ITERATIVE PROJECTION METHODS FOR SPARSE LINEAR SYSTEMS AND EIGENPROBLEMS
CHAPTER 8 : QMR METHODS

Heinrich Voss

voss@tu-harburg.de

Hamburg University of Technology
Institute of Numerical Simulation

TUHH
Technische Universität Hamburg-Harburg

Two-sided Lanczos method

A different approach to the biconjugate gradient method is based on the following two-sided Lanczos or nonsymmetric Lanczos algorithm which was proposed by Lanczos (1950) as a method to transform a general matrix to tridiagonal form.

The two-sided Lanczos process determines basis vectors v^j and w^j of the Krylov spaces $\mathcal{K}_k(v^0, A)$ and $\mathcal{K}_k(w^0, A^T)$, respectively (i.e. for “both sides of the matrix”) such that v^j and w^j are biorthogonal, i.e.

$$(w^j)^T v^k = 0 \quad \text{for } j \neq k.$$

In the following algorithm the coefficients are chosen such that:

$$(w^j)^T v^k = \delta_{jk}.$$

Other “normalizations” are possible and are in use.

Two-sided Lanczos method

Choose v^1, w^1 with $(v^1)^T w^1 \neq 0$; $v^0 = 0$; $w^0 = 0$; $\beta_0 = 0$; $\gamma_0 = 0$

for $k = 1, 2, \dots$ **do**

  $\tilde{v} = Av^k$; $\tilde{w} = A^T w^k$;

  $\alpha_k = \tilde{v}^T w^k$;

  $\tilde{v} = \tilde{v} - \alpha_k v^k - \beta_{k-1} v^{k-1}$

  $\tilde{w} = \tilde{w} - \alpha_k w^k - \gamma_{k-1} w^{k-1}$

  $\gamma_k = \|\tilde{v}\|_2$

  if $\gamma_k == 0$ **then**

    STOP

  end if

  $v^{k+1} = \tilde{v}/\gamma_k$

  $\beta_k = \tilde{w}^T v^{k+1}$

  if $\beta_k == 0$ **then**

    STOP

  end if

  $w^{k+1} = \tilde{w}/\beta_k$

end for

Two-sided Lanczos ct.

With $V_k := (v^1, \dots, v^k)$, $W_k := (w^1, \dots, w^k)$ and

$$H_k = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \dots & 0 & 0 \\ \gamma_1 & \alpha_2 & \beta_2 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ & & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \alpha_{k-1} & \beta_{k-1} \\ 0 & 0 & 0 & \dots & \gamma_{k-1} & \alpha_k \end{pmatrix}$$

the two-sided Lanczos process can be written as

$$AV_k = V_k H_k + \gamma_k v^{k+1} (e^k)^T, \; A^T W_k = W_k H_k^T + \beta_k w^{k+1} (e^k)^T.$$

The biorthogonality obtains the form

$$W_k^T V_k = I_k.$$

Two-sided Lanczos ct.

From

$$\mathcal{K}_k(v^1, A) = \text{span}\{v^1, \dots, v^k\}$$

and

$$\mathcal{K}_k(w^1, A^T) = \text{span}\{w^1, \dots, w^k\}$$

it follows that the k-th approximation of the BiCG method can be determined from the tridiagonal system

$$W_k^T(A(x^0 + V_k y^k) - b) = H_k y^k - W_k^T r^0 = 0,$$

$$x^k = x^0 + V_k y^k.$$

Breakdown

The two-sided Lanczos process can break down for two reasons.

If $\gamma_k = 0$, i.e. $v^{k+1} = 0$, then $\mathcal{K}_k(r^0, A)$ is an invariant subspace of A, and for $v^1 = \beta_0 r^0$

$$AV_k y^k = V_k H_k y^k = r^0 = \beta V_k e^1 \quad \Longleftrightarrow \quad V_k(H_k y^k - \beta_0 e^1) = 0.$$

Hence, the solution of $Ax = b$ has the representation $x^0 + V_k y^k$, and the system is solved exactly. This breakdown is called regular termination or lucky termination.

Secondly the method can break down with $v^{k+1} \neq 0$ and $\beta_k = 0$, i.e. $(v^{k+1})^T \tilde{w} = 0$. This case is called serious breakdown. One does not obtain biorthogonal bases of the Krylov spaces of dimension $k + 1$ or higher from the two-sided Lanczos process (with initial vectors v^1 and w^1).

Two-sided Lanczos ct.

With $r^0 = b - Ax^0 = \beta_0 v^1$ and

$$\tilde{H}_k = \begin{pmatrix} H_k \\ \gamma_k (e^k)^T \end{pmatrix}$$

the residuum of $x := x^0 + V_k y$ is given by

$$\begin{aligned} b - Ax &= b - A(x^0 + V_k y) = r^0 - AV_k y \\ &= \beta_0 v^1 - V_{k+1}\tilde{H}_k y = V_{k+1}(\beta_0 e^1 - \tilde{H}_k y). \end{aligned}$$

The GMRES solution is obtained minimizing

$$\|b - Ax\|_2 = \|V_{k+1}(\beta_0 e^1 - \tilde{H}_k y)\|_2.$$

Observe however, that V_{k+1} does not have orthonormal columns, and therefore this least squares problem is fully populated.

Quasi Minimal Residuum Method

We neglect the matrix V_{k+1}, solve the least square problem

$$\|\beta_0 e^1 - \tilde{H}_k y\|_2 = \min!$$

and set $x^k = x^0 + V_k y^k$.

This can be interpreted as minimizing the residuum with respect to the transformed norm

$$\|W_{k+1}^T r\|_2 = \|W_{k+1}^T V_{k+1}(\beta_0 e^1 - \tilde{H}_k y)\|_2 = \|\beta_0 e^1 - \tilde{H}_k y\|_2 = \min!$$

Hence, to enable a short recurrence the norm is modified in every single step.

QMR method

Similarly as in the GMRES method the least squares problem in the QMR method can be solved using Givens reflections.

Similarly as in MINRES $x^k = x^0 + V_k y^k$ can be updated. We do not have to store all previous search directions, but only two of them.

The residual norms are not monotonely decreasing, however, magnifications usually appear only in very few iterations and are of small magnitude.

QMR with 3 term recurrence

$r = b - Ax$; $\beta = \|r\|$; $v^1 = r/\beta_0$; $w^1 = v^1$; $z_1 = 1$
$\beta_0 = 0$; $\gamma_0 = 0$; $v^0 = 0$; $w^0 = 0$; $p^{-1} = 0$; $p^{-2} = 0$;
for $k = 1, 2, \ldots$ until convergence **do**
  $\tilde{v} = Av^k$; $\tilde{w} = A^T w^k$
  $\alpha_k = \tilde{v}^T w^k$
  $\tilde{v} = \tilde{v} - \alpha_k v^k - \beta_{k-1} v^{k-1}$;
  $\tilde{w} = \tilde{w} - \alpha_k w^k - \gamma_{k-1} w^{k-1}$
  $\gamma_k = \|\tilde{v}\|_2$; $v^{k+1} = \tilde{v}/\gamma_k$
  $\beta_k = \tilde{w}^T v^{k+1}$; $w^{k+1} = \tilde{w}/\beta_k$
  if $k > 2$ **then**

$$\begin{pmatrix} \rho \\ \beta_{k-1} \end{pmatrix} = \begin{pmatrix} c_{k-2} & s_{k-2} \\ s_{k-2} & -c_{k-2} \end{pmatrix} \begin{pmatrix} 0 \\ \beta_{k-1} \end{pmatrix};$$

  end if
  if $k > 1$ **then**

$$\begin{pmatrix} \beta_{k-1} \\ \alpha_k \end{pmatrix} = \begin{pmatrix} c_{k-1} & s_{k-1} \\ s_{k-1} & -c_{k-1} \end{pmatrix} \begin{pmatrix} \beta_{k-1} \\ \alpha_k \end{pmatrix};$$

  end if

QMR algorithm ct.

