal (4.4)	$1.17 \cdot 10^{-2}$ $(\ell = 20)$	$1.23 \cdot 10^{-2}$	$9.55 \cdot 10^{-1}$ $(\ell = 13)$
Tridiagonal (4.5)	$9.93 \cdot 10^{-3}$ $(\ell = 22)$	$3.41 \cdot 10^{-3}$	

Table 4.1

Example 4.1. Relative errors in computed solutions for noise level 10^{-3}. The parameter ℓ denotes the number of steps with Algorithm 2.2 and LSQR.

where

$$k(s,t) = \begin{cases} s\,(t-1), & s < t, \\ t\,(s-1), & s \geq t. \end{cases}$$

We discretize the integral equation by a Galerkin method with orthonormal box test and trial functions using the MATLAB code deriv2 from [12]. This yields the matrix $A \in \mathbb{R}^{1000\times 1000}$. The function deriv2 also produces the "exact" solution $\breve{\boldsymbol{x}} \in \mathbb{R}^{1000}$, which is a scaled discrete approximation of the analytical solution $x(t) = \exp(t)$ of (4.3). We determine the "noise-free" vector $\breve{\boldsymbol{b}} := A\breve{\boldsymbol{x}}$ to which we add an error vector $\boldsymbol{e} \in \mathbb{R}^{1000}$ that corresponds to the noise level 10^{-3}; cf. (1.3). Bidiagonal and tridiagonal regularization matrices that approximate derivatives are applied. Thus, the bidiagonal regularization matrix is a scaled finite difference approximation of the first derivative operator,

$$B = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n}, \tag{4.4}$$

and the tridiagonal regularization matrix is a scaled finite difference approximation of the second derivative operator,

$$B = \begin{bmatrix} -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \end{bmatrix} \in \mathbb{R}^{(n-2)\times n}. \tag{4.5}$$

We apply Algorithm 2.2 to the matrix pair $\{A, B\}$ and determine the regularization parameter with the aid of the discrepancy principle. Figure 4.1 displays the approximate solutions $\boldsymbol{x}_{\mu_{20}}^{(20)}$ and $\boldsymbol{x}_{\mu_{22}}^{(22)}$ of (1.2) determined by the method of Section 3.1 (red continuous curves), the approximate solutions computed by using the GSVD of the matrix pair $\{A, B\}$ (blue continuous curves), and the desired solution $\breve{\boldsymbol{x}}$ (black dash-dotted curves). The regularization parameter for the GSVD approximate solutions is determined by the discrepancy principle using the MATLAB function discrep from [12]. Clearly, the regularization matrix (4.5) gives the best approximation of $\breve{\boldsymbol{x}}$.

Table 4.1 shows numerical values for the relative errors in the computed approximate solutions. The number of steps ℓ with Algorithm 2.2 is determined by the stopping criteria (4.1) and (4.2). When B is defined by (4.4), Algorithm 2.2 determines the iterate $\boldsymbol{x}_{\mu_{20}}^{(20)}$, which is a better approximation of $\breve{\boldsymbol{x}}$ than the approximation determined by GSVD. LSQR delivers solutions with a much larger relative error than either Algorithm 2.2 or the GSVD. This illustrates the benefit of the regularization

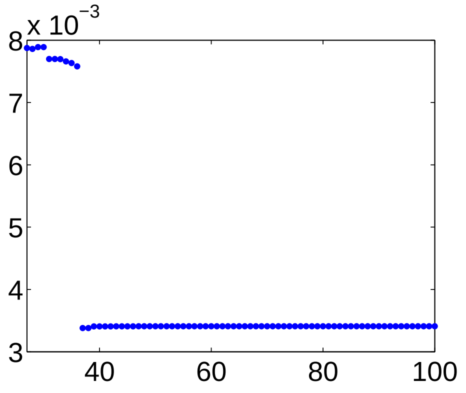

FIG. 4.2. *Example 4.1. Relative error in computed approximate solutions $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ as a function of $\ell \geq 27$ for the test problem* deriv2 *with noise level 10^{-3} and the tridiagonal regularization matrix (4.5).*

matrix (4.4). If instead the regularization matrix B is given by (4.5), then the stopping criteria (4.1) and (4.2) terminates Algorithm 2.2 after 22 steps. For this regularization matrix, GSVD delivers a more accurate approximation of $\breve{\boldsymbol{x}}$ than Algorithm 2.2. The algorithm yields a more accurate approximation of $\breve{\boldsymbol{x}}$ when it is allowed to carry out more iterations. For instance, the relative error in $\boldsymbol{x}_{\mu_{37}}^{(37)}$ is only $3.38 \cdot 10^{-3}$, which is smaller than the relative error in the approximate solution determined by the GSVD.

