Lanczos-Type Solvers for Non-Hermitian Linear Systems

Martin H. Gutknecht

mhg@sam.math.ethz.ch

http://www.sam.math.ethz.ch/~mhg/

Seminar for Applied Mathematics
ETH Zurich

Part II of Short Course at EMS Workshop '99

- Lanczos-type solvers
- Lanczos process: related topics
- The unsymmetric Lanczos or BIO algorithm
- The BIORES form of the BICG method
- The (standard) BIOMIN form of the BICG method
- The BICG method: features, problems, improvements
- Avoiding breakdowns by look-ahead
- (Bi)CGS and Lanczos-Type Product Methods (LTPMs)
- The (standard) BIOMINS form of the BICGS method
- The BICGSTAB algorithm
- BICGSTAB2 and BICG×MR2
- The quasi-minimum residual (QMR) method
- Finite precision effects [= roundoff effects]
- Nonnormality
- Conclusions

The **unsymmetric** (or **nonsymmetric** or **two-sided**) **Lanczos process** [Lanczos **'50**$_{\text{JResNBS}}$] is the generalization to non-Hermitian matrices of the symmetric Lanczos process.

Instead of orthogonal bases of the Krylov spaces $\mathcal{K}_n = \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ pairs of **biorthogonal** (or, **dual**) bases of the pair of dual Krylov spaces $\mathcal{K}_n = \mathcal{K}_n(\mathbf{A}, \mathbf{y}_0)$ and $\widetilde{\mathcal{K}}_n = \mathcal{K}_n(\mathbf{A}^\star, \widetilde{\mathbf{y}}_0)$ are generated.

Instead of orthogonal projections of $\mathbf{A}$ we will now construct **oblique** projections that are still represented by tridiagonal matrices $\mathbf{T}_n$.

Hence, recurrences are three-term (as in the symmetric case). Alternatively, there exist coupled two-term recurrences involving direction vectors [Lanczos **'52**$_{\text{JResNBS}}$].

Like the symmetric Lanczos process, the unsymmetric one can be used to find some eigenvalues, preferably the extremal ones.

Its application to linear systems is a generalization of the three-term version (ORES) of the conjugate-gradient (CG) method (which is limited to Hpd matrices).

Again, when a linear system is solved, $\mathbf{y}_0 := \mathbf{r}_0$ or $\mathbf{y}_0 := \mathbf{r}_0/||\mathbf{r}_0||$. But one is free to choose $\widetilde{\mathbf{y}}_0$.

The polynomials associated with the residuals are here called **Lanczos polynomials**.

Krylov space solvers that make use of them will be called **Lanczos-type solvers**.

The basic Lanczos-type method for solving linear systems is the **biconjugate gradient** (BiCG) **method** [Lanczos **'52**$_{\mathrm{JResNBS}}$, Fletcher *Proc. Dundee '75*/**'76**] , which comes in various forms: BiOMin, BiORes, and BiODir.

There are many other Lanczos-type solvers, including

- the **quasi-minimal residual** (QMR) **method** [Freund/Nachtigal **'91**$_{\mathrm{NM}}$],
- the (**bi-**)**conjugate gradient squared** (CGS or BiCGS) **method** [Sonneveld *'84*/**'89**$_{\mathrm{SISSC}}$],
- BiCGStab [van der Vorst *'90*/**'92**$_{\mathrm{SISSC}}$], BiCGStab2 [Gutknecht *'91*/**'93**$_{\mathrm{SISC}}$], BiCGStab(ℓ) [Sleijpen/Fokkema *'93*/**'93**$_{\mathrm{ETNA}}$],
- **transposed-free QMR** (TFQMR) [Freund **'93**$_{\mathrm{SISC}}$],
- BiCG×MR2 [Zhang *"93*/**'97**$_{\mathrm{SISC}}$ (calls it GPBI-CG); Gutknecht *OW, Apr. '94*; Cao *'95*] .

The Lanczos process is closely related to several topics from classical complex analysis and linear algebra:

- **Formal orthogonal polynomials (FOPs)**
- **Padé approximation**
- **Continued fractions**
- **Inverse LU factorization of Hankel matrices**
- **Fast Hankel solvers**
- **Rutishauser's** QD **algorithm**
- **The QR algorithm for tridiagonal matrices**
- **The classical moment problem**
- **Gauss quadrature** (in spd case)

Aim: For given $\mathbf{A}$, $\mathbf{y}_0$, $\widetilde{\mathbf{y}}_0$, generate a pair of **biorthogonal** (or: **dual**) **bases** of **Lanczos vectors**

$$\mathbf{y}_n \in \mathcal{K}_{n+1} :\equiv \text{span}\,(\mathbf{y}_0, \mathbf{A}\mathbf{y}_0, \ldots, \mathbf{A}^n\mathbf{y}_0),$$
$$\widetilde{\mathbf{y}}_m \in \widetilde{\mathcal{K}}_{m+1} :\equiv \text{span}\,(\widetilde{\mathbf{y}}_0, \mathbf{A}^\star\widetilde{\mathbf{y}}_0, \ldots, (\mathbf{A}^\star)^m\widetilde{\mathbf{y}}_0),$$

$$\boxed{\langle \widetilde{\mathbf{y}}_m, \mathbf{y}_n \rangle = \begin{cases} 0 & \text{if} \quad m \neq n, \\ \delta_n \neq 0 & \text{if} \quad m = n. \end{cases} \qquad (m, n = 0, 1, \ldots, \nu - 1)}$$

that span a pair of finite, nested sequences of Krylov spaces, $\{\mathcal{K}_n\}_{n=1}^{\nu}$ and $\{\widetilde{\mathcal{K}}_n\}_{n=1}^{\nu}$.

Basic idea:

$$\boxed{\textbf{two-sided Gram-Schmidt + Krylov space extension}}$$

ALGORITHM 1 (UNSYMMETRIC LANCZOS OR BIO ALGORITHM). *Choose* $\mathbf{y}_0$, $\widetilde{\mathbf{y}}_0 \in \mathbb{C}^N$ *such that* $\delta_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{y}_0 \rangle \neq 0$, *and set* $\beta_0 := 0$. *For* $n = 0, 1, \ldots$ *compute*

$$\alpha_n := \langle \widetilde{\mathbf{y}}_n, \mathbf{A}\mathbf{y}_n \rangle / \delta_n\,, \tag{1a}$$
$$\beta_{n-1} := \gamma_{n-1}\delta_n/\delta_{n-1} \quad (\textit{if } n > 0), \tag{1b}$$
$$\mathbf{y}_{\text{temp}} := \mathbf{A}\mathbf{y}_n - \mathbf{y}_n\alpha_n - \mathbf{y}_{n-1}\beta_{n-1}\,, \tag{1c}$$
$$\widetilde{\mathbf{y}}_{\text{temp}} := \mathbf{A}^\star\widetilde{\mathbf{y}}_n - \widetilde{\mathbf{y}}_n\overline{\alpha_n} - \widetilde{\mathbf{y}}_{n-1}\overline{\beta_{n-1}}\,, \tag{1d}$$
$$\delta_{\text{temp}} := \langle \widetilde{\mathbf{y}}_{\text{temp}}, \mathbf{y}_{\text{temp}} \rangle; \tag{1e}$$

