SIAM J. SCI. COMPUT.
Vol. 15, No. 2, pp. 297–312, March 1994

004

RESIDUAL SMOOTHING TECHNIQUES FOR ITERATIVE METHODS*

LU ZHOU† AND HOMER F. WALKER†

Abstract. An iterative method for solving a linear system $Ax = b$ produces iterates $\{x_k\}$ with associated residual norms that, in general, need not decrease "smoothly" to zero. "Residual smoothing" techniques are considered that generate a second sequence $\{y_k\}$ via a simple relation $y_k = (1-\eta_k)y_{k-1}+\eta_k x_k$. The authors first review and comment on a technique of this form introduced by Schönauer and Weiss that results in $\{y_k\}$ with monotone decreasing residual norms; this is referred to as *minimal residual smoothing*. Certain relationships between the residuals and residual norms of the biconjugate gradient (BCG) and quasi-minimal residual (QMR) methods are then noted, from which it follows that QMR can be obtained from BCG by a technique of this form; this technique is extended to generally applicable quasi-minimal residual smoothing. The practical performance of these techniques is illustrated in a number of numerical experiments.

Key words. iterative linear algebra methods, Krylov subspace methods, residual smoothing methods, GMRES, CGS, Bi-CGSTAB methods, QMR methods

AMS subject classification. 65F10

1. Introduction.

In recent years, there has been a great deal of interest in iterative methods for solving a general nonsymmetric linear system

$$Ax = b,$$

where $A \in \mathbb{R}^{n\times n}$ and $x, b \in \mathbb{R}^n$. The quality of the iterates $\{x_k\}$ produced by a method is often judged by the behavior of $\{\|r_k\|\}$, where $r_k = b - Ax_k$; in particular, it is usually desirable that $\{\|r_k\|\}$ converge "smoothly" to zero.

In the widely used generalized minimal residual (GMRES) method [13], each x_k is characterized by

$$\|b - Ax_k\|_2 = \min_{x\in x_0+\mathcal{K}_k(r_0,A)} \|b - Ax\|_2,$$

where $\|\cdot\|_2$ is the Euclidean norm and the *Krylov subspace* $\mathcal{K}_k(r_0, A)$ is defined by

$$\mathcal{K}_k(r_0, A) = \text{span}\left\{r_0, Ar_0, \ldots, A^{k-1}r_0\right\}.$$

For GMRES, then, $\{\|r_k\|_2\}$ converges to zero optimally among all *Krylov subspace methods*, for which $x_k \in x_0 + \mathcal{K}_k(r_0, A)$. Other methods, such as biconjugate gradient (BCG) [12], [4] and conjugate gradient squared (CGS) [15], have certain advantages over GMRES but often exhibit very irregular residual-norm behavior. This irregular behavior has provided an incentive for the development of methods that have similar advantages but produce better behaved residual norms, such as the biconjugate gradient stabilized (Bi-CGSTAB) methods [10], [17] and methods based on the quasi-minimal residual (QMR) approach [2], [3], [6]–[8].

Another approach to generating well-behaved residual norms has been pursued in [14] and [18]. In this approach, an auxiliary sequence $\{y_k\}$ is generated from $\{x_k\}$ by a relation

$$\begin{aligned} &y_0 = x_0, \\ &y_k = (1-\eta_k)y_{k-1} + \eta_k x_k, \qquad k = 1, 2, \ldots, \end{aligned} \tag{1.1}$$

*Received by the editors May 18, 1992; accepted for publication (in revised form) December 27, 1992. This work was supported in part by United States Air Force Office of Scientific Research grant AFOSR-91-0294 and United States Department of Energy grant DE-FG02-92ER25136, both with Utah State University.

†Department of Mathematics and Statistics, Utah State University, Logan, Utah 84322-3900.

in which each η_k is chosen to minimize $\|b - A((1-\eta)y_{k-1} + \eta x_k)\|_2$ over $\eta \in \mathbb{R}$, i.e.,

$$\eta_k = -\frac{s_{k-1}^T(r_k - s_{k-1})}{\|r_k - s_{k-1}\|_2^2}, \tag{1.2}$$

where $s_{k-1} = b - Ay_{k-1}$. (A weighted Euclidean norm is allowed in [18], but this is not important here.) The resulting residuals $s_k = b - Ay_k$ clearly have monotone decreasing Euclidean norms, and $\|s_k\|_2 \leq \|r_k\|_2$ for each k.

Our purpose here is to explore residual smoothing techniques of the form (1.1). In §2, we elaborate briefly on the particular technique of [14] and [18] described above, which we refer to here as *minimal residual smoothing* (MRS). In §3, we first note that the residuals and residual norms of the QMR and BCG methods are related in certain ways, from which it follows that QMR can be obtained from BCG by a technique of the form (1.1). We then extend this to a *quasi-minimal residual smoothing* (QMRS) technique applicable to any iterative method.[1] We describe a number of illustrative numerical experiments in §4 and discuss conclusions in §5.

A notational convention: When helpful, we denote iterates and residuals associated with a particular method by superscripts indicating that method.

2. Minimal residual smoothing. Assuming we have some iterative method that generates iterates $\{x_k\}$ and corresponding residuals $\{r_k\}$, we formulate the MRS technique of [14], [18] as follows:

ALGORITHM 2.1. Minimal residual smoothing [14], [18].

INITIALIZE: SET $s_0 = r_0$ AND $y_0 = x_0$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:

COMPUTE x_k AND r_k.
COMPUTE η_k by (1.2).
SET $s_k = s_{k-1} + \eta_k(r_k - s_{k-1})$ AND $y_k = y_{k-1} + \eta_k(x_k - y_{k-1})$.

There is a potential numerical difficulty with Algorithm 2.1. In practice, when the performance of an iterative method has become degraded through numerical error, the computed value of r_k can differ significantly from $b - Ax_k$. When this happens, the value of s_k computed in Algorithm 2.1 can differ significantly from $b - Ay_k$. Algorithm 2.2 below is a mathematically equivalent variation that does not suffer this difficulty provided an accurate value of Ap_k is available for each $p_k = x_k - x_{k-1}$, which is very often the case. In Algorithm 2.2, both s_k and y_k are determined directly from p_k and Ap_k; in particular, r_k is not involved in the computation at all. The intermediate quantities u_k and v_k are maintained so that, after the updating in the final step, we have $r_k = s_k - u_k$ and $x_k = y_k + v_k$. The comparative practical behavior of Algorithms 2.1 and 2.2 is illustrated in §4.1.

ALGORITHM 2.2. Minimal residual smoothing.

INITIALIZE: SET $s_0 = r_0$, $y_0 = x_0$, AND $u_0 = v_0 = 0$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:

COMPUTE $p_k = x_k - x_{k-1}$ AND Ap_k.
SET $u_k = u_{k-1} + Ap_k$ AND $v_k = v_{k-1} + p_k$.
COMPUTE $\eta_k = s_{k-1}^T u_k / u_k^T u_k$.
SET $s_k = s_{k-1} - \eta_k u_k$ AND $y_k = y_{k-1} + \eta_k v_k$.
UPDATE $u_k \leftarrow (1 - \eta_k)u_k$ AND $v_k \leftarrow (1 - \eta_k)v_k$.

[1] This should not be confused with the QMR squared method, designated QMRS in [8].

In MRS, each s_k is the vector of minimal Euclidean norm along the line containing s_{k-1} and r_k. There is an obvious extension, viz., inductively determining s_k as the vector of minimal Euclidean norm in the affine subspace containing s_{k-1} and $r_k, \ldots, r_{k-\ell}$ for some ℓ. This extension can be implemented reasonably economically (in $O(\ell n) + O(\ell^2)$ arithmetic operations for each k). However, there is no guarantee of improvement, and, in experiments that we carried out, it was not more effective than basic MRS in reducing the residual norm.

