NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 0000; **00**:1–18
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nla

GCV for Tikhonov regularization via global Golub–Kahan decomposition

C. Fenu[1], L. Reichel[2] and G. Rodriguez[1*]

[1]*Dipartimento di Matematica e Informatica, Università di Cagliari, viale Merello 92, 09123 Cagliari, Italy.*
[2]*Department of Mathematical Sciences, Kent State University, Kent, OH 44242, USA.*

SUMMARY

Generalized Cross Validation (GCV) is a popular approach to determining the regularization parameter in Tikhonov regularization. The regularization parameter is chosen by minimizing an expression, which is easy to evaluate for small-scale problems, but prohibitively expensive to compute for large-scale ones. This paper describes a novel method, based on Gauss-type quadrature, for determining upper and lower bounds for the desired expression. These bounds are used to determine the regularization parameter for large-scale problems. Computed examples illustrate the performance of the proposed method and demonstrate its competitiveness. Copyright © 0000 John Wiley & Sons, Ltd.

Received ...

KEY WORDS: generalized cross validation; Tikhonov regularization; parameter estimation; global Golub–Kahan decomposition.

1. INTRODUCTION

We are concerned with the solution of least-squares problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x}-\mathbf{b}\|, \tag{1.1}$$

where $A \in \mathbb{R}^{m\times n}$ is a large matrix whose singular values decay gradually to zero without a significant gap. In particular, the norm of $A^\dagger$, the Moore–Penrose pseudoinverse of A, is very large. Least-squares problems of this kind are often referred to as discrete ill-posed problems. The data vector $\mathbf{b} \in \mathbb{R}^m$ is assumed to stem from measurements and be contaminated by a measurement error $\mathbf{e} \in \mathbb{R}^m$ of unknown norm.

Let $\widehat{\mathbf{b}}$ denote the unknown error-free vector associated with $\mathbf{b}$. Hence, $\mathbf{b} = \widehat{\mathbf{b}} + \mathbf{e}$. We would like to compute an accurate approximation of $\widehat{\mathbf{x}} := A^\dagger\widehat{\mathbf{b}}$. However, due to the large norm of $A^\dagger$ and the presence of the error $\mathbf{e}$ in $\mathbf{b}$, the solution of (1.1), given by $A^\dagger\mathbf{b} = \widehat{\mathbf{x}} + A^\dagger\mathbf{e}$, typically is severely contaminated by the propagated error $A^\dagger\mathbf{e}$ and not a useful approximation of $\widehat{\mathbf{x}}$. A better approximation of $\widehat{\mathbf{x}}$ often can be computed by first replacing the least-squares problem (1.1) by a nearby problem, whose solution is less sensitive to the error $\mathbf{e}$ in $\mathbf{b}$. This replacement is known as

*Correspondence to: Dipartimento di Matematica e Informatica, Università di Cagliari, viale Merello 92, 09123 Cagliari, Italy. E-mail: rodriguez@unica.it.

Contract/grant sponsor: University of Cagliari RAS Visiting Professor Program, INdAM-GNCS

Contract/grant sponsor: NSF; contract/grant number: DMS-1115385

regularization. The possibly most popular regularization method is due to Tikhonov. In its simplest form, it replaces the problem (1.1) by a penalized least-squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \left\{ \|A\mathbf{x}-\mathbf{b}\|^2 + \mu^2\|\mathbf{x}\|^2 \right\}, \tag{1.2}$$

where $\mu > 0$ is known as a regularization parameter. It is the purpose of the regularization term $\mu^2\|\mathbf{x}\|^2$ to damp the propagated error in the computed solution; see, e.g., [14, 21] for discussions on Tikhonov regularization. Throughout this paper $\|\cdot\|$ denotes the Euclidean vector norm.

It can be difficult to determine a suitable value of the regularization parameter μ when no accurate bound for the norm of the error $\mathbf{e}$ in $\mathbf{b}$ is known. A too small value of μ gives a solution that is severely contaminated by propagated error, and a too large value yields an unnecessarily poor approximation of $\widehat{\mathbf{x}}$.

When an accurate bound for $\|\mathbf{e}\|$ is available, a suitable value of μ often can be determined with the aid of the discrepancy principle; see [14, 21]. However, for many discrete ill-posed problems (1.1) that arise in science and engineering, such a bound is not known. A large number of methods for determining a suitable value of μ in this situation have been proposed and investigated in the literature; see, e.g., [3, 8, 26, 27, 31, 32]. One of the most popular of these methods is Generalized Cross Validation (GCV); see [11, 18, 21]. This method requires the minimization of the GCV function

$$\mathcal{V}(\mu) := \frac{\|A\mathbf{x}_\mu - \mathbf{b}\|^2}{(\mathrm{trace}(I_m - A(\mu)))^2}, \tag{1.3}$$

where the *influence matrix* is defined by

$$A(\mu) := A(A^TA + \mu^2 I_n)^{-1}A^T \tag{1.4}$$

and $\mathbf{x}_\mu$ is the solution of (1.2),

$$\mathbf{x}_\mu = (A^TA + \mu^2 I_n)^{-1}A^T\mathbf{b}. \tag{1.5}$$

The superscript ^{T} denotes transposition and $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. Thus, $A(\mu)\mathbf{b} = A\mathbf{x}_\mu$. The GCV method prescribes that the minimizer $\mu > 0$ of (1.3), which we denote by μ^*, be determined and that $\mathbf{x}_{\mu^*}$ be used as an approximate solution of (1.1). The minimum of (1.3) generally is unique. It is well known that the derivative of the GCV function $\mathcal{V}(\mu)$ typically is of small magnitude in a neighborhood of μ^*. This can make the accurate determination of μ^* difficult; see, e.g., Hansen et al. [23] for a recent discussion.

For small to medium-sized problems, for which the singular value decomposition (SVD) of A can be computed, the evaluation of (1.3) for several μ-values is straightforward. However, when A is too large for the computation of its SVD to be feasible or attractive, the minimization of the function (1.3) can be quite expensive.

It is well known how to determine upper and lower bounds for the numerator of the GCV function by means of Gauss quadrature formulas; see [10, 19, 20] or Subsection 2.2 for details. In principle, it is possible to employ a similar technique to bound each element on the diagonal of $I_m - A(\mu)$. However, this requires a too large computational effort to be attractive when A is large. Golub and von Matt [20] describe an elegant approach to approximate the denominator of (1.3) based on the application of Hutchinson's trace estimator [24]. The use of this estimator also is discussed by Bai et al. [2], Burrage et al. [9], and Sidje et al. [33]. We will briefly describe the application of the Hutchinson trace estimator in Section 4.

Hutchinson's trace estimator determines an estimate of the denominator of (1.3). The minimization of the approximation of the GCV function with the denominator replaced by the Hutchinson estimate gives suitable values of the regularization parameter μ for many discrete ill-posed problems. However, as we will illustrate in Section 4, this approach may occasional fail to determine a suitable value of the regularization parameter μ. A reason for this may be that, while the expected value of Hutchinson's trace estimator is the denominator of (1.3), the variance of the computed estimate can be fairly large when the matrix A is far from diagonally dominant; see [20, Theorem 1].

DOI: 10.1002/nla

There are several other methods for computing an approximation of the trace of an implicitly defined large symmetric matrix; see, e.g., Brezinski et al. [6, 7] and Tang and Saad [34]. Some methods focus on computing the trace of particular matrix functions. Randomized algorithms for estimating the trace of a large matrix have recently been surveyed by Avron and Toledo [1]. They typically require a very large number of matrix-vector product evaluations to yield an estimate of moderate to high accuracy. Novati and Russo [29] determine an approximation of the trace in the denominator of (1.3) by first reducing the matrix A in (1.1) to a small matrix by carrying out a few steps of the Arnoldi process and then computing the trace of a small matrix so obtained. This approach also is discussed by Gazzola et al. [16], who moreover consider reduction of A by a few steps of Golub–Kahan bidiagonalization. An approximation of the trace in the GCV function then is determined by computing the trace of a small matrix. We are interested in determining upper and lower bounds for the GCV function (1.3). The available methods mentioned require less computational effort than the scheme proposed in this paper, but are not guaranteed to yield bounds for the GCV function. Computed examples in Section 4, and in particular Tables I and II, show that knowledge of accurate upper and lower bounds for the GCV function is helpful for determining a suitable value of the regularization parameter.

This paper describes how the method for computing upper and lower bounds for the trace of a large symmetric matrix presented in [4] can be used to determine upper and lower bounds for the GCV function (1.3). Our approach applies the global Golub–Kahan decomposition method described by Toutounian and Karimi [35]. This decomposition method is a block generalization of the standard Golub–Kahan bidiagonalization method and is closely related to the global Lanczos method for reducing a large symmetric matrix to a block tridiagonal matrix. The latter matrix is the tensor product of a symmetric tridiagonal matrix and an identity matrix. The global Lanczos method was proposed and analyzed by Elbouyahyaoui et al. [12] and Jbilou et al. [25].