if $\delta_{\text{temp}} \approx 0$, *stop (termination or breakdown); otherwise, choose* γ_n *and* δ_{n+1} *such that*

$$\gamma_n^2\delta_{n+1} = \delta_{\text{temp}}\,, \tag{1f}$$

set

$$\mathbf{y}_{n+1} := \mathbf{y}_{\text{temp}}/\gamma_n\,, \qquad \widetilde{\mathbf{y}}_{n+1} := \widetilde{\mathbf{y}}_{\text{temp}}/\overline{\gamma_n}\,, \tag{1g}$$

and proceed with the next step.

Petrov–Galerkin conditions:

$$\boxed{\widetilde{\mathcal{K}}_n \perp \mathbf{y}_n \in \mathcal{K}_{n+1}, \quad \mathcal{K}_n \perp \widetilde{\mathbf{y}}_n \in \widetilde{\mathcal{K}}_{n+1}}$$

Polynomial representation:

$$\mathbf{y}_n = p_n(\mathbf{A})\mathbf{y}_0 \,, \quad \widetilde{\mathbf{y}}_n = \overline{p_n}(\mathbf{A}^\star)\widetilde{\mathbf{y}}_0 \,,$$

where p_n is a polynomial of degree n, the nth **Lanczos polynomial**.

Termination / breakdowns:

If $\mathbf{y}_{\mathrm{temp}} \approx \mathbf{o}$ and $\widetilde{\mathbf{y}}_{\mathrm{temp}} \approx \mathbf{o}$: **full termination**

If either $\mathbf{y}_{\mathrm{temp}} \approx \mathbf{o}$ or $\widetilde{\mathbf{y}}_{\mathrm{temp}} \approx \mathbf{o}$: **one-sided termination**, **benign breakdown**

If $\delta_{\mathrm{temp}} \approx 0$ and $\mathbf{y}_{\mathrm{temp}} \not\approx \mathbf{o}$, $\widetilde{\mathbf{y}}_{\mathrm{temp}} \not\approx \mathbf{o}$: **Lanczos breakdown**, **(serious breakdown)**

Normalization:

For example, may choose γ_n such that $||\mathbf{y}_{n+1}|| = 1$.

$\gamma_n = 1$ $(\forall n)$ can cause overflow or underflow.

BiCG will require that $\gamma_n := -\alpha_n - \beta_{n-1}$.

A more general form of Algorithm 1 allows for $||\widetilde{\mathbf{y}}_{n+1}|| = ||\mathbf{y}_{n+1}|| = 1$.

For the **shorthand notation** or **matrix representation**

$$\boxed{\begin{array}{c} \mathbf{A}\mathbf{Y}_n = \mathbf{Y}_{n+1}\underline{\mathbf{T}}_n\,, \qquad \mathbf{A}^\star\widetilde{\mathbf{Y}}_n = \widetilde{\mathbf{Y}}_{n+1}\overline{\underline{\mathbf{T}}_n} \quad (n \le \bar{\nu})\,, \\ \widetilde{\mathbf{Y}}_n\mathbf{Y}_n \;=\; \mathbf{D}_{\delta;n} \quad (n \le \bar{\nu}) \end{array}}$$

we need to define matrices of right and left Lanczos vectors

$$\mathbf{Y}_n :\equiv (\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1})\,, \quad \widetilde{\mathbf{Y}}_n :\equiv (\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_{n-1})\,,$$

the extended tridiagonal matrix

$$\underline{\mathbf{T}}_n :\equiv \left[\begin{array}{c} \mathbf{T}_m \\ \hline \gamma_{n-1}\mathbf{l}_n^\top \end{array}\right] = \begin{bmatrix} \alpha_0 & \beta_0 & & & \\ \gamma_0 & \alpha_1 & \beta_1 & & \\ & \gamma_1 & \alpha_2 & \ddots & \\ & & \ddots & \ddots & \beta_{n-2} \\ & & & \gamma_{n-2} & \alpha_{n-1} \\ & & & & \gamma_{n-1} \end{bmatrix},$$

and the diagonal Gramian matrix

$$\mathbf{D}_{\delta;n} :\equiv \mathrm{diag}(\delta_0, \ldots, \delta_{n-1}) = \mathbf{Y}_n^\star\mathbf{Y}_n\,.$$

For $\underline{\mathbf{T}}_n$ to be a recurrence coefficient matrix for residuals, it needs to have vanishing column sums (consistency condition), *i.e.*,

$$\gamma_k := -\alpha_k - \beta_{k-1} \quad (0 \le k \le n-1)$$

in which case we call the matrix $\underline{\mathbf{T}}_n^\circ$.

On the other hand, if $\mathbf{y}_n$ and $\tilde{\mathbf{y}}_n$ are scaled independently, $\overline{\underline{\mathbf{T}}_n}$ is replaced by diagonally scaled version $\underline{\widetilde{\mathbf{T}}}_n$.

A first version, BIORES, of the **biconjugate gradient** (BICG) **method** for solving $\mathbf{A}\mathbf{x} = \mathbf{b}$:

Choose $\mathbf{x}_0$, $\widetilde{\mathbf{y}}_0$; set $\mathbf{y}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$.

In addition to the biorthogonal Lanczos vectors

$$\mathbf{y}_n = p_n(\mathbf{A})\mathbf{y}_0 \quad \in \mathcal{K}_{n+1}\,, \qquad \widetilde{\mathbf{y}}_n = \overline{p_n}(\mathbf{A}^\star)\widetilde{\mathbf{y}}_0 \quad \in \widetilde{\mathcal{K}}_{n+1}\,,$$

compute **iterates** $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n$ such that $\widetilde{\mathcal{K}}_n \perp \mathbf{y}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n\,.$

$\Longrightarrow$ **Lanczos polynomials = residual polynomials**; $p_n(0) = 1$.

