Iterative methods and software for systems of equations

EFFICIENCY
ACCURACY
ACCELERATION

Henk van der Vorst

Universiteit Utrecht *Mathematical Institute*

Tuesday, August 29, 2000
Euro-Par 2000, München

OUTLINE

- Krylov methods
- GMRES, Bi-CG hybrids
- Efficiency: reduction overhead
- Irregular convergence: accuracy
- Preconditioning:
  Circuit Simulation

KRYLOV SUBSPACE METHODS

$$Ax = b, \qquad r_0 = b - Ax_0$$

$$x_i = x_0 + \left[r_0, Ar_0, A^2 r_0, \ldots, A^{i-1} r_0\right]$$

a *Ritz-Galerkin*:
$b - Ax_i \perp \mathcal{K}^i(A; r_0)$
CG, Lanczos, **FOM**, GENCG

b *Minimum Residual*:
$\|b - Ax_i\|_2$ minimal over $\mathcal{K}^i(A; r_0)$
MINRES, **GMRES**, ORTHODIR

c *Petrov-Galerkin*:
$b - Ax_i \perp \mathcal{K}^i(A^T; s_0)$
Bi-CG, QMR

HOW TO SELECT?

For $Ax = b$, with $A \neq A^T$, $A \in \mathbb{R}^{n \times n}$.

1. If overhead no problem: **GMRES**

2. Too much overhead:
   QMR, **BiCGSTAB**,
   TFQMR, **CGS**, **Bi-CGSTAB(ℓ)**

3. Variable preconditioning:
   GMRESR, **FGMRES**

SKETCH OF GMRES

FOR $j = 1, 2, \ldots, m$

$w = Av_j$

FOR $i = 1, \ldots, j$

$h_{ij} = (w, v_i)$

$w = w - h_{ij} v_i$

$h_{j+1,j} = \|w\|_2$; $v_{j+1} = w / \|w\|_2$

Solve $H_m y_m = \|r_0\| e_1$

$x = x_0 + [v_1, \ldots, v_m]\, y_m$

if necessary repeat

If A has n_A nonzeros per row:
then overhead dominates if $m \geq \frac{n_A}{2}$

REDUCTION GMRES-ITERATIONS

restrict number of GMRES steps
by combination with other method

1. further reduction of r_i
   with e.g. GMRES itself: GMRESR
   (VDV and Vuik '94)

2. precond of search direction
   FGMRES (Saad '93)

GMRESR: more memory
robust; reductions add

FGMRES: cheaper in memory
not robust

GMRESR

$r_0 = b - Ax_0$;
for $i = 0, 1, 2, \ldots$
  z_m approx. sol. of $Az = r_i$
   (with method of choice)
  $c = Az_m$
  for $k = 0, 1, \ldots, i-1$
    $\alpha = (c_k, c)$
    $c = c - \alpha c_k$
    $z_m = z_m - \alpha u_k$
  $c_i = c/\|c\|_2$; $u_i = z_m/\|c\|_2$
  $x_{i+1} = x_i + (c_i, r_i)u_i$
  $r_{i+1} = r_i - (c_i, r_i)c_i$

(FGMRES works with u_i)

EXAMPLE OF GMRESR

discretized Navier-Stokes problem

	GMRES	GMRES(50)	GMRESR+ GMRES(10) as "solve"
matvec	184	1220	198
daxpy	17000	35000	1386
ddot	17000	35000	1224
memory	184	50	46
cpu-time	7.2	17.0	1.2

CHOICE OF m IN GMRESR

incompressible Navier-Stokes, flow through curved channel ($n = 1000$)

GMRES: 177 it., GMRES(50): 1323 it

GMRESR(m)

m	it	m*it	cpu-time	memory
4	45	180	1.15	94
8	23	184	0.89	54
12	16	192	0.90	44
16	12	192	1.05	40
20	10	200	1.23	40

GCRO

GMRESR-part:
z_m approx. sol. of $Az = r_i$
(with method of choice)

For inner iteration other than GMRES:

sometimes effective to keep iterations orthogonal on previous GMRESR c_i's

Do inner-iterations with

$$A_i \equiv \prod_{j=0}^{i-1} (I - c_j c_j^T) A$$

GCRO (de Sturler and Fokkema '92)
overhead: number of inner products

Bi-CG AND VARIANTS

with short recurrencies we can construct x_i such that $r_i \perp \mathcal{K}^i(A^T; s_0)$

- not optimal in $\mathcal{K}^i(A; r_0)$
- 2 MV's per iteration
- CG-like computational overhead
- CG-like memory requirements
- more iterations than GMRES: $\|Ax_i^{BiCG} - b\|_2 \geq \|Ax_i^{GMRES} - b\|_2$

