IDR-CGS-BiCGSTAB-IDR(s)

A case of serendipity

Student Krylov Day 2015

Peter Sonneveld

febr 2-02-2015

Delft Institute of Applied Mathematics

TU Delft

Delft University of Technology

Outline

- Attempt for N-dimensional secant method.
- Accelerated Gauss-Seidel.
- First IDR.
- CGS.
- BiCGSTAB
- Zemke's mail.
- IDR(s).
- Covergence (analysis at last...)

1-dimensional secant method

Quasi-Newton method for solving $f(x) = 0$.

Constructs a sequence of approximations $\{x_n\}$.

One step ($f_n = f(x_n)$, $e_n = x_n - x$):

Solve

$$c_n f_n + d_n f_{n-1} = 0, \text{ with } c_n + d_n = 1$$

Calculate:

$$x_{n+1} = c_n x_n + d_n x_{n-1}$$

$$e_{n+1} \approx C e_n e_{n-1} \implies |e_{n+1}| \approx \widetilde{C} |e_n|^{\alpha}$$

$\alpha \approx 1.618$, largest root of $\alpha^2 - \alpha - 1 = 0$.

Secant effectively cheaper than Newton:

1. Work(Newton-step) $\approx 2\times$ Work(Secant-step)

2. $\alpha^2 \approx 2.618 > 2$

Poor Man's N-dim. Secant (*PMS*)

Simple generalization for $\mathbb{R}^N$, $N > 1$:

Suppose $\boldsymbol{x}_j, \boldsymbol{x}_{j-1}, \ldots, \boldsymbol{x}_{j-N}$, and $\boldsymbol{f}_j, \boldsymbol{f}_{j-1}, \ldots, \boldsymbol{f}_{j-N}$ are given.

Calculate $\boldsymbol{x}_{j+1}$ by

$$\sum_{k=0}^{N} c_{j,k} \boldsymbol{f}_{j-k} = \boldsymbol{0}, \ \sum_{k=0}^{N} c_{j,k} = 1 \Longrightarrow c_{j,k}$$

$$\boldsymbol{x}_{j+1} = \sum_{k=0}^{N} c_{j,k} \boldsymbol{x}_{j-k}$$

Is ill conditioned for large j. Safer variant:

$$\sum_{k=0}^{1} c_{j,k} \boldsymbol{f}_{j-k} + \sum_{k=2}^{N} c_{j,k} \boldsymbol{f}_{N-k} = \boldsymbol{0}, \ \sum_{k=0}^{N} c_{jk} = 1$$

$$\boldsymbol{x}_{j+1} = \sum_{k=0}^{1} c_{j,k} \boldsymbol{x}_{j-k} + \sum_{k=2}^{N} c_{j,k} \boldsymbol{x}_{N-k}$$

(Only last 2 vectors replaced)

For asymptotic behaviour: Try linear system $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{0}$,

Serendipity moment: experiment 1976:

n	$\|\boldsymbol{f}_n\|$	n	$\|\boldsymbol{f}_n\|$	n	$\|\boldsymbol{f}_n\|$
0	2.2017e+00	6	8.4701e-01	12	3.4924e-04
1	2.6116e+00	7	7.8169e-01	13	1.1295e-04
2	1.3207e+00	8	9.9805e+00	14	4.4870e-14
3	6.7938e-01	9	2.6692e-01	15	8.0980e-16
4	8.1994e-01	10	4.6617e-02	16	1.1736e-16
5	8.6446e-01	11	7.9480e-03		

Idealized secant method, digits=16, N=7

(*Replayed in 2006*)

Extraction of essential recursion

Residuals $\boldsymbol{f}_j$ are related by

$$\boldsymbol{f}_{j+1} = \boldsymbol{B}(\widetilde{c}_{j,0}\boldsymbol{f}_j + \widetilde{c}_{j,1}\boldsymbol{f}_{j-1})$$

, with

$$\widetilde{c}_{j,0}\boldsymbol{f}_j + \widetilde{c}_{j,1}\boldsymbol{f}_{j-1} \perp \boldsymbol{p}, \quad \widetilde{c}_{j,0} + \widetilde{c}_{j,1} = 1$$

$\boldsymbol{B}$ and $\boldsymbol{p}$ depend on $\boldsymbol{f}_0, \boldsymbol{f}_1, \ldots, \boldsymbol{f}_{N-2}$. Are fixed during the process.

Norm-drop at $j = 2N$ is generic property of recurrence.

Similarity with Picard iteration for solving $\boldsymbol{x} = \boldsymbol{B}\boldsymbol{x} + \boldsymbol{b}$:

Let $\boldsymbol{f}_j = \boldsymbol{b} - (\boldsymbol{I} - \boldsymbol{B})\boldsymbol{x}_j$. Picard iteration:

$$\boldsymbol{x}_{j+1} = \boldsymbol{B}\boldsymbol{x}_j + \boldsymbol{b}, \quad \boldsymbol{f}_{j+1} = \boldsymbol{B}\boldsymbol{f}_j.$$

Accelerated Gauss-Seidel (AGS).

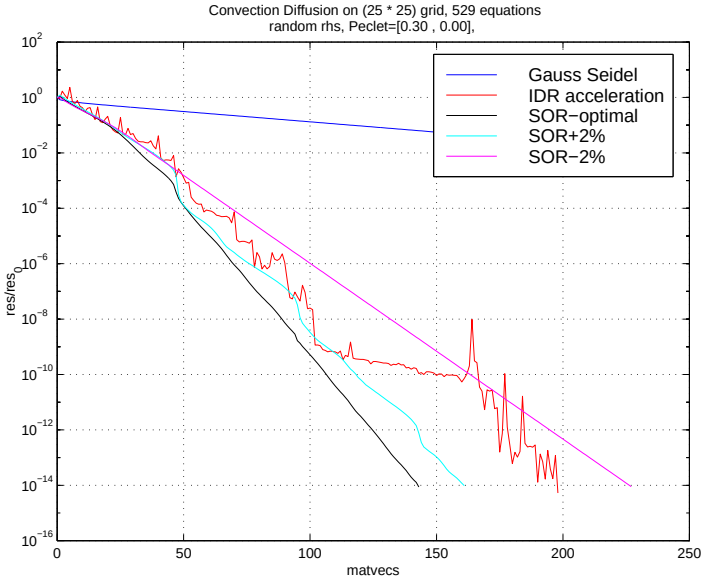
For solving $\boldsymbol{Ax} = \boldsymbol{b}$: Let $\boldsymbol{A} = \boldsymbol{D} - \boldsymbol{L} - \boldsymbol{U}$; given $\boldsymbol{x}_0$, each step solve $(\boldsymbol{D} - \boldsymbol{L})\boldsymbol{x}_{n+1} = \boldsymbol{U}\boldsymbol{x}_n + \boldsymbol{b}$, equivalent to $\boldsymbol{x}_{n+1} = \boldsymbol{B}\boldsymbol{x}_n + \widetilde{\boldsymbol{b}}$, with $\boldsymbol{B} = (\boldsymbol{D} - \boldsymbol{L})^{-1}\boldsymbol{U}$, $\widetilde{\boldsymbol{b}} = (\boldsymbol{D} - \boldsymbol{L})^{-1}\boldsymbol{b}$

New process:

1. Choose arbitrary fixed $\boldsymbol{p} \in \mathbb{R}^N$.
2. Choose $\boldsymbol{x}_0$, let $\boldsymbol{x}_1 = \boldsymbol{B}\boldsymbol{x}_0 + \widetilde{\boldsymbol{b}}$.
3. For $j = 1, 2, \ldots$
4. $\boldsymbol{s}_j = \boldsymbol{f}_j + \beta_j(\boldsymbol{f}_j - \boldsymbol{f}_{j-1})$, with β_j such that $\boldsymbol{s}_j \perp \boldsymbol{p}$
5. $\boldsymbol{x}_{j+1} = \boldsymbol{x}_j + \boldsymbol{s}_j + \beta_j(\boldsymbol{x}_j - \boldsymbol{x}_{j-1})$,
6. Then $\boldsymbol{f}_{j+1} = \boldsymbol{B}\boldsymbol{s}_j = \boldsymbol{B}(\boldsymbol{f}_j + \beta_j(\boldsymbol{f}_j - \boldsymbol{f}_{j-1})$

Unexpected property

Compare AGS with classic competitors:



Work: $\mathrm{SOR}(\omega_{\mathrm{opt}}) \leq \mathrm{SOR}(\omega_{\mathrm{opt}} + 2\%) \leq \mathrm{AGS} \leq \mathrm{SOR}(\omega_{\mathrm{opt}} - 2\%) \lll \mathrm{GS}$

AGS can compete with SOR, ω_{opt} slightly to high.

Determination of ω_{opt} is increasingly difficult at growing size!

Question: Why does it converge so much faster than $2N = 1058$ steps?

First: Why is this solver finite?!

- Consider the sequence $\{\boldsymbol{f}_0, \boldsymbol{f}_1, \ldots\}$, with $\boldsymbol{f}_1 = \boldsymbol{B}\boldsymbol{f}_0$, and

$$\boldsymbol{f}_{j+1} = \boldsymbol{B}[\boldsymbol{f}_j - \beta_j(\boldsymbol{f}_j - \boldsymbol{f}_{j-1})]$$

  with β_j chosen such that $\boldsymbol{f}_j - \beta_j(\boldsymbol{f}_j - \boldsymbol{f}_{j-1}) \perp \boldsymbol{p}$.