ALGORITHM 2 (BIORES FORM OF THE BICG METHOD). .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$, *choose an initial approximation* $\mathbf{x}_0$,
set $\mathbf{y}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$, *and choose* $\widetilde{\mathbf{y}}_0$ *such that* $\delta_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{y}_0 \rangle \neq 0$.
Then apply the BIO *algorithm with*

$$\gamma_n := -\alpha_n - \beta_{n-1}\,, \tag{2a}$$

so that (1c)–(1g) simplify to

$$\begin{aligned}
\mathbf{y}_{n+1} &:= (\mathbf{A}\mathbf{y}_n - \mathbf{y}_n\alpha_n - \mathbf{y}_{n-1}\beta_{n-1})/\gamma_n\,, && \text{(2b)}\\
\widetilde{\mathbf{y}}_{n+1} &:= (\mathbf{A}^\star\widetilde{\mathbf{y}}_n - \widetilde{\mathbf{y}}_n\overline{\alpha_n} - \widetilde{\mathbf{y}}_{n-1}\overline{\beta_{n-1}})/\overline{\gamma_n}\,, && \text{(2c)}\\
\delta_{n+1} &:= \langle \widetilde{\mathbf{y}}_{n+1}, \mathbf{y}_{n+1} \rangle. && \text{(2d)}
\end{aligned}$$

Additionally, compute the vectors

$$\mathbf{x}_{n+1} := -(\mathbf{y}_n + \mathbf{x}_n\alpha_n + \mathbf{x}_{n-1}\beta_{n-1})/\gamma_n\,. \tag{2e}$$

If $\gamma_n \approx 0$, *the algorithm breaks down ("pivot breakdown").*
If $\mathbf{y}_{n+1} \approx \mathbf{o}$, *it terminates and* $\mathbf{x}_{n+1}$ *is the solution.*
If $\mathbf{y}_{n+1} \not\approx \mathbf{o}$, *but* $\delta_{n+1} \approx 0$, *it also breaks down ("Lanczos breakdown").*

Like the (standard) OMIN form of CG, the (standard) BIOMIN form of the BICG method makes use of direction vectors [Lanczos **'52**$_{\text{JResNBS}}$, Fletcher *Proc. Dundee '75*/**'76**] .

In addition to the **biorthogonal Lanczos vectors**

$$\mathbf{y}_n = p_n(\mathbf{A})\mathbf{y}_0 \quad \in \mathcal{K}_{n+1}\,, \qquad \widetilde{\mathbf{y}}_n = \overline{p_n}(\mathbf{A}^\star)\widetilde{\mathbf{y}}_0 \quad \in \widetilde{\mathcal{K}}_{n+1}\,,$$

satisfying

$$\boxed{\langle \widetilde{\mathbf{y}}_m, \mathbf{y}_n \rangle = \begin{cases} 0\,, & m \neq n, \\ \delta_n\,, & m = n, \end{cases}}$$

generate **biconjugate direction vectors**

$$\mathbf{v}_n = \widehat{p}_n(\mathbf{A})\mathbf{y}_0 \quad \in \mathcal{K}_{n+1}\,, \qquad \widetilde{\mathbf{v}}_n = \overline{\widehat{p}_n}(\mathbf{A}^\star)\widetilde{\mathbf{y}}_0 \quad \in \widetilde{\mathcal{K}}_{n+1}\,,$$

satisfying

$$\boxed{\langle \widetilde{\mathbf{v}}_m, \mathbf{A}\mathbf{v}_n \rangle = \begin{cases} 0\,, & m \neq n, \\ \delta'_n = \delta_n \varphi_n\,, & m = n, \end{cases}}$$

where $\delta'_n \neq 0$ for $0 \leq n \leq \dot{\nu} - 1\,;\ \ \dot{\nu} \leq \nu$.

Additionally, compute iterates $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n$ such that $\mathbf{y}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n$.

$\Longrightarrow$ **Lanczos polynomials = residual polynomials**; $p_n(0) = 1$.

$\widehat{p}_n$ is the nth **direction polynomial**.

ALGORITHM 3 (BIOMIN FORM OF THE BICG METHOD). .
For solving $\mathbf{Ax} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x}_0$*, set* $\mathbf{v}_0 := \mathbf{y}_0 := \mathbf{b} - \mathbf{Ax}_0$*, and choose* $\widetilde{\mathbf{y}}_0$ *such that* $\delta_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{y}_0 \rangle \neq 0$ *and* $\delta'_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{Av}_0 \rangle \neq 0$.
Then let $\tilde{\mathbf{v}}_0 := \tilde{\mathbf{y}}_0$ *and apply the* BIOC *algorithm with* $\gamma_n := -\varphi_n$*, so that after substituting* $\omega_n := 1/\varphi_n$ *the following is computed in the* n*th step:*

$$\omega_n := \delta_n / \delta'_n \,, \tag{3a}$$

$$\mathbf{y}_{n+1} := \mathbf{y}_n - \mathbf{Av}_n \omega_n \,, \tag{3b}$$

$$\widetilde{\mathbf{y}}_{n+1} := \widetilde{\mathbf{y}}_n - \mathbf{A}^\star \widetilde{\mathbf{v}}_n \overline{\omega_n} \,, \tag{3c}$$

$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n \,, \tag{3d}$$

$$\delta_{n+1} := \langle \widetilde{\mathbf{y}}_{n+1}, \mathbf{y}_{n+1} \rangle, \tag{3e}$$

$$\psi_n := -\delta_{n+1} / \delta_n \,, \tag{3f}$$

$$\mathbf{v}_{n+1} := \mathbf{y}_{n+1} - \mathbf{v}_n \psi_n \,, \tag{3g}$$

$$\widetilde{\mathbf{v}}_{n+1} := \widetilde{\mathbf{y}}_{n+1} - \widetilde{\mathbf{v}}_n \overline{\psi_n} \,, \tag{3h}$$

$$\delta'_{n+1} := \langle \widetilde{\mathbf{v}}_{n+1}, \mathbf{Av}_{n+1} \rangle. \tag{3i}$$

If $\mathbf{y}_{n+1} \approx \mathbf{o}$ *the process terminates and* $\mathbf{x}_{n+1}$ *is the solution; if* $\delta_{n+1} \approx 0$ *(and hence* $\psi_n \approx 0$*) or* $\delta'_{n+1} \approx 0$*, but* $\mathbf{y}_{n+1} \not\approx \mathbf{o}$*, the algorithm breaks down ("Lanczos and pivot breakdowns", respectively).*