Our method for bounding the GCV function is based on the following decomposition of the trace of a matrix. Let $M \in \mathbb{R}^{m\times m}$ be a symmetric matrix and let f be a function such that $f(M)$ is defined. Denote by $\mathbf{e}_i$ the ith column of the identity matrix I_m and introduce the block vectors

$$E_j = \left[\mathbf{e}_{(j-1)k+1}, \ldots, \mathbf{e}_{\min\{jk,m\}}\right], \qquad j = 1, 2, \ldots, \widetilde{m}, \tag{1.6}$$

with at most k columns. Here $\widetilde{m} := \left\lfloor \frac{m+k-1}{k} \right\rfloor$ and $\lfloor \alpha \rfloor$ denotes the largest integer smaller than or equal to α. Our algorithm computes upper and lower bounds for

$$\mathrm{trace}(E_j^T f(M) E_j), \qquad j = 1, 2, \ldots, \widetilde{m}.$$

These bounds yield upper and lower bounds for

$$\mathrm{trace}(f(M)) = \sum_{j=1}^{\widetilde{m}} \mathrm{trace}(E_j^T f(M) E_j). \tag{1.7}$$

The computation of upper and lower bounds for the $\widetilde{m}$ diagonal blocks is faster for large block sizes k than determining upper and lower bounds for each diagonal entry of M separately, because block methods can be executed efficiently on modern computers with a hierarchical memory structure. This is illustrated in Section 4. Moreover, block methods perform well in multiprocessor computing environments; see, e.g., [15].

In some applications of Tikhonov regularization, the minimization problem (1.2) is replaced by the problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \{\|A\mathbf{x} - \mathbf{b}\|^2 + \mu^2 \|L\mathbf{x}\|^2\} \tag{1.8}$$

with a regularization matrix $L \in \mathbb{R}^{p\times n}$ that is different from the identity. Among the most popular choices of regularization matrices, that are not the identity, are scaled discretizations of a differential operator such as the tridiagonal matrix

$$L = \begin{bmatrix} -1 & 2 & -1 & & & \mathrm{O} \\ & 1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ \mathrm{O} & & & -1 & 2 & -1 \end{bmatrix} \in \mathbb{R}^{(n-2)\times n};$$

Prepared using nlaauth.cls
DOI: 10.1002/nla

see, e.g., [14, 21] for discussions and illustrations. Björck [5] and Eldén [13] describe how the minimization problem (1.8) can be transformed into the form (1.2). This transformation is not very demanding computationally when L has a suitable structure, such as being banded. We will therefore only consider Tikhonov regularization problems of the form (1.2) in the present paper.

This paper is organized as follows. Section 2 discusses how bounds for the denominator of the GCV function can be computed by using the connection between global Golub–Kahan decomposition and Gauss quadrature rules. The numerator of the GCV function is bounded by using the well-known connection between standard Golub–Kahan bidiagonalization and Gauss quadrature. This connection is described, e.g., by Calvetti et al. [10] and Golub and Meurant [19]. Section 3 presents an algorithm for computing upper and lower bounds for the GCV function. This algorithm uses the bounds of Section 2. Computed examples in Section 4 illustrate the performance of the proposed method and compares it to a scheme proposed by Golub and von Matt [20]. Section 5 contains concluding remarks.

2. BOUNDING THE GCV FUNCTION BY GAUSS QUADRATURE

This section describes how upper and lower bounds for the GCV function (1.3) can be computed with the aid of Gauss and Gauss–Radau quadrature rules. These rules are evaluated using the standard Golub–Kahan bidiagonalization method for the numerator and the global Golub–Kahan decomposition method for the denominator. We first describe the latter.

2.1. Bounding the denominator of the GCV function

The form of (1.3) and the splitting (1.7) suggest that we should determine upper and lower bounds for expressions of the form

$$\mathcal{I} f_\mu := \operatorname{trace}(W^T f_\mu(AA^T)W), \tag{2.1}$$

where $W \in \mathbb{R}^{m\times k}$, with $1 \le k \ll m$, is a block vector, $\mu > 0$, and

$$f_\mu(t) := \frac{\mu^2}{t+\mu^2}, \qquad t \ge 0. \tag{2.2}$$

Proposition 1
Let f_μ be defined by (2.2) and let $A(\mu)$ be the influence matrix (1.4). Then

$$f_\mu(AA^T) = I_m - A(\mu).$$

Proof
We have

$$f_\mu(AA^T) = \mu^2(AA^T + \mu^2 I_m)^{-1} = I_m - (AA^T + \mu^2 I_m)^{-1}AA^T. \tag{2.3}$$

Substituting the identity

$$(AA^T + \mu^2 I_m)^{-1}A = A(A^TA + \mu^2 I_n)^{-1} \tag{2.4}$$

into the right-hand side of (2.3) yields the desired result. □

Introduce the spectral factorization $AA^T = Q\Lambda Q^T$, with

$$\Lambda = \operatorname{diag}[\lambda_1, \ldots, \lambda_m], \qquad 0 \le \lambda_1 \le \cdots \le \lambda_m,$$

and $Q \in \mathbb{R}^{m\times m}$ orthogonal. Define for block vectors $U, V \in \mathbb{R}^{m\times k}$ the inner product

$$\langle U, V\rangle := \operatorname{trace}(U^TV),$$

and the induced Frobenius matrix norm $\|U\|_F := \langle U, U\rangle^{1/2}$. Let $\breve{W} := W/\|W\|_F$. Then $\|\breve{W}\|_F = 1$. Substituting the spectral factorization of AA^T into (2.1) gives

$$\mathcal{I} f_\mu = \|W\|_F^2 \operatorname{trace}(\breve{W}^T f_\mu(AA^T)\breve{W}) = \|W\|_F^2 \operatorname{trace}(\widetilde{W}^T f_\mu(\Lambda)\widetilde{W}),$$

DOI: 10.1002/nla

where $\widetilde{W} = [\widetilde{w}_{ij}] = Q^T\breve{W}$. Moreover, for $j = 1, \ldots, k$, we obtain

$$\mathbf{e}_j^T\widetilde{W}^T f_\mu(\Lambda)\widetilde{W}\mathbf{e}_j = \sum_{i=1}^{m} f_\mu(\lambda_i)\widetilde{w}_{ij}^2 = \int_0^\infty f_\mu(\lambda)\, d\widetilde{w}_j(\lambda), \tag{2.5}$$

where $\widetilde{w}_j(\lambda)$ is a nondecreasing piecewise constant distribution function with jumps at the eigenvalues λ_i of AA^T. The relations (2.5) allow us to write the expression (2.1) as a Stieltjes integral,

$$\mathcal{I}f_\mu = \|W\|_F^2 \sum_{j=1}^{k} \int_0^\infty f_\mu(\lambda)\, d\widetilde{w}_j(\lambda) = \|W\|_F^2 \int_0^\infty f_\mu(\lambda)\, d\widetilde{w}(\lambda) \tag{2.6}$$

with the distribution function $\widetilde{w}(\lambda) := \sum_{j=1}^{k} \widetilde{w}_j(\lambda)$.

Using the fact that the function (2.2) is totally monotonic, i.e., all even order derivatives of $\lambda \to f_\mu(\lambda)$ are positive and all odd order derivatives are negative for $\lambda \geq 0$ and $\mu > 0$, allows us to determine upper and lower bounds for the expression (2.6), and therefore for (2.1), with the aid of Gauss-type quadrature rules. This technique is described in [10, 19]. The quadrature rules are evaluated using the global Golub–Kahan decomposition method. This algorithm was introduced by Toutounian and Karimi [35] with the aim of solving least-squares problems with multiple right-hand sides. Its use in conjunction with Gauss-type quadrature is new.

Given $U_1 \in \mathbb{R}^{m\times k}$ such that $\|U_1\|_F = 1$, ℓ steps of the global Golub–Kahan decomposition method applied to A with initial block vector U_1 determines the decompositions

$$\begin{aligned} A[V_1, V_2, \ldots, V_\ell] &= [U_1, U_2, \ldots, U_\ell]\widehat{C}_\ell + \sigma_{\ell+1}U_{\ell+1}E_\ell^T, \\ A^T[U_1, U_2, \ldots, U_\ell] &= [V_1, V_2, \ldots, V_\ell]\widehat{C}_\ell^T, \end{aligned} \tag{2.7}$$

where $U_i \in \mathbb{R}^{m\times k}$, $V_i \in \mathbb{R}^{n\times k}$ and $\langle U_i, U_j\rangle = \langle V_i, V_j\rangle = \delta_{ij}$. Here δ_{ij} denotes the Kronecker delta. Moreover, $\widehat{C}_\ell = C_\ell \otimes I_k$, where $\otimes$ stands for Kronecker product, and C_ℓ is a lower bidiagonal matrix,

$$C_\ell = \begin{bmatrix} \rho_1 & & & & \\ \sigma_2 & \rho_2 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \rho_{\ell-1} & \\ & & & \sigma_\ell & \rho_\ell \end{bmatrix} \in \mathbb{R}^{\ell\times\ell}.$$

We assume that ℓ is small enough so that all entries ρ_j and σ_j of the matrix C_ℓ are nonvanishing. Then the decompositions (2.7) with the stated properties exists. This is the generic situation. For future reference, we define the rectangular bidiagonal matrix

$$C_{\ell+1,\ell} = \begin{bmatrix} C_\ell \\ \sigma_{\ell+1}\mathbf{e}_\ell^T \end{bmatrix} \in \mathbb{R}^{(\ell+1)\times\ell}. \tag{2.8}$$

The computation of the decompositions (2.7) is described by Algorithm 1.

Combining the decompositions (2.7), we obtain the global Lanczos decomposition of the matrix AA^T,

$$AA^T[U_1, U_2, \ldots, U_\ell] = [U_1, U_2, \ldots, U_\ell]\widehat{T}_\ell + \rho_\ell\sigma_{\ell+1}U_{\ell+1}E_\ell^T,$$

where $\widehat{T}_\ell := T_\ell \otimes I_k$ and the matrix $T_\ell := C_\ell C_\ell^T$ is symmetric and tridiagonal.

It follows from [4, Theorem 3.1] that

$$\mathcal{G}_\ell f_\mu = \|W\|_F^2\, \mathbf{e}_1^T f_\mu(T_\ell)\mathbf{e}_1 \tag{2.9}$$

is an ℓ-point Gauss quadrature rule for the approximation of (2.6). Thus,

$$\mathcal{G}_\ell p = \mathcal{I}p \qquad \forall p \in \mathbb{P}_{2\ell-1},$$

Prepared using nlaauth.cls
DOI: 10.1002/nla

Algorithm 1 The global Golub–Kahan decomposition method.

