Iterative Linear Solvers and Jacobian-free Newton-Krylov Methods

Eric de Sturler
Department of Mathematics, Virginia Tech
www.math.vt.edu/people/sturler/index.html
sturler@vt.edu

Efficient Solution of Large Systems of Nonlinear PDEs in Science
ENS Lyon, Lyon, France, October 7 – 9, 2013

Overview

- Inexact Newton Iteration
  - Basics
  - Approximating Jacobian-vector products
- Solver Aspects
  - Krylov subspace methods
  - Matrix-free implementations
- Matrix-free Preconditioning
  - Physics-based preconditioner
  - Approximating the Jacobian
  - Updating Preconditioners – Tuesday
- Useful Solver Variants
  - FGMRES
  - GMRES*, GCRO/GCROT

Inexact Newton

Consider $F(x) = 0$ where $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$

Typical Newton algorithm

```
while $\|F(x)\| > \varepsilon$
    (1) Solve $F'(x_n)s = -F(x_n)$
    (2) $x_{n+1} = x_n + \lambda s$        (with $\lambda$ chosen for sufficient decrease)
end
```

Inexact Newton:

(1) $\|F(x_n) + F'(x_n)s\| \leq \eta \|F(x_n)\|$

Use some suitable solver for computing s

Proceed with line search (2)

Inexact Newton Iteration

Inexact Newton: $\left\| F\left(x_n\right) + F'\left(x_n\right)s \right\| \le \eta \left\| F\left(x_n\right) \right\|$

Use Krylov subspace methods, like GMRES, CG, BiCGStab for approximately solving

$$F'\left(x_n\right)s = -F\left(x_n\right)$$

The great advantage of such iterative methods is that we only need a matrix-vector product to solve the system, not the matrix.

Other methods with that property exist, sometimes (nonlinear) multigrid methods can be implemented in a similar fashion.

Approximating the Jacobian-vector Product

Solve $F'(x_n)s = -F(x_n)$, $\qquad x_0 = 0 \to r_0 = -F(x_n)$

Approximate matrix-vector product

$$F'(x_n)r_0 \approx D_\varepsilon(x, r_0) = \frac{F(x_n + \varepsilon\sigma(x_n, r_0)r_0) - F(x_n)}{\varepsilon\sigma(x_n, r_0)},$$

where $s(x_n, r_0)$ is a scaling factor chosen for accuracy

Sequence of such directional derivative approximations.
Each approximates the Jacobian slightly differently, possibly destroying special structure.
Preference for robust methods.

Krylov Methods Crash Course

Consider $Ax = b$. Initial guess $x_0 \to r_0 = b - Ax_0$

In step m: $x_m = x_0 + z_m$ where

$$z_m \in K_m\left(A, r_0\right) = \text{span}\left\{r_0, Ar_0, A^2 r_0, \ldots, A^{m-1} r_0\right\}$$

Krylov space is a space of polynomials in A times vector r_0

So, $z_m = p_{m-1}\left(A\right) r_0$ and $x_m = x_0 + p_{m-1}\left(A\right) r_0$

$$r_m = b - Ax_m = b - Ax_0 - Ap_{m-1}\left(A\right) r_0 = r_0 - Ap_{m-1}\left(A\right) r_0$$

$$r_m = q_m\left(A\right) r_0 = \left(I - Ap_{m-1}\left(A\right)\right) r_0$$

Error: $e_m = A^{-1} b - x_m = A^{-1}\left(b - Ax_m\right) = A^{-1} r_m$

$$e_m = A^{-1} q_m\left(A\right) r_0 = q_m\left(A\right) A^{-1} r_0 = q_m\left(A\right) e_0$$

Choices from Krylov Space

Given x_0 and $r_0 = b - Ax_0$, pick $z_m \in K_m(A, r_0)$ and $x_m = x_0 + z_m$

Several possibilities. Two particularly important ones are:
Find z_m such that $\| r_m \| = \| r_0 - Az_m \|$ is minimal
Find z_m such that $\| e_m \| = \| \hat{x} - (x_0 + z_m) \|$ is minimal

The second one is possible in practice for special norms, like the
$\|x\|_A = (Ax, x)^{1/2}$ if A Hermitian positive definite

Other possibilities exist, in particular non-optimal ones that allow very cheap iterations (BiCGStab)

Approximation by Matrix Polynomials

Let $A = V \Lambda V^{-1}$, let $\Lambda(A) \subset \Omega \subset \mathbb{C}$.