$$\eta = \sqrt{\alpha_k^2 + \gamma_k^2}$$
$$s_k = \gamma_k/\eta; \quad c_k = \alpha_k/\eta$$
$$\alpha_k = \eta$$
$$z_{k+1} = s_k z_k; \quad z_k = c_k z_k$$
$$p^{k-1} = (v^k - \beta_{k-1} p^{k-2} - \rho p^{k-3})/\alpha_k$$
$$x = x + \beta z_k p^{k-1}$$

end for

Cost of QMR

2 matrix-vector products
3 scalar products
9 _axpy

Storage requirements: 10 vectors

QMR ct.

$\|r^k\|_2$ is not determined in the QMR method.

However, it holds

$$b - Ax = V_{k+1}(\beta e^1 - \tilde{H}_k y).$$

If the two-sided Lanczos process is normalized such that $\|v^k\|_2 = 1$ for every k, then it follows

$$\|r^k\|_2 \leq \|V_{k+1}\|_S \|\beta_0 e^1 - \tilde{H}_k y\|_2 \leq \sqrt{m+1} \|\beta_0 e^1 - \tilde{H}_k y\|_2,$$

and the termination can be based on the moderate overestimation of $\|r^k\|_2$ on the right.

QMR & GMRES

The norm of the QMR residual can be related to that of the optimal GMRES residual as follows:

Theorem 8.1
If r_G^k and r_Q^k denote the GMRES and QMR residual, respectively, then it holds

$$\|r_Q^k\|_2 \le \kappa(V_{k+1})\|r_G^k\|_2,$$

where V_{k+1} is the matrix of basis vectors for $\mathcal{K}_{k+1}(r^0, A)$ constructed by the two-sided Lanczos method and $\kappa(V_{k+1})$ denotes its condition number.

Proof: For the GMRES residual it holds that

$$\|r_G^k\|_2 = \min_y \|V_{k+1}(\beta e^1 - \tilde{H}_k y)\| \ge \sigma_{\min}(V_{k+1}) \min_y \|\beta e^1 - \tilde{H}_k y\|_2,$$

where $\sigma_{\min}(V_{k+1})$ is the smallest singular value of V_{k+1}. Combining this with

$$\|r_Q^k\|_2 \le \|V_{k+1}\|_2\|\beta e^1 - \tilde{H}_k y\|_2 \quad \text{for every } y \in \mathbb{R}^k$$

gives the desired result.

QMR & BiCG

Following the work of Cullum & Greenbaum (1996) we establish a relationship between the residual norms in the BiCG and the QMR method.

To this end we first consider the relationship between upper Hessenberg linear systems and least squares problems. Let H_k, $k = 1, 2, \dots$ be a family of upper Hessenberg matrices where H_{k-1} is the $(k-1)-by-(k-1)$ principal submatrix of H_k, and let

$$\tilde{H}_k = \begin{pmatrix} H_k \\ h_{k+1,k}(e^k)^T \end{pmatrix}.$$

The matrix $\tilde{H}_k$ can be factored in the form $\tilde{H}_k = F^T R$, where $F \in \mathbb{R}^{k+1 \times k+1}$ is an orthogonal matrix and $R \in \mathbb{R}^{k+1 \times k}$ in an upper triangular matrix.

QMR & BiCG ct.

The factorization $\tilde{H}_k = F^T R$ can be performed using plane rotations in the manner described for the GMRES algorithm:

$$(F_k \dots F_1)\tilde{H}_k = R, \quad \text{where } F_j = \begin{pmatrix} I_{j-1} & & & \\ & c_j & -s_j & \\ & s_j & c_j & \\ & & & I_{k-j} \end{pmatrix}.$$

Note that the first $k-1$ sines and cosines s_j, c_j, $j = 1, \dots, k-1$ are those used in the factorization of $\tilde{H}_{k-1}$.

Let $\beta > 0$ be given, and assume that H_k is nonsingular. Let $\tilde{y}^k$ be the solution of the linear system $H_k y = \beta e^1$, let y^k denote the solution of the least squares problem $\|\tilde{H}_k y - \beta e^1\|_2 = \min$, and let

$$\tilde{v}^k = \tilde{H}_k \tilde{y}^k - \beta e^1 \quad \text{and} \quad v^k = \tilde{H}_k y^k - \beta e^1.$$

Lemma 8.2

$$\|v^k\| = \beta|s_1 s_2 \dots s_k| \quad \text{and} \quad \|\tilde{v}^k\| = \beta\frac{1}{|c_k|}|s_1 \dots s_k| \qquad (1)$$

$$\|\tilde{v}^k\| = \frac{\|v^k\|}{\sqrt{1-(\|v^k\|/\|v^{k-1}\|)^2}} \qquad (2).$$

Proof:The least squares problem with $\tilde{H}_k$ can be written in the form

$$\min_y \|\tilde{H}_k y - \beta e^1\| = \min_y \|F(\tilde{H}_k y - \beta y^1)\| = \min_y \|Ry - \beta F e^1\|,$$

and the solution y^k is determined by solving the triangular system corresponding to the leading m components of $Ry - \beta F e^1$.

The remainder is therefore zero except for the last component, which is just the last entry of

$$-\beta F e^1 = -\beta F_k \dots F_1 e^1 \quad \text{i.e.} \ -\beta s_1 \dots s_k.$$

This establishes the first equality in (1).

Proof ct.

For $\tilde{y}^k = \beta H_k^{-1} e^1$ we have

$$\tilde{v}^k = \beta \tilde{H}_k H_k^{-1} e^1 - \beta e^1$$

which is zero except for the last entry, and this one is $\beta h_{k+1,k}$ times the $(m, 1)$ entry of H_k^{-1}.

H_k can be factored in the form $\tilde{F}^T \tilde{R}$, where $\tilde{F} = \tilde{F}_{k-1} \dots \tilde{F}_1$, and $\tilde{F}_j$ is the j-by-j principle submatrix of F_j.

The matrix $\tilde{H}_k$ after applying the first $k - 1$ plane rotations has the form

$$(F_{k-1} \dots F_1) \tilde{H}_k = \begin{pmatrix} * & * & \dots & * \\ & * & \dots & * \\ & & \ddots & \vdots \\ & & & r \\ & & & h \end{pmatrix}$$

where r is the (k, k)-entry of $\tilde{R}$ and $h = h_{k+1,k}$.

Proof ct.

The k-th rotation is chosen such to annihilate the nonzero entry in the last row:

$$c_k = \frac{r}{\sqrt{r^2 + h^2}}, \; s_k = -\frac{h}{\sqrt{r^2 + h^2}}.$$

Since $H_k^{-1} = \tilde{R}^{-1}\tilde{F}$, the $(k, 1)$-entry of this is $\frac{1}{r}$ times the $(k, 1)$-entry of $\tilde{F} = \tilde{F}_{k-1} \dots \tilde{F}_1$, and this is just $s_1 \dots s_{k-1}$.