VARIANTS OF BiCG (1)

- **QMR** (Freund & Nachtigal)

- slightly better than Bi-CG
- more smooth convergence
- more iterations than GMRES

- **CGS** (Sonneveld)

- 2 MV's in BiCG can be used to apply BiCG twice: CGS
- same costs as BiCG
- often twice as fast
- very irregular convergence
- often faster than GMRES
- more MV's than GMRES

CGS

Basis for $K^m(A; r_0)$ and $K^m(A^T; s_0)$
with same 3-term recursions:

$$r_i = P_i(A)r_0$$

but also: $\hat{r}_i = P_i(A^T)s_0$

Bi-CG coëfficients through
innerproducts like $(Ar_i, \hat{r}_j)$

Sonneveld (1984):

$$(Ar_i, \hat{r}_j) = (AP_i(A)r_0, P_j(A^T)s_0)$$

$$= (AP_i(A)P_j(A)r_0, s_0)$$

$\hat{r}_j$ not necessary!!
Simple recursion for $\tilde{r}_i = P_i^2(A)r_0$: CGS

ARE PEAKS BAD?

Bi-CG type processes (Bi-CG, CGS, ...):

$$\begin{aligned} x_i &= x_{i-1} + \alpha_i p_i \\ r_i &= r_{i-1} - \alpha_i A p_i \end{aligned}$$

errors in x_i no effect on r_i

In finite precision:

$$r_i = r_{i-1} - \alpha_i A p_i - \alpha_i \Delta_A p_i$$

$$|\Delta_A| \leq n_A \xi |A|$$

$$r_i - (b - Ax_i) = -\sum_{j=1}^{i} \alpha_j \Delta_A p_j$$

$$|\|r_i\|_2 - \|b - Ax_i\|_2| \leq$$

$$2\, i\, n_A \xi \||A|\| \|A^{-1}\| \max_j \|r_j\|$$

RELIABLE UPDATING

from suggestion by Neumaier '94
made for CGS

$\bar{x} = x_0$; $\bar{r} = r_0$; $x_u = 0$
...
for $i = 0, 1, 2, \ldots$
  ...
  $x_u = x_u + \alpha_i p_i$
  $r = r - \alpha_i A p_i$
  ...
  if $(\|r\| < \|\bar{r}\| \wedge i - i_{prev} < m_i)$
    $\bar{x} = \bar{x} + x_u$
    $\bar{r} = r = b - A\bar{x}$
    $x_u = 0$
  endif

if $\|r\| \approx \xi \|r_0\|$: $r_i \approx b - A x_i$
for analysis: Sleijpen and VDV '94
simple criteria for m_i: Ye and VDV '99

MOTIVATION TO IMPROVE CGS

GOALS:

- smoother convergence
- faster convergence
- maintain quadratic effects
  in converged eigendirections

POSSIBILITIES:

- different s_0
- instead of $r_i = P_i^2(A) r_0$:

$$r_i = \widetilde{P}_i(A) P_i(A) r_0$$

  with "nearby" $\widetilde{P}_i$

NEARBY POLYNOMIALS

$$r_i = \widetilde{P}_i(A) P_i(A) r_0$$

(1) $P_i(A)$ is based on r_0 and s_0

Make $\widetilde{P}_i(A)$ with r_0 and $t_0 \neq s_0$

Often for $\widetilde{P}_i$ and P_i

$$\widetilde{P}_i(\lambda_j) \approx P_i(\lambda_j) \approx 0$$

for some λ_j and i not too small
quadratic convergence in some directions

NEARBY POLYNOMIALS (2)

Take $\widetilde{P}_i(A) = (I - \omega A) P_{i-1}(A)$
Often much smoother behavior

Heuristic explanation:
If A is symmetric then zeros of P_i
separate those of P_{i-1}
$\Rightarrow$ zeros of P_{i-1} kill peaks
of P_i

For converged zeros: $P_i(\lambda_j) \approx P_{i-1}(\lambda_j) \approx 0$
again quadratic effects

details: Fokkema, Sleijpen, VDV (1996)

VARIANTS OF BiCG (2)

BiCGSTAB (Van Der Vorst)

- 2 MV's in BiCG can be used to combine BiCG with repeated GMRES(1)
- same costs as BiCG
- often much faster than BiCG
- much smoother than CGS
- often faster than CGS
- breakdown when GMRES(1) stagnates
- poor when GMRES(1) is very poor

VARIANTS OF BiCG (3)

BiCGSTAB(2)