Shorthand notation: Defining

$$\mathbf{Y}_n :\equiv (\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{n-1}), \quad \widetilde{\mathbf{Y}}_n :\equiv (\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \dots, \tilde{\mathbf{y}}_{n-1}),$$

$$\mathbf{V}_n :\equiv [\ \mathbf{v}_0 \quad \mathbf{v}_1 \quad \cdots \quad \mathbf{v}_{n-1}\ ], \quad \widetilde{\mathbf{V}}_n :\equiv [\ \tilde{\mathbf{v}}_0 \quad \tilde{\mathbf{v}}_1 \quad \cdots \quad \tilde{\mathbf{v}}_{n-1}\ ],$$

$$\underline{\mathbf{L}}_n^\circ :\equiv \begin{bmatrix} \varphi_0 & & & & \\ -\varphi_0 & \varphi_1 & & & \\ & -\varphi_1 & \varphi_2 & & \\ & & \ddots & \ddots & \\ & & & -\varphi_{n-2} & \varphi_{n-1} \\ & & & & -\varphi_{n-1} \end{bmatrix}, \quad \mathbf{U}_n :\equiv \begin{bmatrix} 1 & \psi_0 & & & \\ & 1 & \psi_1 & & \\ & & 1 & \ddots & \\ & & & \ddots & \psi_{n-2} \\ & & & & 1 \end{bmatrix},$$

we can write the recurrences as

$$\boxed{\mathbf{Y}_n = \mathbf{V}_n \mathbf{U}_n\,, \quad \widetilde{\mathbf{Y}}_n = \widetilde{\mathbf{V}}_n \overline{\mathbf{U}_n} \quad (n \le \dot{\nu})}$$

and

$$\boxed{\mathbf{A}\mathbf{V}_n = \mathbf{Y}_{n+1} \underline{\mathbf{L}}_n^\circ\,, \quad \mathbf{A}^\star \widetilde{\mathbf{V}}_n = \widetilde{\mathbf{Y}}_{n+1} \overline{\underline{\mathbf{L}}_n^\circ} \quad (n \le \dot{\nu}).}$$

Connection to three-term Lanczos process:

$$\boxed{\underline{\mathbf{T}}_n^\circ = \underline{\mathbf{L}}_n^\circ \mathbf{U}_n\,.}$$

Biorthogonality and biconjugacy:

$$\boxed{\widetilde{\mathbf{Y}}_n \mathbf{Y}_n = \mathbf{D}_{\delta;n} \quad \widetilde{\mathbf{V}}_n \mathbf{A} \mathbf{V}_n = \mathbf{D}_{\delta';n}\,.}$$

Summary of BICG **features:**

1. **short recurrences** [$\rightsquigarrow$ low cost, no restarts]
2. typically **fast convergence** in practice

Summary of BICG **problems:**

1. Lanczos and pivot **breakdowns** possible
2. **two matrix-vector products** per relevant Krylov space dimension
3. nonmonotone, often **erratic convergence**
4. **inaccurate Lanczos vectors and coefficients** due to roundoff
5. **loss of inherited biorthogonality** due to roundoff effects
6. **gap between recursive and true residual**

Summary of possible improvements:

1. **look-ahead Lanczos**
2. **(bi)conjugate gradient squared** (BICGS) **method, Lanczos-type product methods (LTPMs)**
3. **residual smoothing processes** (MR smoothing, QMR smoothing, QMR method, BICGSTAB family)
4. **local rebiorthogonalization**
5. **partial rebiorthogonalization** ("maintaining duality")
6. **Neumaier trick** and its improved variations

Observation: Exact breakdown causes division by zero;
near-breakdowns may cause slowdown of convergence.
Aims: Eliminate exact and near-breakdowns;
Remedy: Look-ahead (LA)

Look-Ahead for Three-Term Lanczos:

The Lanczos BIO algorithm breaks down at $n = \nu$ when

$$\boxed{\delta_\nu := \langle \widetilde{\mathbf{y}}_\nu, \mathbf{y}_\nu \rangle \approx 0 \quad \text{but} \quad \widetilde{\mathbf{y}}_\nu \not\approx \mathbf{o}, \;\; \mathbf{y}_\nu \not\approx \mathbf{o}.}$$

Look-ahead Lanczos overcomes a breakdown if curable, i.e., if $\langle \widetilde{\mathbf{y}}_\nu, \mathbf{A}^k \mathbf{y}_\nu \rangle \not\approx 0$ for some $k > 0$.

Biorthogonality is replaced by **block biorthogonality**:

$$\boxed{\widetilde{\mathbf{y}}_n \perp \mathcal{K}_{n_j}, \quad \widetilde{\mathcal{K}}_{n_j} \perp \mathbf{y}_n \quad \text{if } n \geq n_j \ (j = 0, \ldots, J),}$$

where $0 = n_0 < n_1 < \cdots < n_J = \nu \leq N$ is the index subsequence of **regular** Lanczos vectors y_{n_j}.
For the other indices the vectors are **inner** vectors.

The Gramian matrix $\mathbf{D} :\equiv (\langle \widetilde{\mathbf{y}}_m, \mathbf{y}_n \rangle)$ becomes **block diagonal** instead of diagonal.

The tridiagonal matrix $\mathbf{T} :\equiv \mathbf{D}^{-1} (\langle \widetilde{\mathbf{y}}_m, \mathbf{A}\mathbf{y}_n \rangle)$ becomes **Hessenberg and block tridiagonal**,
i.e., the recurrences become “block three-term” recurrences.

Simplification: the “older” block can be replaced by a single vector.

In practice, only few look-ahead steps are needed; hence, the recurrences are mostly just three-term.

- Parlett/Taylor/Liu [*'81*/**'85**$_{\mathrm{MC}}$]: *LA 3-term Lanczos for EVals*
- G. [*'89*/**'92**$_{\mathrm{SIMAX}}$, *'90*/**'94**$_{\mathrm{SIMAX}}$]: *LA 3-term & 2-term, exact & near-breakdowns, for systems of linear equations*
- Joubert [**'90**$_{\mathrm{Diss}}$], Boley/Elhay/Golub/G. [*'90*/**'91**$_{\mathrm{NumAlg}}$]
- Freund/G./Nachtigal [*'91*/**'93**$_{\mathrm{SISC}}$]: *details of sophisticated implementation of LA 3-term Lanczos*
- Brezinski/Redivo Zaglia/Sadok [*'91*/**'93**$_{\mathrm{NM}}$ + many more]: ($\approx$ *same as* LA-BIODIR *of* G. [*'90*/**'94**$_{\mathrm{SIMAX}}$] *when exact bd., but different when near-bd.*)
- Freund/Nachtigal [*'92*/**'94**$_{\mathrm{SISC}}$]: *sophisticated implementation of LA 2-term Lanczos*
- Ye [*'92*/**'94**$_{\mathrm{MC}}$]: **T** *of increasing bandwidth*
- Brezinski/Redivo Zaglia [*'94*/**'95**$_{\mathrm{BIT}}$]: LA-BICGSTAB (*complicated, considerable overhead*)
- Cao [*'95*]: LA-BICGSTAB *for exact breakdowns*
- G./Ressel [*'96*]: *LA for LPTMs (incl.* BICGSTAB *family)*
- Hochbruck [**'96**$_{\mathrm{Habil}}$]: *LA simplified in case of successive blocks*

Note: Still some open questions:
How can we improve accuracy of look-ahead steps?
Accuracy vs. overhead decisions.
When do we need look-ahead, i.e., what is a near-breakdown?