Figure 4.2 shows the error in the computed approximate solutions $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ as a function of the number of steps ℓ when the tridiagonal regularization matrix (4.5) is used. The error can be seen to drop significantly at iteration $\ell = 37$. It is clear that for this example it is not crucial when the iterations are terminated as long as sufficiently many of them are carried out. The stopping criteria (4.1) and (4.2) gives quite accurate approximations of $\breve{\boldsymbol{x}}$ for many problems, but it is an open question how to design a stopping criterion that yields the most accurate approximation of $\breve{\boldsymbol{x}}$ and does not require unnecessarily many iterations for a large variety of problems. We remark that the relative errors in the iterates $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ for $1 \leq \ell < 27$ are not shown in Figure 4.2, because some of them are much larger than the errors displayed and would obscure the behavior of the relative errors shown.

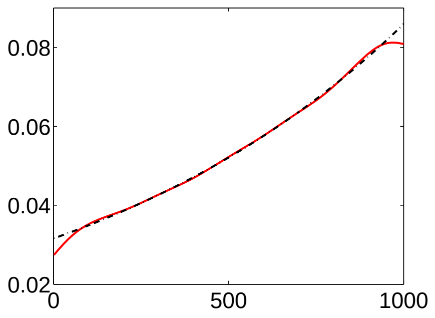

FIG. 4.3. *Example 4.1. Approximate solution $\boldsymbol{x}_{10}$ (continuous red curve) computed by the method [13] for test problem* deriv2 *with noise level 10^{-3} and regularization matrix (4.5). The dashed black curve displays $\hat{\boldsymbol{x}}$.*

We conclude this example with an illustration of the performance of the method described in [13]. The method determines solution subspaces that are the union of the null space of (4.5) and a Krylov subspace of the form (2.2) with $\boldsymbol{u}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$. The regularization matrix (4.5) is projected onto the solution subspaces. The discrepancy principle is satisfied after 10 iterations, and the relative error in the iterate $\boldsymbol{x}_{10}$ is $1.68 \cdot 10^{-2}$. This relative error is larger than the relative error achieved with Algorithm 2.2. Figure 4.3 displays $\boldsymbol{x}_{10}$ (solid red curve) and the desired solution $\hat{\boldsymbol{x}}$ (dashed black curve). $\Box$

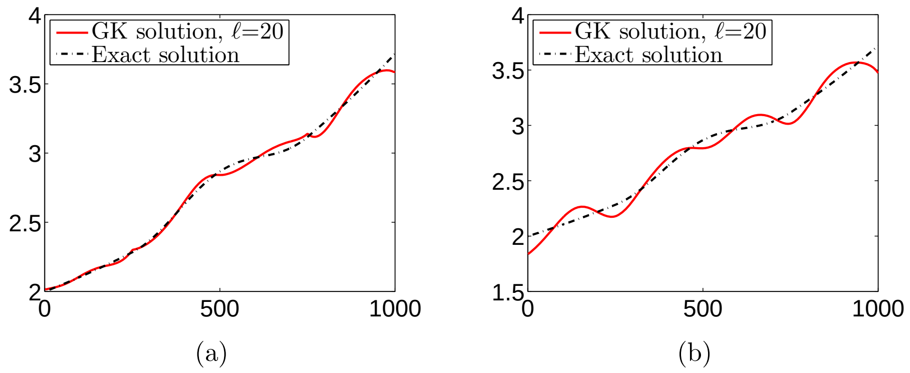

FIG. 4.4. *Example 4.2. Computed approximate solutions for the modified* phillips *test problem with noise level* 10^{-2}. *(a) Tikhonov solution* $\boldsymbol{x}^{(20)}_{\mu_{20}}$ *with* B *defined by* (4.4). *(b) Tikhonov solution* $\boldsymbol{x}^{(20)}_{\mu_{20}}$ *with* B *defined by* (4.5).