3. Quasi-minimal residual smoothing.

3.1. How QMR smoothes BCG. We first develop certain relationships between the QMR and BCG residuals and their norms. We recall the BCG method as follows:

ALGORITHM 3.1.1. Biconjugate gradient method [12], [4].

INITIALIZE:

CHOOSE x_0 AND SET $q_0 = r_0 = b - Ax_0$.
CHOOSE $\tilde{r}_0 \neq 0$, AND SET $\tilde{q}_0 = \tilde{r}_0$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:

COMPUTE $\delta_k = \tilde{r}_{k-1}^T r_{k-1} / \tilde{q}_{k-1}^T A q_{k-1}$ AND SET $x_k = x_{k-1} + \delta_k q_{k-1}$.
SET $r_k = r_{k-1} - \delta_k A q_{k-1}$ AND $\tilde{r}_k = \tilde{r}_{k-1} - \delta_k A^T \tilde{q}_{k-1}$.
COMPUTE $\gamma_k = \tilde{r}_k^T r_k / \tilde{r}_{k-1}^T r_{k-1}$.
SET $q_k = r_k + \gamma_k q_{k-1}$ AND $\tilde{q}_k = \tilde{r}_k + \gamma_k \tilde{q}_{k-1}$.

We consider the basic QMR method obtained from the general method in [7] by omitting diagonal scaling and using the classical nonsymmetric Lanczos process [11] *without* look-ahead. To discuss this, we note that $r_0, \ldots, r_{k-1}$ generated by Algorithm 3.1.1 clearly form a basis of $\mathcal{K}_k(r_0, A)$. Set $V_k = (v_1, \ldots, v_k)$ with $v_i = r_{i-1}/\rho_{i-1}$ and $\rho_{i-1} = \|r_{i-1}\|_2$ for each i. The columns of V_k also form a basis of $\mathcal{K}_k(r_0, A)$, and any $x \in x_0 + \mathcal{K}_k(r_0, A)$ can be written as

$$x = x_0 + V_k z, \qquad z \in \mathbb{R}^k. \tag{3.1.1}$$

By the nonsymmetric Lanczos process, we have

$$AV_k = V_{k+1} H_k, \tag{3.1.2}$$

where $H_k \in \mathbb{R}^{(k+1)\times k}$ is tridiagonal. From (3.1.2), the residual for x in (3.1.1) is

$$\begin{aligned} b - Ax &= r_0 - AV_k z = r_0 - V_{k+1} H_k z \\ &= V_{k+1}(\rho_0 e_1 - H_k z), \end{aligned} \tag{3.1.3}$$

where $e_1 = (1, 0, \ldots, 0)^T \in \mathbb{R}^{k+1}$. The kth QMR iterate is defined by $x_k^{\mathrm{QMR}} = x_0 + V_k z_k$, where z_k satisfies

$$\|\rho_0 e_1 - H_k z_k\|_2 = \tau_k \equiv \min_{z \in \mathbb{R}^k} \|\rho_0 e_1 - H_k z\|_2.$$

For later reference, we recall from [7] that

$$\|r_k^{\mathrm{QMR}}\|_2 \leq \sqrt{k+1}\, \tau_k. \tag{3.1.4}$$

In [7], the upper bound $\sqrt{k+1}\, \tau_k$ is used in a preliminary test for termination in QMR. We comment further on it below.

To develop the desired relationships between QMR and BCG, we note that $q_0, \ldots, q_{k-1}$ generated by Algorithm 3.1.1 also form a basis of $\mathcal{K}_k(r_0, A)$. Set

$$Y_k = (\delta_1 q_0, \ldots, \delta_k q_{k-1})$$

and

$$G_{k+1} = \begin{pmatrix} \rho_0 & & & \\ & \rho_1 & & \\ & & \ddots & \\ & & & \rho_k \end{pmatrix}.$$

We assume that Algorithm 3.1.1 can successfully carry out the kth step, in which case $\delta_1, \ldots, \delta_k$ are all nonzero and the columns of Y_k form a basis of $\mathcal{K}_k(r_0, A)$. From Algorithm 3.1.1, we have

$$AY_k = V_{k+1} G_{k+1} \bar{H}_k, \tag{3.1.5}$$

where

$$\bar{H}_k = \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & -1 \end{pmatrix} \in \mathbb{R}^{(k+1)\times k}.$$

Since the columns of Y_k are a basis for $\mathcal{K}_k(r_0, A)$, we can write $x \in x_0 + \mathcal{K}_k(r_0, A)$ as $x = x_0 + Y_k \bar{z}$ for $\bar{z} \in \mathbb{R}^k$. Then from (3.1.5), we have

$$\begin{aligned} b - Ax &= r_0 - AY_k \bar{z} = r_0 - V_{k+1} G_{k+1} \bar{H}_k \bar{z} \\ &= V_{k+1} \left[G_{k+1} (e_1 - \bar{H}_k \bar{z}) \right]. \end{aligned} \tag{3.1.6}$$

Comparing (3.1.3) and (3.1.6), one easily verifies that

$$\tau_k = \min_{z \in \mathbb{R}^k} \|\rho_0 e_1 - H_k z\|_2 = \min_{\bar{z} \in \mathbb{R}^k} \|G_{k+1}(e_1 - \bar{H}_k \bar{z})\|_2. \tag{3.1.7}$$

The least-squares problem on the right-hand side of (3.1.7) is easily solved. Setting $\bar{z} = (\zeta_1, \zeta_2, \ldots, \zeta_k)^T$ gives

$$\tau_k^2 = \min_{\bar{z} \in \mathbb{R}^k} \rho_0^2 (1 - \zeta_1)^2 + \rho_1^2 (\zeta_1 - \zeta_2)^2 + \cdots + \rho_{k-1}^2 (\zeta_{k-1} - \zeta_k)^2 + \rho_k^2 \zeta_k^2.$$

With a change of variables $\xi_0 = 1 - \zeta_1, \xi_1 = \zeta_1 - \zeta_2, \ldots, \xi_{k-1} = \zeta_{k-1} - \zeta_k, \xi_k = \zeta_k$, we have

$$\tau_k^2 = \min_{\sum_{i=0}^k \xi_i = 1} \sum_{i=0}^k \rho_i^2 \xi_i^2.$$

The unique minimizer is given by

$$\xi_i = \frac{\frac{1}{\rho_i^2}}{\sum_{j=0}^k \frac{1}{\rho_j^2}}, \quad i = 0, 1, \ldots, k,$$

and so

$$\tau_k = \sqrt{\frac{1}{\sum_{j=0}^{k} \frac{1}{\rho_j^2}}}.$$

Then the upper bound in (3.1.4) is

$$\sqrt{k+1}\,\tau_k = \sqrt{\frac{1}{\frac{1}{k+1}\sum_{j=0}^{k} \frac{1}{\rho_j^2}}}, \tag{3.1.8}$$

and it follows from (3.1.6) that the QMR residual vector is

$$r_k^{\mathrm{QMR}} = V_{k+1}(\rho_0\xi_0, \ldots, \rho_k\xi_k)^T = \frac{1}{\sum_{j=0}^{k} \frac{1}{\rho_j^2}} \sum_{i=0}^{k} \frac{1}{\rho_i^2} r_i^{\mathrm{BCG}}. \tag{3.1.9}$$

These results are of interest in their own right. We summarize them in the following theorem and then discuss some consequences.

