Methods based on short recurrence

In this Section we consider methods which determine approximations $x^k \in x^0 + \mathcal{K}_k(r^0, A)$ from short recurrences.

By the Theorem of Faber and Manteuffel these methods can not minimize the residuum. Hence, measured by the number of iteration steps necessary to reduce the residual norm by a given factor they must be outperformed by GMRES.

However, the arithmetic work necessary to obtain a given accuracy can be much smaller than for GMRES.

BiCG method Lanczos (1952), Fletcher (1976)

$r^0 = b - Ax^0$;
Choose $\tilde r^0$ with $\alpha_0 = (\tilde r^0)^T r^0 \neq 0$
$d^1 = r^0$; $\tilde d^1 = \tilde r^0$;
for $k = 1, 2, \ldots$ until convergence

$$
\begin{aligned}
s^k &= Ad^k;\\
\tau_k &= \alpha_{k-1}/(\tilde d^k)^T s^k;\\
x^k &= x^{k-1} + \tau_k d^k;\\
r^k &= r^{k-1} - \tau_k s^k;\\
\tilde r^k &= \tilde r^{k-1} - \tau_k A^T \tilde d^k;\\
\beta_k &= 1/\alpha_{k-1};\\
\alpha_k &= (\tilde r^k)^T r^k;\\
\beta_k &= \alpha_k \beta_k;\\
d^{k+1} &= r^k + \beta_k d^k;\\
\tilde d^{k+1} &= \tilde r^k + \beta_k \tilde d^k;
\end{aligned}
$$
end

Cost: 2 matrix-vector products
2 scalar products
5 _axpy

Storage requirements: 6 vectors

If A is symmetric and positive definite, and if $\tilde{r}^0 = r^0$, then $\tilde{r}^k = r^k$ and $\tilde{d}^k = d^k$ for every k, and the BiCG method coincides with the CG method.

Whereas for SPD matrices (in exact arithmetic) the method can stop only if the solution $x^k = x^*$ of the linear system $Ax = b$ is reached, in the general case it can break down with $(\tilde{r}^k)^T r^k = 0$ or $(\tilde{d}^k)^T A d^k = 0$.

It can be seen easily that the orthogonality of the residuals and the conjugacy of the search directions in the CG method have to be replaced by the following properties of the BiCG method:

4 vector sequences r^k, $\tilde{r}^k$, d^k and $\tilde{d}^k$ are determined such that r^k and $\tilde{r}^k$ are biorthogonal, i.e.

$$(\tilde{r}^i)^T r^j = 0 \quad \text{for } i \neq j$$

and such that d^k und $\tilde{d}^k$ are biconjugate, i.e.

$$(\tilde{d}^i)^T A d^j = 0 \quad \text{for } i \neq j.$$

A direct consequence of the biorthogonality of r^j and $\tilde{r}^j$ is, that the BiCG method stops after at most n Steps.

From $d^{k+1} = r^k + \beta_k d^k$ it follows by induction

$$d^{k+1} = r^k + \sum_{j=0}^{k-1} \delta_j r^j.$$

Hence, d^{k+1} and $\tilde{r}^j$ are orthogonal for $j > k$, and correspondingly $\tilde{d}^{k+1}$ and r^j are orthogonal for $j > k$.

As for the CG method it follows by induction

$$\text{span}\{d^1, \ldots, d^k\} = \text{span}\{r^0, Ar^0, \ldots, A^{k-1}r^0\}.$$

Hence, the iterates are contained in the shifted Krylov space $x^0 + \mathcal{K}_k(r^0, A)$.

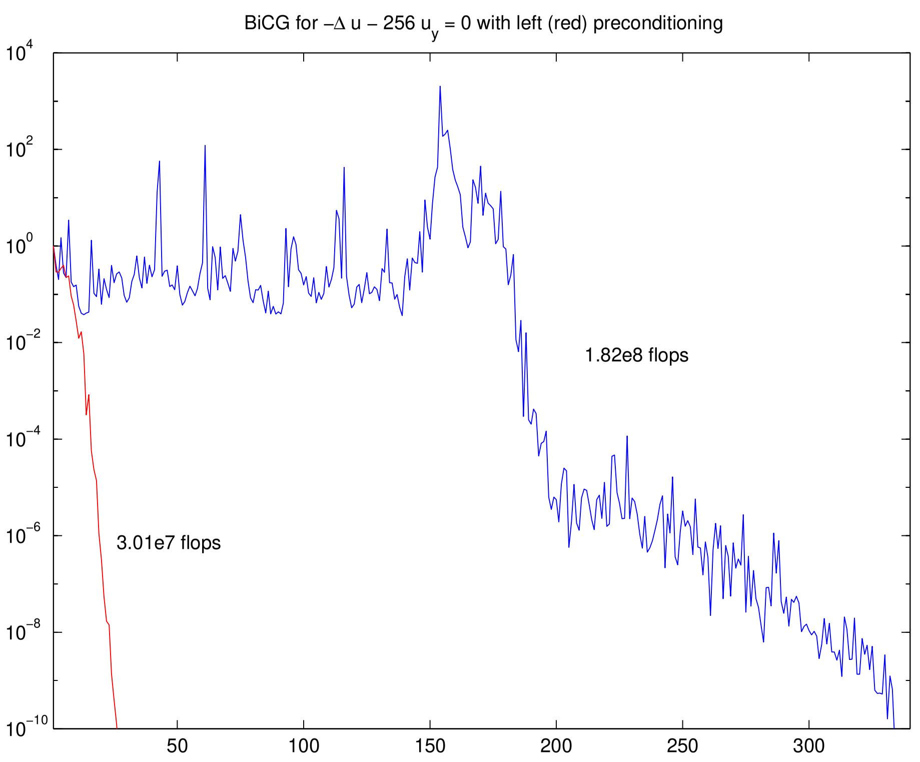
Similarly, it holds that

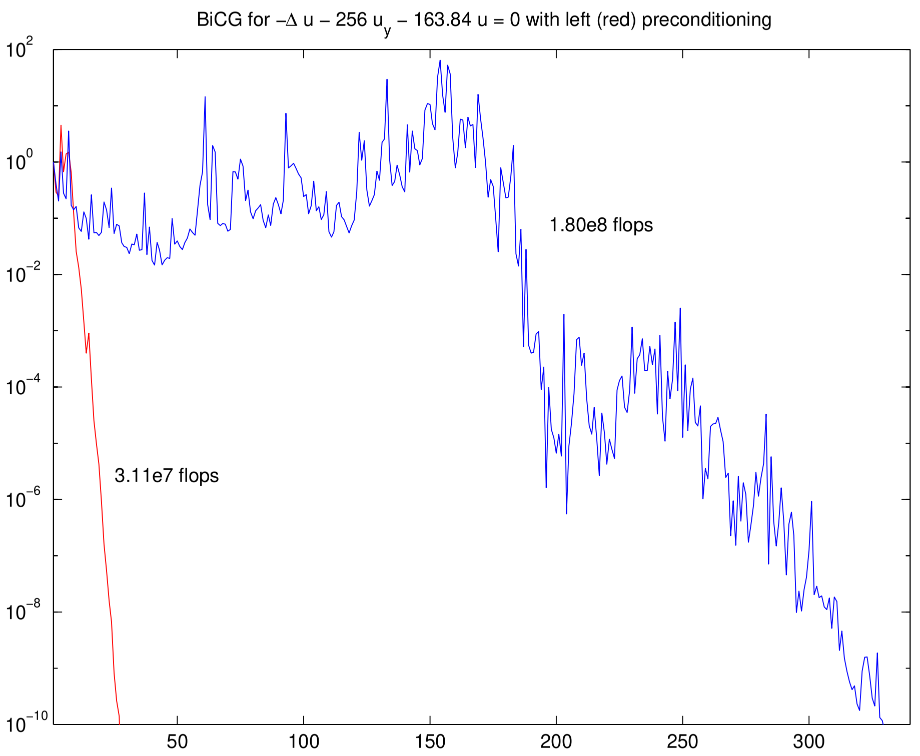
$$\text{span}\{\tilde{r}^0, \ldots, \tilde{r}^{k-1}\} = \text{span}\{\tilde{r}^0, A^T\tilde{r}^0, \ldots, (A^T)^{k-1}\tilde{r}^0\}.$$