If $p_{m-1}(t) \approx \frac{1}{t}$ for all $t \in \Omega$, then $p_{m-1}(A) = V \operatorname{diag}\left(p_{m-1}(\lambda_i)\right) V^{-1} \approx A^{-1}$

Let $r_0 = V\rho$. Then $p_{m-1}(A) r_0 = \sum_i v_i p_{m-1}(\lambda_i) \rho_i \approx \sum_i v_i \frac{\rho_i}{\lambda_i} = A^{-1} r_0$

$$r_m = q_m(A) r_0 = \left(I - A p_{m-1}(A)\right) r_0 = \sum_i v_i \left(1 - \lambda_i p_{m-1}(\lambda_i)\right) \rho_i \approx 0$$

If we can construct such polynomials for modest m, we have an efficient linear solver.

This is possible if the region Ω is nice – small region away from origin: clustered eigenvalues

If this is not the case, we improve by *preconditioning*: $PAx = Pb$ s.t. PA has clustered eigenvalues and product with P is cheap.

Convergence Bounds

Residual at iteration m: $r_m = p_m(A) r_0$ optimal (2-norm)

Eigenvalue bound $\|r_m\| \leq \|V\| \|V^{-1}\| \|r_0\| \min_{\substack{p \in \Pi_m, \\ p(0)=1}} \max_{\lambda \in \Lambda(A)} |p(\lambda)|$

FOV bound $\|r_m\| \leq 2 \|r_0\| \min_{\substack{p \in \Pi_m \\ p(0)=1}} \max_{\gamma \in W(A)} |p(\gamma)|$

Alternative FOV bound

$$\|r_m\| \leq 2 \|r_0\| \min_{\substack{p \in \Pi_m \\ p(0)=1}} \left[\|P_Q\| \max_{\gamma_1 \in W(Q^* A Q)} p(\gamma_1) + \|P_Y\| \max_{\gamma_2 \in W(Y^* A Y)} p(\gamma_2) \right]$$

Pseudospectrum bound $\|r_m\| \leq \|r_0\| \dfrac{\mathcal{L}(\mathcal{C}_\varepsilon)}{2\pi\varepsilon} \min_{\substack{p \in \Pi_m, \\ p(0)=1}} \max_{\gamma \in \mathcal{C}_\varepsilon} |p(\gamma)|$

Krylov Methods Crash Course

Consider $Ax = b$, initial guess x_0, and residual $r_0 = b - Ax_0$

Compute optimal update z_m from

$$K_m\left(A, r_0\right) = \text{span}\{r_0, Ar_0, \ldots, A^{m-1}r_0\}: \qquad \text{(for example)}$$

$$\min\left\{\left\|b - A\left(x_0 + z\right)\right\|_2 \mid z \in K_m\left(A, r_0\right)\right\} \quad \Leftrightarrow \quad \min_{z \in K_m\left(A, r_0\right)} \left\|r_0 - Az\right\|_2$$

Let $K_m = \left[r_0 \; Ar_0 \; A^2 r_0 \cdots A^{m-1} r_0\right]$, then $z = K_m \zeta$,

and we must solve the following least squares problem

$$AK_m \zeta \approx r_0 \quad \Leftrightarrow \quad \left[Ar_0 \; A^2 r_0 \cdots A^m r_0\right]\zeta \approx r_0$$

$$x_m = x_0 + z_m \quad \text{and} \quad r_m = r_0 - Az_m$$

Do this accurately and efficiently every iteration for increasing m.
One such method: GMRES – Saad/Schulz'86

Minimum Residual Solutions: GMRES

Generate iteration-wise an orthogonal basis for $K_{m+1}(A, r_0)$.