B	Algorithm 2.2
Bidiagonal	$1.16 \cdot 10^{-2}$ $(\ell = 20)$
Tridiagonal	$2.64 \cdot 10^{-2}$ $(\ell = 20)$

TABLE 4.2
Example 4.2. Relative error in computed solutions for noise level 10^{-2}. *The parameter* ℓ *denotes the number of steps with Algorithm 2.2.*

B	Algorithm 2.2
Bidiagonal	$6.55 \cdot 10^{-3}$ $(\ell = 28)$
Tridiagonal	$8.52 \cdot 10^{-3}$ $(\ell = 20)$

TABLE 4.3
Example 4.2. Relative error in computed solutions for noise level 10^{-3}. *The parameter* ℓ *denotes the number of steps with Algorithm 2.2.*

EXAMPLE 4.2. Consider the Fredholm integral equation of the first kind

$$\int_{-6}^{6} k(s,t)\, x(t)\, dt = g(s), \qquad -6 \le s \le 6, \tag{4.6}$$

discussed by Phillips [22]. Define the function

$$\phi(s) = \begin{cases} 1 + \cos(\frac{\pi s}{3}), & |s| < 3, \\ 0, & |s| \ge 3. \end{cases}$$

Then the kernel k, the solution x, and the right-hand side g are given by

$$\begin{aligned} k(s,t) &= \phi(s-t), \\ x(t) &= \phi(t), \\ g(s) &= (6-|s|)\big(1+\tfrac{1}{2}\cos(\tfrac{\pi s}{3})\big)+\tfrac{9}{2\pi}\sin(\tfrac{\pi|s|}{3}). \end{aligned}$$

We use the MATLAB code phillips from [12] to discretize (4.6). This yields the matrix $A \in \mathbb{R}^{1000\times 1000}$ and a discretization of a scaled solution $\boldsymbol{z}_0 \in \mathbb{R}^{1000}$. We add a discretization of the function $1+\exp\left(\frac{t+6}{12}\right)$, $t \in [-6,6]$, to $\boldsymbol{z}_0$ to obtain a vector $\breve{\boldsymbol{x}}$ which represents a slowly oscillatory and increasing solution. The vector $\breve{\boldsymbol{b}} := A\breve{\boldsymbol{x}}$ yields "error-free" data to which we add a noise-vector $\boldsymbol{e}$; cf. (1.3). The vector $\boldsymbol{e}$ is chosen to correspond to the noise levels 10^{-2} or 10^{-3}. We apply Algorithm 2.2 to determine approximations of the desired vector $\breve{\boldsymbol{x}}$ using the regularization matrices (4.4) and (4.5). The number of steps, ℓ, is chosen in the same manner as in Example 4.1. The results achieved are displayed by Tables 4.2 and 4.3 for the noise levels 10^{-2} and 10^{-3}, respectively, as well as in Figure 4.4.

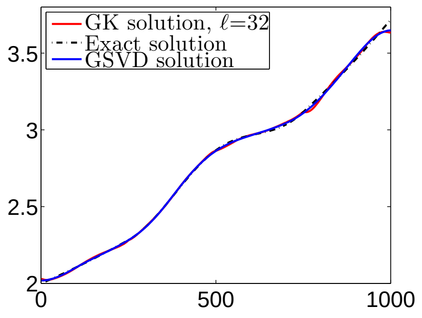

FIG. 4.5. *Example 4.2. Computed approximate solutions for the modified* phillips *test problem with Cauchy noise of level* 10^{-3}. *The approximate solution* $\boldsymbol{x}_{\mu_{32}}^{(32)}$ *determined by Algorithm 2.2, the approximate solution determined by GSVD, and the desired solution* $\breve{\boldsymbol{x}}$ *are shown. The regularization matrix* B *is given by* (4.4).

Finally, we investigate the performance of Algorithm 2.2 when applied to the solution of (1.4) when the matrix A has more rows than columns. Specifically, we define $A \in \mathbb{R}^{2000\times 1000}$ by stacking two copies of the matrix obtained by discretizing (4.6) as described above. The data vector $\boldsymbol{b} \in \mathbb{R}^{2000}$ in (1.4) is determined by stacking two error-free vectors $\breve{\boldsymbol{b}} \in \mathbb{R}^{1000}$ defined as above and then adding 0.1% Cauchy noise. We apply Algorithm 2.2 with regularization matrix (4.4) to determine an approximate solution of the stacked problem. The discrepancy principle and the stopping criteria (4.1) and (4.2) are satisfied after $\ell = 32$ steps with Algorithm 2.2. The computed approximate solution $\boldsymbol{x}_{\mu_{32}}^{(32)}$ has relative error $\|\boldsymbol{x}_{\mu_{32}}^{(32)} - \breve{\boldsymbol{x}}\|/\|\breve{\boldsymbol{x}}\| = 4.77\cdot 10^{-3}$. We remark that the regularization matrix (4.4) gives a more accurate approximation of $\breve{\boldsymbol{x}}$ than the matrix (4.5). Figure 4.5 displays $\boldsymbol{x}_{\mu_{32}}^{(32)}$, the approximate solution determined by GSVD, and the desired solution $\breve{\boldsymbol{x}}$. The approximate solutions determined by Algorithm 2.2 and GSVD are very close. Note that if the noise were Gaussian, the overdetermined system could be reduced to a system with a square matrix. □