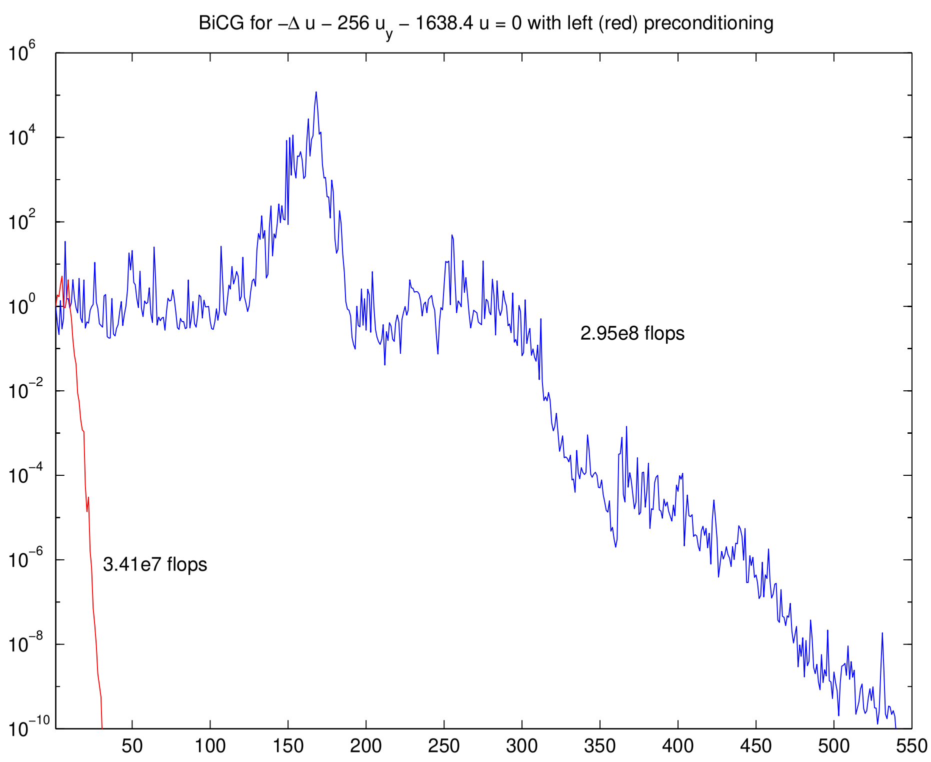
Hence, the biorthogonality $(\tilde{r}^j)^T r^k = 0,\ j = 0, \ldots, k-1$ is equivalent

$$((A^T)^j \tilde{r}^0)^T (b - Ax^k) = 0,\ j = 0, \ldots, k-1.$$

Therefore, the iterate x^k of the BiCG method is the Petrov-Galerkin approximation with respect to the ansatz space $V_k := \mathcal{K}_k(r^0, A)$ and the test space $W_k = \mathcal{K}_k(\tilde{r}^0, A^T)$.


BiCG for –Δ u – 256 u_y = 0 with left (red) preconditioning
1.82e8 flops
3.01e7 flops
10^4
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
50
100
150
200
250
300


BiCG for –Δ u – 256 u_y – 163.84 u = 0 with left (red) preconditioning
1.80e8 flops
3.11e7 flops
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
50
100
150
200
250
300



One drawback of the BiCG method is, that in every step one multiplication with A^T is needed, which is not at hand in many cases (in FEM modelling Ax is accumulated from its element proportions, and A^Tx is not available) or which can not be realized easily (on parallel computers with distributed memory, e.g.).

Sonneveld 1989 presented the CGS method (Conjugate Gradient Squared) which is a transposition free variant of the BiCG method, i.e. which avoids multiplications by A^T.

We first consider a different interpretation of the CG method.

CG method and orthogonal polynomials

Let A be a SPD matrix.

From

$$\text{span}\{d^1, \dots, d^k\} = \text{span}\{r^0, Ar^0, \dots, A^{k-1}r^0\}$$

it follows that there exists a polynomial $\psi_k \in \Pi_{k-1}$ with

$$d^k = \psi_{k-1}(A)r^0,$$

and $r^k = d^{k+1} - \beta_k d^k$ can be written as

$$r^k = \phi_k(A)r^0, \ \phi_k \in \Pi_k.$$

From

$$\begin{aligned} d^{k+1} &= r^k + \beta_k d^k = \phi_k(A)r^0 + \beta_k \psi_{k-1}(A)r^0 \\ r^{k+1} &= r^k - \tau_{k+1} A d^{k+1} = \phi_k(A)r^0 - \tau_{k+1} A \psi_k(A) r^0 \end{aligned}$$

one obtains the recurrences

$$\begin{aligned} \phi_{k+1}(t) &= \phi_k(t) - \tau_{k+1} t \psi_k(t) \\ \psi_k(t) &= \phi_k(t) + \beta_k \psi_{k-1}(t). \end{aligned}$$

The parameters β_k and τ_k can be determined from the polynomials ϕ_k and ψ_k and the bilinear form

$$\langle \phi, \psi \rangle := (\phi(A)r^0)^T \psi(A) r^0.$$

$$\langle \phi, \psi \rangle := (\phi(A)r^0)^T \psi(A)r^0$$

is almost a scalar product. It is bilinear and symmetric and nonnegative. Only the positive definiteness is missing, since a polynomial ϕ might exist such that $\phi(A)r^0 = 0$.

Additionally it holds that

$$\langle \phi\psi, \chi \rangle = \langle \phi, \psi\chi \rangle \quad \text{for every polynomial } \phi, \psi, \chi.$$

Rewritten with polynomials the CG algorithm obtains the following form:

Orthogonal polynomials with CG

$\phi_0 \equiv 1$; $\psi_0 \equiv 1$;
$\alpha_0 = 1$;
for $k = 1, 2, \ldots$
  $\gamma_{k-1} = \langle \psi_{k-1}, \iota\psi_{k-1} \rangle$;
  $\tau_k = \alpha_{k-1}/\gamma_{k-1}$;
  $\phi_k = \phi_{k-1} - \tau_k \iota \psi_{k-1}$;
  $\beta_k = 1/\alpha_{k-1}$;
  $\alpha_k = \langle \phi_k, \phi_k \rangle$;;
  $\beta_k = \alpha_k \beta_k$;
  $\psi_k = \phi_k + \beta_k \psi_{k-1}$;
  end

where the polynomial ι is defined by $\iota(t) := t$.

The orthogonality of the residuals and the conjugacy of the search directions in the CG method yield

$$\langle \phi_j, \phi_k \rangle = \alpha_j \delta_{jk}, \quad \langle \psi_j, \iota\psi_k \rangle = \gamma_k \delta_{jk}$$

for the polynomials.

These equations do not depend on the scalar product $\langle \cdot, \cdot \rangle$ under consideration, but they hold for every symmetric bilinear form $[\,\cdot\,,\,\cdot\,]$ with

$$[\phi\psi, \chi] = [\phi, \psi\chi] \quad \text{for every polynomial } \phi, \psi, \chi.$$

Hence, the CG method can be considered as a method for determining orthogonal polynomials with respect to a given bilinear form $[\cdot, \cdot]$.