- $j \geq 2$, all $\boldsymbol{f}_j$ in $\boldsymbol{B}(\boldsymbol{p}^\perp)$.
- $j \geq 4$, $\boldsymbol{f}_{j-1} - \beta_{j-1}(\boldsymbol{f}_{j-1} - \boldsymbol{f}_{j-2})$ in $\boldsymbol{p}^\perp \cap \boldsymbol{B}(\boldsymbol{p}^\perp)$.
  $\Longrightarrow$ $j \geq 4$: $\boldsymbol{f}_j \in \boldsymbol{B}(\boldsymbol{p}^\perp \cap [\boldsymbol{B}(\boldsymbol{p}^\perp)]) \subset \boldsymbol{B}(\boldsymbol{p}^\perp)$.
- And so on and on and on.
- Define: $\mathcal{G}_0 = \mathbb{R}^N$, $\mathcal{G}_{j+1} = \boldsymbol{B}(\mathcal{G}_j \cap \boldsymbol{p}^\perp)$, then $\boldsymbol{f}_{i \geq 2j} \in \mathcal{G}_j$. $\mathcal{G}$-spaces form a *nest*: $\mathcal{G}_{j+1} \subset \mathcal{G}_j$.

IDR-theorem (1980)

A slightly more general explanation for the norm-drop property:

Theorem 1 (IDR) *Let $\boldsymbol{A}$ be any matrix in $\mathbb{R}^{N\times N}$,*
let $\boldsymbol{v}_0$ be any nonzero vector in $\mathbb{R}^N$,
let $\mathcal{G}_0$ be the full Krylov space $\mathcal{K}^N(\boldsymbol{A}, \boldsymbol{v}_0)$,
let $\mathcal{S}$ denote any (proper) subspace of $\mathbb{R}^N$, and let the sequence
$\mathcal{G}_j$, for $j = 1, 2, \ldots$ be defined by

$$\mathcal{G}_j = (\boldsymbol{I} - \omega_j \boldsymbol{A})(\mathcal{G}_{j-1} \cap \mathcal{S})$$

where ω_j are nonzero numbers. Then
i: $\mathcal{G}_j \subset \mathcal{G}_{j-1}$, ii: $M \leq N$ exists such that
$\mathcal{G}_j = \mathcal{G}_M, \ j = M+1, M+2, \ldots$

For 'almost all' $\mathcal{S}$: $\mathcal{G}_M = \{\boldsymbol{0}\}$.

From theorem to algorithm.

- Choose initial estimate $\boldsymbol{x}_0$, and a suitable vector $\boldsymbol{p} \in \mathbb{R}^N$. Let $\mathcal{S} = \boldsymbol{p}^{\perp}$.
- Start residual $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{x}_1 = \boldsymbol{x}_0 - \boldsymbol{r}_0$, $\boldsymbol{r}_1 = \boldsymbol{r}_0 + \boldsymbol{A}\boldsymbol{r}_0$. Both $\boldsymbol{r}_0$ and $\boldsymbol{r}_1$ are in $\mathcal{G}_0 \ \left(= \mathcal{K}^N(\boldsymbol{A}, \boldsymbol{r}_0)\right)$.
- Assume $\boldsymbol{r}_{n-1}$ and $\boldsymbol{r}_n$ both in $\mathcal{G}_j$. Make $\boldsymbol{s}_n = \boldsymbol{r}_n - \beta(\boldsymbol{r}_n - \boldsymbol{r}_{n-1}) \ \perp \boldsymbol{p}$. Then $\boldsymbol{s}_n \in \mathcal{S} \cap \mathcal{G}_j$
- Then $\boldsymbol{r}_{n+1} = (\boldsymbol{I} - \omega_j \boldsymbol{A})\boldsymbol{s}_n$ is in $\mathcal{G}_{j+1}$
- Before calculating the first residual in $\mathcal{G}_{j+1}$, ω_j may be chosen freely. Mostly used to minimize $\|\boldsymbol{r}_{n+1}\|$
- $\boldsymbol{r}_{2j}$ and $\boldsymbol{r}_{2j+1}$ are in $\mathcal{G}_j$.

Convergence: Analyse IDR-*polynomials*.

- First element in $\mathcal{G}_j$, $\boldsymbol{r}_{2j}$, satisfies $\boldsymbol{r}_{2j} = \Phi_{2j}(\boldsymbol{A})\boldsymbol{r}_0$.
- From the recursions follows $\Phi_{2j}(t) = \Omega_j(t)\phi_j(t)$, with $\Omega_j(t) = (1-\omega_j t)(1-\omega_{j-1}t)\cdots(1-\omega_1 t)$
- From the intersections with $\boldsymbol{p}^{\perp}$ follows for $l < j$: $\boldsymbol{p}^T\Omega_l(\boldsymbol{A})\phi_j(\boldsymbol{A})\boldsymbol{r}_0 = 0, \Longrightarrow \Omega_l(\boldsymbol{A}^T)\boldsymbol{p} \perp \phi_j(\boldsymbol{A})\boldsymbol{r}_0$.
- ϕ_j is the j-th BiCG - polynomial!
- Obtained without calculating $\boldsymbol{A}^T$ products.

(Bi)-CG algorithm

Regular steps in (Bi-)CG algorithm:

$$\rho_n = \widetilde{\boldsymbol{r}}_n^T \boldsymbol{r}_n, \;\; \beta_n = \rho_n / \rho_{n-1}$$

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_n \boldsymbol{p}_{n-1}, \;\; \boldsymbol{q}_n = \boldsymbol{A}\boldsymbol{p}_n;$$

$$\widetilde{\boldsymbol{p}}_n = \widetilde{\boldsymbol{r}}_n + \beta_n \widetilde{\boldsymbol{p}}_{n-1}, \;\; \widetilde{\boldsymbol{q}}_n = \boldsymbol{A}^T \widetilde{\boldsymbol{p}}_n$$

$$\sigma_n = \widetilde{\boldsymbol{p}}_n^T \boldsymbol{q}_n, \;\; \alpha_n = \rho_n / \sigma_n$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n - \alpha_n \boldsymbol{q}_n;$$

$$\widetilde{\boldsymbol{r}}_{n+1} = \widetilde{\boldsymbol{r}}_n - \alpha_n \widetilde{\boldsymbol{q}}_n$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n \boldsymbol{p}_n$$

The polynomial relations

- The relevant vectors satisfy:

$$\boldsymbol{r}_n = \varphi_n(\boldsymbol{A})\boldsymbol{r}_0,\ \boldsymbol{p}_n = \psi_n(\boldsymbol{A})\boldsymbol{r}_0$$

$$\widetilde{\boldsymbol{r}}_n = \varphi_n(\boldsymbol{A}^T)\widetilde{\boldsymbol{r}}_0,\ \widetilde{\boldsymbol{p}}_n = \psi_n(\boldsymbol{A}^T)\widetilde{\boldsymbol{r}}_0$$

  where φ_n and ψ_n are polynomials of degree n.

- Define 'inner product' between polynomials:

$$\langle \phi_1, \phi_2 \rangle = \boldsymbol{r}_0^T \phi_1(\boldsymbol{A})\phi_2(\boldsymbol{A})\boldsymbol{r}_0 = [\phi_1(\boldsymbol{A}^T)\boldsymbol{r}_0]^T \phi_2(\boldsymbol{A})\boldsymbol{r}_0$$

- then bi-orthogonality between $\boldsymbol{r}_n$ and $\widetilde{\boldsymbol{r}}_k$ corresponds to formal orthogonality of φ_n and φ_k.

‘CG’ for orthogonal polynomials

The coefficients and polynomials can also be calculated by

$$\rho_n = \langle \varphi_n, \varphi_n \rangle, \ \beta_n = \rho_n / \rho_{n-1}$$
$$\psi_n(t) = \varphi_n(t) + \beta_n \psi_{n-1}(t),$$
$$\sigma_n = \langle \psi_n, t\psi_n \rangle, \ \alpha_n = \rho_n / \sigma_n$$
$$\varphi_{n+1}(t) = \varphi_n(t) - \alpha_n t \psi_n(t)$$

This is (part of) CG-algorithm for orthogonal polynomials.