THEOREM 3.1. *With* $\rho_i = \|r_i^{\mathrm{BCG}}\|_2$ *for* $i = 0, \ldots, k$, *the* QMR *residual* r_k^{QMR} *is given by* (3.1.9) *as a convex combination of the* BCG *residuals* $r_0^{\mathrm{BCG}}, \ldots, r_k^{\mathrm{BCG}}$. *The upper bound* $\sqrt{k+1}\,\tau_k$ *on* $\|r_k^{\mathrm{QMR}}\|_2$ *in* (3.1.4) *is given by* (3.1.8) *as the square root of the harmonic mean of* $\rho_0^2, \ldots, \rho_k^2$.

Equation (3.1.9) gives considerable insight into how QMR produces relatively smoothly decreasing (if not monotonically decreasing) residual norms. If $\|r_k^{\mathrm{BCG}}\|_2$ is small for some k, then r_k^{BCG} is given large weight in the convex combination (3.1.9) and $\|r_k^{\mathrm{QMR}}\|_2$ is small. If subsequent BCG residual norms increase, then the effect of this increase is mollified by relatively small weights in (3.1.9) and any increase in $\|r_k^{\mathrm{QMR}}\|_2$ is correspondingly small. These observations are borne out in Experiments 3 and 4 in §4.2 below, in which each QMR residual norm is roughly comparable to the best BCG residual norm obtained so far.

It follows from (3.1.8) that the upper bound $\sqrt{k+1}\,\tau_k$ on $\|r_k^{\mathrm{QMR}}\|_2$ is greater than or equal to $\min_{j=0,\ldots,k} \|r_j^{\mathrm{BCG}}\|_2$, with equality if and only if $\|r_0^{\mathrm{BCG}}\|_2 = \cdots = \|r_k^{\mathrm{BCG}}\|_2$. In fact, this upper bound can be a significant overestimate of $\|r_k^{\mathrm{QMR}}\|_2$, e.g., when A is nearly symmetric. Indeed, if A were symmetric, then the nonsymmetric Lanczos process would become the symmetric process, the vectors r_i^{BCG} would be mutually orthogonal, and we would have $\|r_k^{\mathrm{QMR}}\|_2 = \tau_k$ from (3.1.9).

From (3.1.9), we immediately obtain

$$\begin{aligned} r_k^{\mathrm{QMR}} &= \frac{\tau_k^2}{\tau_{k-1}^2} r_{k-1}^{\mathrm{QMR}} + \frac{\tau_k^2}{\rho_k^2} r_k^{\mathrm{BCG}}, \\ x_k^{\mathrm{QMR}} &= \frac{\tau_k^2}{\tau_{k-1}^2} x_{k-1}^{\mathrm{QMR}} + \frac{\tau_k^2}{\rho_k^2} x_k^{\mathrm{BCG}}. \end{aligned} \tag{3.1.10}$$

This has two useful consequences: First, since $\tau_k^2/\tau_{k-1}^2 + \tau_k^2/\rho_k^2 = 1$, we conclude that *the* QMR *method is obtained from the* BCG *method by a smoothing technique of the form* (1.1).[2] We generalize this technique in §3.2 below. Second, (3.1.10) gives a convenient and economical way of obtaining the QMR iterates and residuals from BCG. We note, however, that

[2]This is also implicit in results in [8] derived in a different manner. A different but equivalent way of determining the QMR iterates from BCG is also given in [8, §3].

if r_k^{BCG} and $b - Ax_k^{\mathrm{BCG}}$ differ significantly because of numerical error, then, as with Algorithm 2.1, r_k^{QMR} and $b - Ax_k^{\mathrm{QMR}}$ computed by (3.1.10) can also differ significantly. In this case, an analogue of Algorithm 2.2 obtained by applying the general Algorithm 3.2.2 below should perform better. In §4.2, we describe an experiment in which (3.1.10) performs poorly while the mathematically equivalent implementation of Algorithm 3.2.2 performs satisfactorily.

3.2. General quasi-minimal residual smoothing. Suppose we have some iterative method that generates iterates $\{x_k\}$ and corresponding residuals $\{r_k\}$. Since the difference of any two residuals is in the range of A, we can find for each $k \geq 1$ a w_k such that $r_{k-1} - r_k = Aw_k$. Define $Y_k = (w_1, \ldots, w_k)$ and $\rho_i = \|r_i\|_2$ for $i = 0, \ldots, k$. With V_{k+1}, G_{k+1}, and $\bar{H}_k$ as in §3.1, we again have (3.1.5), and so (3.1.6) holds for $x = x_0 + Y_k\bar{z}$, $\bar{z} \in \mathbb{R}^k$. It follows as before that

$$\tau_k \equiv \min_{\bar{z}\in\mathbb{R}^k} \|G_{k+1}(e_1 - \bar{H}_k\bar{z})\|_2 = \sqrt{\frac{1}{\sum_{j=0}^{k} \frac{1}{\rho_j^2}}}, \tag{3.2.1}$$

and, for the minimizing $\bar{z}$, we have, as in (3.1.9),

$$s_k \equiv V_{k+1}\left[G_{k+1}(e_1 - \bar{H}_k\bar{z})\right] = \frac{1}{\sum_{j=0}^{k} \frac{1}{\rho_j^2}} \sum_{i=0}^{k} \frac{1}{\rho_i^2} r_i. \tag{3.2.2}$$

In analogy with (3.1.10), we can use (3.2.2) to define "smoothed" residuals and iterates by

$$s_k = \frac{\tau_k^2}{\tau_{k-1}^2} s_{k-1} + \frac{\tau_k^2}{\rho_k^2} r_k, \qquad y_k = \frac{\tau_k^2}{\tau_{k-1}^2} y_{k-1} + \frac{\tau_k^2}{\rho_k^2} x_k,$$

and we can determine τ_k simply by $1/\tau_k^2 = 1/\tau_{k-1}^2 + 1/\rho_k^2$. This leads to the following algorithm.

ALGORITHM 3.2.1. Quasi-minimal residual smoothing.

INITIALIZE: SET $s_0 = r_0$, $y_0 = x_0$, AND $\tau_0 = \rho_0 = \|r_0\|_2$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:

COMPUTE x_k AND r_k.

SET $\rho_k = \|r_k\|_2$ AND DEFINE τ_k by $1/\tau_k^2 = 1/\tau_{k-1}^2 + 1/\rho_k^2$.

SET $s_k = (\tau_k^2/\tau_{k-1}^2)s_{k-1} + (\tau_k^2/\rho_k^2)r_k$ AND $y_k = (\tau_k^2/\tau_{k-1}^2)y_{k-1} + (\tau_k^2/\rho_k^2)x_k$.

In this algorithm, as in Algorithm 2.1, the divergence of r_k and $b - Ax_k$ through numerical error can cause s_k and $b - Ay_k$ to differ significantly. We formulate a mathematically equivalent algorithm analogous to Algorithm 2.2 that should avoid this difficulty provided an accurate value of Ap_k is available for each $p_k = x_k - x_{k-1}$.

ALGORITHM 3.2.2. Quasi-minimal residual smoothing.

INITIALIZE: SET $s_0 = r_0$, $y_0 = x_0$, $u_0 = v_0 = 0$, AND $\tau_0 = \rho_0 = \|r_0\|_2$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:

COMPUTE $p_k = x_k - x_{k-1}$ AND Ap_k.

SET $u_k = u_{k-1} + Ap_k$ AND $v_k = v_{k-1} + p_k$.

SET $\rho_k = \|s_{k-1} - u_k\|_2$ AND DEFINE τ_k BY $1/\tau_k^2 = 1/\tau_{k-1}^2 + 1/\rho_k^2$.