The following examples illustrate the application of Algorithm 2.2 to the restoration of 2D gray-scale images that have been contaminated by blur and noise. The

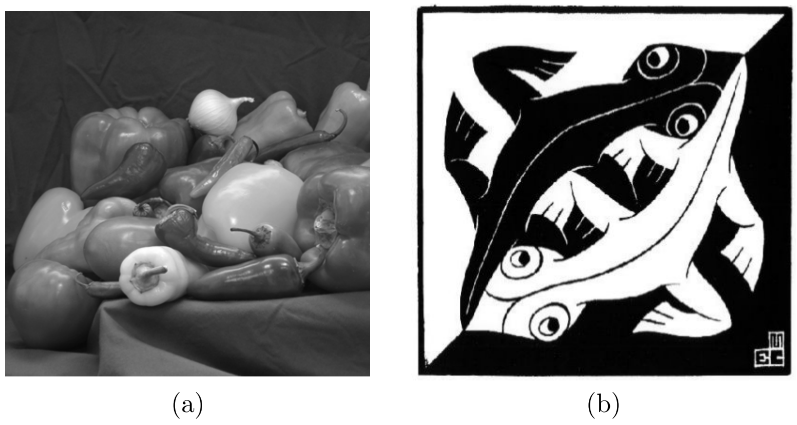

FIG. 4.6. *Blur- and noise-free images for Examples 4.3 and 4.4.*

images are given by an $m \times m$ matrix of pixel values in the set $\{0, 1, \dots, 255\}$. The pixels are ordered row-wise and stored in a vector of dimension $n = m^2$. A blurred but noise-free image $\breve{\boldsymbol{b}}$ is obtained from the original image $\breve{\boldsymbol{x}} \in \mathbb{R}^n$ by multiplication by a blurring matrix $A \in \mathbb{R}^{n \times n}$. In the first two examples below, this matrix represents Gaussian blur and is a symmetric block Toeplitz matrix with Toeplitz blocks. In our last example the matrix models motion blur. The block Toeplitz matrix is generated with the MATLAB function blur from [12]. This matrix depends on the parameters band (the half-bandwidth of the Toeplitz blocks) and sigma (the variance of the Gaussian point spread function). Increasing sigma means more blur, while increasing band means that more storage and computational work is needed for the matrix-vector product evaluations with A and A^*, and also (to some extent) more blur. We obtain a blurred and noisy image $\boldsymbol{b} \in \mathbb{R}^n$ by adding a noise-vector $\boldsymbol{e} \in \mathbb{R}^n$ to $\breve{\boldsymbol{b}}$; cf. (1.3). We assume that the noise level δ is known and seek to approximate the original image $\breve{\boldsymbol{x}}$.

A measure of the quality of the approximations $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$ determined by Algorithm 2.2 and different regularization matrices is provided by the Peak Signal-to-Noise Ratio (PSNR),

$$\mathrm{PSNR}(\boldsymbol{x}_{\mu_\ell}^{(\ell)}, \widehat{\boldsymbol{x}}) = 20 \log_{10} \left(\tfrac{255}{\|\boldsymbol{x}_{\mu_\ell}^{(\ell)} - \widehat{\boldsymbol{x}}\|} \right) \mathrm{dB},$$

where 255 is the maximal pixel-value. While a high PSNR-value indicates a good restoration $\boldsymbol{x}_{\mu_\ell}^{(\ell)}$, we also display the obtained images since PSNR-values do not always coincide with visual judgment.