Algorithm for squared polynomials

Define $\Phi_n = \varphi_n^2, \quad \Theta_n = \varphi_n \psi_{n-1}, \quad \Psi_n = \psi_n^2$, then the CG coefficients could as well have been calculated by

$$\rho_n = \langle 1, \Phi_n \rangle, \ \sigma_n = \langle 1, t\Psi_n \rangle = \langle t, \Psi_n \rangle$$

Complete recursion:

$$\begin{aligned}
\Psi_n(t) &= \Phi_n(t) + 2\beta_n \Theta_n(t) + \beta_n^2 \Psi_{n-1}(t) \\
\Theta_{n+1}(t) &= \Phi_n(t) + \beta_n \Theta_n(t) - \alpha_n t \Psi_n(t) \\
\Phi_{n+1}(t) &= \Phi_n(t) - 2\alpha_n t[\Phi_n(t) + \beta_n \Theta_n(t)] + \alpha_n^2 t^2 \Psi_n(t)
\end{aligned}$$

where $\beta_n = \frac{\rho_n}{\rho_{n-1}}$, $\alpha_n = \frac{\rho_n}{\sigma_n}$, $\rho_n = \langle 1, \Phi_n \rangle$, and $\sigma_n = \langle t, \Psi_n \rangle$

Back to vectors: CGS

- $\widehat{\boldsymbol{r}}_n = \Phi_n(\boldsymbol{A})\boldsymbol{r}_0$, $\widehat{\boldsymbol{p}}_n = \Psi_n(\boldsymbol{A})\boldsymbol{r}_0$, and $\widehat{\boldsymbol{q}}_n = \Theta_n(\boldsymbol{A})\boldsymbol{r}_0$.
- Substitute $\boldsymbol{A}$ for t in the 'squared polynomial algorithm', and apply the obtained operators to $\boldsymbol{r}_0$:

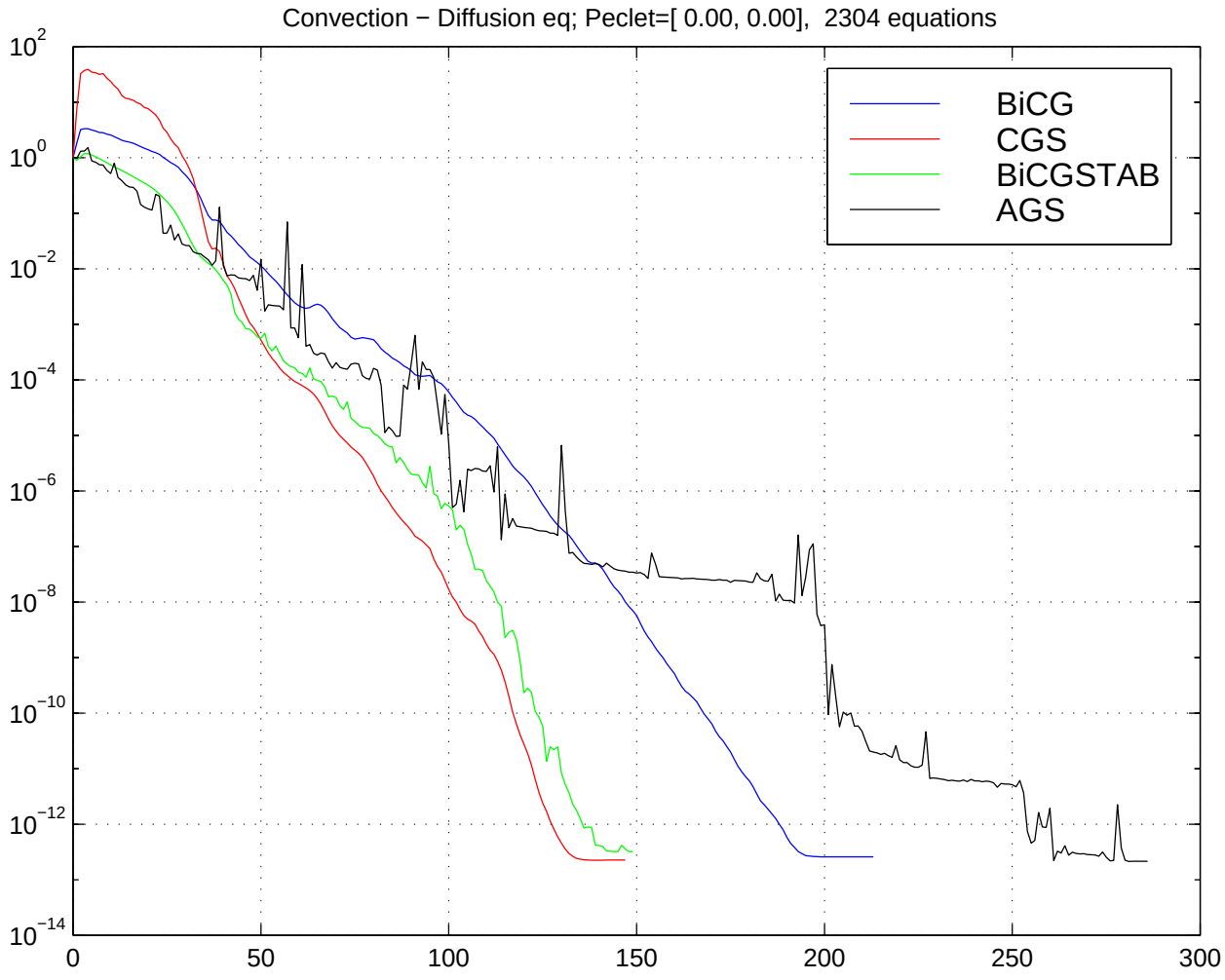
$$
\begin{aligned}
\widehat{\boldsymbol{p}}_n &= \widehat{\boldsymbol{r}}_n + 2\beta_n\widehat{\boldsymbol{q}}_n + \beta_n^2\widehat{\boldsymbol{p}}_{n-1} \\
\widehat{\boldsymbol{q}}_{n+1} &= \widehat{\boldsymbol{r}}_n + \beta_n\widehat{\boldsymbol{q}}_n - \alpha_n\boldsymbol{A}\widehat{\boldsymbol{p}}_n \\
\widehat{\boldsymbol{r}}_{n+1} &= \widehat{\boldsymbol{r}}_n - 2\alpha_n\boldsymbol{A}[\widehat{\boldsymbol{r}}_n + \beta_n\widehat{\boldsymbol{q}}_n] + \alpha_n^2\boldsymbol{A}^2\widehat{\boldsymbol{p}}_n
\end{aligned}
$$

and update corresponding $\widehat{\boldsymbol{x}}$:

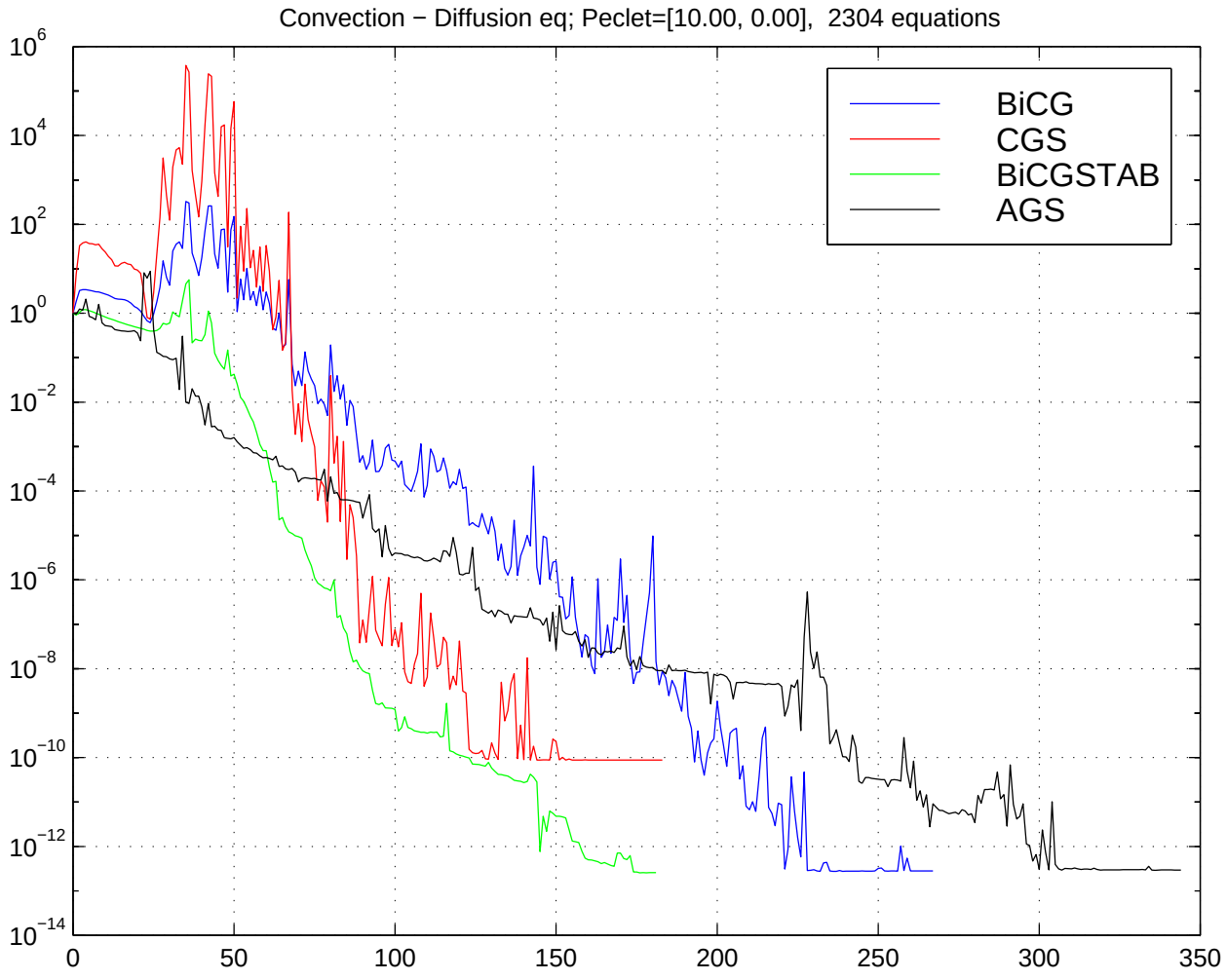
$$
\widehat{\boldsymbol{x}}_{n+1} = \widehat{\boldsymbol{x}}_n + 2\alpha_n[\widehat{\boldsymbol{r}}_n + \beta_n\widehat{\boldsymbol{q}}_n] - \alpha_n^2\boldsymbol{A}\widehat{\boldsymbol{p}}_n
$$

- This is the heart of a (very primitive!) CGS algorithm

CGS on simple problem (2304 equations).



mesh-Péclet=10: The price of squaring



Possible remedy:

Use $\Omega_n\varphi_n$ instead of φ_n^2, with Ω_n defined by any other rule.