SET $s_k = s_{k-1} - (\tau_k^2/\rho_k^2)u_k$ AND $y_k = y_{k-1} + (\tau_k^2/\rho_k^2)v_k$.

UPDATE $u_k \leftarrow (1 - (\tau_k^2/\rho_k^2))u_k$ AND $v_k \leftarrow (1 - (\tau_k^2/\rho_k^2))v_k$.

It would be trivial to incorporate a diagonal scaling matrix

$$\Omega_{k+1} = \operatorname{diag}(\omega_0, \ldots, \omega_k) \in \mathbb{R}^{(k+1)\times(k+1)}$$

into the above developments, as is typically done in deriving QMR-type methods. In place of (3.2.1) and (3.2.2), we would then have

$$\tau_k \equiv \min_{\bar{z}\in\mathbb{R}^k} \|\Omega_{k+1}G_{k+1}(e_1 - \bar{H}_k\bar{z})\|_2 = \sqrt{\frac{1}{\sum_{j=0}^{k}\frac{1}{\omega_j^2\rho_j^2}}},$$

and, for the minimizing $\bar{z}$,

$$s_k \equiv V_{k+1}\Omega_{k+1}^{-1}\left[\Omega_{k+1}G_{k+1}(e_1 - \bar{H}_k\bar{z})\right] = \frac{1}{\sum_{j=0}^{k}\frac{1}{\omega_j^2\rho_j^2}}\sum_{i=0}^{k}\frac{1}{\omega_i^2\rho_i^2}r_i.$$

Algorithms 3.2.1 and 3.2.2 would be modified by replacing each occurrence of ρ_k^2 by $\omega_k^2\rho_k^2$. Similar remarks hold for the developments in §3.1.

As in Theorem 3.1, each s_k generated by Algorithms 3.2.1 and 3.2.2 is given by (3.2.2) as a convex combination of $r_0, \ldots, r_k$. Also, (3.2.1) and (3.2.2) imply immediately an analogue of (3.1.4) and (3.1.8), i.e.,

$$\|s_k\|_2 \le \sqrt{k+1}\,\tau_k = \sqrt{\frac{1}{\frac{1}{k+1}\sum_{j=0}^{k}\frac{1}{\rho_j^2}}}.$$

The remarks in the two paragraphs following Theorem 3.1, with appropriate changes, are valid here.

We mention the possibility of applying Algorithms 3.2.1 and 3.2.2 repeatedly to produce increasingly "smoothed" residuals and iterates. However, in an experiment that we performed, applying these algorithms twice to a BCG sequence (i.e., applying them once to a QMR sequence) showed no practical advantages.

In the case of the CGS method, Algorithms 3.2.1 and 3.2.2 can be applied in a straightforward way; however, we note an interesting alternative in this specific case. We write the CGS method as follows:

ALGORITHM 3.2.3. Conjugate gradient squared method [15].

INITIALIZE:
CHOOSE x_0 AND SET $p_0 = u_0 = r_0 = b - Ax_0$ AND $v_0 = Ap_0$.
CHOOSE $\tilde{r}_0$ SUCH THAT $\rho_0 = \tilde{r}_0^T r_0 \neq 0$.

ITERATE: FOR $k = 1, 2, \ldots,$ DO:
COMPUTE $\sigma_{k-1} = \tilde{r}_0^T v_{k-1}$, $\alpha_{k-1} = \rho_{k-1}/\sigma_{k-1}$, AND
$q_k = u_{k-1} - \alpha_{k-1}v_{k-1}$.
SET $x_k = x_{k-1} + \alpha_{k-1}(u_{k-1} + q_k)$ AND $r_k = r_{k-1} - \alpha_{k-1}A(u_{k-1} + q_k)$.
SET $\rho_k = \tilde{r}_0^T r_k$, $\beta_k = \rho_k/\rho_{k-1}$, $u_k = r_k + \beta_k q_k$,
$p_k = u_k + \beta_k(q_k + \beta_k p_{k-1})$, AND $v_k = Ap_k$.

We define auxiliary iterates and residuals as follows: For $k = 0, 1, \ldots,$ set

$$\begin{aligned} \hat{x}_{2k} &= x_k, & \hat{x}_{2k+1} &= x_k + \alpha_k u_k, \\ \hat{r}_{2k} &= r_k, & \hat{r}_{2k+1} &= r_k - \alpha_k A u_k. \end{aligned} \tag{3.2.3}$$

It is not hard to verify that, if we apply Algorithms 3.2.1 and 3.2.2 (modified, if necessary, to incorporate diagonal scaling) to the iterates $\{\hat{x}_k\}$ and residuals $\{\hat{r}_k\}$, then the resulting method is equivalent to the "transpose-free" QMR-like method derived from CGS in [6]. This is *not* the same as the method obtained by the straightforward application of Algorithms 3.2.1 and 3.2.2 to CGS; in §4.3, we illustrate the practical performance of these two ways of applying QMRS to CGS. This equivalence may help to explain experiments in [3] and also Experiments 5 and 6 in §4.3 below, in which residual norms from the method of [6] are roughly comparable to the best CGS residual norms obtained so far. (However, an example is given in [8] in which CGS diverges while the method of [6] converges.)

One can similarly obtain the QMRCGSTAB method of [3] by applying Algorithms 3.2.1 and 3.2.2 to Bi-CGSTAB [17]. Indeed, in addition to iterates $\{x_k\}$ and residuals $\{r_k\}$, Bi-CGSTAB produces $\{s_k, p_k, \alpha_k, \omega_k\}$ such that

$$x_k = x_{k-1} + \alpha_k p_k + \omega_k s_k,$$
$$s_k = r_{k-1} - \alpha_k A p_k, \qquad r_k = s_k - \omega_k A s_k.$$

In this case, we define auxiliary iterates and residuals by

$$\hat{x}_{2k} = x_k, \qquad \hat{x}_{2k+1} = x_k + \alpha_{k+1} p_{k+1},$$
$$\hat{r}_{2k} = r_k, \qquad \hat{r}_{2k+1} = s_{k+1},$$

for $k = 0, 1, \ldots$. Then the method obtained by applying Algorithms 3.2.1 and 3.2.2 (modified to incorporate diagonal scaling, if necessary) to $\{\hat{x}_k\}$ and $\{\hat{r}_k\}$ is equivalent to QMRCGSTAB.

4. Numerical experiments. We report on numerical experiments that illustrate the performance of the algorithms discussed previously. The test problem used in all but one of the experiments is a discretization of

$$\Delta u + cu + d\frac{\partial u}{\partial x} = f \quad \text{in } D, \qquad u = 0 \quad \text{on } \partial D, \tag{4.1}$$

where $D = [0, 1] \times [0, 1]$ and c and d are constants. In the experiments outlined here, we took $f \equiv 1$ and used a 100×100 mesh of equally spaced discretization points in D, so that the resulting linear systems were of dimension 10,000. Discretization was by the usual second order centered differences. Preconditioning, when used, was with a fast Poisson solver from FISHPACK [16]. In all experiments, computing was done in double precision on Sun Microsystems workstations.

4.1. Comparing Algorithms 2.1 and 2.2. We compared the numerical performance of the mathematically equivalent MRS Algorithms 2.1 and 2.2 in the following two experiments.