1: **Input:** Matrix $A \in \mathbb{R}^{m\times n}$, initial block $W \in \mathbb{R}^{m\times k}$, number of steps ℓ.
2: $V_0 = 0$, $\sigma_1 = \|W\|_F$, $U_1 = W/\sigma_1$
3: **for** $j = 1$ **to** ℓ
4: $\quad \widetilde{V} = A^T U_j - \sigma_j V_{j-1}$, $\rho_j = \|\widetilde{V}\|_F$, $V_j = \widetilde{V}/\rho_j$
5: $\quad \widetilde{U} = AV_j - \rho_j U_j$, $\sigma_{j+1} = \|\widetilde{U}\|_F$, $U_{j+1} = \widetilde{U}/\sigma_{j+1}$
6: **end for**
7: **Output:** Global Golub–Kahan decompositions (2.7).

where $\mathbb{P}_j$ denotes the set of all polynomials of degree at most j.

The fact that (2.9), indeed, is a quadrature rule with ℓ nodes $\theta_1 < \cdots < \theta_\ell$ can be seen by substituting the spectral factorization of T_ℓ into (2.9); the θ_j are the eigenvalues of T_ℓ. Since T_ℓ is an orthogonal projection of AA^T, it follows that the nodes θ_j live in the interval $[\lambda_1, \lambda_m]$.

The even derivatives $f_\mu^{(2\ell)}$ of the function (2.2) are positive for $t \geq 0$. Therefore, it follows from the error formula for Gauss quadrature, shown, e.g., in [17],

$$\mathcal{I}f_\mu - \mathcal{G}_\ell f_\mu = \frac{f_\mu^{(2\ell)}(\xi)}{(2\ell)!} \int_0^\infty \prod_{j=1}^{\ell} (\lambda - \theta_j)^2 \, d\widetilde{w}(\lambda),$$

where $\xi \in (\lambda_1, \lambda_m)$, that $\mathcal{G}_\ell f_\mu \leq \mathcal{I}f_\mu$ for each ℓ. Moreover, $\mathcal{G}_{\ell-1} f_\mu \leq \mathcal{G}_\ell f_\mu$; see [28] for a proof.

An $(\ell+1)$-point Gauss–Radau quadrature rule with one prescribed node at ζ in the closure of the complement of the convex hull of the support of the measure $d\widetilde{w}(\lambda)$ can be obtained by determining $\ell+1$ weights and the free nodes $\theta_{1,\zeta}, \ldots, \theta_{\ell,\zeta}$ so that the rule is exact for polynomials of as high degree as possible; see Gautschi [17] or Golub and Meurant [19] for details. This rule can be expressed as

$$\mathcal{R}_{\ell+1,\zeta} f_\mu = \|W\|_F^2 \mathbf{e}_1^T f_\mu(T_{\ell+1,\zeta}) \mathbf{e}_1, \tag{2.10}$$

where the symmetric tridiagonal matrix $T_{\ell+1,\zeta}$ is obtained by suitably modifying the last diagonal entry of $T_{\ell+1}$; see below. This rule satisfies

$$\mathcal{R}_{\ell+1,\zeta} p = \mathcal{I}p \qquad \forall p \in \mathbb{P}_{2\ell}.$$

The error formula for (2.10) is given by

$$\mathcal{I}f_\mu - \mathcal{R}_{\ell+1,\zeta} f_\mu = \frac{f_\mu^{(2\ell+1)}(\xi_\zeta)}{(2\ell+1)!} \int_0^\infty (\lambda - \zeta) \prod_{j=1}^{\ell} (\lambda - \theta_{j,\zeta})^2 \, d\widetilde{w}(\lambda),$$

where ξ_ζ lies in the smallest open interval that contains ζ and the spectrum of AA^T; see, e.g., Gautschi [17] for a proof. Since the odd derivative $f_\mu^{(2\ell+1)}(t)$ of the function (2.2) is negative for $t \geq 0$, we have $\mathcal{I}f_\mu \leq \mathcal{R}_{\ell+1,\zeta} f$, provided that the prescribed node satisfies $\zeta \leq \lambda_1$. Further, $\mathcal{R}_{\ell+1,\zeta} f_\mu \leq \mathcal{R}_{\ell,\zeta} f_\mu$; see [28].

Since AA^T is positive semidefinite, we may choose the fixed node $\zeta = 0$. Then $T_{\ell+1,0} = C_{\ell+1,\ell} C_{\ell+1,\ell}^T$, where $C_{\ell+1,\ell}$ is defined by (2.8); see, e.g., [10, Proposition 3.1] or [20] for proofs. In particular, the quadrature rule (2.10) can be evaluated after ℓ steps of the global Golub–Kahan decomposition method have been carried out.

In summary, the global Golub–Kahan decomposition (2.7) of A yields upper and lower bounds for the functional (2.1). The bounds get sharper for every increase of ℓ. We have assumed that Algorithm 1 does not break down, i.e., that all $\rho_1, \ldots, \rho_\ell$ and $\sigma_1, \ldots, \sigma_\ell$ are nonvanishing. Breakdown can help bound the integral (2.1). For instance, if $\sigma_{\ell+1} = 0$ in (2.7), then the value of Gauss rule (2.9) equals (2.1). Since breakdown is very rare, we will not dwell on this situation further.

DOI: 10.1002/nla

2.2. Bounding the numerator of the GCV function

Bounds for the numerator of (1.3) can be determined by carrying out a few steps of standard Golub–Kahan bidiagonalization of A. We outline the required computations. Details are described in [10, 19].

Proposition 2
Let $\mathbf{x}_\mu$ be defined by (1.5) and introduce the function

$$g_\mu(t) := \frac{\mu^2}{(t+\mu^2)^2}.$$

Then

$$\|A\mathbf{x}_\mu - \mathbf{b}\|^2 = \mu^2 \mathbf{b}^T g_\mu(AA^T)\mathbf{b}. \tag{2.11}$$

Proof
The result can be established by using (2.4) similarly as in the proof of Proposition 1; see [10] for details. □

Substituting the spectral factorization of AA^T into the right-hand side of (2.11) yields, analogously to (2.6),

$$\mu^2 \mathbf{b}^T g_\mu(AA^T)\mathbf{b} = \mu^2 \int_0^\infty g_\mu(\lambda)\, d\nu(\lambda) =: \mathcal{I}' f_\mu, \tag{2.12}$$

where $\nu(\lambda)$ is a nondecreasing piecewise constant distribution function with jumps at the eigenvalues of AA^T. Since the function $g_\mu(\lambda)$ is totally monotonic, a discussion analogous to the one about the function $f_\mu(\lambda)$ in Subsection 2.1 shows that the right-hand side of (2.6) can be bracketed by pairs of Gauss and Gauss–Radau quadrature rules associated with the measure $d\nu$. These rules are conveniently evaluated with the aid of the standard Golub–Kahan bidiagonalization method applied to A with initial vector $\mathbf{u}_1 := \mathbf{b}/\|\mathbf{b}\|$. Application of ℓ steps of this method determines the decompositions

$$\begin{aligned} A[\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\ell] &= [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_\ell]B_\ell + \beta_{\ell+1}\mathbf{u}_{\ell+1}\mathbf{e}_\ell^T, \\ A^T[\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_\ell] &= [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\ell]B_\ell^T, \end{aligned} \tag{2.13}$$

where the vectors $\{\mathbf{u}_i\}_{i=1}^{\ell+1} \subset \mathbb{R}^m$ and $\{\mathbf{v}_i\}_{i=1}^{\ell} \subset \mathbb{R}^n$ satisfy $\mathbf{u}_i^T\mathbf{u}_j = \delta_{ij}$ and $\mathbf{v}_i^T\mathbf{v}_j = \delta_{ij}$, and B_ℓ is a lower bidiagonal matrix,

$$B_\ell = \begin{bmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \alpha_{\ell-1} & \\ & & & \beta_\ell & \alpha_\ell \end{bmatrix} \in \mathbb{R}^{\ell\times\ell}.$$

We assume that ℓ is small enough so that all entries α_j and β_j of B_ℓ are nonvanishing. Then the decompositions (2.13) with the specified properties exist. We also will need the rectangular bidiagonal matrix obtained by appending a row to the matrix B_ℓ,

$$B_{\ell+1,\ell} = \begin{bmatrix} B_\ell \\ \beta_{\ell+1}\mathbf{e}_\ell^T \end{bmatrix} \in \mathbb{R}^{(\ell+1)\times\ell}. \tag{2.14}$$

Let $T'_\ell = B_\ell B_\ell^T$. Analogously to (2.9), we define the ℓ-point Gauss quadrature rule

$$\mathcal{G}'_\ell g_\mu = \mu^2\|\mathbf{b}\|^2\, \mathbf{e}_1^T g_\mu(T'_\ell)\mathbf{e}_1 \tag{2.15}$$

associated with the measure $d\nu$. Thus,

$$\mathcal{G}'_\ell p = \mathcal{I}' p \qquad \forall p \in \mathbb{P}_{2\ell-1}.$$

DOI: 10.1002/nla

Moreover, similarly as in Subsection 2.1,

$$\mathcal{G}'_{\ell-1} g_\mu \leq \mathcal{G}'_\ell g_\mu \leq \|A\mathbf{x}_\mu - \mathbf{b}\|^2.$$

The symmetric tridiagonal matrix $T'_{\ell+1,0} = B_{\ell+1,\ell} B_{\ell+1,\ell}^T$ determines the $(\ell+1)$-point Gauss–Radau rule

$$\mathcal{R}'_{\ell+1} g_\mu = \mu^2 \|\mathbf{b}\|^2 \, \mathbf{e}_1^T g_\mu(T'_{\ell+1,0}) \mathbf{e}_1 \tag{2.16}$$

associated with the measure $d\nu$ and with a prescribed node at the origin. It follows similarly as in Subsection 2.1 that

$$\|A\mathbf{x}_\mu - \mathbf{b}\|^2 \leq \mathcal{R}'_{\ell+1} g_\mu \leq \mathcal{R}'_\ell g_\mu.$$

Thus, the numerator of the GCV function can be bracketed in terms of Gauss-type quadrature rules. The evaluation of the rules (2.15) and (2.16) requires the execution of ℓ steps of the standard Golub–Kahan bidiagonalization method.