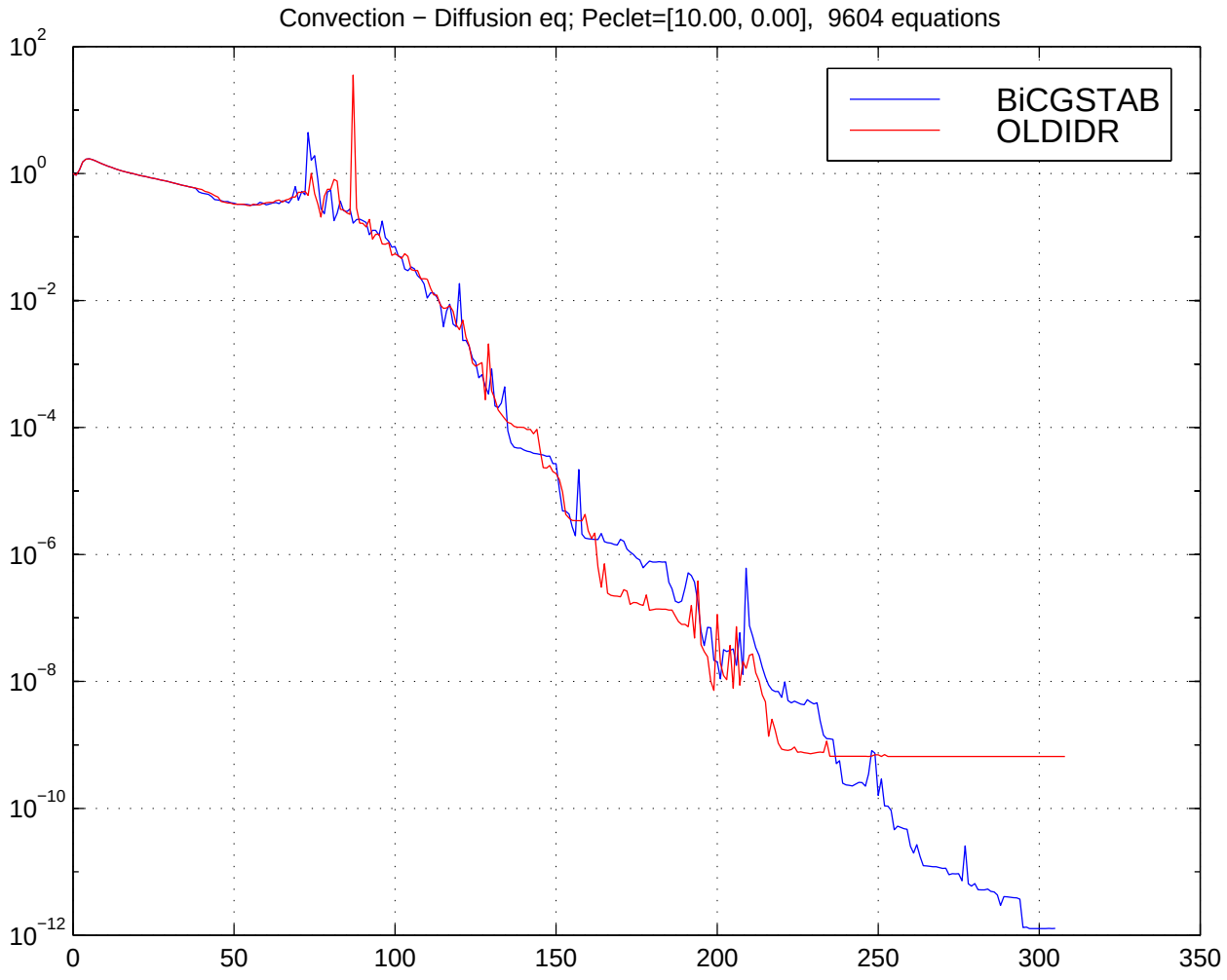
- Main property of φ_n: $\langle \vartheta, \varphi_n \rangle = 0$ for all ϑ of degree lower than n.
- A polynomial Ω of degree n satisfies $\Omega(t) = \gamma\varphi_n(t) + \vartheta(t)$, with ϑ of degree at most $n-1$.
- Alternative calculation: Let $\Omega_n(t) = \gamma\varphi_n(t) + \vartheta(t) = \kappa\psi_n(t) + \upsilon(t)$, then

$$\rho_n = \frac{\langle 1, \Omega_n\varphi_n \rangle}{\gamma}, \quad \sigma_n = \frac{\langle t, \Omega_n\varphi_n \rangle}{\kappa}$$

Birth of BiCGSTAB

- Choose $\Omega_n(t) = (1 - \omega_n t)(1 - \omega_{n-1} t) \cdots (1 - \omega_1 t)$ and determine γ_n and κ_n.
- Choose ω_n to produce minimal growth of the residual.
- Isn't this is mathematically equivalent with IDR??
- Yes, but a completely different implementation!
- This STABilization of CGS was called Bi-CGSTAB.

IDR versus BiCGSTAB



Difference between IDR and Bi-CGSTAB.

Observations:

- The methods behave nearly identical, until IDR starts to suffer from instability.
- A few peaks in IDR lead to loss of about 3 decimal digits compared to BiCGSTAB.
- For difficult problems, this effect becomes more serious.
- Apparently, the BiCG-based construction of the IDR-polynomials is superior.

This, after all, hasn't been a lucky assumption. The problem was a poor *implementation* of the old IDR algorithm.

BiCGStab developments...

- Of course also BiCGSTAB sometimes suffered from problems.
- This led to the development of modifications and generalizations of BiCGSTAB.
- Very clever generalizations have been developed by Martin Gutknecht, and Gerard Sleijpen
- The author had other things on his mind, and went on with a non specifically Krylov-subspace-related life....

Zemke, and a short monologue

- 2006: Jens-Peter Zemke, from Hamburg-Harburg Technical University, mails: What happened to IDR?
- Must read carefully the 1980 version of the theorem. and recover the ancient history.
- Theorem used a space $\mathcal{S}$, not just $\boldsymbol{p}^{\perp}$. Why didn't I use more vectors $\boldsymbol{p}$, say s instead of 1???
- Because it costs $s+1$ matvecs per $\mathcal{G}_j$-space, (Dutch thrift?).
- But maybe there is more dimension reduction per $\mathcal{G}_j$
- Why *@#! didn't I think about that.....

IDR-theorem again

Theorem 2 (IDR) *Let $\boldsymbol{A}$ be any matrix in $\mathbb{R}^{N\times N}$,*
let $\boldsymbol{v}_0$ be any nonzero vector in $\mathbb{R}^N$,
let $\mathcal{G}_0$ be the full Krylov space $\mathcal{K}^N(\boldsymbol{A}, \boldsymbol{v}_0)$,
let $\mathcal{S}$ denote any (proper) subspace of $\mathbb{R}^N$, and let the sequence
$\mathcal{G}_j$, for $j = 1, 2, \ldots$ be defined by

$$\mathcal{G}_j = (\boldsymbol{I} - \omega_j \boldsymbol{A})(\mathcal{G}_{j-1} \cap \mathcal{S})$$

where ω_j are nonzero numbers. Then
i: $\mathcal{G}_j \subset \mathcal{G}_{j-1}$, ii: $M \leq N$ exists such that
$\mathcal{G}_j = \mathcal{G}_M,\ j = M+1, M+2, \ldots$

For 'almost all' $\mathcal{S}$: $\mathcal{G}_M = \{\mathbf{0}\}$.