Experiment 1. This was a controlled experiment in which we artificially simulated the numerical breakdown of a convergent algorithm through exponential error growth. In each simulation, we first generated random[3] A and b and computed $x_* = A^{-1}b$ using a direct method; then, for $k = 0, \ldots, k_{\max}$, we generated $x_k = x_* + 2^{-k}u_k$ for random u_k, computed $r_k = b - Ax_k$, and perturbed $x_k \leftarrow x_k + \epsilon(k/k_{\max})^\nu v_k$ and $r_k \leftarrow r_k + \epsilon(k/k_{\max})^\nu w_k$ for random

[3]Here, "random" means having components that are sampled independently from a normal distribution with zero mean and unit variance. Random normal components were generated using the URAND subroutine from [5] followed by a Box–Muller transformation; see [5, p. 247]. In the particular experiment reported in Fig. 1, the seed 2468 was used in URAND.

v_k and w_k and fixed $\epsilon > 0$ and positive integer ν. We plotted $\log_{10} \|s_k\|_2$ and $\log_{10} \|b - Ay_k\|_2$ versus k for both Algorithms 2.1 and 2.2 in a number of trials. Typical results (with $n = 10$, $\epsilon = 10^2$, $\nu = 10$, and $k_{\max} = 50$) are shown in Fig. 1. Note that in Fig. 1, the solid curve is actually a superposition of the curves for $\log \|s_k\|_2$ and $\log \|b - Ay_k\|_2$ generated by Algorithm 2.2; they are visually indistinguishable. In contrast, the corresponding curves for Algorithm 2.1 diverge strongly.

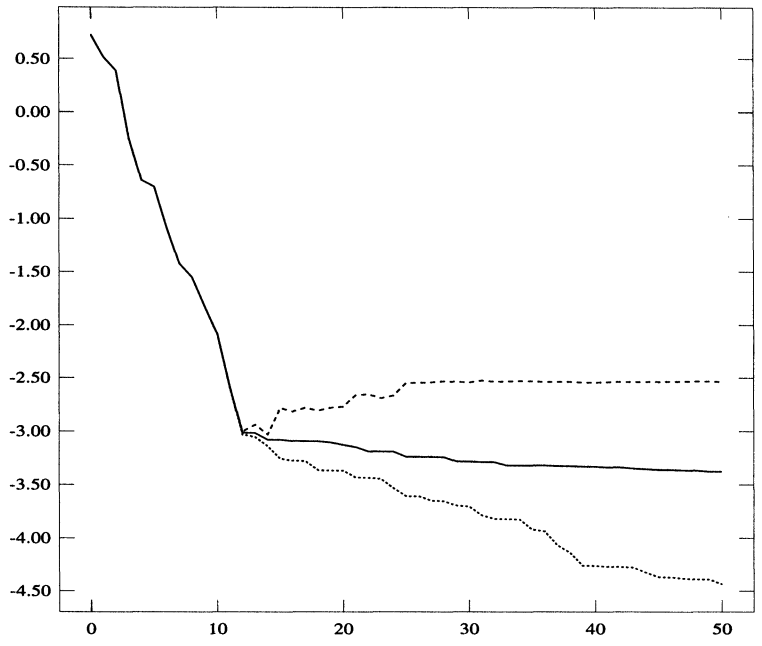

FIG. 1. *Residual norms versus the number of iterations for Experiment* 1. *Solid curve*: $\log_{10} \|s_k\|_2$ *and* $\log_{10} \|b - Ay_k\|_2$ *for Algorithm* 2.2; *dotted curve*: $\log_{10} \|s_k\|_2$ *for Algorithm* 2.1; *dashed curve*: $\log_{10} \|b - Ay_k\|_2$ *for Algorithm* 2.1.

Experiment 2. In this, we applied BCG without preconditioning to problem (4.1) with $c = d = 50$. For insight here and in Experiment 3 below, we first plotted $\log_{10} \|r_k^{\mathrm{BCG}}\|_2$ and $\log_{10} \|b - Ax_k^{\mathrm{BCG}}\|_2$ versus k in Fig. 2. Note the divergence of $\|r_k^{\mathrm{BCG}}\|_2$ and $\|b - Ax_k^{\mathrm{BCG}}\|_2$ beyond about $k = 280$; this is apparently due to rounding errors introduced in earlier iterations when $\|r_k^{\mathrm{BCG}}\|_2$ was very large. We next plotted $\log_{10} \|s_k\|_2$ and $\log_{10} \|b - Ay_k\|_2$ versus k for both Algorithms 2.1 and 2.2. The results are shown in Fig. 3, in which the graphs of $\log_{10} \|b - Ay_k\|_2$ for Algorithm 2.1 (solid curve) and for Algorithm 2.2 (long-dashed curve) are nearly superimposed. Note that for Algorithm 2.2, the values of $\log_{10} \|s_k\|_2$ stay fairly close to the values of $\log_{10} \|b - Ay_k\|_2$ in the later iterations, while for Algorithm 2.1, the values of $\log_{10} \|s_k\|_2$ are wrongly forced downward by the (inaccurate) decreasing values of $\|r_k^{\mathrm{BCG}}\|_2$. In view of the superior performance of Algorithm 2.2, it was used to implement MRS in the remaining experiments discussed below.

4.2. Smoothing methods for BCG. We compared the performance of BCG, QMR, and MRS applied to BCG on problem (4.1) in the two experiments below. In these, MRS was implemented as in Algorithm 2.2, and the QMR iterates and their residuals were generated from the BCG iterates and residuals using either (3.1.10) or a mathematically equivalent implementation of Algorithm 3.2.2. Caution: Comments below regarding the numerical

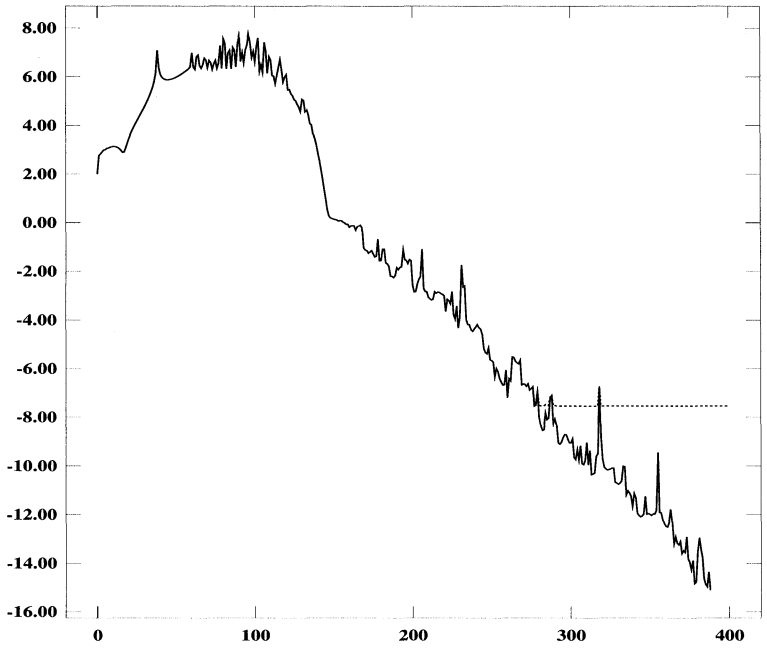

FIG. 2. *Recursive and true* BCG *residual norms versus the number of iterations for Experiments* 2 *and* 3. *Solid curve*: $\log_{10} \|r_k^{\mathrm{BCG}}\|_2$; *dotted curve*: $\log_{10} \|b - Ax_k^{\mathrm{BCG}}\|_2$.