It follows that the nonzero entry of $\tilde{v}^k$ is $\beta \frac{h_{k+1,k}}{r} s_1 \dots s_{k-1}$. Finally, using the fact that $|s_k/c_k| = |h/r| = |h_{k+1,k}/r|$ we obtain the second equality in (1).

From equation (1) it is clear that

$$\frac{\|v^k\|}{\|v^{k-1}\|} = |s_k| \quad \text{and} \quad \frac{\|\tilde{v}^k\|}{\|v^k\|} = \frac{1}{|c_k|}.$$

The result (2) follows upon replacing $|c_k|$ by $\sqrt{1 - |s_k|^2}$.

Quasi-residual

An immediate consequence of Lemma 8.2 is the following relationship between the BiCG residual and the quantity

$$z_Q^k = \beta e^1 - \tilde{H}_k y_Q^k,$$

which is related to the residual r_Q^k of the QMR algorithm:

$$r_Q^k = V_{k+1} z_Q^k, \quad \|r_Q^k\| \leq \sqrt{m+1}\|z_Q^k\|.$$

We will refer to z_Q^k as the QMR quasi-residual, the vector whose norm is actually minimized in the QMR algorithm.

Theorem 8.3

Assume that the Lanczos vectors at step 1 through m are defined and that the triangular matrix generated by the Lanczos algorithm at step m is nonsingular.

Then The BiCG residual r_B^k and the QMR quasi-residual z_Q^k are related by

$$\|r_B^k\| \le \frac{\|z_Q^k\|}{\sqrt{1 - (\|z_Q^k\|/\|z_Q^{k-1}\|)^2}}.$$

Proof: From

$$AV_k = V_{k+1}\tilde{H}_k, \quad x^k = x^0 + V_k y^k \quad \text{and} \quad H_k y^k = \beta e^1$$

it follows that the BiCG residual can be written as

$$r_B^k = r^0 - AV_k y_B^k \; = \; r^0 - V_{k+1}\tilde{H}_k y_B^k = V_{k+1}(\beta e^1 - \beta \tilde{H}_k H_k^{-1} e^1).$$

The quantity in parentheses has only one nonzero entry (in its last component), and from $\|v^{k-1}\| = 1$ we get

$$\|r_B^k\| = \|\beta e^1 - \beta \tilde{H}_k H_k^{-1} e^1\|.$$

The desired result follows from Lemma 8.2 and the definition of z_Q^k.

Discussion

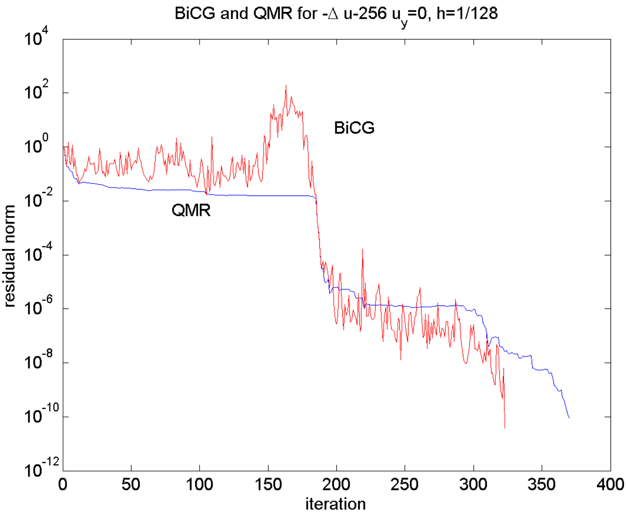
In most cases, the quasi-residual norms and the actual residual norms in the QMR algorithm are of the same order of magnitude:

$$\sigma_{\min}(V_{k+1})\|z_Q^k\| \le \|r_Q^k\| \le \sqrt{m+1}\|z_Q^k\|.$$

Theorem 8.3 shows that if the QMR quasi-residual norm is reduced by a significant factor at step m, then the BiCG residual norm will be approximately equal to the QMR quasi-residual norm at step m, since the denominator in the right hand side will be close to 1.
If the QMR quasi-residual norm remains almost constant, then the denominator will be close to 0, and the BiCG residual norm will be much larger.

Note that the graph of the QMR quasi-residual norm must be very flat before the BiCG residual norm is orders-of-magnitude larger. Peaks in the BiCG residual norm always correspond to plateaus of the QMR quasi-residual norm.

Example



BiCG and QMR for -Δ u-256 u_y=0, h=1/128
residual norm
iteration
BiCG
QMR
10^4
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
0
50
100
150
200
250
300
350
400

Preconditioning

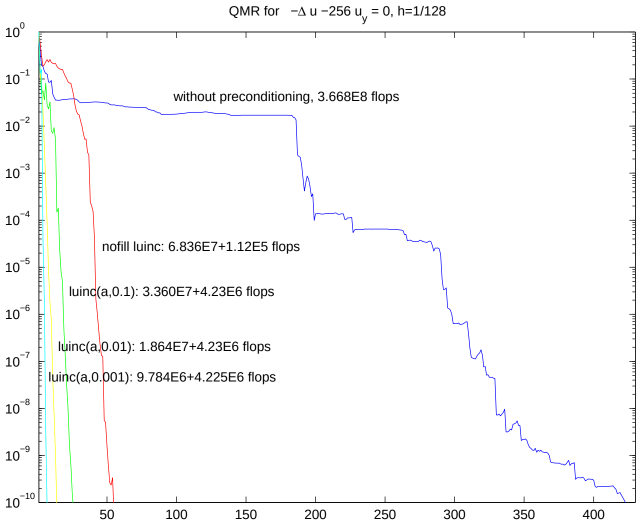
QMR can be preconditioned as before solving the linear system

$$M^{-1}Ax = M^{-1}b \quad \text{or} \quad AM^{-1}y = b,\ x = M^{-1}y,$$

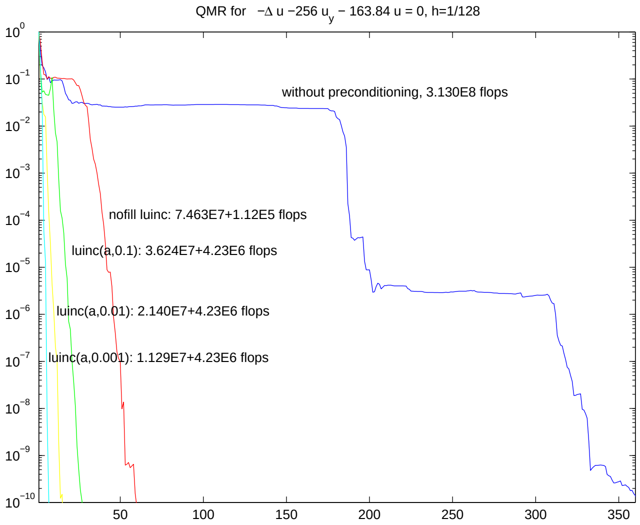
where $M \approx A$ and linear systems with coefficient matrix M can be solved inexpensively.

The following examples demonstrate the efficiency of left preconditioning.