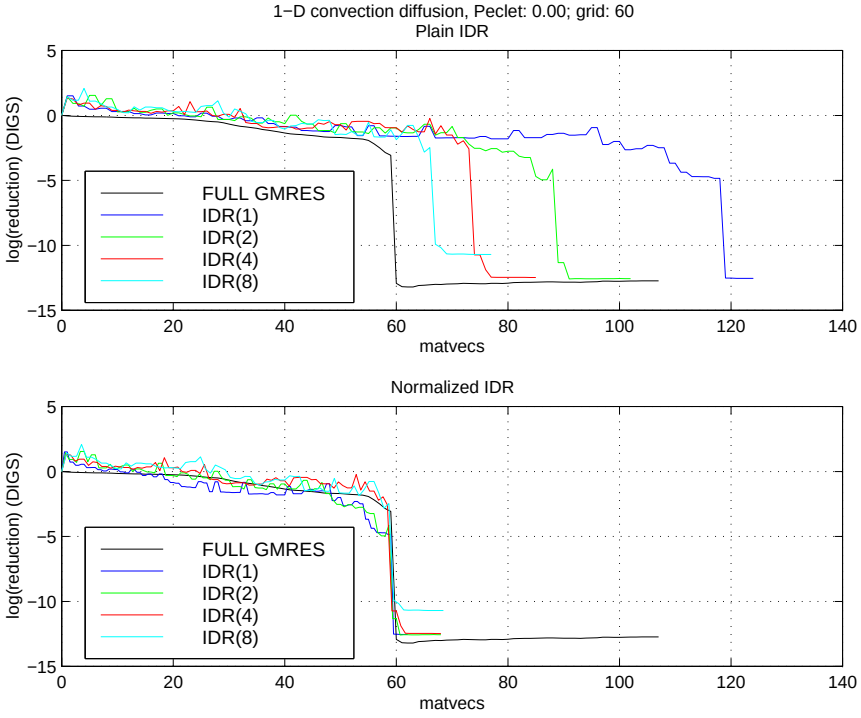
Principle of IDR(s) algorithms

1. Suppose $\boldsymbol{P}$ is some $N \times s$ matrix $\boldsymbol{P}$,and let $\mathcal{S} = \mathcal{N}(\boldsymbol{P}^T)$.
2. Suppose we have $s+1$ independent vectors $\boldsymbol{r}^{(n)}, \boldsymbol{r}^{(n-1)}, \boldsymbol{r}^{(n-2)}, \ldots, \boldsymbol{r}^{(n-s)}$ in $\mathcal{G}_{j-1}$.
3. Define $\boldsymbol{R}_n = [\boldsymbol{r}^{(n)}, \boldsymbol{r}^{(n-1)}, \boldsymbol{r}^{(n-2)}, \ldots, \boldsymbol{r}^{(n-s)}]$
4. Determine a solution of $\boldsymbol{P}^T \boldsymbol{R}_n \boldsymbol{c} = \boldsymbol{0}$, with $\sum c_j = 1$.
5. Then $\boldsymbol{R}_n \boldsymbol{c}$ is in $\boldsymbol{S} \cap \mathcal{G}_{j-1}$, and therefore
6. $(\boldsymbol{I} - \omega_j \boldsymbol{A})(\boldsymbol{R}_n \boldsymbol{c})$ is in G_j.
7. Since $\mathcal{G}_j \subset \mathcal{G}_{j-1}$, this can be repeated to generate more vectors in $\mathcal{G}_j$.
8. Since $\sum c_j = 1$, an x-update can be made.

Termination of IDR(s) on 60×60 system

Finite behavior for IDR(s): Apparently after $\frac{s+1}{s}N$ steps.

This suggests a rescaling of the matvecs count with a factor $\frac{s}{s+1}$:



Upper picture is unscaled

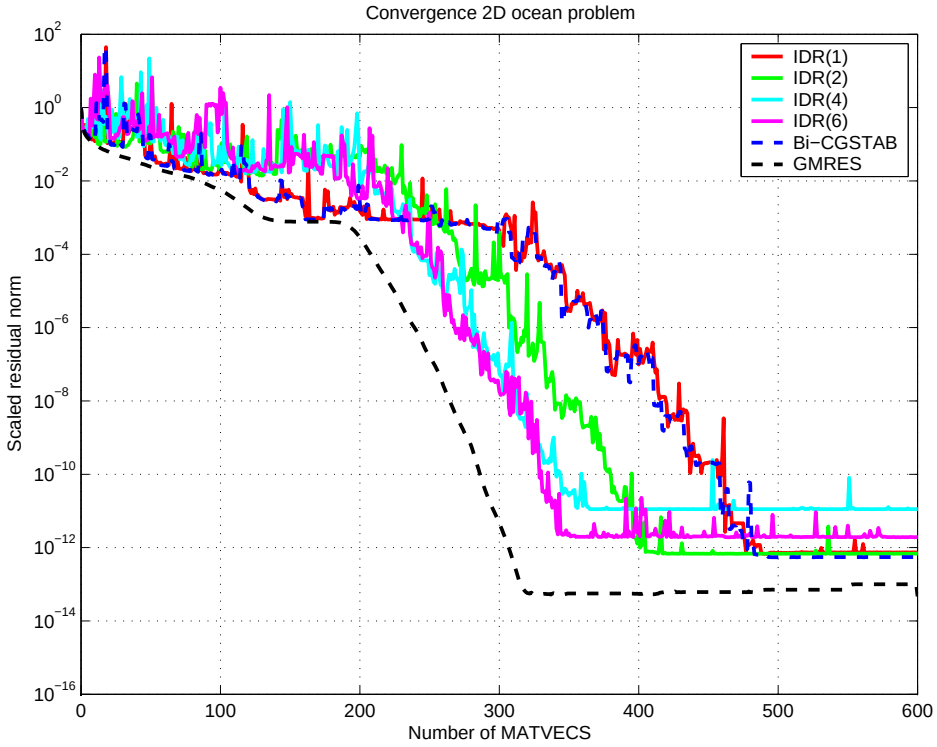
Lower picture:

Horizontal axis displays $s/(s+1)\times$ #matvecs

IDR(s) on a realistic problem

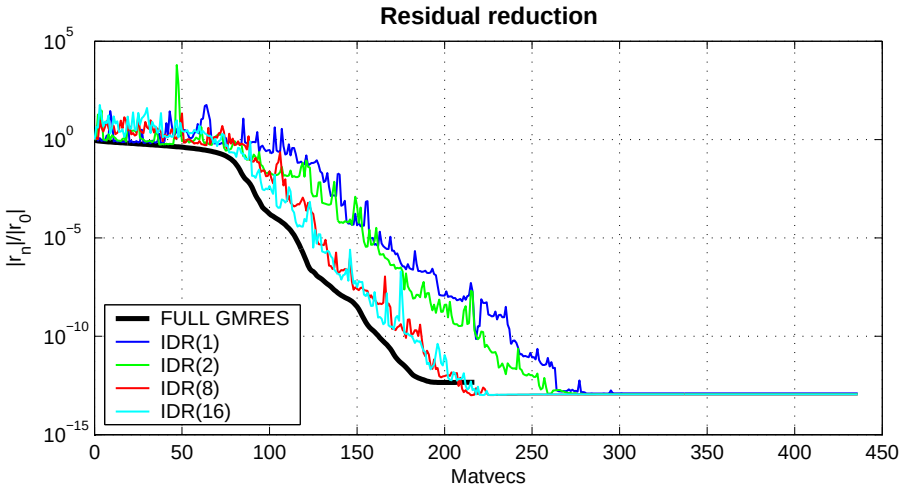
Problem: Convection diffusion equation from oceanography

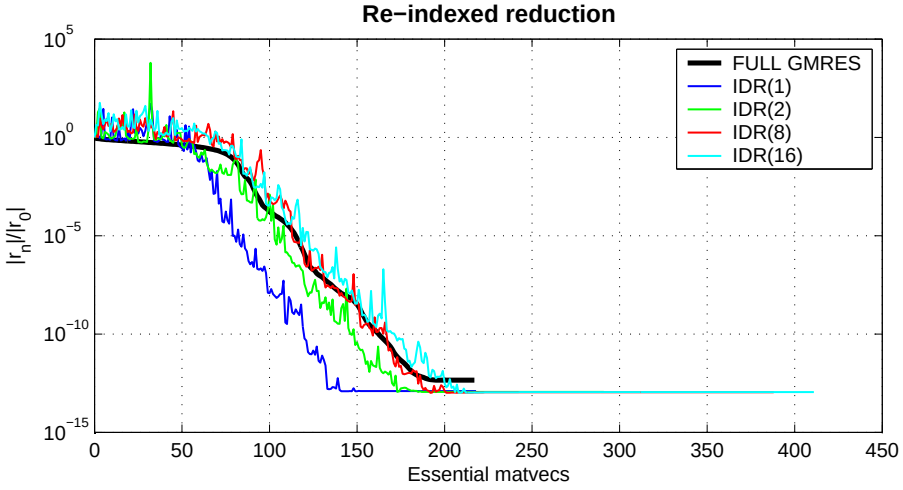
Size: 42248 equations. **Sparseness:** About 300000 nonzeros.



The convergence at different s is similar as in the finite termination behaviour. Scaling the matvec axis?

Convergence ‘Good’ problem



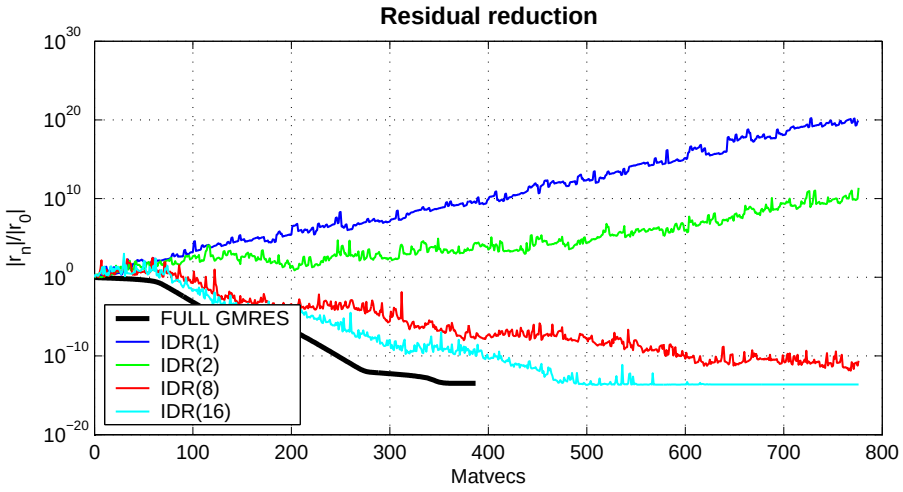


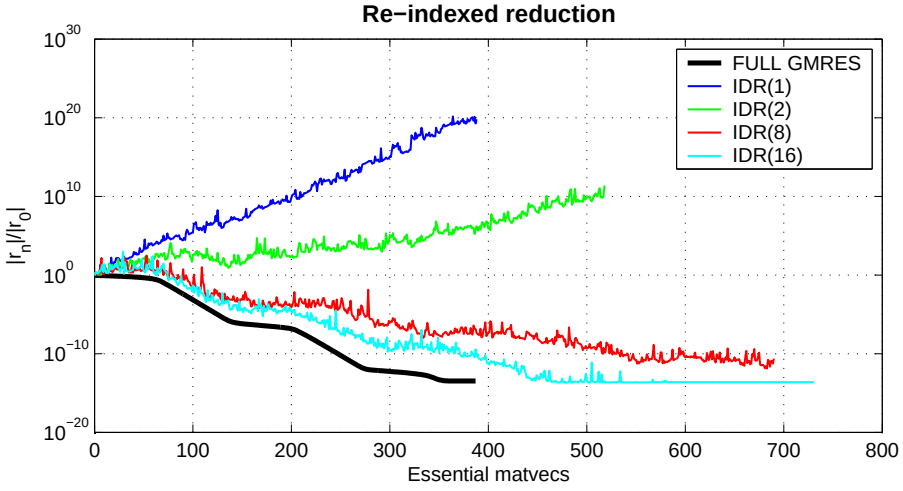
Upper plot: Basic convergence plot for Convection diffusion equation, 60×65 grid, mesh Péclet number 0.50.