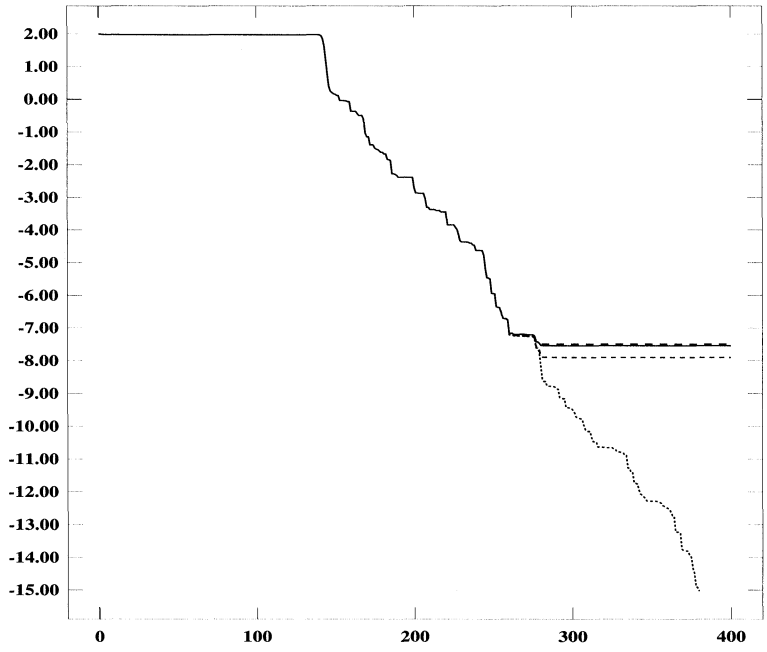

FIG. 3. *Residual norms versus the number of iterations for Experiment* 2. *Solid curve and nearly superimposed long-dashed curve*: $\log_{10} \|b - Ay_k\|_2$ *for Algorithms* 2.1 *and* 2.2; *dotted curve*: $\log_{10} \|s_k\|_2$ *for Algorithm* 2.1; *short-dashed curve*: $\log_{10} \|s_k\|_2$ *for Algorithm* 2.2.

accuracy of QMR iterates are *not* intended to apply to the general QMR method of [7], which uses the look-ahead Lanczos process; they only pertain to the soundness of (3.1.10) and Algorithm 3.2.2 applied to the BCG iterates.

Experiment 3. As in Experiment 2, we considered problem (4.1) with $c = d = 50$, without preconditioning. We first compared the accuracy of (3.1.10) and Algorithm 3.2.2 for obtaining QMR iterates from BCG iterates. The results, which are analogous to those in Fig. 3, are shown in Fig. 4, in which the graphs of $\log_{10} \|b - Ax_k^{\mathrm{QMR}}\|_2$ for (3.1.10) (solid curve) and for Algorithm 3.2.2 (long-dashed curve) are nearly indistinguishable. The remarks made in Experiment 2 about Fig. 3 are valid here, with the appropriate changes. In view of the superior performance of Algorithm 3.2.2, it was used instead of (3.1.10) or Algorithm 3.2.1 in the remaining experiments reported below, except where noted.

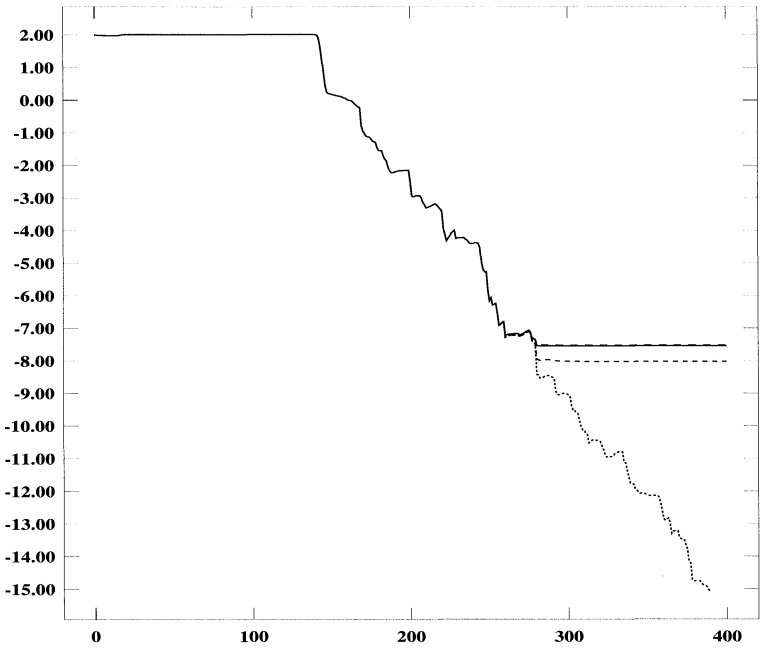

FIG. 4. *Residual norms versus the number of iterations for Experiment* 3. *Solid curve and nearly indistinguishable long-dashed curve*: $\log_{10} \|b - Ax_k^{\mathrm{QMR}}\|_2$ *for* (3.1.10) *and Algorithm* 3.2.2; *dotted curve*: $\log_{10} \|r_k^{\mathrm{QMR}}\|_2$ *for* (3.1.10); *short-dashed curve*: $\log_{10} \|r_k^{\mathrm{QMR}}\|_2$ *for Algorithm* 3.2.2.

We next compared the performance of BCG, QMR, and MRS applied to BCG on this problem. The results are shown in Fig. 5. Note that the solid curve in Fig. 5 is the *recursive* residual value $\log_{10} \|r_k^{\mathrm{BCG}}\|_2$, rather than the *true* value $\log_{10} \|b - Ax_k^{\mathrm{BCG}}\|_2$; cf. Fig. 2. Both QMR and MRS were very effective in smoothing the BCG iterates. Although fine details are hard to make out in parts of Fig. 5, the performance of QMR and MRS was, in fact, very similar over all iterations; of course, the MRS residual norms were monotone decreasing while those of QMR, although trending downward fairly smoothly, were not. Note that both QMR and MRS very effectively stabilized the iterates and residuals at about the point of greatest *true* residual reduction of BCG.

Experiment 4. We applied the three methods with preconditioning to problem (4.1) with $c = d = 100$. The preconditioning resulted in relatively well-behaved BCG residual norms, and QMR and MRS applied to BCG performed very similarly. There was no evidence of numerical error. The results are shown in Fig. 6.

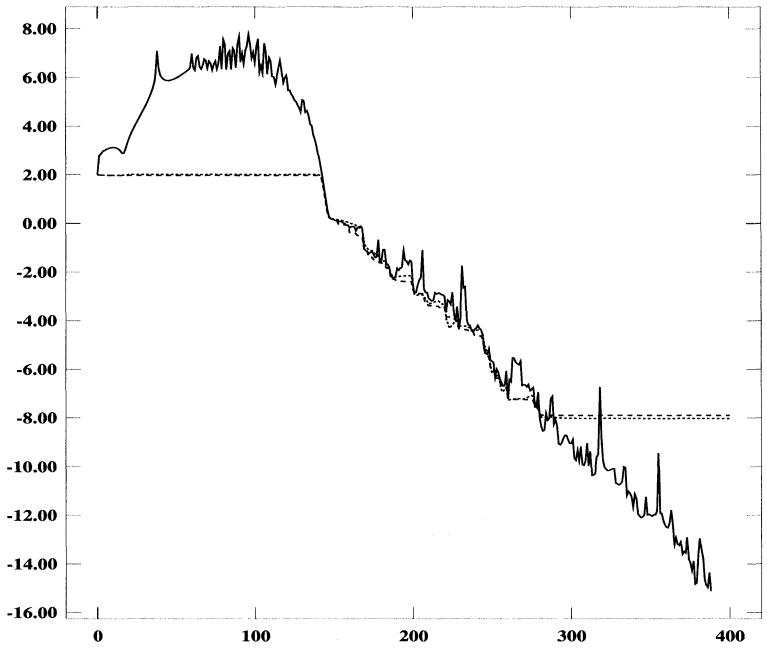

FIG. 5. *Residual norms versus the number of iterations for Experiment* 3. *Solid curve*: $\log_{10} \|r_k^{\mathrm{BCG}}\|_2$; *dotted curve*: $\log_{10} \|r_k^{\mathrm{QMR}}\|_2$; *dashed curve*: $\log_{10} \|s_k\|_2$ *for* MRS *Algorithm* 2.2 *applied to the* BCG *iterates*.