3. AN ALGORITHM FOR BRACKETING THE GCV FUNCTION

We described in Subsection 2.2 how a sequence of lower and upper bounds ℓ_p and u_p such that

$$\ell_p \leq \|A\mathbf{x}_\mu - \mathbf{b}\|^2 \leq u_p, \qquad p = 1, 2, \ldots \,, \tag{3.1}$$

can be computed with the aid of p steps of standard Golub–Kahan bidiagonalization.

Let E_j, $j = 1, 2, \ldots, \widetilde{m}$, denote the block vectors defined by (1.6). Each block vector has k columns, except possibly the last one, $E_{\widetilde{m}}$, which may have fewer columns. Subsection 2.1 describes how q steps of the global Golub–Kahan decomposition method with initial block vector $W = E_j$ can be applied to compute bounds

$$v_q^{(j)} \leq \operatorname{trace}(E_j^T f_\mu(AA^T) E_j) \leq w_q^{(j)}, \quad j = 1, 2, \ldots, \widetilde{m}, \quad q = 1, 2, \ldots \,. \tag{3.2}$$

Define the sums

$$v_q = \left(\sum_{j=1}^{\widetilde{m}} v_q^{(j)} \right)^2, \qquad w_q = \left(\sum_{j=1}^{\widetilde{m}} w_q^{(j)} \right)^2, \qquad q = 1, 2, \ldots \,.$$

For notational simplicity, we assume that the same number of steps with the global Golub–Kahan decomposition method are carried out for bounding the trace of each submatrix $E_j^T f_\mu(AA^T) E_j$, $j = 1, 2, \ldots, \widetilde{m}$. This, of course, is not necessary; see below.

Since

$$\operatorname{trace}(f_\mu(AA^T)) = \sum_{j=1}^{\widetilde{m}} \operatorname{trace}(E_j^T f_\mu(AA^T) E_j), \tag{3.3}$$

cf. (1.7), we have

$$v_q \leq \left(\operatorname{trace}(f_\mu(AA^T)) \right)^2 \leq w_q, \qquad q = 1, 2, \ldots \,. \tag{3.4}$$

Combining (3.1) and (3.4) gives the following bounds for the GCV function,

$$\mathcal{L}_{p,q} \leq \mathcal{V}(\mu) \leq \mathcal{U}_{p,q},$$

where $\mathcal{L}_{p,q} := \ell_p / w_q$ and $\mathcal{U}_{p,q} := u_p / v_q$.

We would like to approximate $\mathcal{V}(\mu)$ with a fairly small relative error. Therefore, we require the bounds $\mathcal{L}_{p,q}$ and $\mathcal{U}_{p,q}$ to satisfy the right-hand side inequality

$$\frac{\mathcal{U}_{p,q} - \mathcal{L}_{p,q}}{\mathcal{V}(\mu)} \leq \frac{\mathcal{U}_{p,q} - \mathcal{L}_{p,q}}{\mathcal{L}_{p,q}} < \tau \tag{3.5}$$

DOI: 10.1002/nla

for some user-supplied constant $\tau > 0$. To meet this requirement, we first compute bounds for the numerator of the GCV function. We carry out p steps with the standard Golub–Kahan bidiagonalization method as described in Subsection 2.2, where we choose p as small as possible so that the bounds (3.1) satisfy

$$\frac{u_p - \ell_p}{\ell_p} < \alpha\tau, \qquad \alpha \in (0,1); \tag{3.6}$$

see below for a comment on the choice of α.

In view of that

$$\frac{\mathcal{U}_{p,q} - \mathcal{L}_{p,q}}{\mathcal{L}_{p,q}} = \frac{w_q}{\ell_p}\left[\frac{u_p}{v_q} - \frac{\ell_p}{w_q}\right] = \frac{u_p - \ell_p}{\ell_p} + \frac{w_q - v_q}{v_q}\frac{u_p}{\ell_p},$$

we carry out q iterations with the global Golub–Kahan decomposition method, where q is the smallest positive integer such that

$$\frac{w_q - v_q}{v_q} < \frac{\ell_p(1-\alpha)}{u_p}\tau. \tag{3.7}$$

Then the computed bounds for (3.3) satisfy (3.5).

Since executing one step of standard Golub–Kahan bidiagonalization is cheaper than executing one step of global Golub–Kahan bidiagonalization, we set $\alpha = 1/10$ in (3.6) to reduce the computational effort required to compute acceptable bounds for the denominator.

We have noticed that some of the above inequalities give too pessimistic estimates of the errors when the lower bounds converge very slowly with increasing p and q. In our implementation, we therefore approximate the numerator error by $2(u_p - \ell_p)/(\ell_p + u_p)$ rather than by $(u_p - \ell_p)/\ell_p$. This is equivalent to approximating the residual $\|A\mathbf{x}_\mu - \mathbf{b}\|^2$ in the denominator of the relative error by the average of the bounds ℓ_p and u_p. The denominator error bounds are adjusted similarly.

Assume for the moment that all lower and upper bounds in (3.2) satisfy

$$\frac{w_q^{(j)} - v_q^{(j)}}{v_q^{(j)}} < \frac{\ell_p(1-\alpha)}{u_p}\tau, \qquad j = 1, 2, \ldots, \widetilde{m}.$$

This can be used as a stop condition for the computation of the terms in the sum (3.3), as it implies

$$\frac{w_q - v_q}{w_q + v_q} \le \frac{w_q - v_q}{\sqrt{v_q w_q} + v_q} = \frac{\sqrt{w_q} - \sqrt{v_q}}{\sqrt{v_q}} < \frac{\ell_p(1-\alpha)}{u_p}.$$

The computations required for obtaining upper and lower bounds for the GCV function are outlined in Algorithm 2. We remark that there are two significant differences between this algorithm and our MATLAB code used in actual computations: First, our MATLAB code computes bounds for the GCV function on a grid of regularization parameter values in a single call. Specifically, the MATLAB code first computes bounds on a coarse grid $\{\mu_j\}_{j=1}^{13}$ of logarithmically equispaced points between 10^{-10} and 10^2. If the minimum is found at one endpoint of the interval $[10^{-10}, 10^2]$, then the grid is shifted to the left or to the right in order for the minimum to be strictly between the extreme grid points, and the computations are restarted. If the minimum is located at an interior point, say μ_s, then the search is repeated on a finer grid of 100 logarithmically equispaced points between μ_{s-1} and μ_{s+1}. Secondly, the number of steps q of the global Golub–Kahan decomposition method is chosen so that the quotients

$$\frac{w_{q_j}^{(j)} - v_{q_j}^{(j)}}{v_{q_j}^{(j)}}, \qquad j = 1, 2, \ldots, \widetilde{m},$$

are roughly independent of j.

When the regularization parameter $\mu > 0$ is close to zero, the quadrature errors for the rules $\mathcal{G}_\ell f_\mu$, $\mathcal{R}_{\ell+1} f_\mu$, $\mathcal{G}'_\ell g_\mu$, and $\mathcal{R}'_{\ell+1} g_\mu$ are typically larger than when μ is further away from the origin. This

DOI: 10.1002/nla

Algorithm 2 GCV computation by Gauss quadrature.

1: **Input:** Matrix $A \in \mathbb{R}^{m\times n}$, noisy right hand side $\mathbf{b}$, block size k.
2: regularization parameter μ, accuracy τ, factor α.
3: Compute bounds ℓ and u for the numerator with accuracy $\alpha\tau$.
4: $\widetilde{m} = \left\lfloor \frac{m+k-1}{k} \right\rfloor$, $v = 0$, $w = 0$
5: **for** $j = 1$ **to** $\widetilde{m}$
6: $E_j = \left[\mathbf{e}_{k(j-1)+1}, \ldots, \mathbf{e}_{\min\{jk,m\}}\right]$
7: Compute bounds $v^{(j)}$ and $w^{(j)}$ for $\mathrm{trace}(E_j^T f_\mu(AA^T)E_j)$ with accuracy $\frac{\ell(1-\alpha)}{u}\tau$
8: $v = v + v^{(j)}$, $w = w + w^{(j)}$
9: **end for**
10: $\mathcal{L} = \frac{\ell}{w^2}$, $\mathcal{U} = \frac{u}{v^2}$, $\mathcal{V} = \frac{\mathcal{L}+\mathcal{U}}{2}$
11: **Output:** Approximate GCV value $\mathcal{V} = \mathcal{V}(\mu)$, lower and upper bounds $\mathcal{L}, \mathcal{U}$.

depends on that the integrands f_μ and g_μ in (2.6) and (2.12), respectively, have a singularity at $t = -\mu^2$, and this singularity is close to the convex hull of the support of the measures $d\widetilde{w}_j(\lambda)$ and $d\nu(\lambda)$ when $\mu > 0$ is "tiny." On the other hand, the minimum of the GCV function is seldom achieved for tiny positive values of μ for discrete ill-posed problems (1.1) that arise in applications.