Lower plot: Rescaled matvec-axis with $s/(s+1)$

What is it with the blue and green curves??

Convergence ‘Bad’ problem





Upper plot: Basic convergence plot for Convection diffusion equation, 60×65 grid, mesh Péclet number 20.

Lower plot: Rescaled matvec-axis with $s/(s+1)$ IDR(1) and IDR(2) don’t converge!

The IDR polynomial

The residuals satisfy

$$\boldsymbol{r}^{(n)} = \Phi_n(\boldsymbol{A})\boldsymbol{r}^{(0)}$$

The polynomial Φ_n is the IDR-polynomial.

For $n = j(s+1), \ldots, j(s+1)+s$:

$\Phi_n(\boldsymbol{A}) = \Omega_j(\boldsymbol{A})\Psi_{n-j}(\boldsymbol{A})$, with

$\Omega_j(t) = \prod_{k=1}^{j}(1-\omega_k t)$: stabilization polynomial,

$\Psi_{n-j}(t) = 1 - \sum_{l=1}^{n-j} c_l t^l$: Lanczos-type polynomial,

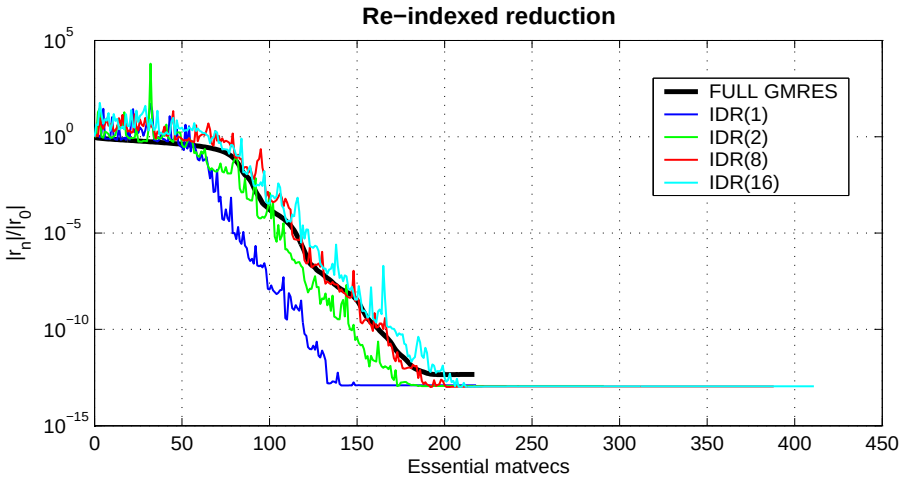
$\widetilde{\boldsymbol{r}}^{(n-j)} = \Psi_{n-j}(\boldsymbol{A})\boldsymbol{r}^{(0)}$: Lanczos residual.

The Lanczos residuals are independent from the choices of ω_j.

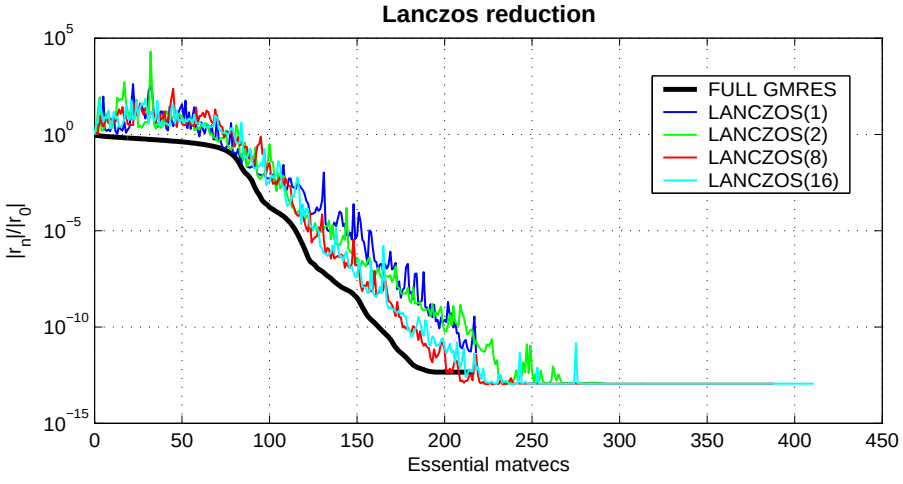
At the cost of $n-j$ extra 'matvecs', they can be obtained from

Martin's implementation of IDR(s).

Lanczos residuals for 'good' problem.

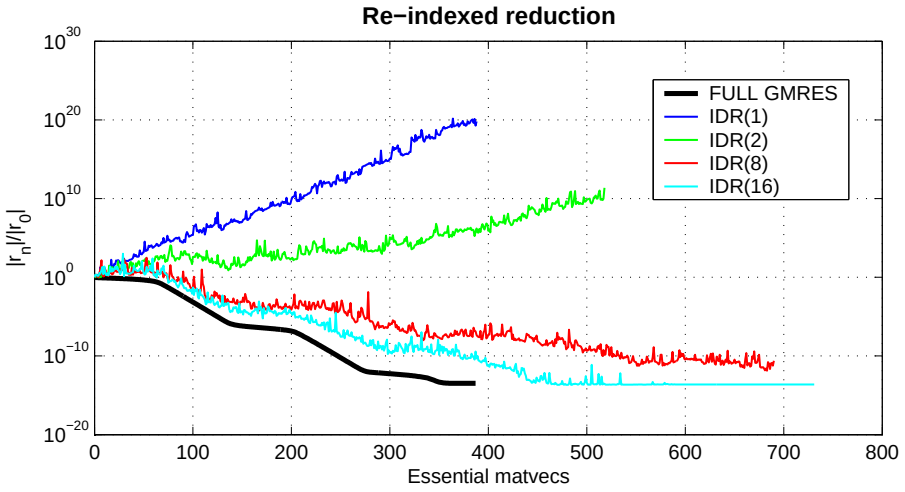


Upper plot: IDR(s) residuals on re-scaled matvec-axis. (Watch the blue and green graphs; Is GMRES not the fastet converger?

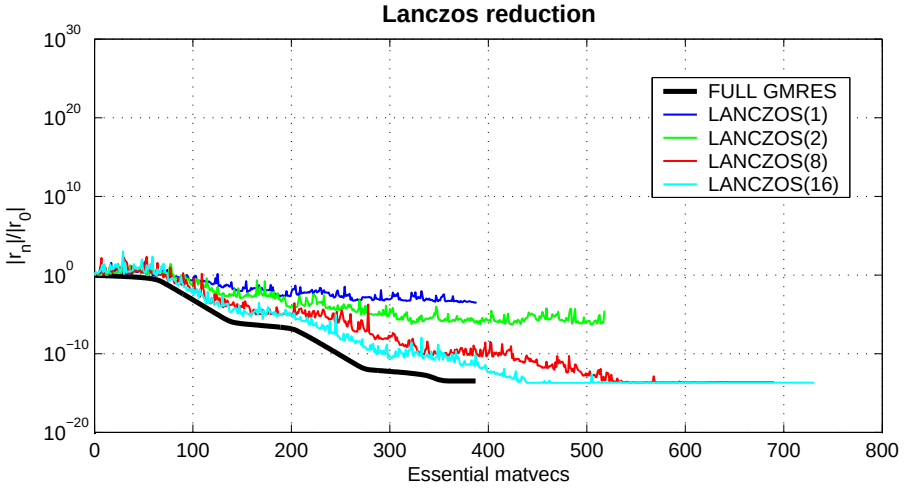


Lower plot: Lanczos residuals. At increasing s, the curves gradually tend to the GMRES curve.

Lanczos residuals for ‘bad’ problem.



Upper plot: IDR(s) residuals on re-scaled matvec-axis.



Lower plot: Lanczos residuals

The stabilizers don’t stabilize at all!

Galerkin interpretation.