Aims: Faster and smoother convergence; avoiding use of $\mathbf{A}^\star$.
Ideas: Squaring of residual polynomials, or using products with 2nd set determined by local minimization of residual.

- (Bi)CGS [Sonneveld *'84*/**'89**$_{\text{SISSC}}$] $(2, 2\times 2)$ p_n^2
- BiCGStab [Van der Vorst *'91*/**'92**$_{\text{SISSC}}$]
  1-dim. minimization every step $(2, 2\times 2)$ $p_n t_n$
- BiCGStab2 [Gutknecht *'91*/**'93**$_{\text{SISC}}$]
  2-dim. minimization every other step $(3, 2\times 2)$ $p_n t_n$
- BiCGStab(ℓ) [Sleijpen/Fokkema *'93*/**'93**$_{\text{ETNA}}$, Sleijpen/Van der Vorst/Fokkema *'93*/**'94**$_{\text{NumAlg}}$]
  ℓ-dim. minimization every ℓth step $(\ell, 2\times 2)$ $p_n t_n$
- CGS2 [Fokkema/Sleijpen/Van der Vorst *'94*/**'96**$_{\text{JCAM}}$]
  $(2, 2\times 2)$ $p_n \tilde{p}_n$
- shifted CGS [Fok./Sle./VdV *'94*/**'96**$_{\text{JCAM}}$]
  $(2, 2\times 2)$ $(1-\gamma\zeta)p_n p_{n-1}$
- BiCG×MR2 [G., *OW, Apr. '94*, Cao **'97**$_{\text{IJNME}}$]
  2-dim. minimization at **every** *step* $(3, 2\times 2)$ $p_n t_n$
- GPBI-CG [Zhang *'93*/**'97**$_{\text{SISC}}$]
  2-dim. minimization at **every** *step* $(2\times 2, 2\times 2)$ $p_n t_n$

Lanczos-type product methods (LPTMs) are characterized by residual polynomials of the form

$$\boxed{p_n(\zeta)t_n(\zeta)}$$

Residual polynomials: $\boxed{p_n^2(\zeta)}$

Vector sequences:

$$\mathbf{r}_n :\equiv p_n^2(\mathbf{A})\mathbf{r}_0 \quad \in \mathcal{K}_{2n+1}, \qquad \mathbf{s}_n :\equiv p_n(\mathbf{A})\widehat{p}_n(\mathbf{A})\mathbf{r}_0 \quad \in \mathcal{K}_{2n+1},$$
$$\mathbf{s}'_n :\equiv p_{n+1}(\mathbf{A})\widehat{p}_n(\mathbf{A})\mathbf{r}_0 \quad \in \mathcal{K}_{2n+2}, \qquad \widehat{\mathbf{r}}_n :\equiv \widehat{p}_n^2(\mathbf{A})\mathbf{r}_0 \quad \in \mathcal{K}_{2n+1},$$

and iterates $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_{2n}$ such that $\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n$.

Note that

$$\delta_n = \langle \widetilde{\mathbf{y}}_n, \mathbf{y}_n \rangle = \langle \overline{p_n}(\mathbf{A}^\star)\widetilde{\mathbf{y}}_0, p_n(\mathbf{A})\mathbf{r}_0 \rangle = \langle \widetilde{\mathbf{y}}_0, p_n^2(\mathbf{A})\mathbf{r}_0 \rangle = \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_n \rangle,$$
$$\delta'_n = \langle \widetilde{\mathbf{v}}_n, \mathbf{A}\mathbf{v}_n \rangle = \langle \overline{\widehat{p}_n}(\mathbf{A}^\star)\widetilde{\mathbf{v}}_0, \mathbf{A}\widehat{p}_n(\mathbf{A})\mathbf{r}_0 \rangle = \langle \widetilde{\mathbf{y}}_0, \widehat{p}_n^2(\mathbf{A})\mathbf{A}\mathbf{r}_0 \rangle = \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_n \rangle.$$

ALGORITHM 4 (BIOMINS FORM OF THE (BI)CGS METHOD). .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x}_0 \in \mathbb{C}^N$ *and set* $\mathbf{s}_0 := \mathbf{r}_0 := \widehat{\mathbf{r}}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
Choose $\widetilde{\mathbf{y}}_0 \in \mathbb{C}^N$ *such that* $\delta_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_0 \rangle \neq 0$ *and* $\delta'_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_0 \rangle \neq 0$.
Then, for $n = 0, 1, \ldots$, *compute*

$$\omega_n := \delta_n / \delta'_n, \tag{4a}$$
$$\mathbf{s}'_n := \mathbf{s}_n - \mathbf{A}\widehat{\mathbf{r}}_n\omega_n, \tag{4b}$$
$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}(\mathbf{s}_n + \mathbf{s}'_n)\,\omega_n, \tag{4c}$$
$$\mathbf{x}_{n+1} := \mathbf{x}_n + (\mathbf{s}_n + \mathbf{s}'_n)\,\omega_n, \tag{4d}$$
$$\delta_{n+1} := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_{n+1} \rangle, \tag{4e}$$
$$\psi_n := -\delta_{n+1} / \delta_n, \tag{4f}$$
$$\mathbf{s}_{n+1} := \mathbf{r}_{n+1} - \mathbf{s}'_n\psi_n, \tag{4g}$$
$$\widehat{\mathbf{r}}_{n+1} := \mathbf{s}_{n+1} - \mathbf{s}'_n\psi_n + \widehat{\mathbf{r}}_n\psi_{n+1}^2, \tag{4h}$$
$$\delta'_{n+1} := \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_{n+1} \rangle. \tag{4i}$$

If $\mathbf{r}_{n+1} \approx \mathbf{o}$, *the process terminates and* $\mathbf{x}_{n+1}$ *is the solution.*
If $\mathbf{r}_{n+1} \not\approx \mathbf{o}$ *but* $\delta_{n+1} \approx 0$ *or* $\delta'_{n+1} \approx 0$, *the algorithm breaks down.*

Residual polynomials: $\boxed{p_n(\zeta)t_n(\zeta)}$, where

$$\boxed{t_{n+1}(\zeta) = (1 - \chi_{n+1}\zeta)t_n(\zeta)\,.}$$

Vector sequences:

$\mathbf{r}_n :\equiv p_n(\mathbf{A})t_n(\mathbf{A})\mathbf{r}_0\,,\quad \widehat{\mathbf{r}}_n :\equiv \widehat{p}_n(\mathbf{A})t_n(\mathbf{A})\mathbf{r}_0\,,\quad \mathbf{w}_n :\equiv p_{n+1}(\mathbf{A})t_n(\mathbf{A})\mathbf{r}_0\,,$

and iterates $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n$ such that $\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n\,.$