Correspondingly the BiCG method can be considered as a method for determining polynomials $\phi_k, \tilde{\phi}_k \in \Pi_k$ such that $r^k = \phi_k(A) r^0$ and $\tilde{r}^k = \tilde{\phi}_k(A^T)\tilde{r}^0$, and the algorithm indicates that for every k it holds $\phi_k = \tilde{\phi}_k$. Therefore,

$$r^k = \phi_k(A) r^0, \quad \tilde{r}^k = \phi_k(A^T)\tilde{r}^0.$$

Comparing BiCG with the CG Method it follows that this polynomial is the residual polynomial of the CG method.

Likewise,

$$d^k = \psi_{k-1}(A) r^0, \quad \tilde{d}^k = \psi_{k-1}(A^T)\tilde{r}^0,$$

where ψ_k is obtained from the CG algorithm.

The parameters α_k and γ_k can be determined from

$$\alpha_k = (\tilde{r}^k)^T r^k = (\phi_k(A^T)\tilde{r}^0)^T \phi_k(A) r^0 = (\tilde{r}^0)^T (\phi_k(A))^2 r^0$$

and

$$\gamma_k = (\tilde{d}^k)^T A d^k = (\psi_{k-1}(A^T)\tilde{r}^0)^T A \psi_{k-1}(A) r^0 = \tilde{r}^0 A (\psi_{k-1}(A))^2 r^0,$$

i.e. without using A^T.

With

$$[\phi, \psi] := (\phi(A^T) r^0)^T \psi(A) r^0$$

α_k and γ_k can be calculated from

$$\alpha_k = [1, \phi_k^2], \quad \gamma_k = [1, \iota\psi_{k-1}^2].$$

From

$$\phi_k = \phi_{k-1} - \tau_k \psi_{k-1}, \;\; \psi_k = \phi_k + \beta_k \psi_{k-1}$$

we get for

$$\Phi_k := \phi_k^2, \;\; \Psi_k := \psi_k^2, \;\; \Xi_k := \phi_k \psi_{k-1}, \;\; \Theta_k := \phi_k \psi_k = \Phi_k + \beta_k \Xi_k :$$

$$\begin{aligned}
\Xi_k &= (\phi_{k-1} - \tau_k \iota \psi_{k-1})\psi_{k-1} = \Theta_{k-1} - \tau_k \iota \Psi_{k-1} \\
\Phi_k &= \Phi_{k-1}^2 - 2\tau_k \iota \Theta_{k-1} + \tau_k^2 \iota^2 \Psi_{k-1} = \Phi_{k-1} - \tau_k \iota (\Theta_{k-1} + \Xi_k) \\
\Theta_k &= \Phi_k + \beta_k \Xi_k \\
\Psi_k &= \Phi_k + 2\beta_k \Xi_k + \beta_k^2 \Psi_{k-1} = \Theta_k + \beta_k (\Xi_k + \beta_k \Psi_{k-1}).
\end{aligned}$$

Hence, the following algorithm allows to compute Φ_k, Ψ_k, Ξ_k and Θ_k, and thus α_k and γ_k, without using A^T.

Orthogonal polynomials; transposition free

$\Phi_0 \equiv 1;\ \Psi_0 \equiv 1;\ \Theta_0 \equiv 1;$
$\alpha_0 = 1;$
for $k = 1, 2, \ldots$
$\quad \gamma_k = [1, \iota\Psi_{k-1}];$
$\quad \tau_k = \alpha_{k-1}/\gamma_k;$
$\quad \Xi_k = \Theta_{k-1} - \tau_k \iota \Psi_{k-1};$
$\quad \Phi_k = \Phi_{k-1} - \tau_k \iota (\Theta_{k-1} + \Xi_k);$
$\quad \beta_k = 1/\alpha_{k-1};$
$\quad \alpha_k = [1, \Phi_k];;$
$\quad \beta_k = \alpha_k \beta_k;$
$\quad \Theta_k = \Phi_k + \beta_k \Xi_k;$
$\quad \Psi_k = \Theta_k + \beta_k(\Xi_k + \beta_k \Psi_{k-1});$
end

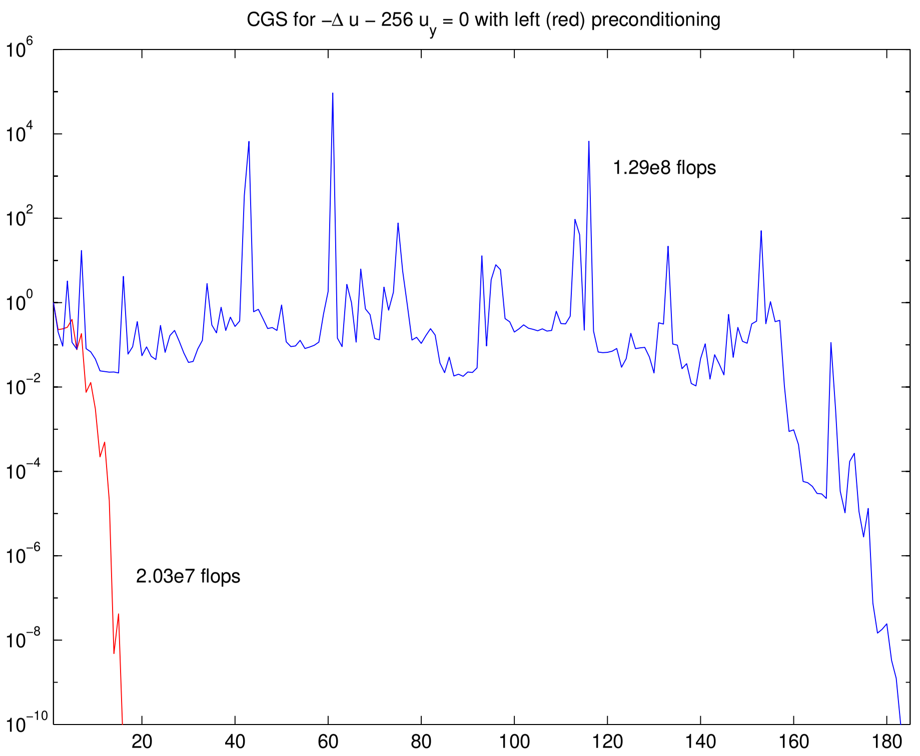
Substituting A into the polynomials and defining $r^k = \Phi_k(A)r^0$, $d^k = \Psi_{k-1}(A)r^0$, $u^k = \Theta_k(A)r^0$, $q^k = \Xi_k(A)r^0$, we get:

CGS method Sonneveld (1989)