EXAMPLE 4.3. Consider the restoration of a blurred and noisy 384×384-pixel image. The desired noise- and blur-free image is displayed in Figure 4.6(a). We compare three regularization matrices B: the identity, a discretization of the Laplacian

$$L(x) = \Delta x, \tag{4.7}$$

and a discretization of the Perona–Malik operator

$$L(x) = \mathrm{div}(g(|\nabla x|^2)\, \nabla x), \tag{4.8}$$

where ∇x is the gradient of x viewed as a function on $\mathbb{R}^2$. We define the diffusivity

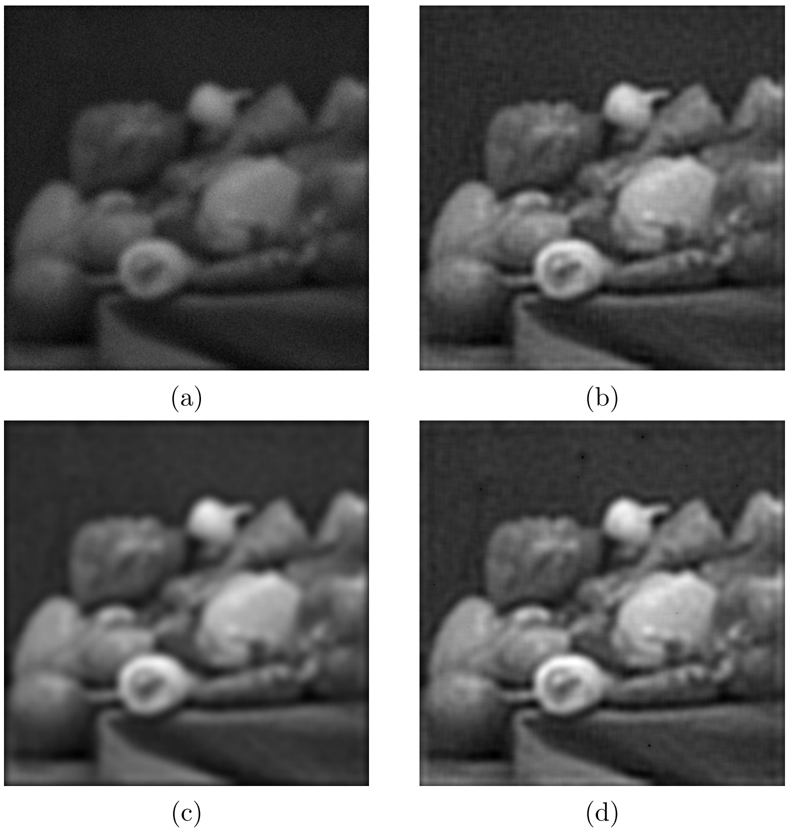

Fig. 4.7. *Example 4.3. (a) Blurred and noisy image. (b) Image restored with $B = I$. (c) Image restored with B the discretized Laplacian. (d) Image restored with B the discretized Perona–Malik operator.*

by

$$g(s) = \frac{\rho}{\rho + s}$$

where $\rho > 0$ is small; see [21] for a discussion. We use the value $\rho = 10^{-5}$ in the computed examples. Discretizing (4.8) by finite differences yields a matrix B with five nonzero entries per row; we refer to [24] for more details.

The regularization matrix B in (1.4) plays an important role. The discretization of the Perona–Malik operator (4.8) often provides better restorations with sharper edges than the discretization of the Laplacian (4.7), which usually yields over-smoothed restorations. We determine the discretized Perona–Malik operator B from the available image.

An image with blur and noise is generated with the MATLAB function blur from [12] as described above with band = 7 and sigma = 5. The noise level is $\delta = 10^{-1}$; see Figure 4.7(a). The best restorations for the regularization matrices $B = I$ and B determined by (4.7) are generated with $\ell = 6$ and $\ell = 7$ steps of Algorithm 2.2 and have PSNR-values 25.82 and 25.68, respectively; see Figures 4.7(b) and (c). The best restoration with B being a discretization of the Perona–Malik operator (4.8) is obtained with $\ell = 15$ steps of Algorithm 2.2. It has PSNR-value 28.32 and is shown in Figure 4.7(d). The higher PSNR-value of the latter restoration is in agreement with visual perception. While the evaluation of matrix-vector products with B is straightforward, the use of the inverse of B is not. This makes it attractive to apply the method of the present paper. $\square$