The Arnoldi algorithm (iteration): Let $v_1 = r_0 / \|r_0\|_2$;

```
for $k = 1 \dots m$,
    $\tilde{v}_{k+1} = A v_k$;
    for $j = 1 \dots k$,
        $h_{j,k} = v_j^* \tilde{v}_{k+1}$; $\tilde{v}_{k+1} = \tilde{v}_{k+1} - h_{j,k} v_j$;
    end
    $h_{k+1,k} = \|\tilde{v}_{k+1}\|_2$; $v_{k+1} = \tilde{v}_{k+1} / h_{k+1,k}$;
end
```

Arnoldi recurrence: $A V_m = V_{m+1} \underline{H}_m$

$V_{m+1}^* V_{m+1} = I_{m+1}$ (orthogonal),

$\underline{H}_m = V_{m+1}^* A V_m$ (upper Hessenberg)

Minimum Residual Solutions: GMRES

Using $AV_m = V_{m+1}\underline{H}_m$, we solve $\min\left\{\left\|r_0 - Az\right\|_2 \mid z \in K_m\left(A, r_0\right)\right\}$ as follows.
Let $z = V_m\zeta$, and minimize $\left\|r_0 - AV_m\zeta\right\|_2$ over all m-vectors ζ.
Note that this is an $n \times m$ least squares problem (as before).

Now substitute $r_0 = V_{m+1}\eta_1\left\|r_0\right\|_2$ and $AV_m = V_{m+1}\underline{H}_m$. This gives

$$\left\|V_{m+1}\eta_1\left\|r_0\right\|_2 - V_{m+1}\underline{H}_m\zeta\right\|_2 = \left\|V_{m+1}\left(\eta_1\left\|r_0\right\|_2 - \underline{H}_m\zeta\right)\right\|_2 = \left\|\eta_1\left\|r_0\right\|_2 - \underline{H}_m\zeta\right\|_2$$

The latter is a small $(m+1)\times m$ least squares problem we can solve by standard dense linear algebra techniques (e.g. using LAPACK)

We can exploit the structure of $\underline{H}_m$ and the least squares problem to
1. do this efficiently,
2. compute the residual norm without computing the residual

Minimum Residual Solutions: GMRES

```
GMRES: $Ax = b$
Choose $x_0$, tolerance $\varepsilon$; set $r_0 = b - Ax_0$; $v_1 = r_0 / \|r_0\|_2$, $k = 0$.
while $\|r_k\|_2 \geq \varepsilon$ do
    $k = k + 1$
    $\tilde{v}_{k+1} = Av_k$;
    for $j = 1 \ldots k$,
        $h_{j,k} = v_j^* \tilde{v}_{k+1}$; $\tilde{v}_{k+1} = \tilde{v}_{k+1} - h_{j,k} v_j$;
    end
    $h_{k+1,k} = \|\tilde{v}_{k+1}\|_2$; $v_{k+1} = \tilde{v}_{k+1} / h_{k+1,k}$;
    Solve LS $\min_\zeta \left\| \eta_1 \|r_0\|_2 - \underline{H}_k \zeta \right\|_2$ $\left( = \|r_k\|_2 \right)$ by construction
    (actually we update the solution rather than solve from scratch – see later)
end
$x_k = x_0 + V_k \zeta_k$;
$r_k = r_0 - V_{k+1} \underline{H}_k \zeta_k = V_{k+1} \left( \eta_1 \|r_0\| - \underline{H}_k \zeta_k \right)$ or simply $r_k = b - Ax_k$
```

Conjugate Gradient Method

Hermitian matrices: Error minimization in the A-norm

We are solving $\boldsymbol{Ax} = \boldsymbol{b}$ with initial guess $\boldsymbol{x}_0 \rightarrow \boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{Ax}_0$ and $\hat{x}$ is the solution to $\boldsymbol{Ax} = \boldsymbol{b}$.
The error at iteration $\boldsymbol{i}$ is $\varepsilon_i = \hat{x} - (x_0 + z_i)$,
where $z_i \in \boldsymbol{K}^i(A, \boldsymbol{r}_0)$ is the *ith* update to the initial guess.

Theorem:
Let $\boldsymbol{A}$ be Hermitian, then the vector $z_i \in \boldsymbol{K}^i(A, \boldsymbol{r}_0)$ satisfies
$z_i = \textbf{arg min}\{ \|\hat{x} - (x_0 + z)\|_A : z \in K^i(A, r_0)\}$ iff $\boldsymbol{r}_i \equiv \boldsymbol{r}_0 - \boldsymbol{A}z_i$ satisfies $\boldsymbol{r}_i \perp \boldsymbol{K}^i(A, \boldsymbol{r}_0)$.

The most important algorithm of this class is the Conjugate Gradient Algorithm.