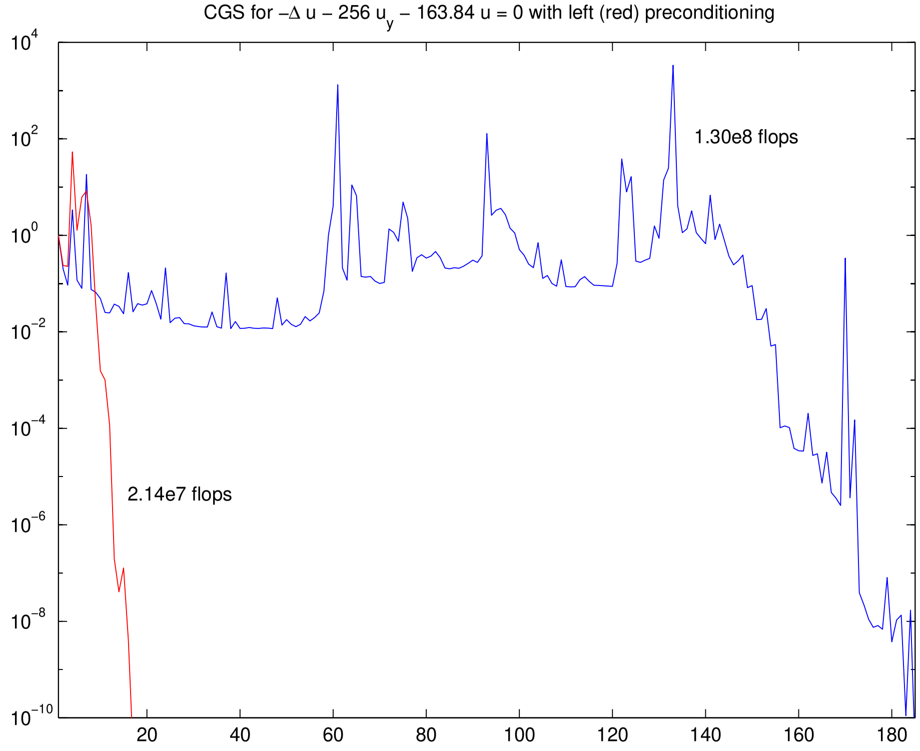
$r^0 = b - Ax^0;\ d^1 = r^0;\ u^0 = r^0;$
Choose $\tilde{r}^0$ with $\alpha_0 = (\tilde{r}^0)^T r^0 \neq 0$
for $k = 1, 2, \ldots$ until convergence
$\quad s^k = Ad^k;$
$\quad \gamma_k = (\tilde{r}^0)^T s^k; \qquad \tau_k = \alpha_{k-1}/\gamma_k;$
$\quad q^k = u^{k-1} - \tau_k s^k;$
$\quad w^k = u^{k-1}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

Cost: 2 matrix-vector products (not with A^T)
2 scalar products
7 _axpy

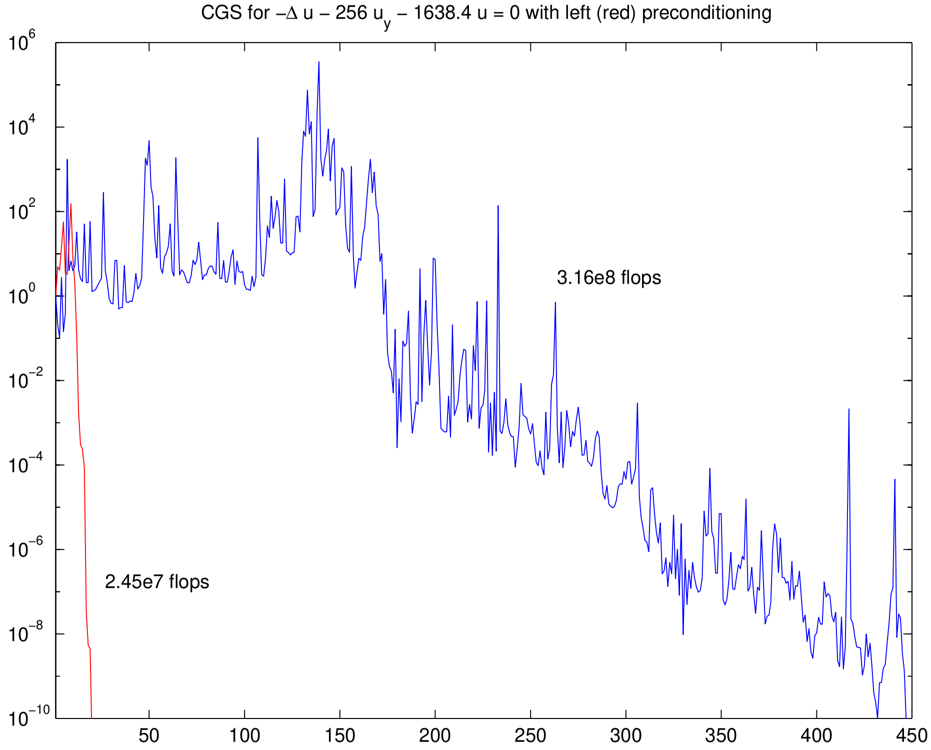
Storage requirements: 8 vectors



CGS for $-\Delta u - 256 u_y - 163.84 u = 0$ with left (red) preconditioning

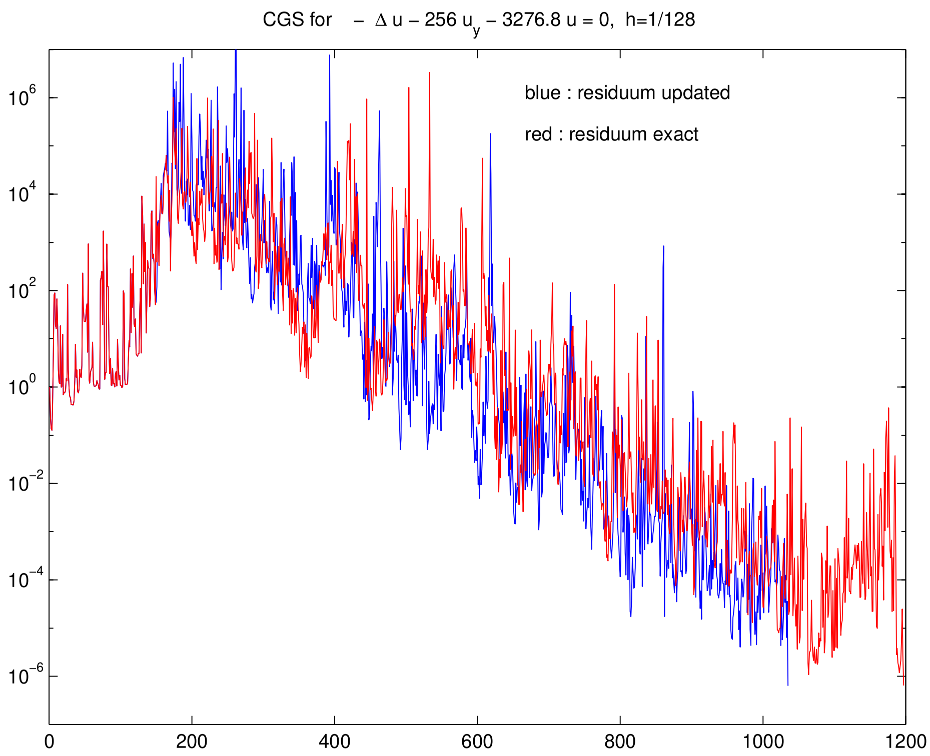


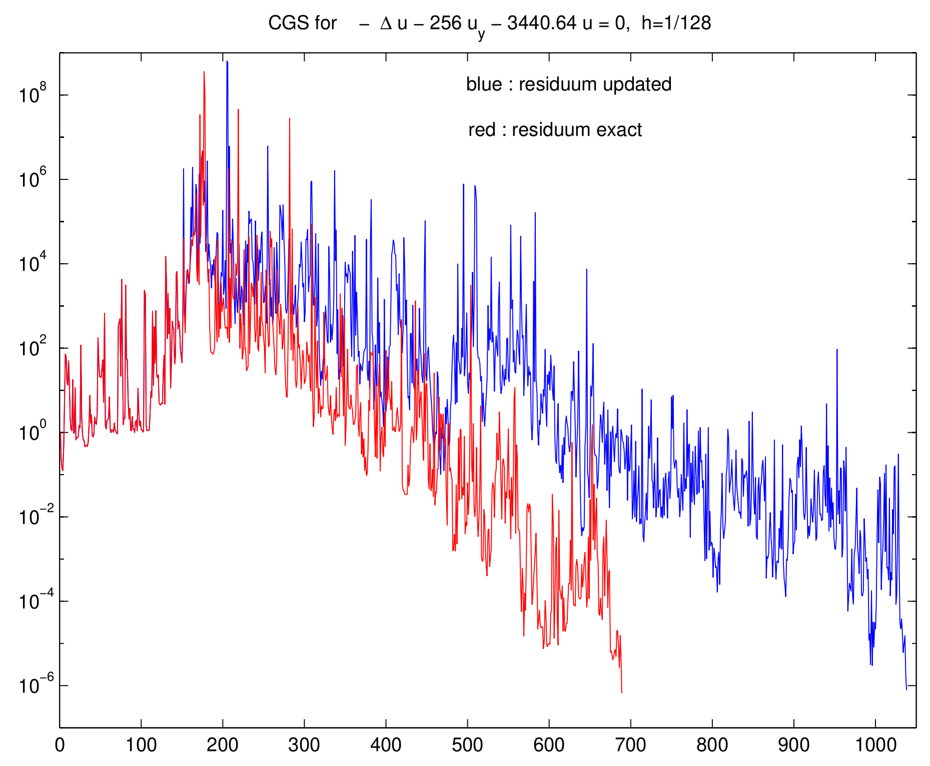
CGS for $-\Delta u - 256 u_y - 1638.4 u = 0$ with left (red) preconditioning