(Sleijpen, Fokkema, vdVorst '94)

- after each two BiCG steps: GMRES(2)
- often faster than BiCGSTAB
- also for near skew-symm. matrices
- $\approx$ same costs as BiCG
- easy to generalize: BiCGSTAB(ℓ)
- BiCGSTAB(4): fast and robust
- breakdown when GMRES(ℓ) stagnates

AVOIDING BREAKDOWN

two reasons for breakdown
in Bi-CGSTAB methods

(1) Bi-CG part may break-down:
Look-ahead techniques (complicated)

(2) GMRES-part gives no reduction
no expansion of Krylov subspace

Sleijpen and VDV '95:
Combination of GMRES and FOM
available from Web-page

PRECONDITIONING REQUIRED

Convergence behavior depends on
spectral properties

Iterative methods often applied to

ℓ *left-preconditioned system*
$K^{-1}Ax = K^{-1}b$

r *right-preconditioned system*
$AK^{-1}z = b$

c *central-preconditioned system*
$L^{-1}AU^{-1}w = L^{-1}b$

If K $(= LU)$ is a good aproximation to A
then **all iterative methods robust**

PRECONDITIONING

1. Idea: replace A by some easier to solve M

2. Point ILU and MILU

3. Variants
   - (block, RILU, ILQ, drop-tol,...)

4. Parallel aspects

---

* Overview of ideas, no details

ILU Preconditioners

Advantages:

- general
- robust for elliptic PDE's
- bridge between GE and basic iteration

Disadvantages:

- does not exploit analytical properties of A
- not optimal (h-dependent) for PDE

POINT ILU (Meijerink-VDV 77)

* Idea: in GE, drop disallowed fill-ins in L, U.

* Define S such that $l_{i,j}, u_{i,j} = 0$ if $(i,j) \notin S$.

* Let $M = LU$. Want: $m_{i,j} = a_{i,j}$ if $(i,j) \in S$.

* E.g. $S = \{(i,j) | a_{i,j} \neq 0\}$

Stencil for difference operators:

$$
\begin{array}{ccccc}
 & & s_{i,j} & & \\
 & & | & & \\
b_{i,j} & -- & a_{i,j} & -- & c_{i,j} \\
 & & | & & \\
 & & t_{i,j} & &
\end{array}
$$

ILU: $d_{i,j} = a_{i,j} - c_{i,j} b_{i-1,j} / d_{i-1,j} - s_{i,j} t_{i,j-1} / d_{i,j-1}$

MODIFIED ILU (Gustafson 78)

* Idea: add dropped terms to main diagonal.

Stencil for difference operators:

$$
\begin{array}{ccccc}
 & & s_{i,j} & & \\
 & & | & & \\
b_{i,j} & -- & a_{i,j} & -- & c_{i,j} \\
 & & | & & \\
 & & t_{i,j} & &
\end{array}
$$

MILU:

$$
\begin{aligned}
d_{i,j} = a_{i,j} \; & - \; c_{i,j}(b_{i-1,j} + t_{i-1,j}) / d_{i-1,j} \\
& - \; s_{i,j}(t_{i,j-1} + b_{i,j-1}) / d_{i,j-1} + ch^2
\end{aligned}
$$

ILU-dropped terms are in red

Let $A = D + L + U$.
Result: $M = (\tilde{D} + L)\tilde{D}^{-1}(\tilde{D} + U)$

EISENSTAT TRICK

Eisenstat '81

a. Make (M)ILU:
$M = (L + \widetilde{D})\widetilde{D}^{-1}(\widetilde{D} + U)$

b. Scale given system $Ax = b$ to:

$$\widetilde{D}^{-1/2} A \widetilde{D}^{-1/2} = \widetilde{D}^{-1/2} b$$

or $\widetilde{A}\widetilde{x} = \widetilde{b}$.

Nice effect: $\widetilde{M} = (\widetilde{L} + I)(I + \widetilde{U})$

Savings in:
$(\widetilde{L} + I)^{-1}\widetilde{A}(I + \widetilde{U})^{-1}v =$
$(\widetilde{L} + I)^{-1}\left[v + (diag(\widetilde{A} - 2I)t\right] + t$
with $t = (I + \widetilde{U})^{-1}v$

keeps cost about same as for Av.