Example

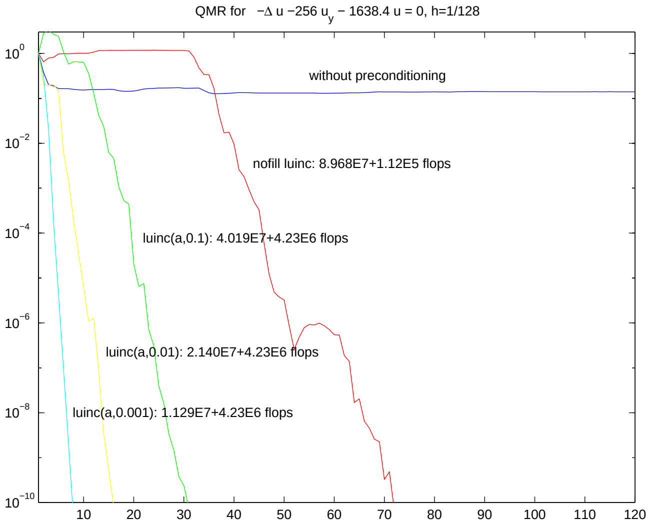

QMR for −Δ u −256 u_y = 0, h=1/128
without preconditioning, 3.668E8 flops
nofill luinc: 6.836E7+1.12E5 flops
luinc(a,0.1): 3.360E7+4.23E6 flops
luinc(a,0.01): 1.864E7+4.23E6 flops
luinc(a,0.001): 9.784E6+4.225E6 flops
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
10^−7
10^−8
10^−9
10^−10
50
100
150
200
250
300
350
400

Example



QMR for −Δ u −256 u_y − 163.84 u = 0, h=1/128
without preconditioning, 3.130E8 flops
nofill luinc: 7.463E7+1.12E5 flops
luinc(a,0.1): 3.624E7+4.23E6 flops
luinc(a,0.01): 2.140E7+4.23E6 flops
luinc(a,0.001): 1.129E7+4.23E6 flops
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
10^−7
10^−8
10^−9
10^−10
50
100
150
200
250
300
350

Example


QMR for −Δ u −256 u_y − 1638.4 u = 0, h=1/128
without preconditioning
nofill luinc: 8.968E7+1.12E5 flops
luinc(a,0.1): 4.019E7+4.23E6 flops
luinc(a,0.01): 2.140E7+4.23E6 flops
luinc(a,0.001): 1.129E7+4.23E6 flops
10^0
10^−2
10^−4
10^−6
10^−8
10^−10
10
20
30
40
50
60
70
80
90
100
110
120

Two-term recurrence

In addition to the two-sided Lanczos process with 3–term recurrence a coupled 2–term Lanczos process was introduced by Freund & Nachtigal (1994) to determine biorthogonal bases $\{v^j\}$ and $\{w^j\}$. In this case additional bases $\{p^j\}$ of $\mathcal{K}_k(v^1, A)$ and $\{q^j\}$ of $\mathcal{K}_k(w^1, A^T)$ are determined.

Let $P_k = [p^1, \dots, p^k]$, $Q_k = [q^1, \dots, q^k]$. We want to recover from these matrices the Lanczos bases v^j, w^j, $j = 1, \dots, k$. That is, we have to find $R_k, \tilde{R}_k \in \mathbb{R}^{k \times k}$ und $L, \tilde{L} \in \mathbb{R}^{k+1 \times k}$ such that

$$V_k = P_k R_k \quad \text{and} \quad W_k = Q_k \tilde{R}_k$$
$$AP_k = V_{k+1} L_k \quad \text{and} \quad A^T Q_k = Q_{k+1} \tilde{L}_k$$

From $\text{span}\{v^1, \dots, v^j\} = \text{span}\{p^1, \dots, p^j\} = \mathcal{K}_j(v^1, A)$ for $j = 1, \dots, k$ it follows, that R_k (for analogous reasons $\tilde{R}_k$) is an upper triangular matrix, and

$$Ap^j = A\Big(\sum_{i=1}^{j} r_{ij}^{(-1)} v^i\Big) = \sum_{i=1}^{j} r_{ij}^{(-1)} Av^i = \sum_{i=1}^{j+1} \rho_{ij} v^i$$

demonstrates that L (and correspondingly $\tilde{L}$) is an upper Hessenberg matrix.

Two-term recurrence ct.

Let ℓ^k be the last column of L. From $W_k^T V_k = I_k$ we get

$$Ap^k = V_{k+1}\ell^k \quad \text{and} \quad V_k W_k^T A p^k = V_k W_k^T V_{k+1}\ell^k = V_k[I_k, 0]\ell^k,$$

and assuming $\ell_{k+1,k} = 1$ which is just a scaling of p^k it follows that

$$v^{k+1} = Ap^k - V_k W_k^T A p^k = Ap^k - V_k \tilde{R}_k^T Q_k^T A p^k,$$

and analogously

$$w^{k+1} = A^T q^k - W_k V_k^T A^T q_k = A^T q^k - W_k R_k^T P_k^T A^T q^k.$$

These two equations demonstrate, that $Q_k^T A p^k$ determines the length of the recurrences for the Lanczos vectors v^k and w^k.

To obtain a short recurrence for v^{k+1} and w^{k+1} we require A-orthogonality of P_k and Q_k:

$$Q_k^T A P_k = D_k = \text{diag}\{d_1, \ldots, d_k\}.$$

Two-term recurrence ct.

From

$$v^k = \sum_{j=1}^{k} r_{jk} p^j \quad \Rightarrow \quad p^k = \frac{1}{r_{kk}} \Big(v^k - \sum_{j=1}^{k-1} r_{jk} p^j \Big)$$

it follows for $i = 1, \dots, k-2$

$$\begin{aligned} 0 &= (q^i)^T A p^k = \frac{1}{r_{kk}} q^i A \Big(v^k - \sum_{j=1}^{k-1} r_{jk} p^j \Big) \\ &= \frac{1}{r_{kk}} (q^i)^T A v^k - \frac{1}{r_{kk}} \sum_{j=1}^{k-1} r_{jk} (q^i)^T A p^k = \frac{1}{r_{kk}} (A q^i)^T v^k - \frac{r_{ik}}{r_{kk}} d_k. \end{aligned}$$

$Aq^i \in \mathcal{K}_{i+1}(v^1, A)$ implies $(Aq^i)^T v^k = 0$, from which we obtain $r_{ik} = 0$ for $i = 1, \dots, k-2$.

Hence, R (and analogously $\tilde{R}$) is a bidiagonal upper triangular matrix.

Two-term recurrence ct.

$p^k = v^k + \alpha p^{k-1}$, and multiplying by $(q^{k-1})^T A$ from the left yields

$$0 = (q^{k-1})^T A p^k = (q^{k-1})^T A v^k + \alpha (q^{k-1})^T A p^{k-1}$$

i.e.

$$p^k = v^k - p^{k-1} \frac{(q^{k-1})^T A v^k}{(q^{k-1})^T A p^{k-1}}.$$

Analogously, we obtain

$$q^k = w^k - q^{k-1} \frac{(p^{k-1})^T A^T w^k}{(q^{k-1})^T A p^{k-1}}.$$

Two-term recurrence ct.