Conjugate Gradients method

```
Solve $Ax = b$, Choose $x_0 \to r_0 = b - Ax_0$
$p_1 = r_0; i = 0$
While $\|r_0\| > \varepsilon$ do
    $i = i + 1;$
    $\alpha_i = \langle r_{i-1}, r_{i-1} \rangle / \langle Ap_{i-1}, p_{i-1} \rangle$
    $x_i = x_{i-1} + \alpha_i p_i;\ r_i = r_{i-1} - \alpha_i Ap_i$
    $\beta_i = \langle r_i, r_i \rangle / \langle r_{i-1}, r_{i-1} \rangle$
    $p_i = r_i - \beta_i p_{i-1}$
End
```

Preconditioning needs to maintain symmetry:

- Precondition on both sides: $\tilde{L}^{-1} A \tilde{L}^{-T}$
- Maintain symmetry wrt to inner products including preconditioner

Preconditioned CG

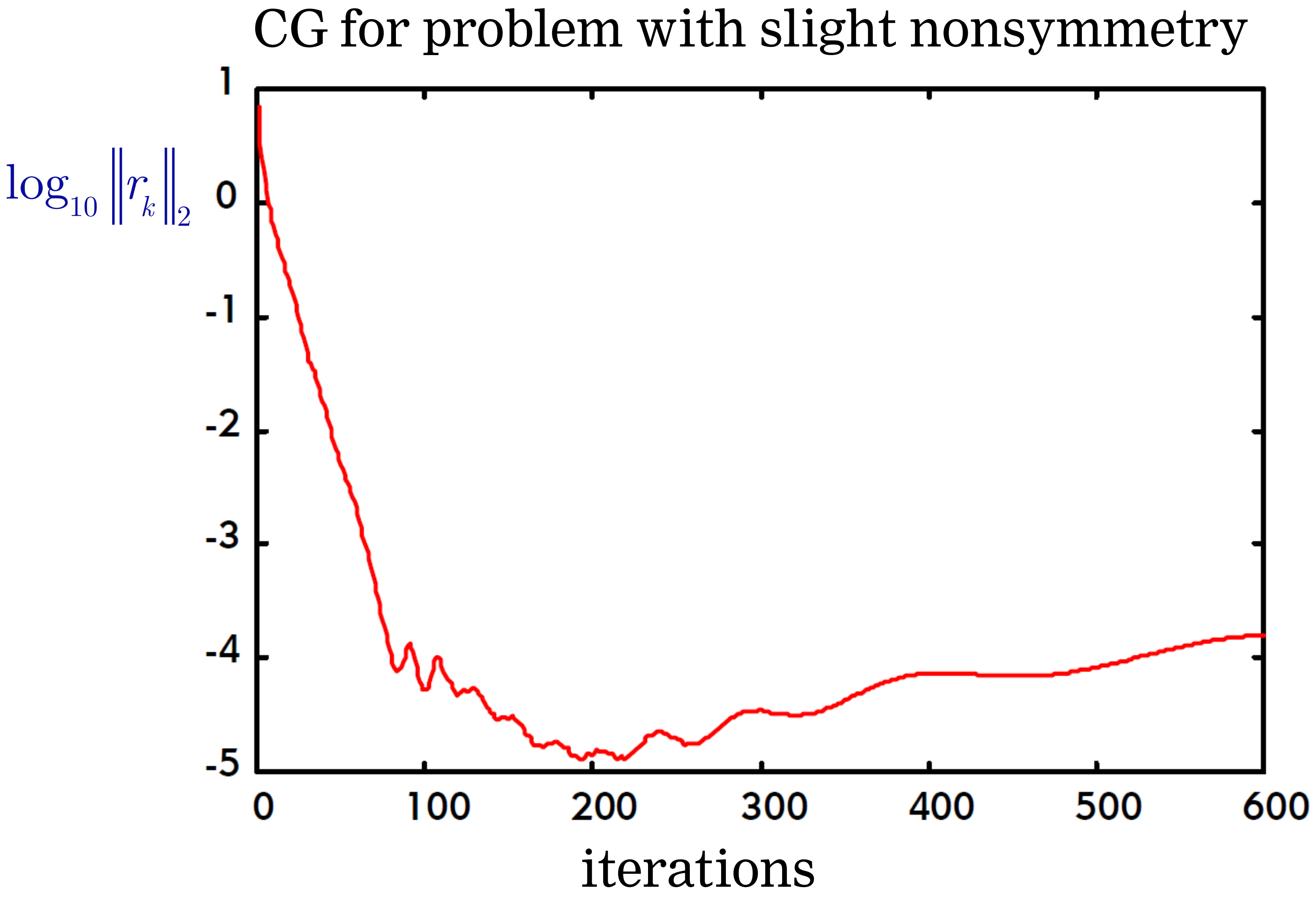
- Preconditioner may be based on some linear operator, say, fast solve for differential operator (Laplacian).
- Hard to 'split' such an operation.
- Change of inner product can help.