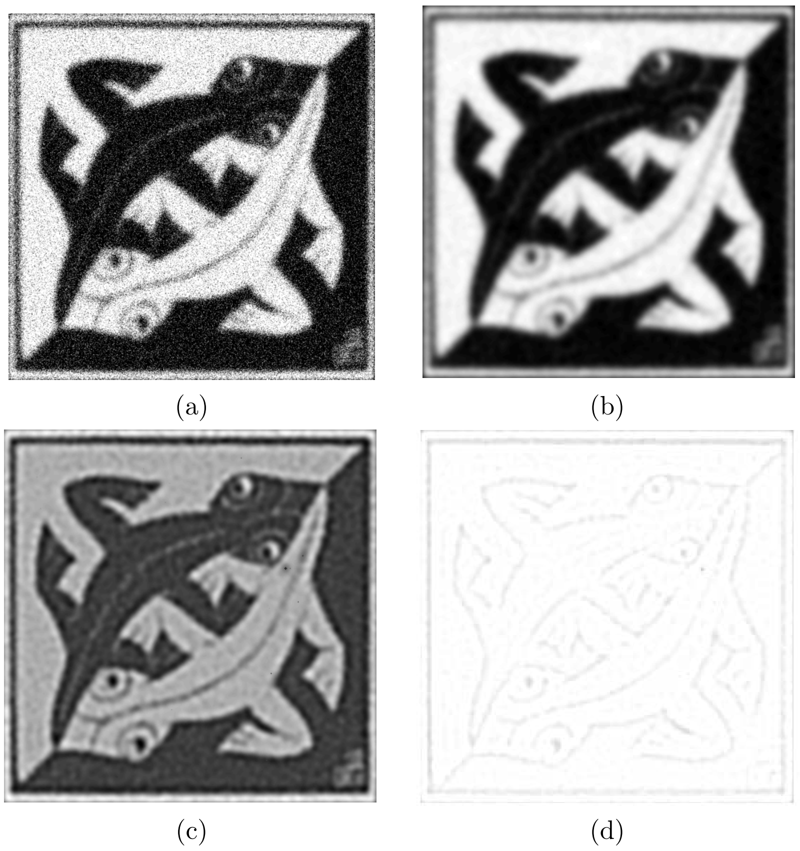

FIG. 4.8. *Example 4.4. (a) Blurred and noisy image. (b) Image restored with B being the discretized Laplacian. (c) Image restored with B being the discretized Perona–Malik operator. (d) Edge map for the restoration in (c).*

EXAMPLE 4.4. We apply the regularization matrices obtained by discretizing the operators (4.7) and (4.8) to the 412×412-pixel image shown in Figure 4.6(b). This image has many fine details and well-defined edges. We severely blur the image using $\mathsf{band} = 9$ and $\mathsf{sigma} = 3$ in the function blur from [12], and add 30% Gaussian noise; see Figure 4.8(a).

The discretizations of the Laplace and Perona–Malik operators (4.7) and (4.8) are the same as in Example 4.3. For B being the discrete Laplacian, the best restoration is obtained after $\ell = 1$ iterations with Algorithm 2.2; for B being the discretized Perona–Malik operator, the most accurate restoration is obtained after $\ell = 7$ iterations. We get the PSNR-values 12.11 and 13.56, respectively, for the restored images; see Figures 4.8(b) and (c). We see the over-smoothing caused by the discretized Laplacian.

We also use the regularization matrix $B_\otimes$ defined by

$$B_\otimes = \begin{bmatrix} I \otimes B \\ B \otimes I \end{bmatrix}, \tag{4.9}$$

where the matrix B is given by (4.4) with $n = 412$. The matrix $B_\otimes \in \mathbb{R}^{338664\times169744}$ has almost twice as many rows as columns. This kind of regularization matrix also is used in [15]. It yields fairly accurate restorations for many examples. However, for the present example, we obtain a restoration of the same quality as with the discrete Laplace operator; the best restoration is determined after $\ell = 1$ iteration and has PSNR 12.11. Nevertheless, it is of interest that our iterative method can handle regularization matrices with more rows than columns.

Figure 4.8(d) shows an edge map for the image restored with B being the discretized Perona–Malik operator. The edge map, determined with `gimp`, a public domain edge detector for image processing, is accurate also in the presence of the severe noise and blur present in the image.