From $Q_k A P_k = D_k$ and

$$v^{k+1} = Ap^k - V_k \tilde{R}_k^T Q_k^T A p^k,$$

it follows $v^{k+1} = Ap^k + \alpha v^k$ for some $\alpha \in \mathbb{R}$, and $(w^k)^T v^{k+1} = 0$ yields

$$v^{k+1} = Ap^k - v^k \frac{(w^k)^T A p^k}{(w^k)^T v_k},$$

and similarly

$$w^{k+1} = A^T q^k - w^k \frac{(w^k)^T A^T q^k}{(w^k)^T v^k}.$$

Hence, L_k and $\tilde{L}_k$ are lower bidiagonal $(k+1) \times k$ Hessenberg matrices.

Two-term recurrence ct.

In conclusion, we obtain the following coupled two-term recurrence

$$\begin{aligned}
p^k &= v^k - p^{k-1}\frac{(q^{k-1})^T A v^k}{(q^{k-1})^T A p^{k-1}} \\
q^k &= w^k - q^{k-1}\frac{(p^{k-1})^T A^T w^k}{(q^{k-1})^T A p^{k-1}} \\
v^{k+1} &= Ap^k - v^k\frac{(w^k)^T A p^k}{(w^k)^T v_k} \\
w^{k+1} &= A^T q^k - w^k\frac{(w^k)^T A^T q^k}{(w^k)^T v^k}.
\end{aligned}$$

$x^k = x^0 + P_k y^k$, $y^k \in \mathbb{R}^k$ implies

$$r^k = r^0 - AP_k y^k = r^0 - V_{k+1}L_k y^k = V_{k+1}(e^1 - L_k y^k),$$

and the QMR method can be performed by

$$\|e^1 - L_k y^k\| = \min!, \quad x^k = x^0 + P_k y^k.$$

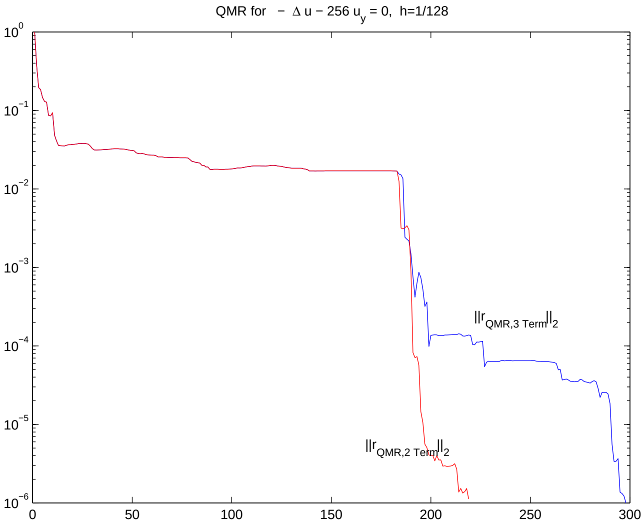
Properties of 2-term QMR

Work and storage requirements are low and comparable to the original implementation.

Direction vectors used to update x^k also have a two-term recurrence.

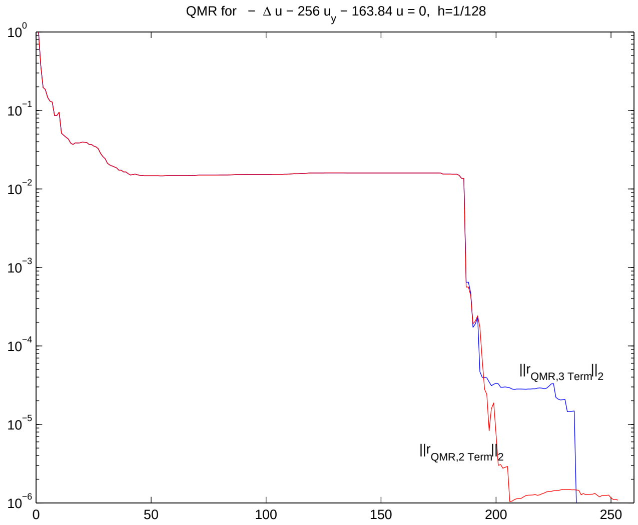
Numerical examples indicate that the numerical properties are better. Typically, $\kappa(L_k) < \kappa(H_k)$, and sometimes $\kappa(L_k) \ll \kappa(H_k)$

Example

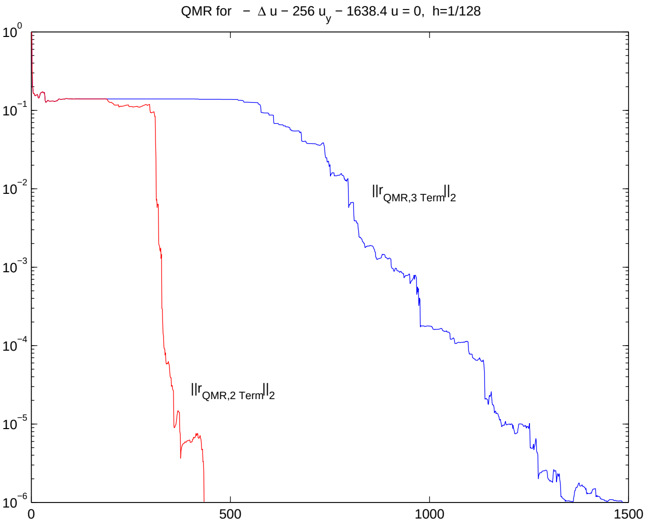


QMR for − Δ u − 256 u_y = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
50
100
150
200
250
300
||r_QMR,3 Term||_2
||r_QMR,2 Term||_2

Example


QMR for − Δ u − 256 u_y − 163.84 u = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
50
100
150
200
250
||r_QMR,3 Term||_2
||r_QMR,2 Term||_2

Example



QMR for − Δ u − 256 u_y − 1638.4 u = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
500
1000
1500
||r_QMR,3 Term||_2
||r_QMR,2 Term||_2

Look–ahead

The look-ahead Lanczos process prevents possible break downs of the classical Lanczos method by relaxing the biorthogonality condition.
Look–ahead Lanczos methods were introduced by Parlett, Taylor & Liu (1985) solving eigenvalue problems numerically; in connection with QMR methods for sparse linear systems they were studied by Freund, Gutknecht & Nachtigal (1993,1994).

It fills the gap between regular vectors v^{k_j} and $v^{k_{j+1}}$ (and between w^{k_j} and $w^{k_{j+1}}$) such that $k_{j+1} > k_j + 1$ adding vectors

$$v^k \in \mathcal{K}_k(v^1, A) \setminus \mathcal{K}_{k-1}(v^1, A), \; w^k \in \mathcal{K}_k(w^1, A^T) \setminus \mathcal{K}_{k-1}(w^1, A^T),$$

$k = k_j, \dots, k_{j+1} - 1,$.

These inner vectors can be chosen such that the vectors v^j and w^k from different blocks are biorthogonal, where the vectors v^i, $i = k_j, \dots, k_{j+1} - 1$, are referred to as the j-th block.

Two-sided Lanczos method

Choose v^1, w^1 with $(v^1)^T w^1 \neq 0$; $v^0 = 0$; $w^0 = 0$; $\beta_0 = 0$; $\gamma_0 = 0$
for $k = 1, 2, \dots$ **do**
  $\tilde{v} = Av^k$; $\tilde{w} = A^T w^k$;
  $\alpha_k = \tilde{v}^T w^k$;
  $\tilde{v} = \tilde{v} - \alpha_k v^k - \beta_{k-1} v^{k-1}$
  $\tilde{w} = \tilde{w} - \alpha_k w^k - \gamma_{k-1} w^{k-1}$
  $\gamma_k = \|\tilde{v}\|_2$
  if $\gamma_k == 0$ **then**
    STOP
  end if
  $v^{k+1} = \tilde{v}/\gamma_k$
  $\beta_k = \tilde{w}^T v^{k+1}$
  if $\beta_k == 0$ **then**
    STOP
  end if
  $w^{k+1} = \tilde{w}/\beta_k$
end for

Look–ahead ct.