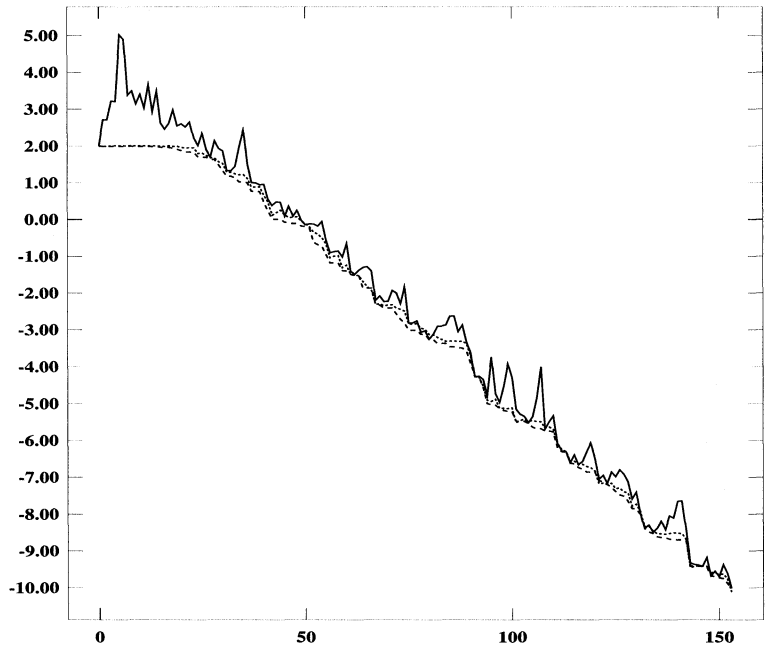

FIG. 6. *Residual norms versus the number of iterations for Experiment* 4. *Solid curve*: $\log_{10} \|r_k^{\mathrm{BCG}}\|_2$; *dotted curve*: $\log_{10} \|r_k^{\mathrm{QMR}}\|_2$; *dashed curve*: $\log_{10} \|s_k\|_2$ *for* MRS *Algorithm* 2.2 *applied to the* BCG *iterates*.

4.3. Smoothing methods for CGS. In the two experiments below, we compared the performance on problem (4.1) of CGS, the "transpose-free" QMR-like method of [6] (referred to as TFQMR below), CGS with QMRS, and CGS with MRS. We implemented TFQMR by applying QMRS to $\{\hat{x}_k\}$ and $\{\hat{r}_k\}$ defined in (3.2.3); see §3.2. The implementation of CGS with QMRS was straightforward.

Experiment 5. We applied the methods without preconditioning to problem (4.1) with $c = d = 5$. The results are given in Fig. 7. The three smoothing methods very effectively smoothed the very ill-behaved CGS residuals and performed roughly the same, although MRS tended to give slightly smaller residual norms than the other two methods. Although not shown in Fig. 7, significant divergence of $\|r_k^{\mathrm{CGS}}\|_2$ and $\|b - Ax_k^{\mathrm{CGS}}\|_2$ began at about iteration 262 and, as in Experiment 3 and Fig. 5, all three smoothing methods effectively stabilized the iterates and residuals at about the point of greatest *true* CGS residual reduction.

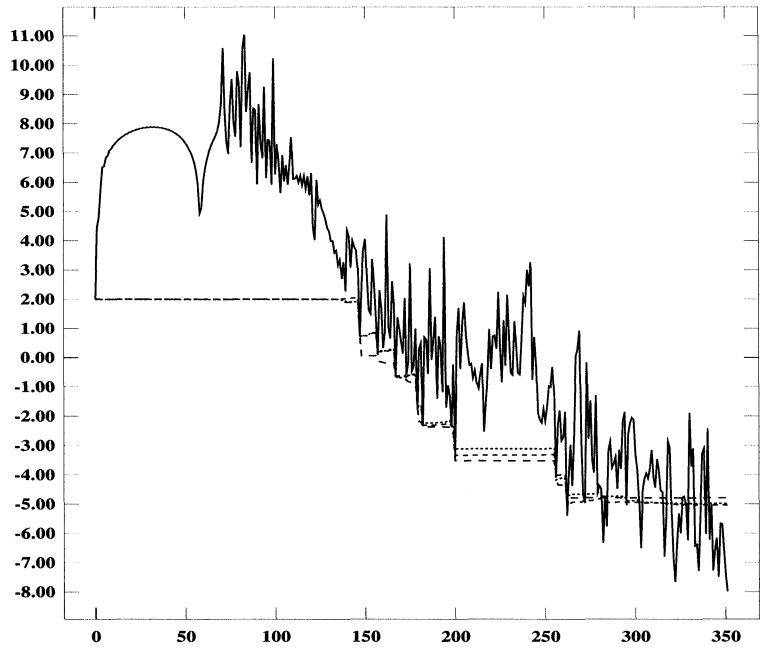

FIG. 7. *Residual norms versus the number of iterations for Experiment* 5. *Solid curve*: $\log_{10}\|r_k^{\mathrm{CGS}}\|_2$; *dotted curve*: $\log_{10}$ *of the* TFQMR *residual norms*; *short-dashed curve*: $\log_{10}\|s_k\|_2$ *for* QMRS *Algorithm* 3.2.2 *applied to the* CGS *iterates*; *long-dashed curve*: $\log_{10}\|s_k\|_2$ *for* MRS *Algorithm* 2.2 *applied to the* CGS *iterates*.

Experiment 6. We applied the methods with preconditioning to problem (4.1) with $c = d = 50$. Because r_k^{CGS} and $b - Ax_k^{\mathrm{CGS}}$ remained very close throughout this experiment, we used Algorithm 3.2.1 rather than Algorithm 3.2.2. The results are given in Fig. 8. It is evident that all three smoothing methods worked very effectively, especially in stabilizing the iterates and residuals once the limits of residual reduction were reached. It is also notable how similarly QMRS and MRS behaved, although MRS ultimately gave the smallest residual norms of all methods.

5. Summary and conclusions. We have focused on two residual smoothing techniques of the general form (1.1), minimum residual smoothing (MRS) and quasi-minimal residual smoothing (QMRS). The former generates from given iterates a sequence of auxiliary iterates with monotone decreasing residual norms that are no greater than those of the original sequence. The latter generates auxiliary iterates with residual norms that typically decrease