Preconditioned matrix $\tilde{L}^{-T}\tilde{L}^{-1}A$, inner product $\left\langle x, y \right\rangle_{\tilde{L}\tilde{L}^T} = y^T \tilde{L}\tilde{L}^T x$

$$\begin{aligned}\left\langle \tilde{L}^{-T}\tilde{L}^{-1}Ax, y \right\rangle_{\tilde{L}\tilde{L}^T} &= y^T \tilde{L}\tilde{L}^T \tilde{L}^{-T}\tilde{L}^{-1}Ax = y^T Ax \\ &= y^T A^T \tilde{L}^{-T}\tilde{L}^{-1}\tilde{L}\tilde{L}^T x = \left\langle x, \tilde{L}^{-T}\tilde{L}^{-1}Ay \right\rangle_{\tilde{L}\tilde{L}^T}\end{aligned}$$

Using $\tilde{L}\tilde{L}^T$ inner product allows to do one-sided preconditioning in CG

CG is Sensitive



BiCGStab

```
Choose $x_0 \rightarrow r_0 = b - Ax_0;\ i = 0;\ \omega_0 = 1$
Choose $\tilde{r}$, typically best random but also $\tilde{r}_0 = r_0$
while $\left\| r_i \right\| > \varepsilon \ \&\ \omega_i \neq 0$, do
    $\rho_{i-1} = \tilde{r}^T r_{i-1}$; if $\rho_{i-1} = 0$ stop (method fails)
    if $i = 1,\ p_i = r_{i-1}$
    else $\beta_{i-1} = \left(\rho_{i-1} / \rho_{i-2}\right)\left(\alpha_{i-1} / \omega_{i-1}\right);\ \ p_i = r_{i-1} + \beta_{i-1}\left(p_{i-1} - \omega_{i-1} v_{i-1}\right)$
    $v_i = Ap_i$
    $\alpha_i = \rho_{i-i} \Big/ \tilde{r}^T v_i;\ s = r_{i-1} - \alpha_i v_i$
    if $\left\| s \right\| < \varepsilon$ break; $x_i = x_{i-1} + \alpha_i p_i$ and stop
    $t = As;\ \omega_i = t^T s \Big/ t^T t$
    $x_i = x_{i-1} + \alpha_i p_i = \omega_i s;\ r_i = s - \omega_i t$
end
```

BiCGStab

- BiCGStab is not based on any optimality property
- In general, convergence is often fairly fast (not much worse than GMRES)
- Method may behave erratically and breakdown
- Iterations, cost per dimension of Krylov space or per matvec , are cheap

- Method does not need symmetry or any other property

Preconditioning

- $PAx = Pb$ or $AP\tilde{x} = b$ or $P_1AP_2\tilde{x} = P_1b$
- Generating the polynomials (basis) requires only a matrix-vector product (matvec)
- Matvec can be done approximately by function evaluation
- Preconditioning is complicated if we do not have the matrix available
    - Precondition inside the function evaluation (limited possibilities)
    - Approximate the linear operator and use approximation to compute a preconditioner
- Many ways to approximate matrix using only matvecs (function evaluations)
- May be expensive but we can update once we have it

Preconditioning

Solve $F'(x_n)s = -F(x_n)$:

Right preconditioning $F'(x_n)M\tilde{s} = -F(x_n)$ and $s = M\tilde{s}$

Left preconditioning $MF'(x_n)s = -MF(x_n)$

Left preconditioner: $MF'(x_n)w \approx (1/\varepsilon)M\left[F(x_n + \varepsilon w) - F(x_n)\right]$