In the above examples, we saved the blur- and noise-contaminated image in the format `uint8`, i.e., each pixel is stored as a nonnegative integer in eight bits. It is interesting to note that when the blur- and noise-contaminated image is saved in "double precision", i.e., as a real floating point numbers with 64 bits, the restored images are of much higher quality with PSNR-values about 16. Thus, the storage of each pixel in the `uint8` format adds additional error. The latter format requires much less computer storage than the double precision format. The choice of format typically depends on the computer hardware and application. □

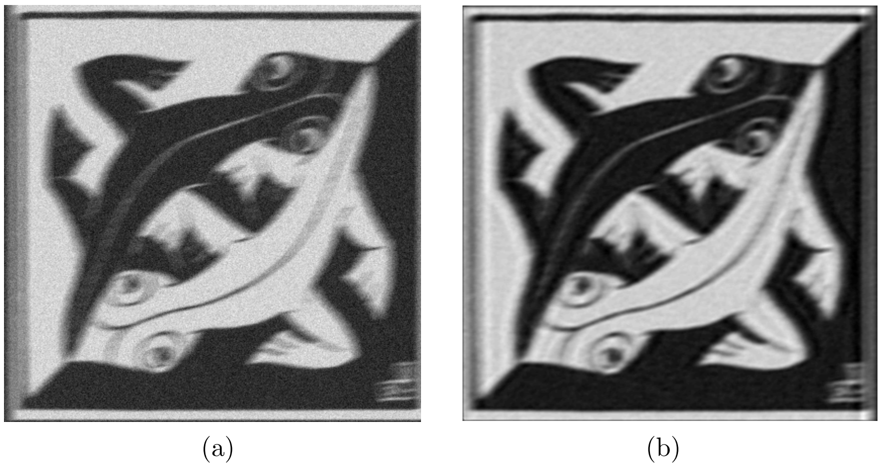

Fig. 4.9. *Example 4.5. (a) Blurred and noisy image. (b) Image restored with* 26 *steps of Algorithm 2.2.*

Example 4.5. Our last example is concerned with the restoration of an image that has been contaminated by linear motion blur and noise. The point spread function (PSF) for motion blur is represented by a line segment of length r pixels in the direction of the motion. The angle θ (in degrees) specifies the direction; it is measured counter-clockwise from the positive horizontal axis. The PSF takes on the value r^{-1} on this segment and vanishes elsewhere. The blurring matrix A is defined by this PSF. The larger r, the more ill-conditioned is A, and the more difficult is the restoration task. In this example, we let $r = 15$ and $\theta = 10$, and use the same test image as in Example 4.4. Thus, the original blur- and noise-free image, shown in Figure 4.6(b), is represented by 412×412 pixels. This image is contaminated by motion blur and by 10% Gaussian noise. Figure 4.9(a) displays the contaminated image, which is represented by the vector $\boldsymbol{b} \in \mathbb{R}^{412^2}$. The regularization matrix B is the Laplacian operator.

Algorithm 2.2 requires 26 iterations to satisfy the discrepancy principle and the stopping criteria (4.1) and (4.2). Figure 4.9(b) shows the restoration obtained; it has PSNR-value 14.09. The PSNR-values of the computed restorations increase with the iteration number. For instance, 100 steps give a restoration with PSNR 14.20. However, the latter restoration is not visually more pleasing than the restoration shown.

We remark that there are other approaches to solve minimization problems of the form (1.4) when the matrices A and B are square, such as iterative methods based on the flexible Arnoldi process [25, Algorithm 2.1]. However, for the present restoration problem [25, Algorithm 2.1] does not provide useful restorations for a variety of square regularization matrices. This depends on that the algorithm produces an unsuitable solution subspace; see [25, Example 5.5] for a discussion. The flexible Arnoldi process [25, Algorithm 2.1] can be modified to yield a more suitable solution subspace; however, the method of the present paper has the advantage of being applicable to a large variety of problems without having to be adapted to the problem at hand by the user. $\square$

5. Conclusion and extension.

We have described a generalized Krylov subspace method for reducing a pair of large, possibly rectangular, matrices to a pair of small matrices. An extension of this method for the reduction of a $(q+1)$-tuplet of large matrices to a $(q+1)$-tuplet of small matrices also is presented. These methods are generalizations of partial Golub–Kahan bidiagonalization of a single large matrix. The large matrices are accessed via matrix-vector product evaluations only.