For $n = js$, the Lanczos residuals can be written as

$\widetilde{\boldsymbol{r}}^{(n)} = \boldsymbol{r}^{(0)} - \boldsymbol{M}\boldsymbol{c}$, with $\boldsymbol{M} = \left(\boldsymbol{A}\boldsymbol{r}^{(0)}\ \boldsymbol{A}^2\boldsymbol{r}^{(0)}\ \ldots\ \boldsymbol{A}^n\boldsymbol{r}^{(0)}\right)$

They satisfy the relations

$\boldsymbol{p}_r^H \boldsymbol{A}^l \widetilde{\boldsymbol{r}}^{(n)} = 0$, for $l = 0, 1, \ldots, j-1$, $r = 1, 2, \ldots, s$

which can be written as

$\boldsymbol{t}_{l,r}^H \widetilde{\boldsymbol{r}}^{(n)} = 0$ for $l = 0, 1, \ldots, j-1$, $r = 1, 2, \ldots, s$

with $\boldsymbol{t}_{l,r} = (\boldsymbol{A}^H)^l \boldsymbol{p}_r$

This can be interpreted as a Galerkin solution, with test vectors $\boldsymbol{t}_{l,r}$, for the overdetermined system of equations $\boldsymbol{M}\boldsymbol{c} - \boldsymbol{r}^{(0)} = \boldsymbol{0}$.

We call this Krylov Galerkin.

Explicit Galerkin solution.

Let $\boldsymbol{T}$ be the matrix of test vectors:

$$\boldsymbol{T} = (\boldsymbol{t}_{0,1}, \boldsymbol{t}_{0,2} \ \ldots, \boldsymbol{t}_{l,r}, \ldots, \ \boldsymbol{t}_{j-1,s})$$

Then the Galerkin solution for $\boldsymbol{Mc} = \boldsymbol{r}^{(0)}$ reads

$$\boldsymbol{c} = (\boldsymbol{T}^H \boldsymbol{M})^{-1} \cdot \boldsymbol{T}^H \boldsymbol{r}^{(0)}$$

Then

$$\widetilde{\boldsymbol{r}}^{(n)} = \boldsymbol{r}^{(0)} - \boldsymbol{Mc} = (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{r}^{(0)}$$

where $\boldsymbol{P} = \boldsymbol{M}(\boldsymbol{T}^H \boldsymbol{M})^{-1}\boldsymbol{T}^H$ is an oblique projection, (only satisfying $\boldsymbol{P}^2 = \boldsymbol{P}$).

GMRES also produces a Galerkin solution, with $\boldsymbol{T} = \boldsymbol{M}$, the least squares solution for $\boldsymbol{Mc} = \boldsymbol{r}^{(0)}$.

Relation to full GMRES.

Denote the GMRES residuals by $\widehat{\boldsymbol{r}}^{(n)}$, then

$$\widehat{\boldsymbol{r}}^{(n)} = (\boldsymbol{I} - \widehat{\boldsymbol{P}})\boldsymbol{r}^{(0)}$$

where $\widehat{\boldsymbol{P}} = \boldsymbol{M}(\boldsymbol{M}^H\boldsymbol{M})^{-1}\boldsymbol{M}^H$.

For any 2 projections on the same space we have $\boldsymbol{P}_1\boldsymbol{P}_2 = \boldsymbol{P}_2$, since the result of $\boldsymbol{P}_2$ is already in this space. Then we also have: $(\boldsymbol{I} - \boldsymbol{P}_1)(\boldsymbol{I} - \boldsymbol{P}_2) = (\boldsymbol{I} - \boldsymbol{P}_1)$. Therefore

$$\widetilde{\boldsymbol{r}}^{(n)} = (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{r}^{(0)} = (\boldsymbol{I} - \boldsymbol{P})(\boldsymbol{I} - \widehat{\boldsymbol{P}})\boldsymbol{r}^{(0)} = (\boldsymbol{I} - \boldsymbol{P})\widehat{\boldsymbol{r}}^{(n)}$$

Hence

$$\widetilde{\boldsymbol{r}}^{(n)} - \widehat{\boldsymbol{r}}^{(n)} = \boldsymbol{P}\widehat{\boldsymbol{r}}^{(n)} = \boldsymbol{M}(\boldsymbol{T}^H\boldsymbol{M})^{-1}\boldsymbol{T}^H\widehat{\boldsymbol{r}}^{(n)}$$

Random testvectors (1).

- In IDR(s), the shadow vectors, the columns of P, are chosen randomly, since a better criterium cannot be found in general.
- For $n \leq s$, the corresponding Galerkin problem has a testmatrix that has random entries.
- IDR(s) converges faster when s is large. Maybe the randomness is the cause.
- So study the Galerkin-GMRES relation for random testvectors.

Random testvectors (2)

Denote $\widetilde{\boldsymbol{r}}^{(n)} - \widehat{\boldsymbol{r}}^{(n)}$ by $d\widetilde{\boldsymbol{r}}^{(n)}$, then

$$d\widetilde{\boldsymbol{r}}^{(n)} = \boldsymbol{R}(\boldsymbol{T}^H \boldsymbol{R})^{-1} \boldsymbol{T}^H \widehat{\boldsymbol{r}}^{(n)}$$

where $\boldsymbol{T}$ is an $N \times n$ random matrix. By a random matrix we mean:

a matrix of which all entries are stochastically independent, standard normally distributed stochastic variables.

It can be proved that

$$d\widetilde{\boldsymbol{r}}^{(n)} = \|\widehat{\boldsymbol{r}}^{(n)}\| \cdot \boldsymbol{v}$$

where $\boldsymbol{v}$ is the solution of a linear $n \times n$ system with random matrix and a random righthandside, a so-called 'completely random system'.

Of course we are interested in $\|d\widetilde{\boldsymbol{r}}^{(n)}\|$ compared to $\|\widehat{\boldsymbol{r}}^{(n)}\|$.

Completely random systems

If $\boldsymbol{Bx} = \boldsymbol{b}$ is a completely random $n \times n$ linear system, then the solution $\boldsymbol{x}$ has a simultaneous probability density:

$$f_n(\boldsymbol{x}) = \frac{C}{(1 + \|\boldsymbol{x}\|^2)^{\frac{1}{2}(n+1)}}$$

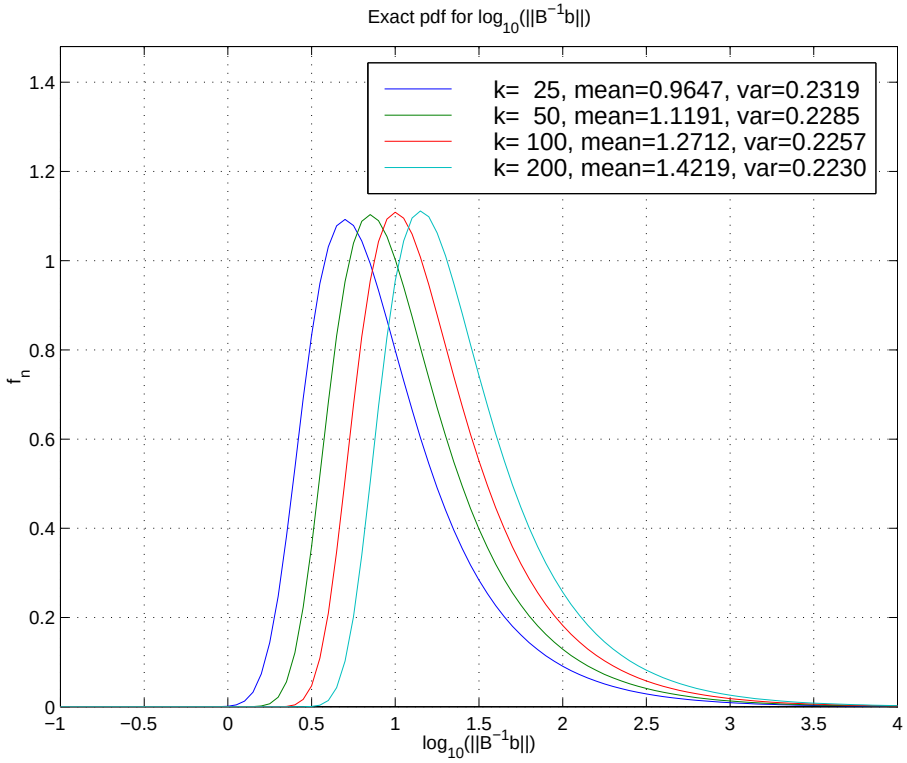
By integrating f_n over a hyper sphere at radius $x = \|\boldsymbol{x}\|$, we get the distribution density F_n of $\|\boldsymbol{x}\|$

$$F_n(x) = C' \frac{x^{n-1}}{(1 + x^2)^{\frac{n+1}{2}}},$$

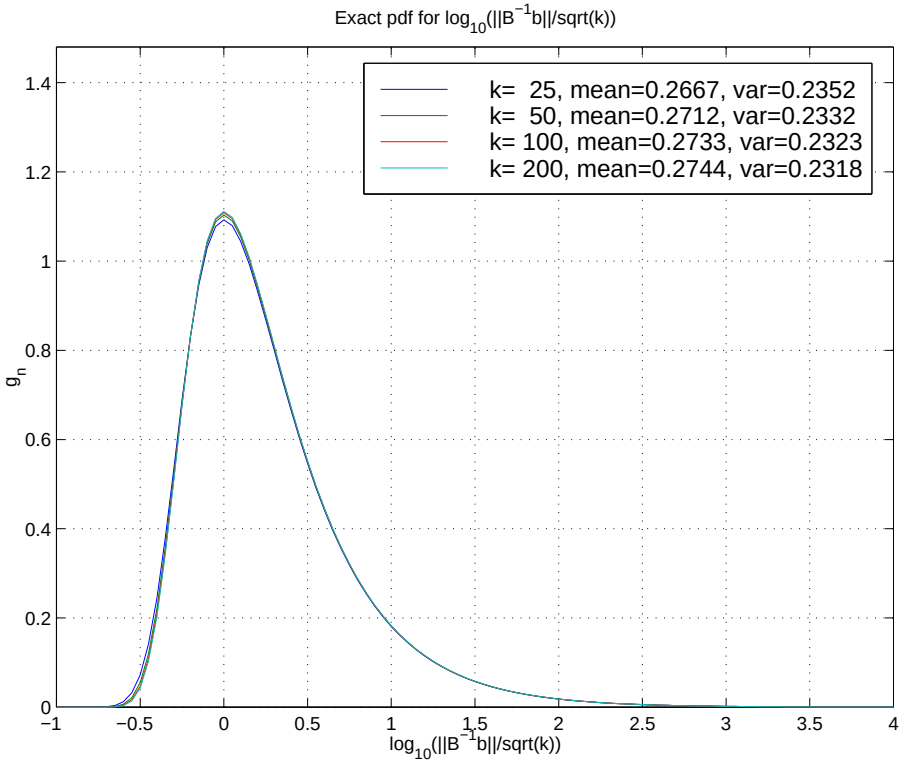
A proof, also involving complex distributions, and written for non-statisticians, can be found in DIAM-report 10.09.