Right preconditioner: $F'(x_n)Mw \approx (1/\varepsilon)\left[F(x_n + \varepsilon Mw) - F(x_n)\right]$

Often include the preconditioning in nonlinear system:

$G(x) = F(Mx) = 0$ or $G(x) = MF(x)$

$F'(Mx)Ms = -F(Mx)$ $\qquad G'(x) = MF'(x)$

Drawback is that preconditioner fixed over nonlinear iterations

Further discussion

- GMRES robust but expensive unless convergence fast – $O(nm^2)$
- Can restart but often bad for convergence
- This has prompted methods with smarter 'restart'
  - GMRESDR (Morgan), GCROT (dS), GMRESR/* (vdVorst/Vuik)
- Implicit preconditioning may lead to varying preconditioner per iteration – FGMRES (Saad),
- GMRESR/* (vdVorst/Vuik'94), GCRO(T) (dS'95 '99) allow varying preconditioner in so-called inner iterations
- Basic idea underlying these variants is that some of the algebraic relations (optimality over subspace) in Krylov methods are preserved, whereas others (which search space) are relaxed

Probing

We can think of the previously generated search spaces and their image under the matrix (Jacobian) as building a preconditioner (using only matvecs).

This idea can be exploited more generally. Use matvecs with selected vectors to approximate or reconstruct the matrix and then build a preconditioner:

Probing or sparse Jacobian/Hessian approximation

How to build a matrix approximation using only matvecs?

- Know or compute or guess structure of the matrix and define an (approximate) nonzero pattern
- Use discrete optimization to compute the ‘probing’ vectors and minimize (approximately) the number of vectors.
- Compute approximate inverse of approximate Jacobian

Probing

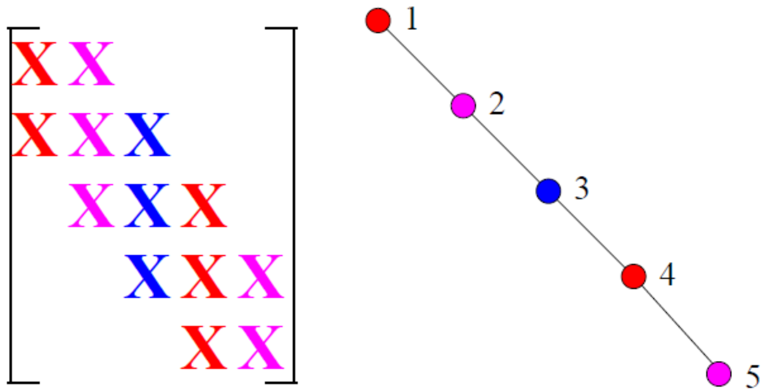
- Idea: Multiply with carefully chosen vectors of 1's and 0's (Curtis, Powell & Reid 1974).
- Example:

$$\begin{bmatrix} a_1 & b_2 & & & \\ c_1 & a_2 & b_3 & & \\ & c_2 & a_3 & b_4 & \\ & & c_3 & a_4 & b_5 \\ & & & c_4 & a_5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} a_1 & b_2 & 0 \\ c_1 & a_2 & b_3 \\ b_4 & c_2 & a_3 \\ a_4 & b_5 & c_3 \\ c_4 & a_5 & 0 \end{bmatrix}$$

Uses vectors $e_1 + e_4$, $e_2 + e_5$ and e_3.