To avoid excessive computing times, the convergence test is coupled to a check for stagnation of the upper bounds. Specifically, we terminate the iterations for bounding the numerator of the GCV function after p steps when

$$2\frac{u_p - \ell_p}{\ell_p + u_p} < \alpha\widetilde{\tau} \qquad \text{or} \qquad \frac{u_{p-1} - u_p}{u_p} < \widetilde{\rho}. \tag{3.8}$$

The denominator is treated similarly. Unless explicitly stated otherwise, we set $\widetilde{\tau} = 10^{-1}$ and $\widetilde{\rho} = 10^{-3}$ in the computed examples of the following section.

Once an estimate μ^* of the regularization parameter that minimizes the GCV function is available, an approximate solution of (1.2) with $\mu = \mu^*$ can be computed very cheaply. Indeed, the square of the residual norm corresponding to $\mathbf{x}_{\mu^*}$ is the numerator of the function $\mathcal{V}(\mu)$ at the minimum. The Galerkin equation

$$V_\ell^T(A^TA + (\mu^*)^2 I)V_\ell \mathbf{y} = V_\ell^T A^T \mathbf{b} \tag{3.9}$$

is the projection of the normal equations associated with (1.2), with $\mu = \mu^*$, onto the subspace range(V_ℓ) determined by ℓ steps of Golub–Kahan bidiagonalization applied to A with initial vector $\mathbf{b}$; see (2.13). Let $\mathbf{y}_{\mu^*,\ell}$ be the solution of (3.9). Theorem 5.1 of [10] shows that the approximate solution $\mathbf{x}_{\mu^*,\ell} = V_\ell \mathbf{y}_{\mu^*,\ell}$ of (1.2) satisfies

$$\|A\mathbf{x}_{\mu^*,\ell} - \mathbf{b}\| = \mathcal{R}'_{\ell+1} g_{\mu^*}.$$

For ℓ sufficiently large, $\mathbf{x}_{\mu^*,\ell}$ is an accurate approximation of $\mathbf{x}_{\mu^*}$. We compute the solution $\mathbf{y}_{\mu^*,\ell}$ of the Galerkin equations (3.9) by solving the associated least-squares problem

$$\min_{\mathbf{y}\in\mathbb{R}^\ell} \left\| \begin{bmatrix} B_{\ell+1,\ell} \\ \mu^* I_\ell \end{bmatrix} \mathbf{y} - \beta_1 \mathbf{e}_1 \right\|,$$

where $B_{\ell+1,\ell}$ is given by (2.14) and $\beta_1 = \|\mathbf{b}\|$. In all computed examples of the following section, the value of ℓ used for computing bounds for the numerator of the GCV function was sufficient to determine an accurate approximation $\mathbf{x}_{\mu^*,\ell}$ of $\mathbf{x}_{\mu^*}$.

A final remark, of particular importance for large-scale problems, is that the computation of the bounds at lines 3 and 7 of Algorithm 2 can be carried out independently. Therefore, the execution time can be reduced substantially in a parallel computing environment.

DOI: 10.1002/nla

4. COMPUTED EXAMPLES

The algorithm introduced in the previous section to bound the GCV function has been implemented in the MATLAB programming language. To investigate its performance, we have applied it to the computation of the Tikhonov regularization parameter in a set of test discrete ill-posed problems that are widely used in the literature. The numerical experiments were executed in double precision, that is, with about 15 significant decimal digits, using MATLAB R2014a on an Intel Core i7 computer with 8 Gbyte RAM, under the Linux operating system.

The test problems are listed in Table I. A majority of them are from the Regularization Tools package [22]. The Hilbert and Lotkin test matrices are available in the standard MATLAB distribution. Two additional structured test matrices are considered. Tomo is a sparse matrix resulting from the discretization of a 2D tomography problem [22]; sparse matrix operations are directly supported in MATLAB. Prolate is an ill-conditioned Toeplitz matrix described in [36]. We handle this example by using the toolbox `smt` [30], which stores the matrix in an optimized format and provides algorithms for fast matrix operations. In particular, matrix-vector products are evaluated by means of the fast Fourier transform (FFT).

Most of the above test problems define both a matrix A and a solution $\widehat{\mathbf{x}}$; to those missing a test solution, we associate the solution of the Shaw problem from [22]. We compute the exact right-hand side as $\widehat{\mathbf{b}} := A\widehat{\mathbf{x}}$. The perturbed data vector $\mathbf{b} \in \mathbb{R}^m$ in (1.1) is assumed to be contaminated by Gaussian noise $\mathbf{e} \in \mathbb{R}^m$ with mean zero and variance one, according to the formula

$$\mathbf{b} = \widehat{\mathbf{b}} + \mathbf{e}\,\|\widehat{\mathbf{b}}\|\frac{\sigma}{\sqrt{m}},$$

where σ is a chosen noise level. In our experiments we set $\sigma = 10^{-3}, 10^{-2}, 10^{-1}$.

Our method, which we below refer to as the Quadrature method, is compared to the method proposed by Golub and von Matt [20].† Their method applies Hutchinson's stochastic trace estimator [24] to compute an estimate of $\mathrm{trace}(I_m - A(\mu))$ in (1.3). This entails the computation of upper and lower bounds for expressions of the form $\mathbf{z}^T(AA^T + \mu^2 I_m)^{-1}\mathbf{z}$, where $\mathbf{z} \in \mathbb{R}^m$ is a random vector with entries ± 1 with equal probability. These bounds are determined by the technique described in Section 3. The software by Golub and von Matt is mostly written in MATLAB, with some subroutines coded in the C language and linked to MATLAB via the MEX (MATLAB executable) interface. We will refer to this software as `gcv_lanczos`, the name of its main function.

Figure 1 displays the upper and lower bounds produced by the Quadrature method of Section 3 when $\ell = 2, 4, \ldots, 10$ steps of both standard and global Golub-Kahan bidiagonalization are carried out for each $j = 1, 2, \ldots, \widetilde{m}$; see (3.2). The plots show the exact GCV function (thick curve), and upper and lower bounds computed with quadrature rules. The bounds improve monotonically as ℓ is increased. Thus, the bounds closest to the thick curves are for $\ell = 10$, while the bounds furthest away from the thick curves are for $\ell = 2$. The left-hand side plot is for the Phillips test problem with $A \in \mathbb{R}^{200\times 200}$, while the right-hand side plot is for the Shaw test problem with $A \in \mathbb{R}^{2000\times 2000}$. Both test problems are discrete ill-posed problems from [22]. The block size used for global Golub–Kahan bidiagonalization is $k = 100$. The exact GCV function is evaluated by means of the singular value decomposition of the matrix A. The noise level is $\sigma = 10^{-3}$.

The (blue) bullets in the plots of Figure 1 mark approximate minima of the GCV function. They are determined by minimizing the computed upper bounds for $\ell = 10$. The location of these minima is not very sensitive to the number of steps ℓ. Indeed, it is remarkable how few steps are required to give a useful approximation of the minimum of the GCV function. The graphs for the Phillips and the Shaw test problems are typical for many discrete ill-posed problems in that the GCV function is very flat around its minimum. The flatness and propagated errors due to finite precision arithmetic can make it difficult to compute the minimizer μ^* of the GCV function accurately. It is considerably

†We thank Urs von Matt for making software available. The MATLAB MMQ toolbox by Gerard Meurant also contains an implementation.

DOI: 10.1002/nla

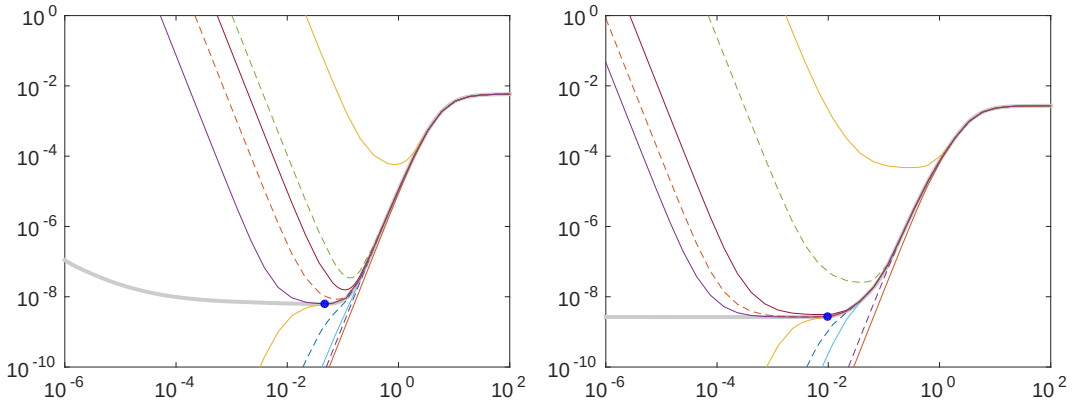

Figure 1. Upper and lower bounds produced by the Quadrature method at $\ell = 2, 4, \ldots, 10$ steps of the standard and global Golub–Kahan bidiagonalization methods (for each $j = 1, 2, \ldots, \widetilde{m}$). The left-hand side plot is for the Phillips example with a matrix A of size 200×200; the right-hand side plot is for the Shaw problem with a matrix A of size 2000×2000. The noise level is $\sigma = 10^{-3}$. The thick curves show the GCV function, while the other curves display upper and lower bounds. The quality of the bounds improves as ℓ increases.

easier to determine the minimizer of a computed upper bound for the GCV function. The difficulty of minimizing the GCV function has recently also been discussed by Hansen et al. [23]. We will in the following illustrate that the minimization of the computed upper bounds for the GCV function produces accurate approximations of the desired solution $\widehat{\mathbf{x}}$.