VARIANTS

1. **More fill-in**:

- According to *size*: drop-tolerance
- According to *pattern*
- According to *level*

2. **Relaxed ILU**: (Axelsson-Lindskog 86)

- interpolate betw ILU and MILU
- (Elman 89): nonsymmetric A

MULTILEVEL AND RE-ORDERED ILU

Idea: re-order to improve convergence rate
e.g. use MG to order grid pts but use ILU to factor A.
van der Ploeg 93

parallel orderings (working from the corners)
van der Vorst 1986; DDSV 98

Red-black-ordering + Schur complement
Sameh et al, Elman and Golub

PARALLEL ASPECTS

* General principles:

- Extract max parallelism from existing meths
- Modify or approx existing meths
  to increase parallelism

(convergence rate may suffer)

* Bottleneck: backsolves with L and U
  (same issues for ILU and MILU)

APPROACHES TO PARALLELIZATION

1. Re-ordering

2. Domain Decomposition

3. Approximate Inverse

MULTI-WAVEFRONTS ORDERINGS

- Meurant 84: 2 fronts $\rightarrow$ Twisted factors
- VDV 87: nested twisted factors
- VDV 89: multi-wavefronts

Notes:

- M and convergence is different from N.O.
- (Eijkhout 92) MILU may breakdown

MULTI-COLORING

* **Red-Black**:
- Degree of //ism = half of grid pts.
- Completely localized
- (Kuo-Chan 90) $K_{RB} \approx 1/4K_{NO}$ for ILU, MILU, SSOR
- Verified by Tong's experiments 89

* **Multi-color**:
- Doi-Lichnewsky 90, 91 (Cray 2), Doi 91
- Doi-Hoshi 92 (NEC SX-3/14, 76^3 grid, 2 Gflops), better than hyperplane order, up to 75 colors
- More colors $\rightarrow$ faster convergence $\rightarrow$ less //ism
- Choose no. of colors to match no. of processors

DOMAIN DECOMPOSITION

* Idea: Compute ILU in each Ω_i independently; special treatment on interface.

* (Radicati-Robert 87): ILU within overlapped diag-blocks of A; average.

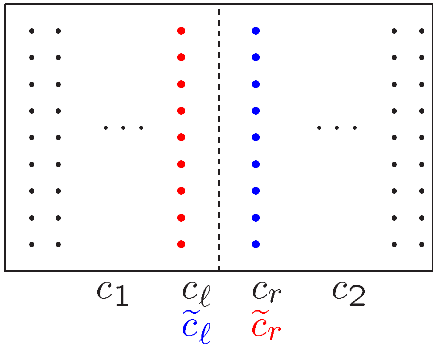
* (Meurant 89, de Sturler 96): repeated twist factorization

* (Chan-Goovaert 90): DD/ILU can converge faster even sequentially (divide and conquer effect)

* (Washio-Hayami 94): SSOR on domains, modified for interface
inverse for coupling by truncated Neumann

* (Tan, Tang): interface conditions; enhanced matrices

DOMAIN DEC. - ENHANCED SYSTEMS



Linear system enhanced with $\tilde{c}_\ell$ and $\tilde{c}_r$

Simplest case:

$$I \begin{pmatrix} c_\ell \\ \tilde{c}_r \end{pmatrix} = I \begin{pmatrix} \tilde{c}_\ell \\ c_r \end{pmatrix}$$

More general:

$$\begin{pmatrix} C_{\ell\ell} & C_{\ell r} \\ C_{r\ell} & C_{rr} \end{pmatrix} \begin{pmatrix} c_\ell \\ \tilde{c}_r \end{pmatrix} = \begin{pmatrix} C_{\ell\ell} & C_{\ell r} \\ C_{r\ell} & C_{rr} \end{pmatrix} \begin{pmatrix} \tilde{c}_\ell \\ c_r \end{pmatrix}$$

TWO DOMAINS

$$\begin{pmatrix} C_{\ell\ell} & C_{\ell r} \\ C_{r\ell} & C_{rr} \end{pmatrix} \begin{pmatrix} c_\ell \\ \tilde{c}_r \end{pmatrix} = \begin{pmatrix} C_{\ell\ell} & C_{\ell r} \\ C_{r\ell} & C_{rr} \end{pmatrix} \begin{pmatrix} \tilde{c}_\ell \\ c_r \end{pmatrix}$$

left Domain :
a. interior unknowns c_1
b. boundary unknowns c_ℓ
c. overlap points $\tilde{c}_r$

right DOMAIN :
a. interior unknowns c_2
b. boundary unknowns c_r
c. overlap points $\tilde{c}_\ell$

Idea stems from W-P Tang (92):
duplicates complete overlap
Virtual overlap: Tan (95)