ALGORITHM 5 (BICGSTAB ALGORITHM). .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x} \in \mathbb{C}^N$ *and set* $\mathbf{r}_0 := \widehat{\mathbf{r}}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$. *Choose* $\widetilde{\mathbf{y}}_0 \in \mathbb{C}^N$ *such that* $\tilde{\delta}_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_0 \rangle \neq 0$ *and* $\tilde{\delta}'_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_0 \rangle \neq 0$. *Then compute, for* $n = 0, 1, \ldots,$

$$\begin{array}{llcl}
\text{(5a)} & \omega_n & := & \tilde{\delta}_n / \tilde{\delta}'_n\,, \\
\text{(5b)} & \mathbf{w}_n & := & \mathbf{r}_n - \mathbf{A}\widehat{\mathbf{r}}_n \omega_n\,, \\
\text{(5c)} & \chi_{n+1} & := & \langle \mathbf{A}\mathbf{w}_n, \mathbf{w}_n \rangle / ||\mathbf{A}\mathbf{w}_n||^2, \\
\text{(5d)} & \mathbf{r}_{n+1} & := & \mathbf{w}_n - \mathbf{A}\mathbf{w}_n \chi_{n+1}\,, \\
\text{(5e)} & \mathbf{x}_{n+1} & := & \mathbf{x}_n + \widehat{\mathbf{r}}_n \omega_n + \mathbf{w}_n \chi_{n+1}\,, \\
\text{(5f)} & \tilde{\delta}_{n+1} & := & \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_{n+1} \rangle, \\
\text{(5g)} & \psi_n & := & -\,\tilde{\delta}_{n+1} / (\tilde{\delta}'_n \chi_{n+1})\,, \\
\text{(5h)} & \widehat{\mathbf{r}}_{n+1} & := & \mathbf{r}_{n+1} - (\widehat{\mathbf{r}}_n - \mathbf{A}\widehat{\mathbf{r}}_n \chi_{n+1})\,\psi_n\,, \\
\text{(5i)} & \delta'_{n+1} & := & \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_{n+1} \rangle\,.
\end{array}$$

If $\mathbf{r}_{n+1} \approx \mathbf{o}$, *the process terminates and* $\mathbf{x}_{n+1}$ *is the solution.*
If $\mathbf{r}_{n+1} \not\approx \mathbf{o}$ *but* $\tilde{\delta}_{n+1} \approx 0$ *or* $\tilde{\delta}'_{n+1} \approx 0$, *the algorithm breaks down.*

χ_{n+1} is chosen such that

$$\boxed{||\mathbf{r}_{n+1}|| = \min_{\chi}\; ||\mathbf{w}_n - \mathbf{A}\mathbf{w}_n \chi||.}$$

Residual polynomials: $\boxed{p_n(\zeta)t_n(\zeta)}$, where

in BiCGStab2

$$\boxed{t_{n+1}(\zeta) = \begin{cases} (1-\chi_{n+1}\zeta)t_n(\zeta)\,, & \text{if } n \text{ is even} \\ (\xi_n + \eta_n\zeta)\,t_n(\zeta) + (1-\xi_n)\,t_{n-1}(\zeta)\,, & \text{if } n \text{ is odd} \end{cases}}$$

in BiCG×MR2

$$\boxed{t_{n+1}(\zeta) = (\xi_n + \eta_n\zeta)\,t_n(\zeta) + (1-\xi_n)\,t_{n-1}(\zeta)}$$

Again, η_n and ξ_n are chosen to make $||\mathbf{r}_{n+1}||$ as small as as possible.

BiCGStab2 enforces a two-dimensional local residual minimization in every other step. (More flexible versions can be defined easily.)

BiCG×MR2 enforces a two-dimensional local residual minimization in every step.

Algorithm 6. *(BiCGStab2 and BiCG×MR2)* .
For solving $\mathbf{Ax} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x}_0 \in \mathbb{C}^N$ *and set* $\mathbf{r}_0 := \widehat{\mathbf{r}}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
Choose $\widetilde{\mathbf{y}}_0 \in \mathbb{C}^N$ *such that* $\tilde{\delta}_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_0 \rangle \neq 0$ *and* $\tilde{\delta}'_0 := \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_0 \rangle \neq 0$.
Then, for $n = 0, 1, \ldots,$ *compute*

$$\omega_n := \tilde{\delta}_n / \tilde{\delta}'_n, \tag{6a}$$

$$\mathbf{w}'_n := \mathbf{w}_{n-1} - \mathbf{A}\mathbf{t}_n \omega_n \quad (\textit{if } n \geq 1), \tag{6b}$$

$$\mathbf{w}_n := \mathbf{r}_n - \mathbf{A}\widehat{\mathbf{r}}_n \omega_n; \tag{6c}$$

if n is even (BiCGStab2) or $n = 1$ (BiCG×MR2), compute χ_n *and let*

$$\mathbf{r}_{n+1} := \mathbf{w}_n - \mathbf{A}\mathbf{w}_n \chi_n, \tag{6d}$$

$$\mathbf{x}_{n+1} := \mathbf{x}_n + \widehat{\mathbf{r}}_n \omega_n + \mathbf{w}_n \chi_n \ , \tag{6e}$$

$$\tilde{\delta}_{n+1} := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_{n+1} \rangle, \tag{6f}$$

$$\psi_n := -\,\omega_n \tilde{\delta}_{n+1} / (\tilde{\delta}_n \chi_n), \tag{6g}$$

$$\widehat{\mathbf{r}}_{n+1} := \mathbf{r}_{n+1} - (\widehat{\mathbf{r}}_n - \mathbf{A}\widehat{\mathbf{r}}_n \chi_n)\, \psi_n, \tag{6h}$$

else compute ξ_n *and* η_n *and let*

$$\mathbf{r}_{n+1} := \mathbf{w}'_n (1 - \xi_n) + \mathbf{w}_n \xi_n + \mathbf{A}\mathbf{w}_n \eta_n, \tag{6i}$$

$$\mathbf{x}_{n+1} := [\mathbf{x}_{n-1} + \widehat{\mathbf{r}}_{n-1}\omega_{n-1} + \mathbf{t}_n \omega_n](1 - \xi_n) \tag{6j}$$

$$+ [\mathbf{x}_n + \widehat{\mathbf{r}}_n \omega_n]\xi_n - \mathbf{w}_n \eta_n, \tag{6k}$$

$$\tilde{\delta}_{n+1} := \langle \widetilde{\mathbf{y}}_0, \mathbf{r}_{n+1} \rangle, \tag{6l}$$

$$\psi_n := -\tilde{\delta}_{n+1} / (\tilde{\delta}'_n \eta_n), \tag{6m}$$

$$\widehat{\mathbf{r}}_{n+1} := \mathbf{r}_{n+1} - [\mathbf{t}_n (1 - \xi_n) + \widehat{\mathbf{r}}_n \xi_n + \mathbf{A}\widehat{\mathbf{r}}_n \eta_n]\psi_n \tag{6n}$$

endif;

$$\mathbf{t}_{n+1} := \mathbf{w}_n - \widehat{\mathbf{r}}_n \psi_n, \tag{6o}$$

$$\mathbf{A}\mathbf{t}_{n+1} := \mathbf{A}\mathbf{w}_n - \mathbf{A}\widehat{\mathbf{r}}_n \psi_n, \tag{6p}$$

$$\tilde{\delta}'_{n+1} := \langle \widetilde{\mathbf{y}}_0, \mathbf{A}\widehat{\mathbf{r}}_{n+1} \rangle \, . \tag{6q}$$