Basic idea:
Compute pairs of inner vectors

$$v^j \quad \text{and} \quad w^j, \quad j = k+1, \dots, k'$$

that satisfy a relaxed biorthogonality condition

$$(w^i)^T v^j = (w^j)^T v^i = 0, \quad i = 1, \dots, k-1$$

until it is safe to compute a next pair of regular vectors $v^{k'+1}$ and $w^{k'+1}$.

The look–ahead Lanczos method generates blocks of vectors of length $k' + 1 - k$

$$V^\ell = [v^k \; v^{k+1} \; \dots \; v^{k'}], \; W^\ell = [w^k \; w^{k+1} \; \dots \; w^{k'}]$$

($k' = k$ for the standard Lanczos algorithm) such that

$$[v^1 \; v^2 \; \dots \; v^k] = [V^{(1)} \; V^{(2)} \; \dots V^{(\ell)}], \quad [w^1 \; w^2 \; \dots \; w^k] = [W^{(1)} \; W^{(2)} \; \dots W^{(\ell)}].$$

The biorthogonality requirement for the standard Lanczos method is replaced by the block biorthogonality

$$(W^{(j)})^T V^{(\ell)} = (W^{(\ell)})^T V^{(j)} = 0, \; j = 1, \dots, \ell - 1,$$

and $H_k = W_k^T A V_k$ becomes block-tridiagonal.

Look ahead Lanczos method

Choose v^1, w^1 with $\|v^1\|_2 = 1$; $\|w^1\|_2 = 1$
$\tilde{V}_1 = v^1$; $\tilde{W}_1 = w^1$; $\tilde{D}_1 := \tilde{W}_1^T \tilde{V}_1$
$\tilde{V}_0 = \emptyset$; $\tilde{W}_0 = \emptyset$; $v^0 = 0$; $w^0 = 0$
$k_1 = 1$; $p = 1$; $\rho_1 = 1$; $\xi_1 = 1$
for $k = 1, 2, \dots$ **do**
  if det $\tilde{D}_p \neq 0$ **then**
    $\tilde{v}^{k+1} = Av^k - \tilde{V}_p \tilde{D}_p^{-1} \tilde{W}_p^T A v^k - \tilde{V}_{p-1} \tilde{D}_{p-1}^{-1} \tilde{W}_{p-1}^T A v^k$
    $\tilde{w}^{k+1} = A^T w^k - \tilde{W}_p \tilde{D}_p^{-1} \tilde{V}_p^T A^T w^k - \tilde{W}_{p-1} \tilde{D}_{p-1}^{-1} \tilde{V}_{p-1}^T A^T w^k$
    $k_{p+1} = k + 1$; $p = p + 1$; $\tilde{V}_p = \emptyset$; $\tilde{W}_p = \emptyset$
  else
    $\tilde{v}^{k+1} = Av^k - \zeta_k v^k - (\eta_k/\rho_k) v^{k-1} - \tilde{V}_{p-1} \tilde{D}_{p-1}^{-1} \tilde{W}_{p-1}^T A v^k$
    $\tilde{w}^{k+1} = A^T w^k - \zeta_k w^k - (\eta_k/\rho_k) w^{k-1} - \tilde{W}_{p-1} \tilde{D}_{p-1}^{-1} \tilde{V}_{p-1}^T A^T w^k$
  end if

Look ahead Lanczos ct.

$\rho_{k+1} = \|\tilde{v}^{k+1}\|_2$
$\xi_{k+1} = \|\tilde{w}^{k+1}\|_2$
if $\rho_{k+1}\xi_{k+1} == 0$ **then**
  STOP
end if
$v^{k+1} = \tilde{v}^{k+1}/\rho_{k+1}$
$w^{k+1} = \tilde{w}^{k+1}/\xi_{k+1}$
$\tilde{V}_p = (\tilde{V}_p, v^{k+1})$
$\tilde{W}_p = (\tilde{W}_p, w^{k+1})$
$\tilde{D}_p = \tilde{W}_p^T \tilde{V}_p$
end for

Remarks

1. ζ_k and η_k are coefficients which (with the exception of $\eta_{k_j} = 0$) can be chosen arbitrarily. Since the blocks, which are built in the look-ahead Lanczos method, are usually small (Freund, Gutknecht & Nachtigal report that the maximum size that they observed in their numerical examples was 4) the choice of ζ_k and η_k is not of great influence on the total iteration.

2. In the algorithm above inner vectors are only determined if a serious breakdown would appear in exact arithmetic. In practice a look-ahead Lanczos algorithm must also be able to handle near break-downs, i.e. $((w^k)^T v^k \approx 0$, but $v^k \not\approx 0$, $w^k \not\approx 0)$.
To check the regularity of $\tilde{D}_p$ in QMRPACK the minimal singular value of $\tilde{D}_p$ is monitored.

3. If only regular steps are performed then the look-ahead Lanczos process reduces to the two-sided Lanczos method.

Two-term recurrence

For the coupled 2-term recurrences breakdown is possible not only since $(w^k)^T v^k \approx 0$, but also since $d_k = (q^k)^T A p^k \approx 0$.

The remedy: Look-ahead in both sequences

There exists (public domain) FORTRAN implementation of Freund & Nachtigal (1993).

TFQMR

There exists a transpose free QMR method (Freund (1993)).

The name is a little misleading. Actually, TFQMR is not a transpose free variant of the QMR method, but a QMR variant of the transpose free CGS method.

TFQMR ct.

The CGS method is given by

$r^0 = b - Ax^0;\ d^1 = r^0;\ u^0 = r^0$
Choose $\tilde{r}^0$ with $\alpha_0 = (\tilde{r}^0)^T r^0 \neq 0$
for $k = 1, 2, \dots$ until convergence **do**
    $s^k = Ad^k$
    $\gamma_k = (\tilde{r}^0)^T s^k$
    $\tau_k = \alpha_{k-1}/\gamma_k$
    $q^k = u^{k-1} - \tau_k s^k$
    $w^k = u^{k-1} + q^k$
    $x^k = x^{k-1} + \tau_k w^k$
    $r^k = r^{k-1} - \tau_k A w^k$
    $\beta_k = 1/\alpha_{k-1};\quad \alpha_k = (\tilde{r}^0)^T r^k;\quad \beta_k = \alpha_k \beta_k$
    $u^k = r^k + \beta_k q^k$
    $d^{k+1} = u^k + \beta_k (q^k + \beta_k d^k)$
end for

TFQMR ct.