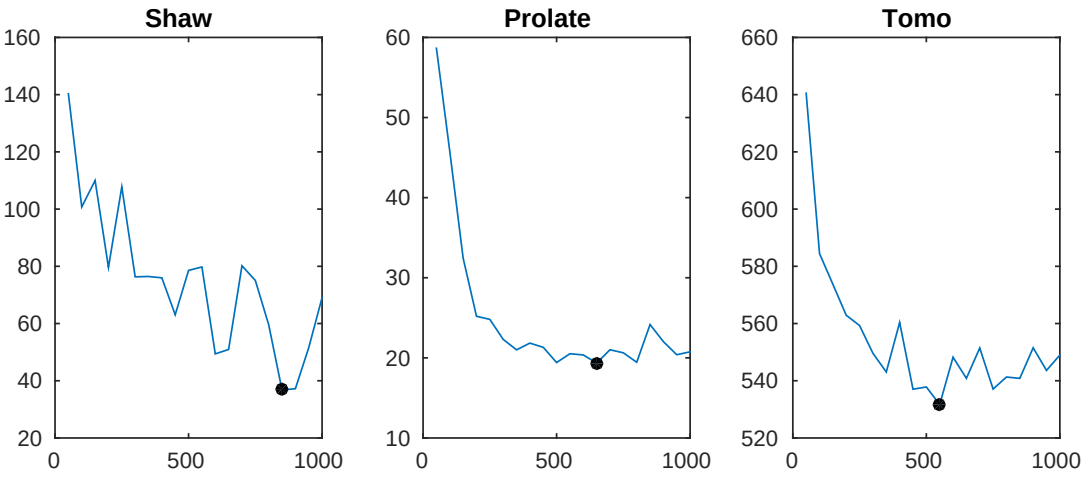

Figure 2. Computing time (in seconds) versus block size for three problems of size 4096. The Shaw matrix is dense, the Prolate matrix has Toeplitz structure, and the Tomo matrix is sparse.

Figure 2 reports the computing times for three different examples as a function of the block size used in the global Golub–Kahan bidiagonalization method. The matrix A is of size 4096×4096 and the block size ranges from 50 till 1000. The black bullet indicates the minimal computing time for each example. It is clear from the figure that increasing the block size generally reduces the computing time. The speedup due to a large block size is particularly pronounced for the Shaw test problem, for which the matrix A is dense.

DOI: 10.1002/nla

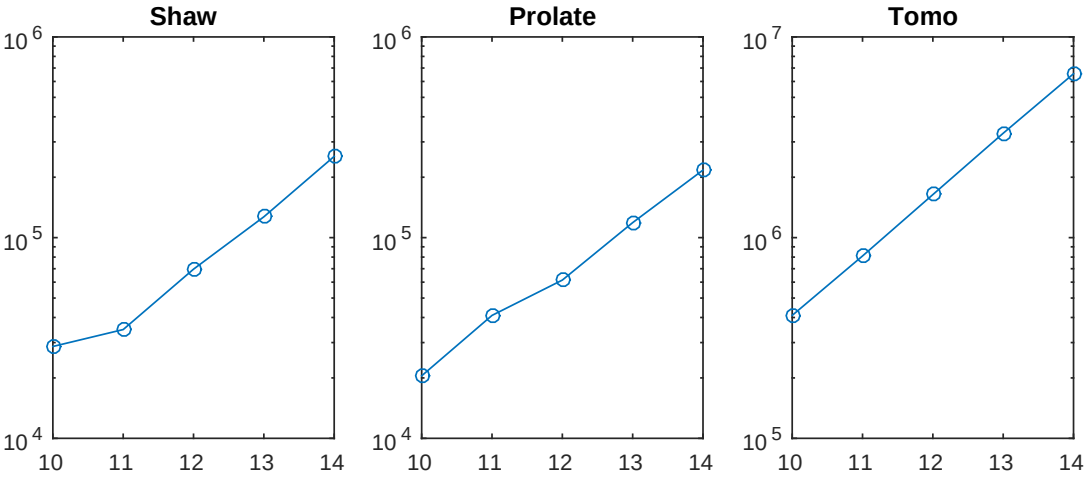

Figure 3. Number of matrix-vector product evaluations versus problem order n in log-log scale for the three problems of Figure 2. The matrices are of order 2^r, $r = 10, \ldots, 14$; the block size is 512.

The dominant computational work for our algorithm, when applied to large-scale problems, is the evaluation of matrix-vector products with A and A^T. Figure 3 shows the number of matrix-vector product evaluations required for the test problems of Figure 2 with matrices $A \in \mathbb{R}^{n\times n}$ of order $n = 1024, 2048, \ldots, 16384$. The block size is chosen to be 512. The number of matrix-vector product evaluations grows linearly with n, that is, proportionally to the number of steps $\widetilde{m}$ of the **for**-loop in Algorithm 2. The graphs indicate that when the matrix order n is doubled, the number of matrix-vector product evaluations increases by a factor not larger than 2. This shows that the number of iterations required to compute each pair of bounds (3.4) is approximately independent of n.

Table I. Comparison between the `gcv_lanczos` code by Golub and von Matt, and the Quadrature method of this paper. For each test problem, we define 60 experiments with different problem sizes, noise levels, and noise realizations; see text. The index F_κ, defined by (4.1), shows the percentage of failures of each method. The computing time is measured in seconds.

	`gcv_lanczos`			Quadrature			
matrix	F_5	F_{10}	time	F_5	F_{10}	time $k = 100$	time $k = 500$
Baart	27	22	7.0e-01	0	0	2.5e+01	2.0e+01
Deriv2	0	0	7.4e-01	0	0	4.3e+01	2.9e+01
Foxgood	33	30	4.9e-01	3	0	2.4e+01	1.7e+01
Gravity	9	7	6.2e-01	1	0	4.6e+01	3.4e+01
Heat(1)	0	0	1.4	0	0	6.1e+01	4.1e+01
Hilbert	0	0	4.9e-01	0	0	3.8e+01	1.7e+01
Lotkin	11	8	4.0e-01	0	0	1.9e+01	1.4e+01
Phillips	4	1	4.0e-01	0	0	3.9e+01	3.3e+01
Shaw	25	21	9.1e-01	0	0	3.1e+01	2.1e+01
Wing	19	18	6.0	0	0	2.1e+01	1.8e+01

To better understand how accurately the method by Golub and von Matt and the Quadrature method approximate the desired solution $\widehat{\mathbf{x}}$, we constructed a set of examples for each one of the matrices listed in Table I. For each matrix, we consider a square system of size 2000×2000 and an overdetermined system of size 4000×2000. The latter matrix is obtained by removing the last 2000 columns from a 4000×4000 matrix. The noise levels $\sigma = 10^{-3}, 10^{-2}, 10^{-1}$ are used, and we generated 10 realizations of the noise vector $\mathbf{e}$ for each noise level. This yields 60 examples for

DOI: 10.1002/nla

each one of the matrices in Table I. We determined the Tikhonov regularization parameter by the `gcv_lanczos` routine by Golub and von Matt, and by the Quadrature method, for every one of these examples.

Let μ_{best} denote the regularization parameter that yields the smallest error, i.e.,

$$\|\mathbf{x}_{\mu_{\text{best}}} - \widehat{\mathbf{x}}\| = \min_{\mu>0} \|\mathbf{x}_{\mu} - \widehat{\mathbf{x}}\|.$$

We let F_κ denote the number of experiments that produced a regularized solution $\mathbf{x}_\mu$ with an error larger than κ times the optimal one, that is, F_κ counts the number of times the inequality

$$\|\mathbf{x}_{\mu} - \widehat{\mathbf{x}}\| > \kappa \, \|\mathbf{x}_{\mu_{\text{best}}} - \widehat{\mathbf{x}}\| \tag{4.1}$$

holds.

The second to fourth columns of Table I show the values of F_5 and F_{10} obtained with the `gcv_lanczos` routine, together with the computing time required. The fifth and sixth columns display F_5 and F_{10} for the Quadrature method, and the last two columns report the computing times for block sizes 100 and 500. The Quadrature method can be seen to be slower and much more robust than the method by Golub and von Matt. Only rarely is the error in the regularized solution determined by the Quadrature method more than 5 times larger than the optimal error, while for several of the examples, the `gcv_lanczos` method gives approximate solutions with an error 5 times larger than the optimal error in about 50% of the runs.

Table II. Performance of the Quadrature method. The experimental setting is similar to that of Table I. In this table the value of $\widetilde{\rho}$ in (3.8) is increased from 10^{-3} to 10^{-1}. The computing time is measured in seconds.

	Quadrature $\widetilde{\rho} = 10^{-1}$			
matrix	F_5	F_{10}	time $k = 100$	time $k = 500$
Baart	0	0	1.2e+01	1.3e+01
Deriv2(2)	0	0	1.2e+01	1.2e+01
Foxgood	8	1	1.4e+01	1.3e+01
Gravity	1	0	1.3e+01	1.1e+01
Heat(1)	3	0	1.3e+01	1.2e+01
Hilbert	0	0	1.4e+01	1.2e+01
Lotkin	0	0	1.3e+01	1.2e+01
Phillips	0	0	1.2e+01	1.2e+01
Shaw	6	0	1.3e+01	1.3e+01
Wing	0	0	1.3e+01	1.2e+01

The computing time for the Quadrature method can be reduced of a factor between 2 and 3 if one is willing to accept less accuracy. This is illustrated by Table II. The results of Table I were obtained by setting $\widetilde{\tau} = 10^{-1}$ and $\widetilde{\rho} = 10^{-3}$ in the stopping condition (3.8). Table II shows the corresponding results for $\widetilde{\tau} = 10^{-1}$ and $\widetilde{\rho} = 10^{-1}$. This increase of $\widetilde{\rho}$ reduces the computing time, and the number of times the computed solution has an error larger than 5 or 10 times the smallest possible error increases only slightly.