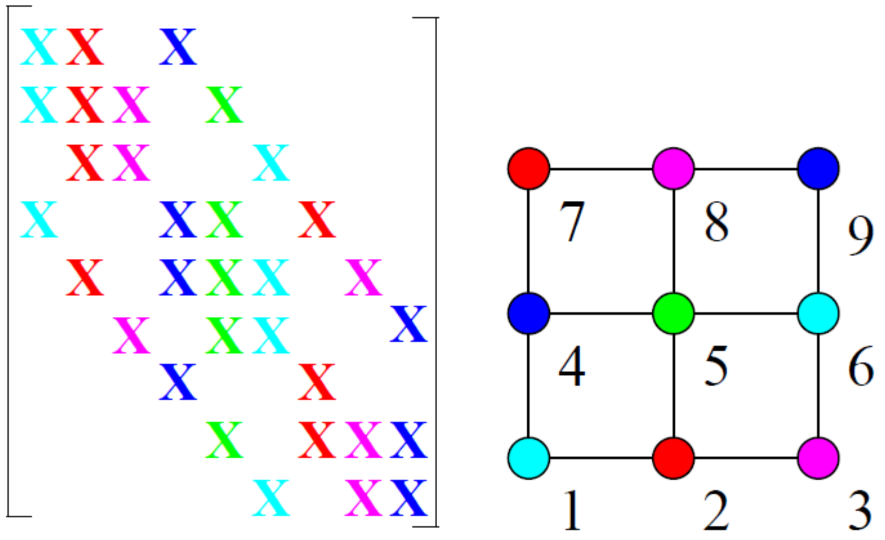
Probing

- This is a graph coloring problem (Coleman & Moré 1983).
- Adjacency graph of A: Has (i, j) edge if $A_{i,j}$ is non-zero.
- Problem: Distance-2 coloring of adjacency graph of matrix (McCormick 1983).



Probing

- For non-banded matrices, we can use the same technique (Coleman & Moré 1983; Cullum & Tuma 2004).



- Survey Article: Gebremedhin, Manne & Pothen, 2005.

Probing methods for saddle-point problems, Siefert & de Sturler, Elec. Trans. Numer. Anal. (ETNA), 2006

Probing

$$\tilde{A} = \textbf{Structured Probing}(A \in \mathbb{R}^{n \times n})$$

1. Choose a sparsity pattern H for the output matrix $\tilde{A}$.
2. Color adjacency graph of H and generate vector of colors.
3. Generate probing vectors $x_1, \ldots, x_p$, one for each of p colors.
4. For each probing vector, x_i, multiply $w_i = Ax_i$.
5. Build $\tilde{A}$ using sparsity pattern H and vectors $w_1, \ldots, w_p$.

Note: If H and A have same sparsity pattern, $\tilde{A} = A$.

Note: Number of colors (p) small, regardless of problem size

$\Rightarrow$ SP is cheap!

FGMRES

FGMRES maintains an Arnoldi-like recurrence

$AZ_m = V_{m+1}\underline{H}_m$ where $V_{m+1}^T V_{m+1} = I_m$ and $Z_m = \left[P_1 v_1 \ldots P_m v_m\right]$

Update $z_m = Z_m \zeta$ is not from $K_m\left(A, r_0\right)$ or $K_m\left(PA, Pr_0\right)$

Algebraic structure and orthonormal columns of V_{m+1} allow similar minimization as for GMRES

Varying preconditioner may be useful

- when some function is applied to multiply vector by preconditioner and this leads to variation in exact operator
- When iteration is used to approximate a linear operator (for example multigrid iteration for fast Poisson solver)

FGMRES (Saad'93)

```
Solve $Ax = b$        allow variable preconditioner
Choose $x_0$, tolerance $\varepsilon$; set $r_0 = b - Ax_0$; $v_1 = r_0 / \|r_0\|_2$, $k = 0$.
while $\|r_k\|_2 \geq \varepsilon$ do
    $k = k + 1$
    $z_k = P_k v_k$;   $\tilde{v}_{k+1} = Az_k$
    for $j = 1 \ldots k$,
        $h_{j,k} = v_j^* \tilde{v}_{k+1}$; $\tilde{v}_{k+1} = \tilde{v}_{k+1} - h_{j,k} v_j$;
    end
    $h_{k+1,k} = \left\|\tilde{v}_{k+1}\right\|_2$; $v_{k+1} = \tilde{v}_{k+1} / h_{k+1,k}$;        $\{AZ_k = V_{k+1} \underline{H}_k\}$
    Pick update $z_k = Z_k \zeta_k$
    Solve LS $\min_\zeta \left\|V_{k+1}\eta_1 \left\|r_0\right\| - AZ_k \zeta\right\| = \min_\zeta \left\|\eta_1 \left\|r_0\right\|_2 - \underline{H}_k \zeta\right\|_2 \;\left( = \left\|r_k\right\|_2 \right)$
end
$x_k = x_0 + Z_k \zeta_k$;
$r_k = r_0 - V_{k+1} \underline{H}_k \zeta_k = V_{k+1}\left(\eta_1 \left\|r_0\right\| - \underline{H}_k \zeta_k\right)$ or simply $r_k = b - Ax_k$
```