If $\mathbf{r}_{n+1} \approx \mathbf{o}$*, the process terminates and* $\mathbf{x}_{n+1}$ *is the solution;*
if $\mathbf{r}_{n+1} \not\approx \mathbf{o}$ *but* $\tilde{\delta}_{n+1} \approx 0$ *or* $\tilde{\delta}'_{n+1} \approx 0$*, the algorithm breaks down.*

Since $\mathbf{x}_n - \mathbf{x}_0 \in \mathcal{K}_n$, we can represent it in terms of Lanczos vectors:

$$\boxed{\mathbf{x}_n = \mathbf{x}_0 + \mathbf{Y}_n \mathbf{k}_n , \quad \mathbf{r}_n = \mathbf{r}_0 - \mathbf{A}\mathbf{Y}_n \mathbf{k}_n ,} \tag{7}$$

with some coordinate vector $\mathbf{k}_n$.

Using $\mathbf{A}\mathbf{Y}_n = \mathbf{Y}_{n+1}\underline{\mathbf{T}}_n$ and $\mathbf{r}_0 = \mathbf{y}_0\rho_0 = \mathbf{Y}_{n+1}\underline{\mathbf{e}}_1\rho_0$, we see that

$$\boxed{\mathbf{r}_n = \mathbf{Y}_{n+1} \underbrace{(\underline{\mathbf{e}}_1\rho_0 - \underline{\mathbf{T}}_n\mathbf{k}_n)}_{\textbf{quasi-residual } \mathbf{q}_n} .} \tag{8}$$

- Instead of $\mathbf{r}_n$ the QMR method minimizes $\mathbf{q}_n$.
  Since the columns of $\mathbf{Y}_{n+1}$ are not orthonormal
  (*i.e.*, coordinate map no isometry), this is not the same.
- Minimal $\mathbf{q}_n$ is found by updating QR-decomposition of $\underline{\mathbf{T}}_n$ (like in MINRES).
- The BICG residuals are not used; but the Lanczos vectors are multiples of them, and the scalar factors are easy to update ($\rightsquigarrow$ inconsistent BIORES).
- Even the QMR residuals are not needed as long as convergence is judged by the norm the of quasi-residuals.
- There are short recurrences for updating $\mathbf{x}_n$, but they lead to reduction of the ultimate accuracy.

Refs.:
Freund/Nachtigal [*'91*/**'91**$_{\mathrm{NM}}$]: *for 3-term Lanczos*
Freund/Nachtigal [*'92*/**'94**$_{\mathrm{SISC}}$]: *for 2-term Lanczos*
Freund [*'91*/**'93**$_{\mathrm{SISC}}$]: *for* (BI)CGS ($\rightsquigarrow$ TFQMR)
Chan/Gallopoulos/Simoncini/Szeto/Tong [*'92*/**'94**$_{\mathrm{SIMAX}}$]:
for BICGSTAB ($\rightsquigarrow$ QMRCGSTAB)
Ressel/G. [*'96*]: *for LTPMs*

Observation: Roundoff errors have strong effects on Lanczos-type methods (including CG) [similar to, but worse than Gram–Schmidt]:

- **inaccurate recurrence coefficients**
- **near -breakdowns**
- **loss of global (bi)orthogonality**
- **gap between true and updated residuals**

Questions: What are the effects on the performance?

- **slowdown of convergence**
- **reduced ultimate accuracy**

How can we counteract these effects?
How can we estimate these effects?

Remarks: Forward roundoff analysis hopeless!
Methods are to some extent self-correcting.

1. **Inaccurate recurrence coefficients / near-breakdowns**

   In BIORES, recurrence coefficients α_n and β_{n-1} are inaccurate if the inner products

   $$\langle \tilde{\mathbf{y}}_n, \mathbf{y}_n \rangle \quad \text{and/or} \quad \langle \tilde{\mathbf{y}}_n, \mathbf{A}\mathbf{y}_n \rangle$$

   have large relative error, *i.e.*, small absolute value.

   $\gamma_n := -\alpha_n - \beta_{n-1}$ is inaccurate if $|\gamma_n| \ll |\alpha_n|$.

   Similarly, in BIOMIN, coefficients ψ_n and ω_n are inaccurate if

   $$\delta_{n+1} := \langle \tilde{\mathbf{y}}_{n+1}, \mathbf{y}_{n+1} \rangle, \quad \text{and/or} \quad \delta'_{n+1} := \langle \tilde{\mathbf{v}}_{n+1}, \mathbf{A}\mathbf{v}_{n+1} \rangle$$

   have large relative error.

   Notes:

   An inner product of nearly orthogonal vectors is inherently prone to large relative roundoff error.

   Three of these four inner products cause near-breakdowns if very small.
   A "pivot near-breakdown" ($\gamma_n \approx 0$ or $\delta'_n \approx 0$) causes not only large local errors, but also very large $\mathbf{x}_n$ and $\mathbf{r}_n$.

   Remedies:

   - **Multiple precision** (not fashionable).
   - **Choice of left vectors**
     $\rightsquigarrow$ one-sided Lanczos algorithm,
     $\rightsquigarrow$ Lanczos-type product methods (LTPMs);
     only one-sided orthogonality is really needed.
   - **Look-ahead** (numerical effects not yet well studied).

2. **Loss of global (bi)orthogonality**

   Short recurrences are cheap and effective, but the tiny errors mentioned under 1. and those of the SAXPYs in the Gram-Schmidt process cause loss of global (bi)orthogonality:

   $$\boxed{\langle \tilde{\mathbf{y}}_m, \mathbf{y}_n \rangle \not\approx 0 \ \text{ if } \ |m-n| \ \text{ large.}}$$

   Note:

   Gram-Schmidt is inherently prone to large relative roundoff errors.

   Further remedies:

   - **Modified** Gram-Schmidt (little effect, since only 3 terms)
   - **Repeated** (classical) Gram-Schmidt (= local reorthogonalization) (little effect).
   - **Full reorthogonalization** is too expensive.
   - **Partial reorthogonalization** (= "maintaining duality") justified for eigenvalue computations (see Day [**'93**$_{\text{Diss}}$, *'95*/**'97**$_{\text{SIMAX}}$]), too cumbersome for linear equation solvers — and, impossible for the more effective LTPMs.