The CGS algorithm can be interpreted in the following way: x^k is updated in two steps, $x^{k+1/2} = x^k + \tau_k u^k$ and $x^{k+1} = x^{k+1/2} + \tau_k q^k$ corresponding to the increase of the degree of the residual polynomial in each step by 2. To avoid indices which are multiples of $1/2$ we double the indices in the CGS method. Then the statements in the for-loop obtain the following form:

1: $\tau_{2k} = (\tilde{r}^0)^T r^{2k} / (\tilde{r}^0)^T A d^{2k}$
2: $q^{2k} = u^{2k} - \tau_{2k} A d^{2k}$
3: $x^{2k+2} = x^{2k} + \tau_{2k}(u^{2k} + q^{2k})$
4: $r^{2k+2} = r^{2k} - \tau_{2k} A(u^{2k} + q^{2k})$
5: $\beta_{2k} = (\tilde{r}^0)^T r^{2k+2} / (\tilde{r}^0)^T r^{2k}$
6: $u^{2k+2} = r^{2k+2} + \beta_{2k} q^{2k}$;
7: $d^{2k+2} = u^{2k+2} + \beta_{2k}(q^{2k} + \beta_{2k} d^{2k})$

TFQMR ct.

x^{2k+2} is updated in two steps:

$$\begin{aligned} x^{2k+1} &= x^{2k} + \tau_{2k} u^{2k}; \\ x^{2k+2} &= x^{2k+1} + \tau_{2k} q^{2k}; \end{aligned}$$

Putting $u^{2k+1} := q^{2k}$ and $\tau_{2k+1} := \tau_{2k}$ both statements read

$$x^k = x^{k-1} + \tau_{k-1} u^{k-1}.$$

For even m this is exactly the approximation from the CGS method, for odd m these approximations of the solution do not appear in the CGS method.

TFQMR ct.

With the matrix

$$U_k = (u^0, \dots, u^{k-1})$$

and the vector

$$z^k = (\tau_0, \tau_1, \dots, \tau_{k-1})^T$$

the approximation x^k can be written as

$$x^k = x^0 + U_k z^k = x^{k-1} + \tau_{k-1} u^{k-1},$$

and the residuum is

$$r^k = r^0 - AU_k z^k = r^{k-1} - \tau_{k-1} A u^{k-1}.$$

The last equation yields

$$Au^k = \frac{1}{\tau_k}(r^k - r^{k+1}).$$

TFQMR ct.

Hence,

$$AU_k = R_{k+1}\tilde{B}_k$$

with

$$R_{k+1} = (r^0, r^1, \dots, r^k) \quad \text{and}$$

$$\tilde{B}_k = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -1 & 1 & \dots & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & & & -1 & 1 \\ 0 & 0 & \dots & \dots & -1 \end{pmatrix} \text{diag}\left\{\frac{1}{\tau_0}, \frac{1}{\tau_1}, \dots, \frac{1}{\tau_{k-1}}\right\},$$

and the residuum is

$$r^k = r^0 - AU_k z_k = R_{k+1}(e^1 - \tilde{B}_k z^k).$$

If the columns of R_{k+1} are scaled by the diagonal matrix $\Delta_{k+1} = \mathrm{diag}\{\delta_0, \dots, \delta_k\}$, then one obtains

$$r^k = R_{k+1}\Delta_{k+1}^{-1}\Delta_{k+1}(e^1 - \tilde{B}_k z^k) = R_{k+1}\Delta_{k-1}^{-1}(\delta_0 e^1 - \tilde{H}_k)$$

with $\tilde{H}_k := \Delta_{k+1}\tilde{B}_k$.

It is easily seen that the approximations x^{2k} of the CGS method are given by

$$x^{2k} = x^0 + U_{2k}H_{2k}^{-1}(\delta_0 e^1),$$

where H_{2k} is obtained from $\tilde{H}_{2k}$ by deleting the last row.

TFQMR ct.

Solving (similarly as in the transition from the BiCG to the GMRES method) the least squares problem

$$\|\delta e^1 - \tilde{H}_k z^k\| = \min!,$$

then one obtains the approximation

$$x^k = x^0 + U_k z^k$$

of the TFQMR method.

To implement this method we still have to deduce how to update x^k for otherwise all previous vectors u^k would have to be stored. This can be found in
Saad: Iterative Methods for Sparse Linear Systems.

TFQMR algorithm

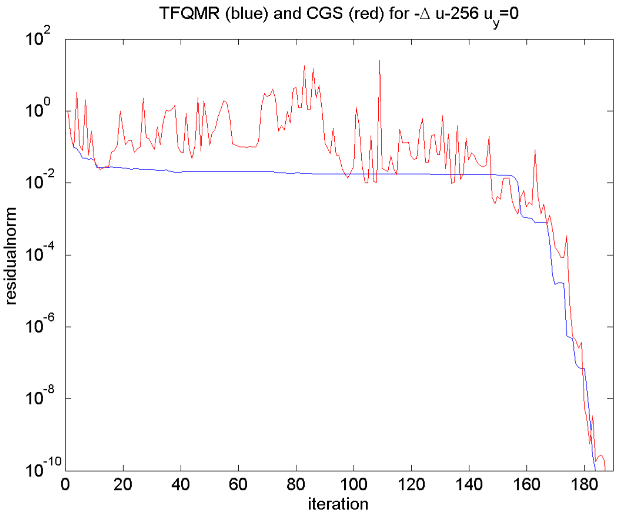
$r = b - Ax$; $\tilde{r} = r$; $w = r$; $u = r$; $v = Au$; $d = 0$
$\tau = \|r\|_2$; $\theta = 0$; $\eta = 0$; $\rho = \tilde{r}^T r$; $z^0 = Au$
for $k = 1, 2, \dots$ **do**
  $\alpha_1 = \rho/(v^T \tilde{r})$; $\alpha_0 = \alpha_1$
  $u^0 = u$; $u = u - \alpha_0 v$
  $w = w - \alpha_0 z^0$
  $d = u^0 + \theta^2 \eta d/\alpha_0$
  $\theta = \|w\|_2/\tau$; $c = 1/\sqrt{1+\theta^2}$; $\tau = \tau\theta c$; $\eta = c^2\alpha_0$
  $x = x + \eta d$
  $z^1 = Au$
  $w = w - \alpha_1 z^1$
  $d = u + \theta^2 \eta d/\alpha_1$
  $\theta = \|w\|_2/\tau$; $c = 1/\sqrt{1+\theta^2}$; $\tau = \tau\theta c$; $\eta = c^2\alpha_1$
  $x = x + \eta d$
  $\beta = 1/\rho$; $\rho = \tilde{r}^T w$; $\beta = \beta\rho$
  $u = w + \beta u$
  $z^0 = Au$
  $v = z^0 + \beta(z^1 + \beta v)$
end for