Figure 4 shows two particular experiments with the Baart test problem from [22] with a matrix $A \in \mathbb{R}^{1024\times 1024}$. This is a discrete ill-posed problem. The noise level is $\sigma = 10^{-1}$ and two different realizations of the noise vector $\mathbf{e}$ are used. The figure depicts the desired solution $\widehat{\mathbf{x}}$ and the Tikhonov solutions $\mathbf{x}_\mu$ corresponding to the μ-values determined by the `gcv_lanczos` code and the Quadrature method. These μ-values, together with the relative errors (measured by the Euclidean vector norm), the block size, and the noise level, are reported in the first two lines of Table III. The plot on the left-hand side of Figure 4 shows a situation when the two methods produce approximations of $\widehat{\mathbf{x}}$ of comparable quality, while the right-hand side plot displays a typical failure

DOI: 10.1002/nla

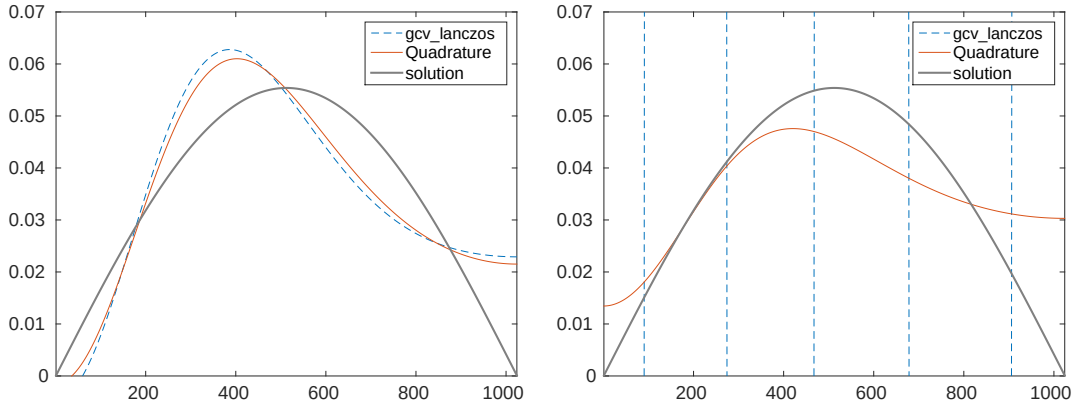

Figure 4. Solution of the Baart test problem of size 1024×1024 by Tikhonov regularization with noise level $\sigma = 10^{-1}$ and two realizations of the noise. The value of the regularization parameter is determined by the Quadrature and the `gcv_lanczos` methods. The parameters characterizing the experiments are reported in Table III.

of the `gcv_lanczos` routine. The large error is caused by under-estimation of the regularization parameter. Figure 5 reports two similar experiments for the Prolate example. The parameters for the two graphs are listed in the third and fourth lines of Table III.

Table III. Parameters which characterize the numerical experiments of Figures 4, 5, and 6.

				Quadrature		`gcv_lanczos`	
matrix	m	k	σ	μ	error	μ	error
Baart	1024	512	10^{-1}	$2.3 \cdot 10^{-2}$	$2.1 \cdot 10^{-1}$	$1.6 \cdot 10^{-2}$	$2.5 \cdot 10^{-1}$
				$4.0 \cdot 10^{-2}$	$2.5 \cdot 10^{-1}$	$5.1 \cdot 10^{-6}$	$1.25 \cdot 10^{2}$
Prolate	512	256	10^{-1}	$9.1 \cdot 10^{-2}$	$8.0 \cdot 10^{-2}$	$7.3 \cdot 10^{-2}$	$8.2 \cdot 10^{-2}$
				$9.1 \cdot 10^{-2}$	$8.7 \cdot 10^{-2}$	$8.1 \cdot 10^{-3}$	$6.2 \cdot 10^{-1}$
Tomo	1024	512	10^{-2}	$4.1 \cdot 10^{-1}$	$1.8 \cdot 10^{-1}$	$6.1 \cdot 10^{-2}$	$2.2 \cdot 10^{-1}$

A particular situation happens with the Tomo example from [22]. This sparse matrix arises from the discretization of a 2D tomography problem. We chose an image solution of size 32×32, leading to a least-squares problem (1.1) with a matrix $A \in \mathbb{R}^{1024\times1024}$. The original (error-free) image is shown in the upper left picture of Figure 6. The upper right picture shows the approximate solution obtained by Tikhonov regularization with the regularization parameter determined by the Quadrature method, while the lower right picture displays the approximate solution computed with Tikhonov regularization with μ determined by `gcv_lanczos`. The parameters for this experiment are reported in the last line of Table III. The two reconstructed images are seen to be of roughly the same quality and the computing times for both methods is comparable: the Quadrature method required 21.2 seconds, while `gcv_lanczos` took 5.6 seconds. The fairly close computing times may be due to the spectral properties of the matrix A. The plot of the singular values of A in the lower left graph of Figure 6 shows that A is rank deficient but differs from the matrices in the other computed examples in that only a small number of the singular values of A are close to zero.

To better illustrate the computing time required, we display in Figure 7 the computing times for the two methods as functions of the problem size and noise level. The graphs on the left-hand side show the computing time of the Quadrature method for the noise levels $\sigma = 10^{-3}, 10^{-2}, 10^{-1}$. They indicate that the computing time does not depend significantly on the noise level. The graphs

DOI: 10.1002/nla

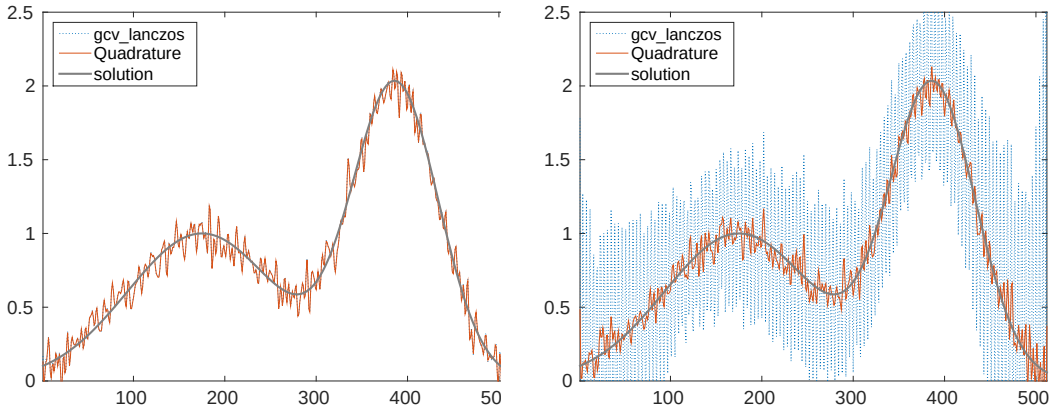

Figure 5. Solution of the Prolate test problem of size 512×512 by Tikhonov regularization with noise level $\sigma = 10^{-1}$ and two realizations of the noise. The value of the regularization parameter is determined by the Quadrature and the `gcv_lanczos` methods. The parameters characterizing the experiments are reported in Table III.

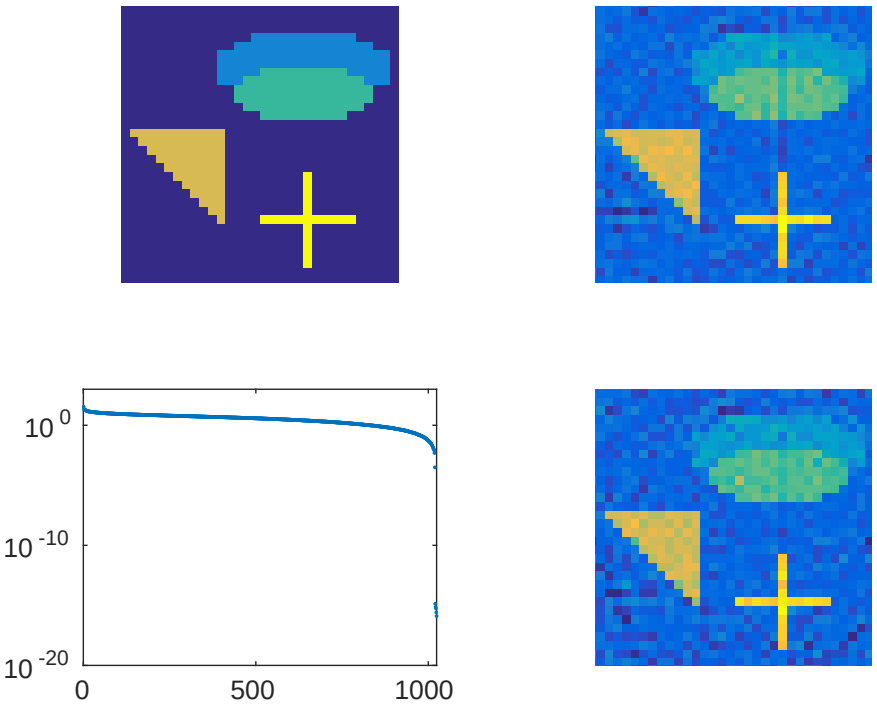

Figure 6. Solution of the Tomo problem of size 1024 by Tikhonov regularization, with noise level $\sigma = 10^{-2}$. The original image (top left) is an image of 32×32 pixels. The value of the regularization parameter is estimated either by the Quadrature (top right) and the `gcv_lanczos` (bottom right) methods. The parameters characterizing the experiments are reported in Table III. The singular values of the Tomo matrix are represented in the bottom left graph.

on the right-hand side display the computing time for the `gcv_lanczos` method for the same noise levels. The computing time is seen to be fairly sensitive to the noise level, but quite insensitive to the problem size. The dashed curve in the right-hand side plot shows timings for the Quadrature

DOI: 10.1002/nla

method for $\sigma = 10^{-3}$. Thus, the Quadrature method is faster than the `gcv_lanczos` method for this noise level and problem sizes up to about 2000.