The vector Aw^k is only used to update the residuum r^k. At the same expense one can determine the residuum $r^k = b - Ax^k$ directly.

Van der Vorst (1992) reports that in many examples it has been observed that the direct calculation of the residuum has a negative effect on the iteration process, i.e. it slows down the convergence of CGS.





The BiCG method shows a very erratic convergence behaviour with wildly oscillating residual norm $\|r^k_{BiCG}\|_2 = \|\phi_k(A)r^0\|_2$, and for $\|r^k_{CGS}\|_2 = \|(\phi_k(A))^2 r^0\|_2$ this behaviour is even worse.

The polynomials ϕ_k are characterized by $\phi_k(0) = 1$ and

$$(\phi_j(A^T)\tilde{r}^0)^T \phi_k(A)r^0 = 0,\ j = 0, \ldots, k-1,$$

i.e. by the fact that $\phi_k(A)r^0$ is orthogonal to the Krylov space $\mathcal{K}_k(\tilde{r}^0, A^T)$.

Equivalent is the requirement that $\phi_k(A)r^0$ is orthogonal to $\tilde{\phi}_j(A^T)\tilde{r}^0$ for suitably chosen polynomials $\tilde{\phi}_j$ of degree j für $j = 0, \ldots, k-1$.

Van der Vorst proposed the choice

$$\tilde{\phi}_k(t) = \prod_{j=1}^{k}(1-\omega_j t) = (1-\omega_k)\tilde{\phi}_{k-1}(t),$$

where ω_k is determined such that the residual norm is $\|r^k\|_2$ minimal.

The algorithm BiCGStab results which shows a much smoother convergence behaviour as BiCG or CGS.

The method is deduced in a similar way as CGS.

For

$$\Phi_j := \tilde{\phi}_j \phi_j, \;\; \Psi_j := \tilde{\phi}_j \psi_j$$

it holds

$$\begin{aligned}
\Phi_k &= \tilde{\phi}_k \phi_k = (1-\omega_k \iota)\tilde{\phi}_{k-1}(\phi_{k-1} - \tau_k \iota \psi_{k-1}) \\
&= \Phi_{k-1} - \tau_k \iota \Psi_{k-1} - \omega_k \iota (\Phi_{k-1} - \tau_k \iota \Psi_{k-1}), \\
\Psi_k &= \tilde{\phi}_k(\phi_k + \beta_k \psi_{k-1}) = \tilde{\phi}_k \phi_k + \beta_k (1-\omega_k \iota)\tilde{\phi}_{k-1}\psi_{k-1} \\
&= \Phi_k + \beta_k \Psi_{k-1} - \beta_k \omega_k \iota \Psi_{k-1}
\end{aligned}$$

To execute these recurrences, we have to represent β_k and τ_k of the BiCG method as scalar products of the polynomials Φ_j and Ψ_j. Let

$$\tilde{\alpha}_k := \langle 1, \Phi_k \rangle = \langle \tilde{\phi}_k, \phi_k \rangle.$$

From

$$\tilde{\phi}_k = (-1)^k \prod_{j=1}^k \omega_j \iota^k + \hat{\phi}_{k-1}, \quad \hat{\phi}_{k-1} \in \Pi_{k-1},$$

and the orthogonality of ϕ_k and Π_{k-1} we get

$$\tilde{\alpha}_k = (-1)^k \prod_{j=1}^k \omega_j \langle \iota^k, \phi_k \rangle,$$

and from

$$\phi_k = (-1)^k \prod_{j=1}^k \tau_j \iota^k + \bar{\phi}_{k-1}, \quad \bar{\phi}_{k-1} \in \Pi_{k-1},$$

it follows

$$\alpha_k = \langle \phi_k, \phi_k \rangle = (-1)^k \prod_{j=1}^k \tau_j \langle \iota^k, \phi_k \rangle = \prod_{j=1}^k \frac{\tau_j}{\omega_j} \tilde{\alpha}_k,$$

$$\alpha_k = \prod_{j=1}^k \frac{\tau_j}{\omega_j} \tilde{\alpha}_k \quad \Longrightarrow \quad \beta_k = \frac{\alpha_k}{\alpha_{k-1}} = \frac{\tilde{\alpha}_k}{\tilde{\alpha}_{k-1}} \cdot \frac{\tau_k}{\omega_k}.$$

Similarly from $\langle \chi, \iota\psi_{k-1} \rangle = 0$ for every $\chi \in \Pi_{k-2}$ one gets

$$\begin{aligned} \tilde{\gamma}_k &:= \langle 1, \iota\Psi_{k-1} \rangle = \langle \tilde{\phi}_{k-1}, \iota\psi_{k-1} \rangle = (-1)^{k-1} \prod_{j=1}^{k-1} \omega_j \langle \iota^k, \iota\psi_{k-1} \rangle \\ \gamma_k &= \langle \psi_{k-1}, \iota\psi_{k-1} \rangle = (-1)^{k-1} \prod_{j=1}^{k-1} \tau_j \langle \iota^{k-1}, \iota\psi_{k-1} \rangle = \prod_{j=1}^{k-1} \frac{\tau_j}{\omega_j} \tilde{\gamma}_j, \end{aligned}$$

and therefore

$$\tau_k = \frac{\alpha_{k-1}}{\gamma_k} = \frac{\tilde{\alpha}_{k-1}}{\tilde{\gamma}_k}.$$

Finally

$$\begin{aligned} r^k &= \Phi_k(A)r^0 \\ &= (1-\omega_k A)\tilde{\phi}_{k-1}(A)(\phi_{k-1}(A) - \tau_k A\psi_{k-1}(A))r^0 \\ &= (1-\omega_k A)(\Phi_{k-1}(A) - \tau_k A\Psi_{k-1}(A))r^0 \\ &=: (1-\omega_k A)u^k \end{aligned}$$