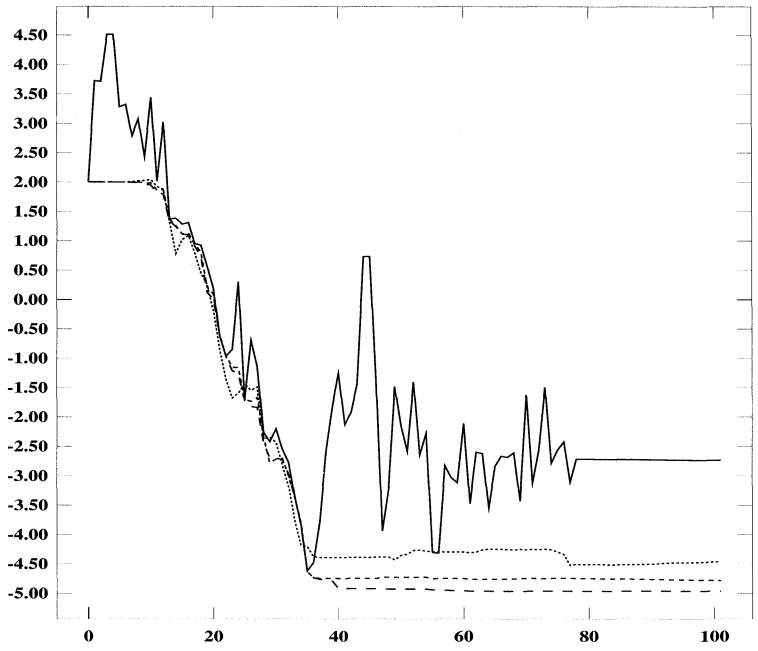

FIG. 8. *Residual norms versus the number of iterations for Experiment* 6. *Solid curve*: $\log_{10} \|r_k^{\mathrm{CGS}}\|_2$; *dotted curve*: $\log_{10}$ *of the* TFQMR *residual norms*; *short-dashed curve*: $\log_{10} \|s_k\|_2$ *for* QMRS *Algorithm* 3.2.1 *applied to the* CGS *iterates*; *long-dashed curve*: $\log_{10} \|s_k\|_2$ *for* MRS *Algorithm* 2.2 *applied to the* CGS *iterates*.

fairly smoothly, if not necessarily monotonically. It also provides insight into the workings of QMR-type methods and, in particular, indicates how they are related to and smooth the residual norms of underlying methods such as BCG and CGS.

These residual smoothing techniques provide important practical tools. Their performance is illustrated in a number of numerical experiments in §4; these indicate that both MRS and QMRS are reliable and very effective in smoothing the residuals of the underlying method when applied as in Algorithms 2.2 and 3.2.2, respectively. The performance of MRS and QMRS was roughly similar in our experiments. However, the MRS residual norms were monotone decreasing while those of QMRS, although typically trending fairly smoothly downward, were not; also, the MRS residual norms were often, although not always, slightly smaller than the QMRS residual norms and, in some cases, tended to remain a little more stable in the final iterations. On the basis of work to date, we have some preference for MRS over QMRS for general use.

Having smoothly decreasing or monotone decreasing residual norms may be of real importance or just a nicety, depending on the application. In fact, this can always be achieved trivially just by saving the best iterate obtained so far; in some of our experiments, MRS and QMRS did not produce significantly smaller residual norms than this simple technique. Thus the good behavior of the MRS and QMRS residual norms alone may not always be enough to justify their use. However, a strongly compelling reason for using MRS or QMRS with methods such as BCG and CGS is their effectiveness in stabilizing the iterates and residuals once the limits of residual reduction have been reached. This is perhaps most clearly seen in Figs. 5 and 7, in which the recursive residuals generated by the underlying method continue to decrease long after they have lost accuracy, while the MRS and QMRS residuals become stable and remain fairly accurate. In addition to helping to avoid misleading results, this stabilizing effect could be useful in obtaining further accuracy from the method. Indeed, one might be

able to detect the onset of stability and restart the method with a fresh, more accurate residual, thereby making further accurate residual reduction possible. We have successfully carried out this strategy in experiments.

We conclude by noting recent related work. In [9] it is shown that iterates produced by certain pairs of "orthogonal error" methods can be related through (1.1), extending Theorem 4.2 of [18, p. 78]. In [1], it is assumed that two sequences $\{x'_k\}$ and $\{x''_k\}$ are given, and an auxiliary sequence is generated by

$$y_k = (1 - \eta_k)x'_k + \eta_k x''_k,$$

where η_k is chosen to minimize the residual at y_k. If a single sequence $\{x_k\}$ is given, then the choice $x'_k = y_{k-1}$, $x''_k = x_k$ gives MRS as a special case. Many other possibilities are explored in [1], and the possibility is raised of combining given $\{x_k^{(1)}\}, \ldots, \{x_k^{(m)}\}$ to produce $\{y_k\}$ by

$$y_k = \eta_k^{(1)} x_k^{(1)} + \cdots + \eta_k^{(m)} x_k^{(m)}, \qquad \sum_{i=1}^{m} \eta_k^{(i)} = 1. \tag{5.1}$$

In [1], it is suggested that the $\eta_k^{(i)}$'s be chosen to minimize the residual at y_k, but more general choices may be useful. For example, the QMR squared method of [8] can be obtained from the CGS iterates and certain auxiliary quantities through a relation of the form (5.1) in which $x_k^{(1)} = y_{k-1}$; see (4.11) of [8, p. 9].

REFERENCES

[1] C. Brezinski and M. Redivo Zaglia, *A hybrid procedure for solving linear systems*, Tech. Rep. ANO-283, Laboratoire d'Analyse Numerérique et d'Optimization, Univ. des Sciences et Technologies de Lille, Sept. 1992.

[2] T. F. Chan, L. de Pillis, and H. van der Vorst, *A transpose-free squared Lanczos algorithm and application to solving non-symmetric linear systems*, Tech. Rep. CAM 91-17, Dept. of Mathematics, Univ. of California at Los Angeles, Sept. 1991.

[3] T. F. Chan, E. Gallopoulos, V. Simoncini, T. Szeto, and C. H. Tong, QMRCGSTAB: *a quasi-minimal residual variant of the* Bi-CGSTAB *algorithm for nonsymmetric systems*, Tech. Rep. CAM 92-26, Dept. of Mathematics, Univ. of California at Los Angeles, June 1992.

[4] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis Dundee 1975, Scotland, Lecture Notes in Math. 506, Springer-Verlag, Berlin, Heidelberg, New York, 1976, pp. 73–89.

[5] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computations*, Series in Automatic Computation, Prentice–Hall, Englewood Cliffs, NJ, 1977.

[6] R. W. Freund, *A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems*, Tech. Rep. 91.18, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA, Sept. 1991; SIAM J. Sci. Comput., 14 (1993), pp. 470–482.

[7] R. W. Freund and N. M. Nachtigal, QMR: *a quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[8] R. W. Freund and T. Szeto, *A quasi-minimal residual squared algorithm for non-Hermitian linear systems*, Tech. Rep. 91.26, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA, Dec. 1991.

[9] M. H. Gutknecht, *Changing the norm in conjugate gradient type algorithms*, Tech. Rep. 91-15, Interdisziplinäres Projektzentrum für Supercomputing, Eidgenössische Technische Hochschule, Zürich, Aug. 1991.

[10] ———, *Variants of* BICGSTAB *for matrices with complex spectrum*, Tech. Rep. 91-14, Interdisziplinäres Projektzentrum für Supercomputing, Eidgenössische Technische Hochschule, Zürich, Aug. 1991.

[11] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Stand., 45 (1950), pp. 255–282.

[12] ———, *Solution of systems of linear equations by minimized iterations*, J. Res. Nat. Bur. Stand., 49 (1952), pp. 33–53.

[13] Y. Saad and M. H. Schultz, GMRES: *a generalized minimal residual method for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[14] W. SCHÖNAUER, *Scientific Computing on Vector Computers*, North–Holland, Amsterdam, New York, Oxford, Tokyo, 1987.

[15] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[16] P. N. SWARZTRAUBER AND R. A. SWEET, *Efficient* FORTRAN *subprograms for the solution of elliptic partial differential equations*, ACM Trans. Math. Software, 5 (1979), pp. 352–364.

[17] H. A. VAN DER VORST, BI-CGSTAB: *a fast and smoothly converging variant of* Bi-CG *for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[18] R. WEISS, *Convergence Behavior of Generalized Conjugate Gradient Methods*, Ph.D. thesis, Univ. of Karlsruhe, Germany, 1990.