The relation between the large and reduced matrices makes our reduction methods well suited for application to one-parameter and multi-parameter Tikhonov regularization of linear discrete ill-posed problems with large, possibly rectangular, matrices. The main reason for developing the one-parameter Tikhonov regularization method of the present paper is to obtain a new alternative to the GSVD when the problems are so large that the latter is expensive or infeasible to apply. We note that the numerical examples of one-parameter Tikhonov regularization show that for problems that are small enough to allow the evaluation of the GSVD of the matrix pair, our reduction method and the GSVD can give computed approximate solutions of the about same accuracy. Example 4.5 illustrates that the method of the present paper can be applied to a larger class of problems than Algorithm 2.1 of [25].

Acknowledgment. We would like to thank Kazim Khan for discussions and the referees for suggestions.

REFERENCES

[1] M. Arioli and D. Orban, *Iterative methods for symmetric quasi-definite linear systems. Part I: theory*, Cahier du GERAD G-2013-32, GERAD, 2013.

[2] M. Belge, M. E. Kilmer, and E. L. Miller, *Efficient selection of multiple regularization parameters in a generalized L-curve framework*, Inverse Problems, 28 (2002), pp. 1161–1183.

[3] C. Brezinski, M. Redivo–Zaglia, G. Rodriguez, and S. Seatzu, *Multi-parameter regularization techniques for ill-conditioned linear systems*, Numer. Math., 94 (2003), pp. 203–224.

[4] C. Brezinski, G. Rodriguez, and S. Seatzu, *Error estimates for the regularization of least squares problems*, Numer. Algorithms, 51 (2009), pp. 61–76.

[5] D. Calvetti, S. Morigi, L. Reichel, and F. Sgallari, *Tikhonov regularization and the L-curve for large discrete ill-posed problems*, J. Comput. Appl. Math, 123 (2000), pp. 423–446.

[6] D. Calvetti and L. Reichel, *Tikhonov regularization of large linear problems*, BIT, 43 (2003), pp. 263–283.

[7] L. Eldén, *A weighted pseudoinverse, generalized singular values, and constrained least squares problems*, BIT, 22 (1982), pp. 487–501.
[8] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer, Dordrecht, 1996.
[9] S. Gazzola and P. Novati, *Multi-parameter Arnoldi–Tikhonov methods*, Electron. Trans. Numer. Anal., 40 (2013), pp. 452–475.
[10] C. W. Groetsch, *The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind*, Pitman, Boston, 1984.
[11] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, 1998.
[12] P. C. Hansen, *Regularization tools version 4.0 for Matlab 7.3*, Numer. Algorithms, 46 (2007), pp. 189–194.
[13] M. E. Hochstenbach and L. Reichel, *An iterative method for Tikhonov regularization with a general linear regularization operator*, J. Integral Equations Appl., 22 (2010), pp. 465–482.
[14] L. Hoffnung, R.-C. Li, and Q. Ye, *Krylov type subspace methods for matrix polynomials*, Linear Algebra Appl., 415 (2006), pp. 52–81.
[15] M. E. Kilmer, P. C. Hansen, and M. I. Español, *A projection-based approach to general-form Tikhonov regularization*, SIAM J. Sci. Comput., 29 (2007), pp. 315–330.
[16] S. Kindermann, *Convergence analysis of minimization-based noise level-free parameter choice rules for linear ill-posed problems*, Electron. Trans. Numer. Anal., 38 (2011), pp. 233–257.
[17] R.-C. Li and Q. Ye, *A Krylov subspace method for quadratic matrix polynomials with application to constrained least squares problems*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 405–428.
[18] S. Lu and S. V. Pereverzev, *Multi-parameter regularization and its numerical realization*, Numer. Math., 118 (2011), pp. 1–31.
[19] S. Morigi, L. Reichel, and F. Sgallari, *Orthogonal projection regularization operators*, Numer. Algorithms, 44 (2007), pp. 99–114.
[20] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[21] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, IEEE Trans. Pattern Analysis and Machine Intelligence, 12 (1990), pp. 629–639.
[22] D. L. Phillips, *A Technique for the numerical solution of certain integral equations of the first kind*, J. ACM, 9 (1962), pp. 84–97.
[23] L. Reichel and G. Rodriguez, *Old and new parameter choice rules for discrete ill-posed problems*, Numer. Algorithms, 63 (2013), pp. 65–87.
[24] L. Reichel, F. Sgallari, and Q. Ye, *Tikhonov regularization based on generalized Krylov subspace methods*, Appl. Numer. Math., 62 (2012), pp. 1215–1228.
[25] L. Reichel and X. Yu, *Tikhonov regularization via flexible Arnoldi reduction*, BIT, in press.