yields that $\|r^k\|_2$ is minimal if and only if

$$\omega_k = \frac{(u^k)^T A u^k}{\|Au^k\|_2^2}.$$

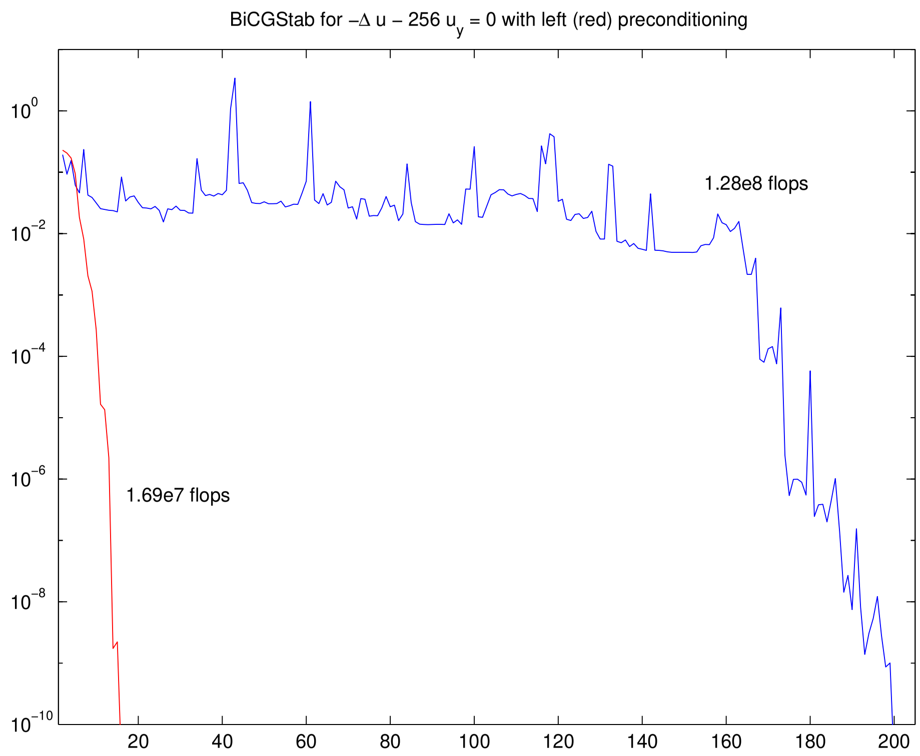
BiCGStab method Van der Vorst (1992)

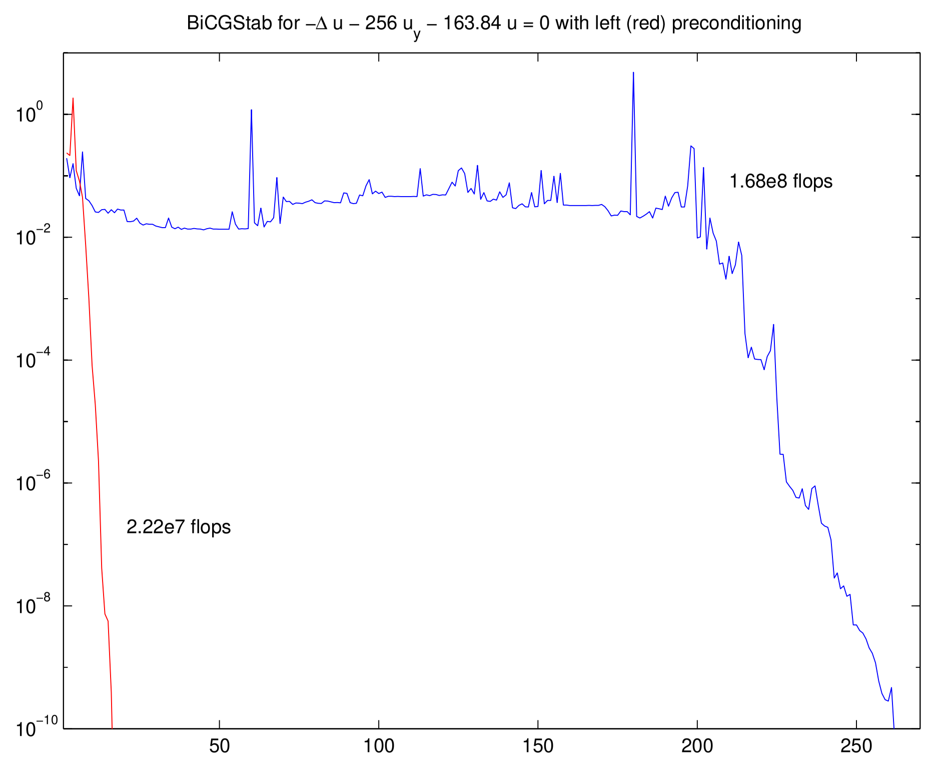
$r^0 = b - Ax^0$; $d^1 = r^0$;
Choose $\tilde{r}^0$ with $\tilde{\alpha}_0 = (\tilde{r}^0)^T r^0 \neq 0$
for $k = 1, 2, \dots$ until convergence

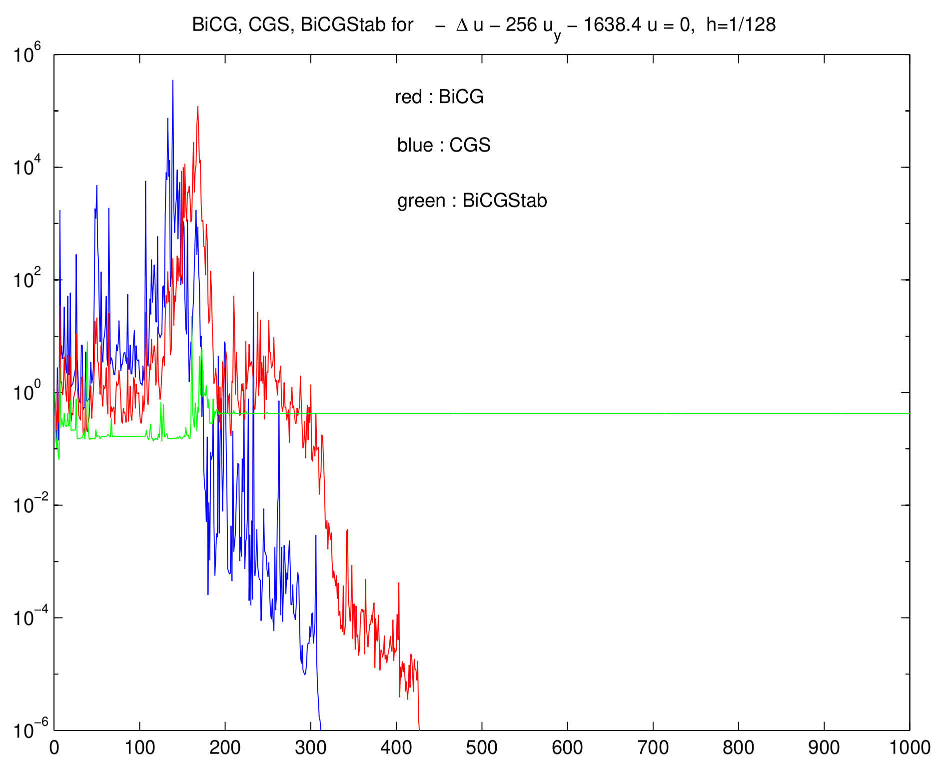
$s^k = Ad^k$;
$\tilde{\gamma}_k = (\tilde{r}^0)^T s^k$; $\tau_k = \tilde{\alpha}_{k-1}/\tilde{\gamma}_k$;
$u^k = r^{k-1} - \tau_k s^k$;
$t^k = Au^k$;
$\omega_k = (t^k)^T u^k / \|t^k\|_2^2$;
$x^k = x^{k-1} + \tau_k d^k + \omega_k u^k$;
$r^k = u^k - \omega_k t^k$;
$\beta_k = 1/\tilde{\alpha}_{k-1}$;
$\tilde{\alpha}_k = (\tilde{r}^0)^T r^k$;;
$\beta_k = \tilde{\alpha}_k \beta_k \tau_k / \omega_k$;
$d^{k+1} = r^k + \beta_k(d^k - \omega_k s^k)$; end