   Notes:

   The above effects of finite precision arithmetic concern cause a **slowdown of the convergence**.

   In the symmetric case, the consequences are well understood due to Greenbaum [**'89**$_{\text{LAA}}$]:
   Lanczos process in finite precision arithmetic behaves like one for a bigger problem in exact arithmetic.

3. **Gap between updated and true residuals**

   If residuals $\mathbf{r}_n$ are updated recursively, they differ from the 'true' residuals $\mathbf{b} - \mathbf{A}\mathbf{x}_n$.

   Gap known to be particularly large if residual norms fluctuate.

   Moreover, worse for ORES/BIORES than for OMIN/BIOMIN.

   Note:

   This gap spoils the **the ultimate accuracy of the solution**: true residuals may stagnate long before they should.

   In fact, for many algorithms the updated residuals become much smaller than the roundoff in evaluating $\mathbf{b} - \mathbf{A}\mathbf{x}_n$. Hence, the gap determines the level of stagnation of the true residual.

   Analysis:

   - Greenbaum [**'97**$_{\mathrm{SIMAX}}$]: *two-term recurrences: norm of largest iterate matters; global gap is sum of local errors.*
   - G./Strakoš ['97]: *three-term recurrences: worse; local errors may get amplified dramatically.*

   Remedies:

   - Consequent usage of true residuals instead of updated residuals leads to inacceptable loss of orthogonality and related slowdown of convergence.
   - **Occasional synchronization of true and updated residual combined with shift of origin**:
     - Neumaier [*OW, Apr. 94*]: *use shifts and true residual at every step where residual norm is reduced.*
     - Sleijpen/Van der Vorst [*'94*/**'96**$_{\mathrm{Computing}}$]: *use fewer shifts and true residuals; various strategies.*
     - Van der Vorst/Ye ['99]: *improved and simplified strategy.*

Strong nonnormality of a matrix is a source of numerical difficulties.

Eigenvectors for different eigenvalues may be close to linear dependent: some eigenspaces nearly collapse.

The spectrum may change much under small perturbation of $\mathbf{A}$.

$\mathbf{A} - \zeta\mathbf{I}$ is nearly singular in large regions of the complex plane.

$\|\mathbf{A}\|_2 \gg \rho(\mathbf{A})$ (= spectral radius)

Henrici number of $\mathbf{A}$ [Henrici **'62**$_{\mathrm{NM}}$, Chaitin-Chatelin *'94*]:

$$\mathrm{He}(\mathbf{A}) :\equiv \frac{\|\mathbf{A}\mathbf{A}^\star - \mathbf{A}^\star\mathbf{A}\|}{\|\mathbf{A}^2\|}$$

Norm dependent. For simplicity often computed with respect to Frobenius norm

$$\|\mathbf{A}\|_F :\equiv \sqrt{\sum_{j,k=1}^{N} |a_{j,k}|^2}\,.$$

ϵ-pseudospectrum of $\mathbf{A}$ [Trefethen *AlgApprII Proc*/**'90**]:

$$\begin{aligned}\sigma_\epsilon(\mathbf{A}) \quad &:\equiv \quad \{\zeta \in \mathbb{C};\ z \text{ eigenvalue of } \mathbf{A} + \delta\mathbf{A},\ \|\delta\mathbf{A}\| \leq \epsilon\|\mathbf{A}\|\} \\ &= \quad \left\{\zeta \in \mathbb{C};\ \|(\mathbf{A} - \zeta\mathbf{I})^{-1}\| \geq \frac{1}{\epsilon\|\mathbf{A}\|}\right\}.\end{aligned}$$

Spectral portrait of $\mathbf{A}$ [Gudonov *IMACS Proc*/**'92**]:

$$\zeta \mapsto \log_{10}\left(\|(\mathbf{A} - \zeta\mathbf{I})^{-1}\|\,\|\mathbf{A}\|\right)$$

- Lanczos-type methods, in particular LTPMs, are normally very efficient solvers for linear systems:
  - small memory requirement,
  - non-increasing cost per step,
  - quick convergence.
- If applied to difficult problems, Lanczos-type methods require a number of safeguards against roundoff effects in order to perform reliably.
- Though CG and BiCG have been used for decades, some accuracy issues have been studied only recently, or not at all. Regarding LTPMs there remains much to do.
- There is a need for reliable and accurate implementations on several platforms. They should feature, *e.g.*,
  - coupled 2-term versions of LTPMs, preferably with look-ahead,
  - other measures to keep local errors under control,
  - occasional synchronization of updated and true residual, along with shift of origin.
- No Krylov space method will ever be able to solve all problems.
- Even the best Krylov space method will not make preconditioning / multilevel approaches / ... obsolete.
- Issues vary if a method is used for different purposes: *e.g.*, as multigrid smoother, with multigrid as 'preconditioner', for Newton corrections, ...

References

Z.-H. Cao (1998), 'On the QMR approach for iterative methods including coupled three-term recurrences for solving nonsymmetric linear systems', *ANM* **27**, 123–140.

R. Fletcher (1976), Conjugate gradient methods for indefinite systems, in *Numerical Analysis, Dundee, 1975* (G. A. Watson, ed.), Vol. 506 of *Lecture Notes in Mathematics*, Springer, Berlin, pp. 73–89.

R. W. Freund (1993), 'A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems', *SIAM J. Sci. Comput.* **14**, 470–482. .

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

M. H. Gutknecht (1993), 'Variants of BiCGStab for matrices with complex spectrum', *SIAM J. Sci. Comput.* **14**, 1020–1033. .

M. H. Gutknecht (1994), 'Local minimum residual smoothing', Talk at Oberwolfach, Germany.

C. Lanczos (1950), 'An iteration method for the solution of the eigenvalue problem of linear differential and integral operators', *J. Res. Nat. Bureau Standards* **45**, 255–281.

C. Lanczos (1952), 'Solution of systems of linear equations by minimized iterations', *J. Res. Nat. Bureau Standards* **49**, 33–53.

G. L. G. Sleijpen and D. R. Fokkema (1993), 'BiCGstab(l) for linear equations involving unsymmetric matrices with complex spectrum', *Electronic Trans. Numer. Anal.* **1**, 11–32. .

P. Sonneveld (1989), 'CGS, a fast Lanczos-type solver for nonsymmetric linear systems', *SIAM J. Sci. Statist. Comput.* **10**, 36–52.

H. A. van der Vorst (1992), 'Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems', *SIAM J. Sci. Statist. Comput.* **13**, 631–644. .

S.-L. Zhang (1997), 'GPBI-CG: generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems', *SIAM J. Sci. Comput.* **18**(2), 537–551. .