Analytic distribution

Probability density for the number of digits that the Lanczos-residuals are away from the GMRES residuals.

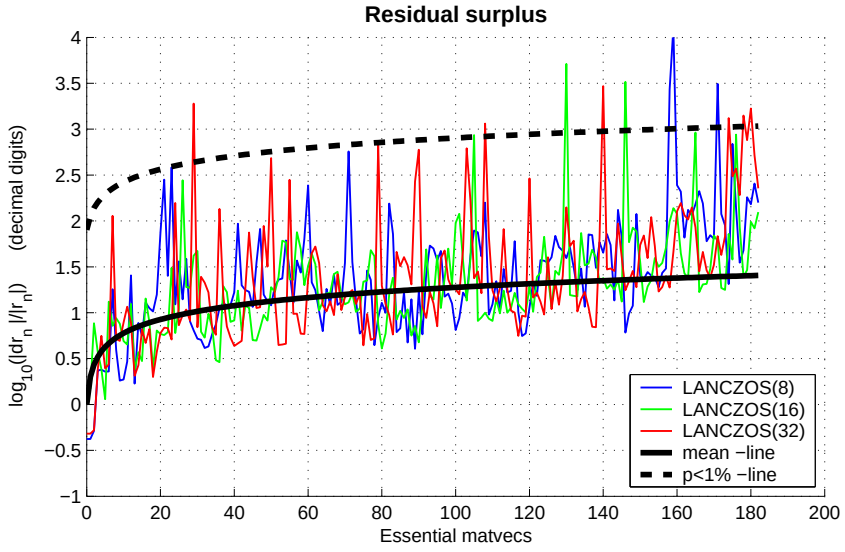


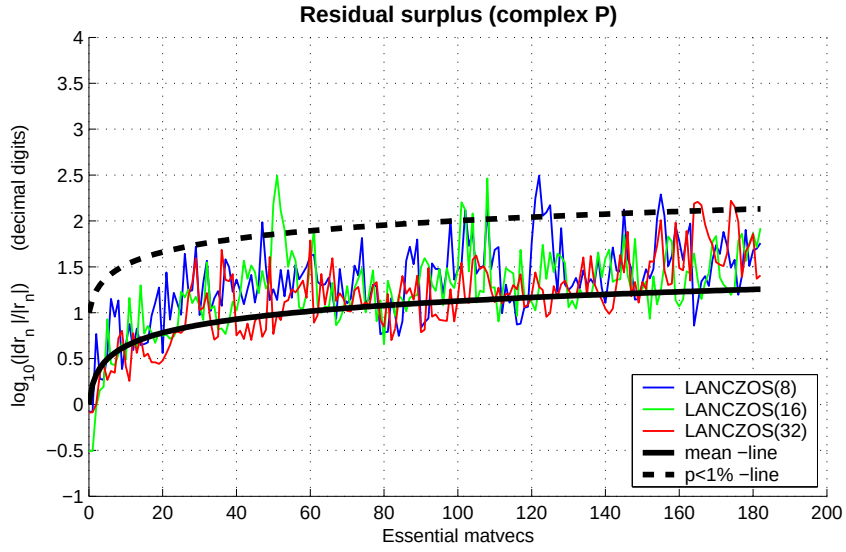
Analytic distribution, scaled by $\sqrt{k}$



n	mean	variance
25	.2667	.2352
50	.2712	.2332
100	.2733	.2323
200	.2744	.2318

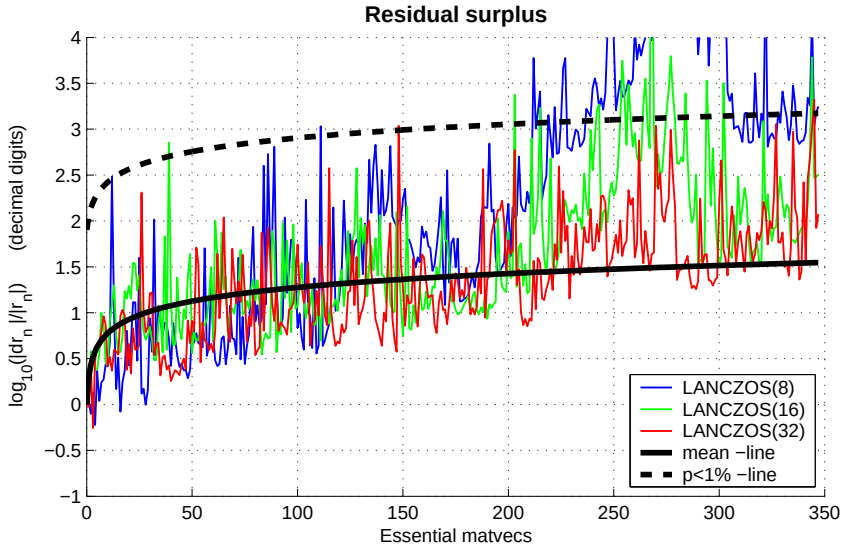
Experiments with ‘good’problem.

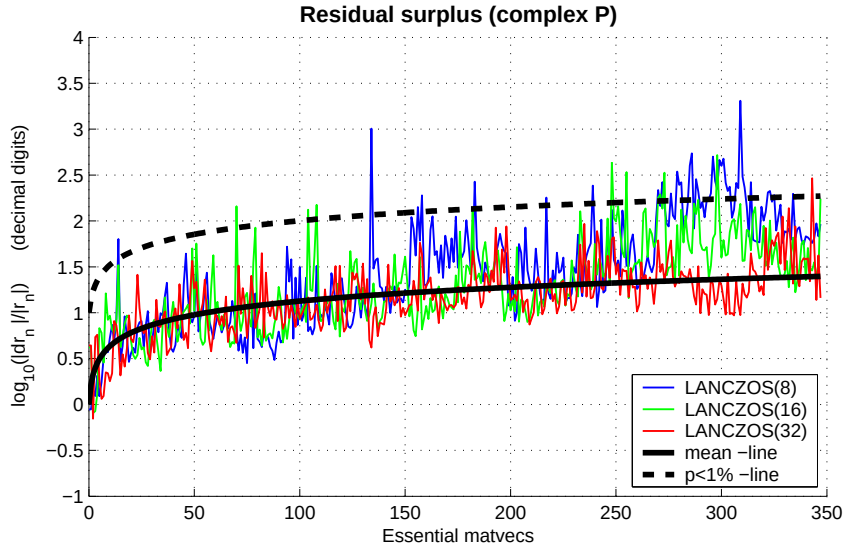




In the right plot, the shadow vectors are chosen complex.

Experiments with ‘bad’ problem.





In the right plot, the shadow vectors are chosen complex.

Final remarks

The author thinks that serendipity is an important part of scientific research, and at least it is an extremely satisfying part. According to Peter Wynn, ‘numerical analysis is much of an experimental science’, and in the IDR-CGS-IDR(s) development, the experimental part was the main source of serendipity.
So the numerical mathematician should never hesitate to do numerical experiments, nor hesitate to look not only to his/her results, but also the non-results. There may be something in it!

Without Martin van Gijzen, this story wouldn’t have been told before I was 80 years old. Thank you Martin.

References

[1] Michael Eierman and Oliver G, Ernst: *Geometric aspects of the theory of Krylov subspace methods;* Acta Numerica : pp. 251-312, (2001)

[2] Y. Saad and M.H. Schultz: *GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems;* SIAM J. Sci. Statist. Comput. **7**: pp. 856-869, (1986)

[3] P. Sonneveld: *On the convergence behaviour of IDR(s);* Delft University of Technology, Reports of the Department of Applied Mathematical Analysis **10-08**: (2010)

[4] P. Sonneveld: *On the statistical properties of solutions of completely random linear systems.;* Delft University of Technology, Reports of the Department of Applied Mathematical Analysis **10-09**: (2010)

[5] Peter Sonneveld and Martin B. van Gijzen: *IDR(s): a family of simple and fast algorithms for solving large nonsymmetric systems of linear equations ;* SIAM J. Sci. and Statist. Comput. **31:2**: pp. 1035-1062, (2008)

[6] Martin B. van Gijzen and Peter Sonneveld: *Algorithm 913: An Elegant IDR(s) Variant that Efficiently Exploits Bi-orthogonality Properties;* ACM Transactions on Mathematical Software **38(1)**: pp. 5:1-5:19, (2011)