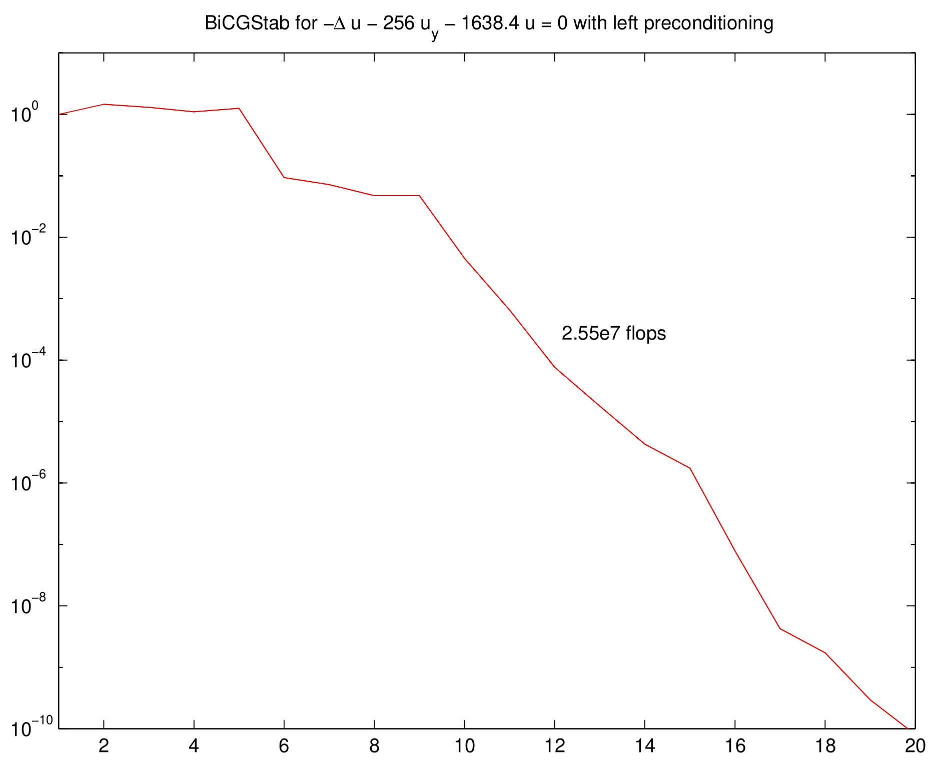
Cost: 2 matrix-vector products (not with A^T)
4 scalar products
6 _axpy

Storage requirements: 8 vectors




BiCGStab for –Δ u – 256 u_y – 163.84 u = 0 with left (red) preconditioning
1.68e8 flops
2.22e7 flops
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
50
100
150
200
250


BiCG, CGS, BiCGStab for – Δ u – 256 u_y – 1638.4 u = 0, h=1/128
red : BiCG
blue : CGS
green : BiCGStab
10^6
10^4
10^2
10^0
10^-2
10^-4
10^-6
0
100
200
300
400
500
600
700
800
900
1000



The stagnation of the BiCGStab algorithm can be avoided by the following generalization due to Sleijpen und Fokkema (1993):

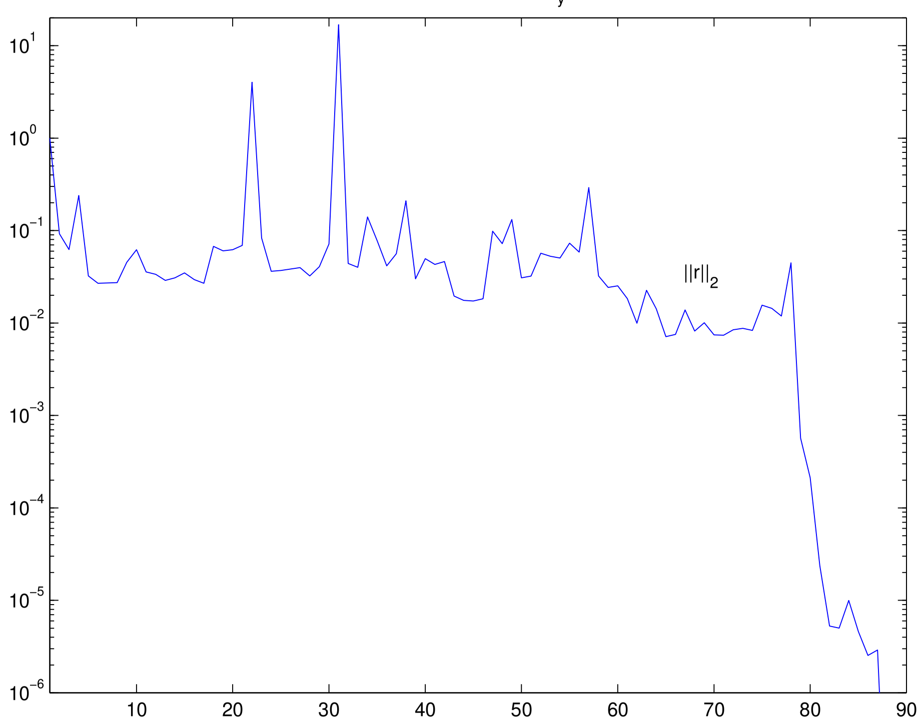
Let $\ell \geq 2$. Then in the k-th step, $k = m\ell + \ell$, $m = 0, 1, \dots$ the polynomial ψ_k is chosen as $\psi_k = \chi_m \psi_{k-\ell}$, where $\chi_m \in \Pi_\ell$ is determined such that $\chi_m(0) = 1$ and

$$\|r^k\|_2 = \|\chi(A)\psi_{k-\ell}(A)\phi_k(A)r^0\|_2,\ \ \chi \in \Pi_\ell,\ \ \chi(0) = 1,$$

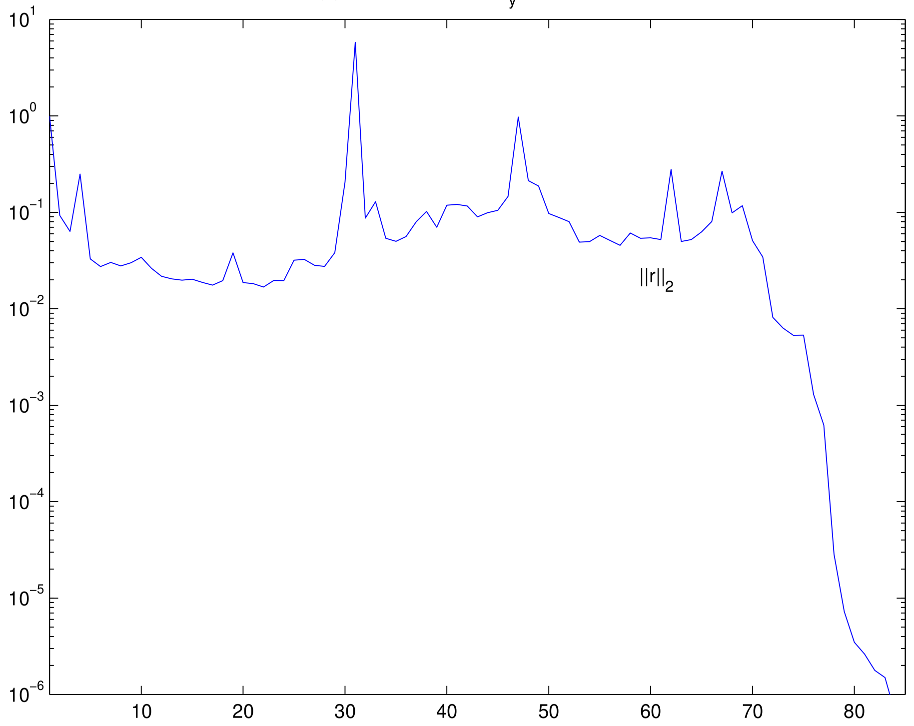
is minimal.