Cost of TFQMR

2 matrix-vector products
4 scalar products
10 _axpy

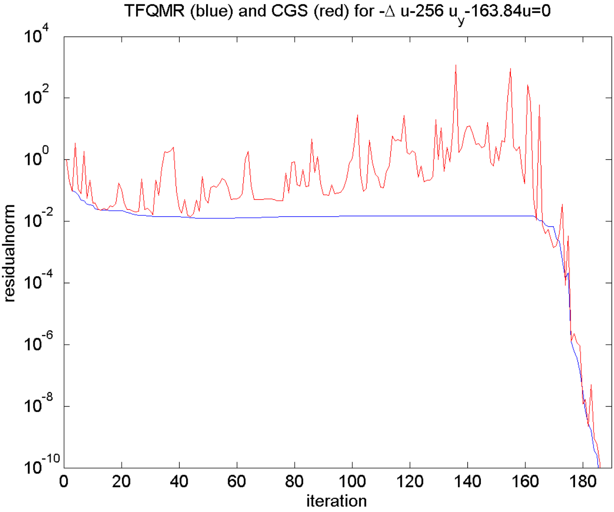
Storage requirements: 10 Vectors

Example

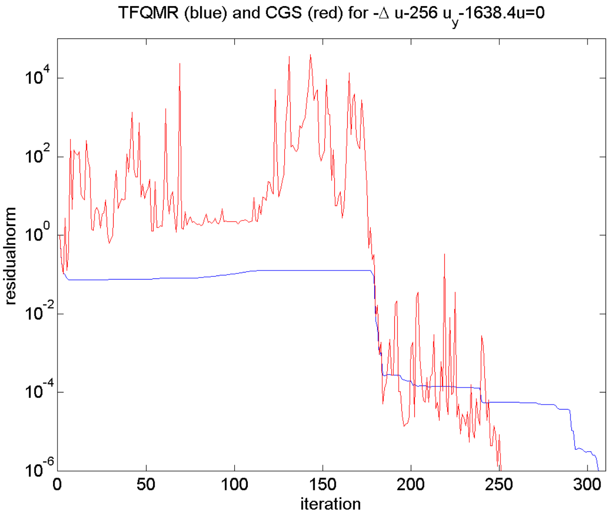


TFQMR (blue) and CGS (red) for -Δ u-256 u_y=0
residualnorm
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
0
20
40
60
80
100
120
140
160
180
iteration

Example


TFQMR (blue) and CGS (red) for -Δ u-256 u_y-163.84u=0
residualnorm
10^4
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
0
20
40
60
80
100
120
140
160
180
iteration

Example



TFQMR (blue) and CGS (red) for -Δ u-256 u_y-1638.4u=0
residualnorm
10^4
10^2
10^0
10^-2
10^-4
10^-6
0
50
100
150
200
250
300
iteration

QMRBiCGStab

In the same way as the QMR variant TFQMR was deduced from the CGS method Chan, Gallopoulos, Simoncini, Szeto und Tong (1994) derived the QMR variant QMRBiCGStab from the BiCGStab method.

QMRBiCGStab algorithm

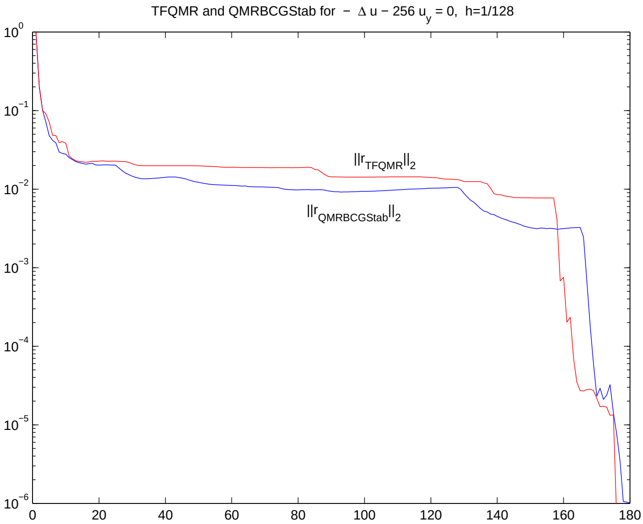
$r = b - Ax$; $\tilde{r} = r$; $u = r$; $v = Au$; $d = 0$
$\tau = \|r\|_2$; $\rho = \tilde{r}^T r$; $\theta = 0$; $\eta = 0$
for $k = 1, 2, \dots$ until convergence **do**
    $\alpha = \rho/(\tilde{r}^T v)$; $s = r - \alpha v$;
    $\tilde{\theta} = \|s\|_2/\tau$; $c = 1/\sqrt{1 + \tilde{\theta}^2}$
    $d = u + \theta^2 \eta d/\alpha$
    $\eta = c^2 \alpha$; $x = x + \eta d$
    $\tau = \tau \tilde{\theta} c$
    $w = As$
    $\omega = w^T s/(w^T w)$
    $r = s - \omega w$
    $\theta = \|r\|_2/\tau$; $c = 1/\sqrt{1 + \theta^2}$
    $d = s + \tilde{\theta}^2 \eta d/\omega$
    $\eta = c^2 \omega$
    $x = x + \eta d$
    $\tau = \tau \theta c$; $\beta = 1/\rho$; $\rho = \tilde{r}^T r$; $\beta = \beta \rho \alpha/\omega$
    $u = r + \beta(u - \omega v)$
    $v = Au$
end for

Cost of QMRBiCGStab

2 matrix-vector products
6 scalar products
8 _axpy

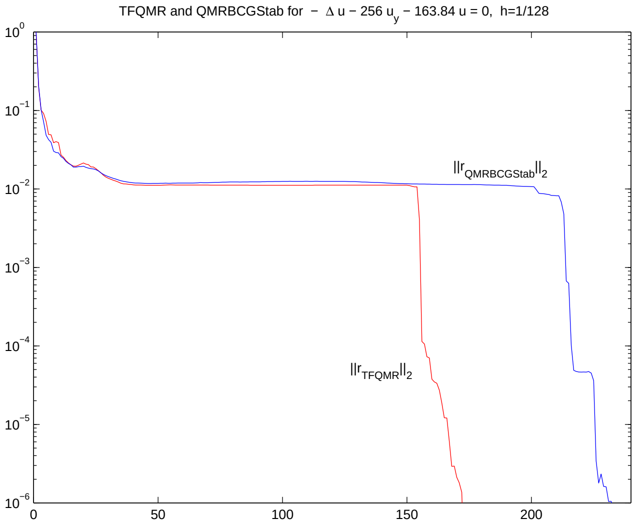
Storage requirements: 8 vectors

Example



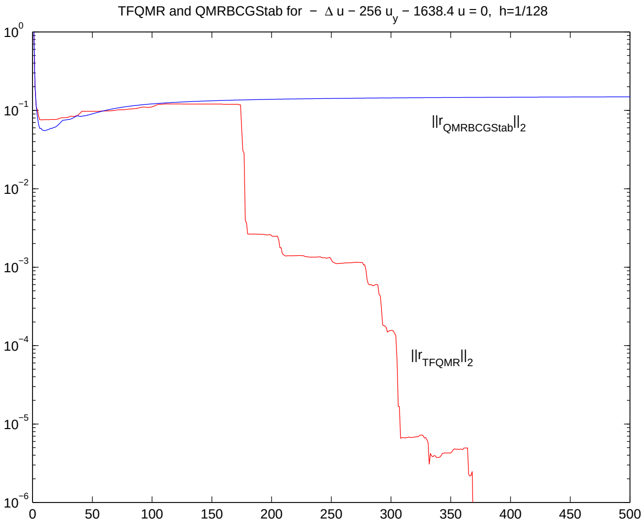
TFQMR and QMRBCGStab for − Δ u − 256 u_y = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
20
40
60
80
100
120
140
160
180
||r_TFQMR||_2
||r_QMRBCGStab||_2

Example



TFQMR and QMRBCGStab for − Δ u − 256 u_y − 163.84 u = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
50
100
150
200
||r_QMRBCGStab||_2
||r_TFQMR||_2

Example



TFQMR and QMRBCGStab for − Δ u − 256 u_y − 1638.4 u = 0, h=1/128
10^0
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
50
100
150
200
250
300
350
400
450
500
||r_QMRBCGStab||_2
||r_TFQMR||_2