Universiteit Utrecht   *Mathematical Institute*

ENHANCED MATRIX

Enhanced system:

$$\begin{pmatrix} A_{11} & A_{1\ell} & 0 & 0 & 0 & 0 \\ A_{2\ell} & A_{\ell\ell} & A_{\ell r} & 0 & 0 & 0 \\ 0 & C_{\ell\ell} & C_{\ell r} & -C_{\ell\ell} & -C_{\ell r} & 0 \\ 0 & -C_{r\ell} & -C_{rr} & C_{r\ell} & C_{rr} & 0 \\ 0 & 0 & 0 & A_{r\ell} & A_{rr} & A_{r2} \\ 0 & 0 & 0 & 0 & A_{2r} & A_{22} \end{pmatrix} \begin{pmatrix} c_1 \\ c_\ell \\ \tilde{c}_r \\ \tilde{c}_\ell \\ c_r \\ c_2 \end{pmatrix}$$

In ‘Schwarz’ iteration:
submatrices outside boxes $\Rightarrow$
right-hand side (with c-values of prev. it.)

$C_{\ell\ell} = C_{rr} = I$, $C_{\ell r} = C_{r\ell} = 0$:
classical Dirichlet-Dirichlet coupling

Tan (95): general couplings for 6 point stencils along interface

Universiteit Utrecht   *Mathematical Institute*

APPROXIMATE INVERSES

* No backsolves, simply mat-vec,

* Choose M, with given sparsity pattern, to min $||MA - I||$ in some weighted norm.

* M can often be computed in parallel

* Kolotilina-Yeremin 86, 93: factored M

* (Chow-Saad 94): M computed by GMRES

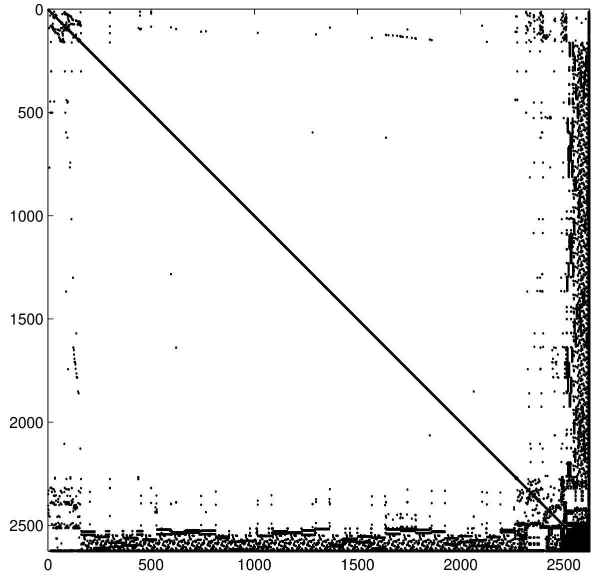
* (Grote-Huckle 97): fill guided by $M-$residual

* see Benzi et al for recent results

* refs in Duff-van der Vorst '99

CIRCUIT SIMULATION

$$Ax = b$$



problem	n	nnz(A)	flops LU	nnz$(L+U)$
A	2624	35823	769817	41385
B	4510	21199	445287	31948
C	12127	48137	373453	67920
D	80209	307604	14225589	461155

PRECONDITIONING SCHUR COMPLEMENT

Stop forming Schur complement if
q iterations with S less expensive
than reduction

Sparsify:
skip small elements (weak couplings) in S:

$$c_{ij} = \begin{cases} s_{ij} & \text{if } |s_{ij}| > t|s_{ii}| \text{ or } |s_{ij}| > t|s_{jj}| \\ 0 & \text{otherwise} \end{cases}$$

Parameter t, e.g. $t = 0.02$

Construct LU-decomposition for
sparsified S

Solve Schur complement with prec. GMRES

DIRECT VERSUS HYBRID

Parameters:
Sparsification: $t = 0.02$
Switch Direct/iterative: $q = 25$
GMRES tol $= 10^{-8}$

problem	n	direct	hybrid
A	2624	865948	459502
B	4510	520526	394315
C	12127	554000	550750
D	80209	15391761	3327380

Note:
1. CPU_time$\not\sim$number_flops
2. Hybrid is parallelizable

SOME REFERENCES

* Dongarra, Duff, Sorensen, and
  van der Vorst, SIAM 1998

* Barrett, Berry, Chan, Demmel, etc
  Templates, SIAM 1994, WWW

* Qiang Ye and van der Vorst:
  Reliable updating, 1999, WWW

* Bomhof and van der Vorst
  Circuit simulation problems, 1999, WWW

* Duff and van der Vorst
  Parallel solution methods, 1999, WWW

* Magolu and van der Vorst
  Parallel incomplete LU, 2000 WWW

WWW:
`http://www.math.uu.nl/people/vorst`