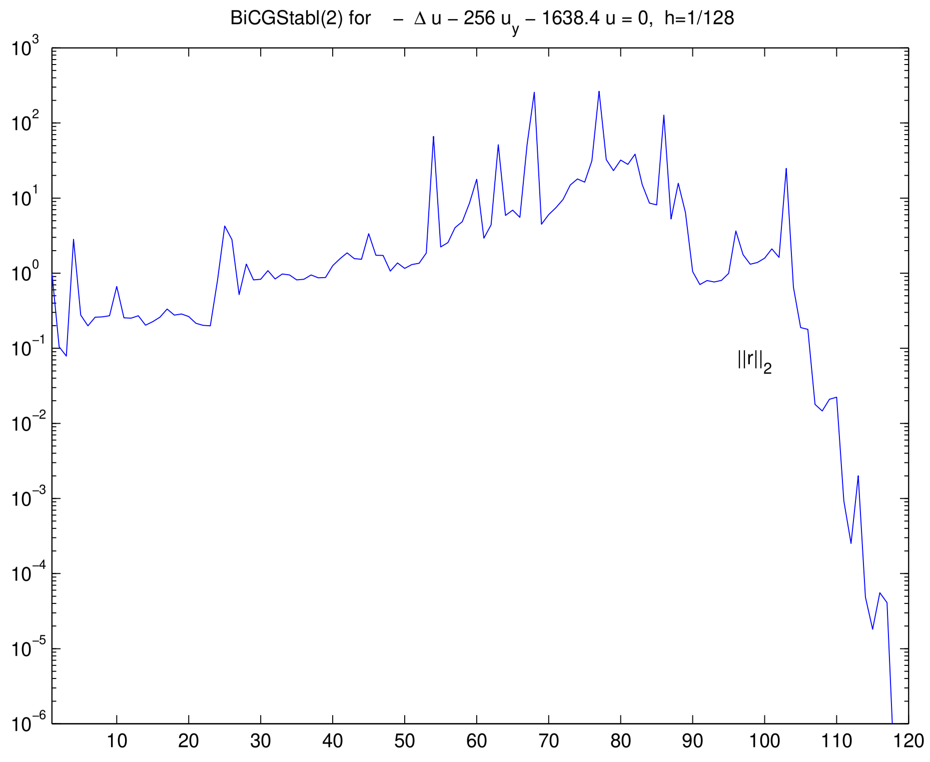
The vectors $r^{m\ell+i}$, $i = 1, \dots, \ell - 1$, are constructed by (transpose free) BiCG steps, where $\psi_{m\ell+i}$ is chosen as $\psi_{m\ell+i} = \iota^i \psi_{m\ell}$.

BiCGStabl(2) for $-\Delta u - 256 u_y = 0$, $h=1/128$

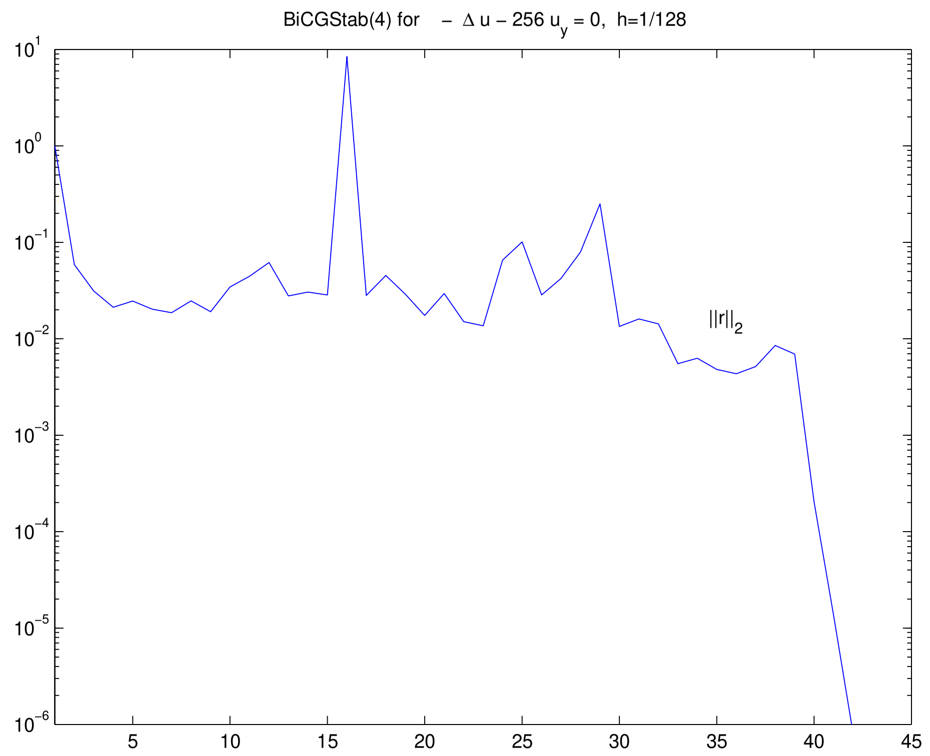


BiCGStabl(2) for $-\Delta u - 256 u_y - 163.84 u = 0$, $h=1/128$



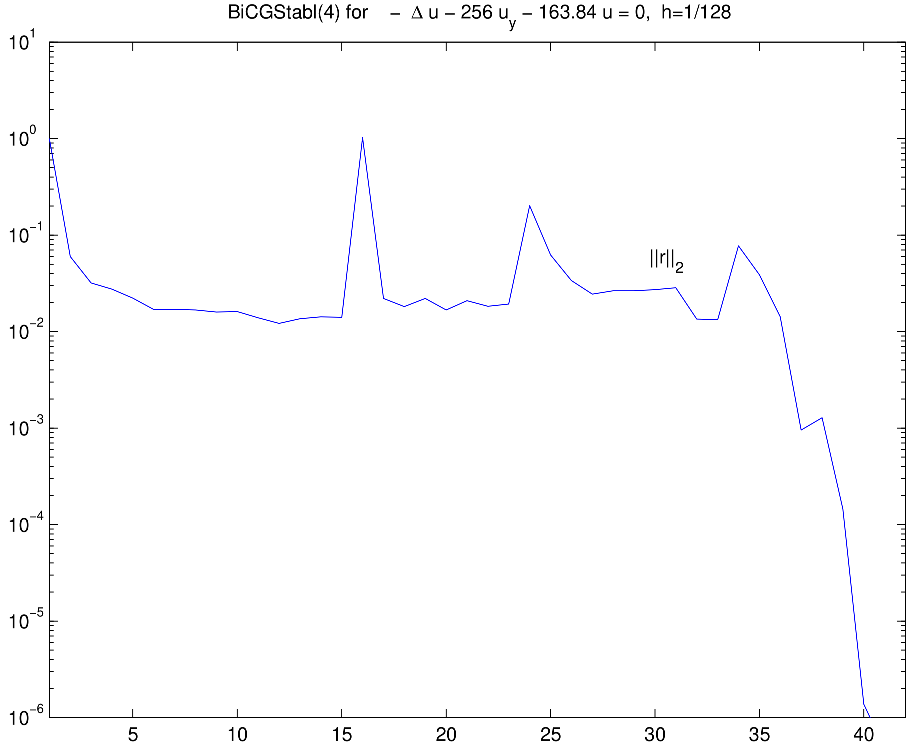


BiCGStabl(2) for − Δ u − 256 u_y − 1638.4 u = 0, h=1/128
||r||_2



BiCGStab(4) for − Δ u − 256 u_y = 0, h=1/128
||r||_2

BiCGStabl(4) for $-\Delta u - 256\, u_y - 163.84\, u = 0$, h=1/128



BiCGStabl(4) for $-\Delta u - 256\, u_y - 1638.4\, u = 0$, h=1/128