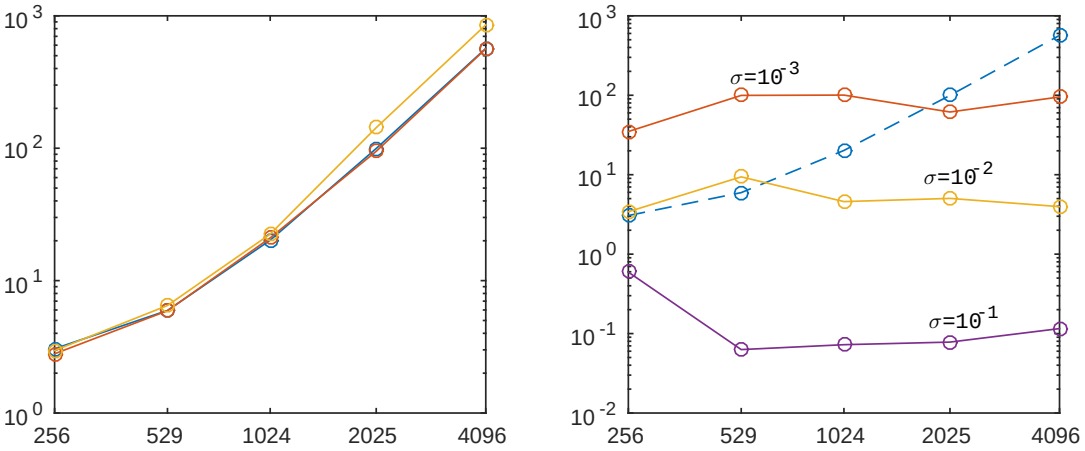

Figure 7. Computing time (in seconds) for estimating the regularization parameter in the Tomo example versus the size of the problem. The graphs on the left-hand side depict timings of the Quadrature method for $\sigma = 10^{-3}, 10^{-2}, 10^{-1}$. The graphs on the right-hand side shows timings for the `gcv_lanczos` routine for the same noise levels. The dashed line in the right-hand side plot shows Quadrature timings for $\sigma = 10^{-3}$.

5. CONCLUSION

A new approach to computing the regularization parameter in Tikhonov regularization by the GCV method is described. This approach exploits the connection between global Golub–Kahan bidiagonalization and Gauss-type quadrature, and is well suited for bounding the GCV function for large-scale problems. Many available methods for determining the Tikhonov regularization parameter by GCV for large-scale problems apply Hutchinson's stochastic trace estimator. Our new approach is much more reliable, but often slower, than the latter approach. Since reliability generally is more important than speed, we feel the proposed approach to be of interest.

ACKNOWLEDGEMENT

Work of L.R. was supported by the University of Cagliari RAS Visiting Professor Program and by NSF grant DMS-1115385. Work of C.F and G.R. was partially supported by INdAM-GNCS.

References

1. H. Avron and S. Toledo, *Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix*, J. ACM, 58 (2011), pp. 8:1–8:34.
2. Z. Bai, M. Fahey, and G. Golub, *Some large-scale matrix computation problems*, J. Comput. Appl. Math., 74 (1996), pp. 71–89.
3. F. Bauer and M. A. Lukas, *Comparing parameter choice methods for regularization of ill-posed problem*, Math. Comput. Simulat., 81 (2011), pp. 1795–1841.
4. M. Bellalij, L. Reichel, G. Rodriguez, and H. Sadok, *Bounding matrix functionals via partial global block Lanczos decomposition*, Appl. Numer. Math., 94 (2015), pp. 127–139.
5. Å. Björck, *A bidiagonalization algorithm for solving large and sparse ill-posed systems of linear equations*, BIT, 18 (1988), pp. 659–670.
6. C. Brezinski, P. Fika, and M. Mitrouli, *Estimations of the trace of powers of self-adjoint operators by extrapolation of the moments*, Electron. Trans. Numer. Anal., 39 (2012), pp. 144–159.
7. C. Brezinski, P. Fika, and M. Mitrouli, *Moments of a linear operator on a Hilbert space, with applications to the trace of the inverse of matrices and the solution of equations*, Numer. Linear Algebra Appl., 19 (2012), pp. 937–953.

DOI: 10.1002/nla

8. C. BREZINSKI, G. RODRIGUEZ, AND S. SEATZU, *Error estimates for the regularization of least squares problems*, Numer. Algorithms, 51 (2009), pp. 61–76.
9. K. BURRAGE, A. WILLIAMS, J. ERHEL, AND B. POHL, *The implementation of a Generalized Cross Validation algorithm using deflation techniques for linear systems*, Appl. Numer. Math., 19 (1995), pp. 17–31.
10. D. CALVETTI, G. H. GOLUB, AND L. REICHEL, *Estimation of the L-curve via Lanczos bidiagonalization*, BIT, 39 (1999), pp. 603–619.
11. P. CRAVEN AND G. WAHBA, *Smoothing noisy data with spline functions*, Numer. Math., 31 (1979), pp. 377–403.
12. L. ELBOUYAHYAOUI, A. MESSAOUDI, AND H. SADOK, *Algebraic properties of the block GMRES and block Arnoldi methods*, Electron. Trans. Numer. Anal., 33 (2009), pp. 207–220.
13. L. ELDÉN, *A weighted pseudoinverse, generalized singular values, and constrained least squares problems*, BIT, 22 (1982), pp. 487–501.
14. H. W. ENGL, M. HANKE, AND A. NEUBAUER, *Regularization of Inverse Problems*, Kluwer, Dordrecht, 1996.
15. K. A. GALLIVAN, M. HEATH, E. NG, B. PEYTON, R. PLEMMONS, C. ROMINE, A. SAMEH, AND R. VOIGT, *Parallel Algorithms for Matrix Computations*, SIAM, Philadelphia, 1990.
16. S. GAZZOLA, P. NOVATI, AND M. R. RUSSO, *On Krylov projection methods and Tikhonov regularization*, Electron. Trans. Numer. Anal., 44 (2015), pp. 83–123.
17. W. GAUTSCHI, *Orthogonal Polynomials: Approximation and Computation*, Oxford University Press, Oxford, 2004.
18. G. H. GOLUB, M. HEATH, AND G. WAHBA, *Generalized cross-validation as a method for choosing a good ridge parameter*, Technometrics, 21 (1979), pp. 215–223.
19. G. H. GOLUB AND G. MEURANT, *Matrices, Moments and Quadrature with Applications*, Princeton University Press, Princeton, 2010.
20. G. H. GOLUB AND U. VON MATT, *Generalized cross-validation for large-scale problems*, J. Comput. Graph. Stat., 6 (1997), pp. 1–34.
21. P. C. HANSEN, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, 1998.
22. P. C. HANSEN, *Regularization tools version 4.0 for MATLAB 7.3*, Numer. Algorithms, 46 (2007), pp. 189–194.
23. P. C. HANSEN, J. G. NAGY, AND K. TIGKOS, *Rotational image deblurring with sparse matrices*, BIT, 54 (2014), pp. 649–671.
24. M. F. HUTCHINSON, *A stochastic estimator of the trace of the infuence matrix for Laplacian smoothing splines*, Commun. Statist. Simula., 18 (1989), pp. 1059–1076.
25. K. JBILOU, A. MESSAOUDI, AND H. SADOK, *Global FOM and GMRES algorithms for matrix equations*, Appl. Numer. Math., 31 (1999), pp. 49–63.
26. 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.
27. S. KINDERMANN, *Discretization independent convergence rates for noise level-free parameter choice rules for the regularization of ill-conditioned problems*, Electron. Trans. Numer. Anal., 40 (2013), pp. 58–81.
28. G. LÓPEZ LAGOMASINO, L. REICHEL, AND L. WUNDERLICH, *Matrices, moments, and rational quadrature*, Linear Algebra Appl., 429 (2008), pp. 2540–2554.
29. P. NOVATI AND M. R. RUSSO, *A GCV based Arnoldi–Tikhonov regularization method*, BIT, 54 (2014), pp. 501–521.
30. M. REDIVO-ZAGLIA AND G. RODRIGUEZ, *smt: a Matlab toolbox for structured matrices*, Numer. Algorithms, 59 (2012), pp. 639–659.
31. L. REICHEL AND G. RODRIGUEZ, *Old and new parameter choice rules for discrete ill-posed problems*, Numer. Algorithms, 63 (2013), pp. 65–87.
32. L. REICHEL, G. RODRIGUEZ, AND S. SEATZU, *Error estimates for large-scale ill-posed problems*, Numer. Algorithms, 51 (2009), pp. 341–361.
33. R. B. SIDJE, A. B. WILLIAMS, AND K. BURRAGE, *Fast generalized cross validation using Krylov subspace method*, Numer. Algorithms, 47 (2008), pp. 109–131.
34. J. TANG AND Y. SAAD, *A probing method for computing the diagonal of the matrix inverse*, Numer. Linear Algebra Appl., 19 (2012), pp. 485–501.
35. F. TOUTOUNIAN AND S. KARIMI, *Global least squares method (Gl-LSQR) for solving general linear systems with several right-hand sides*, Appl. Math. Comput., 178 (2006), pp. 452–460.
36. J. M. VARAH, *The prolate matrix*, Linear Algebra Appl., 187 (1993), pp. 269–278.

DOI: 10.1002/nla