Other Variants

Similar ideas underly other solver variants. Consider GCR method (Eisenstat, Elman, Schultz'83) – algebraically equivalent to GMRES but more expensive, possible breakdown.

```
GCR: $Ax = b$
Choose $x_0$ (e.g. $x_0 = 0$) and tolerance $\varepsilon$; set $r_0 = b - Ax_0$; $i = 0$
while $\|r_i\|_2 \geq \varepsilon$ do
    $i = i + 1$; $u_i = r_{i-1}$; $c_i = Au_i$
    for $j = 1, \ldots, i-1$ do
        $u_i = u_i - u_j c_j^* c_i$; $c_i = c_i - c_j c_j^* c_i$
    end
    $u_i = u_i / \|c_i\|_2$; $c_i = c_i / \|c_i\|_2$
    $x_i = x_{i-1} + u_i c_i^* r_{i-1}$; $r_i = r_{i-1} - c_i c_i^* r_{i-1}$
end
```

Solver Variants

GCR builds following algebraic relations:

$$\text{Range}(U_m) = K_m(A, r_0), \quad AU_m = C_m, \quad \text{and} \quad C_m^T C_m = I_m$$

At each step the method computes minimum 2-norm residual by setting $r_m \perp C_m$.

$$r_m = r_0 - C_m C_m^T r_0 \quad \text{and} \quad x_m = x_0 + U_m C_m^T r_0 = x_0 + A^{-1} C_m C_m^T r_0$$

Krylov space as search space arises because $u_k = r_k$, but this can be generalized. Algebraically, any vector is okay (though probably not equally successful).

So, replace residual in $u_k = r_k$ by any good approximation to the error (the error would give convergence in one step).

Solver Variants

In GMRES* $u_k = p_m\left(A\right)r_0$ or $u_k = p_m\left(AP\right)r_0$ computed by m steps of an inner Krylov method (if GMRES then called GMRESR)

In GCRO $u_k = A^{-1}p_m\left(\left(I - C_kC_k^T\right)A\right)r_0$ or

$$u_k = A^{-1}p_m\left(\left(I - C_kC_k^T\right)AM\right)r_0$$

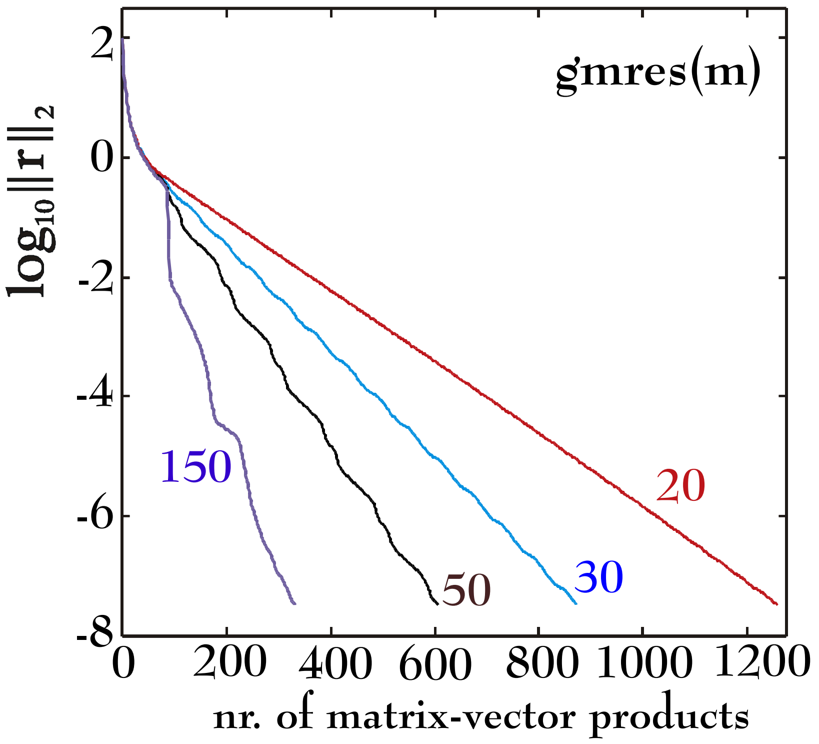
again computed by some inner Krylov method, while maintaining orthogonality to outer search space (optimality)
If GMRES used inside then optimality over entire search space
$\text{Range}\left(U_k\right) + \text{Range}\left(V_m\right)$

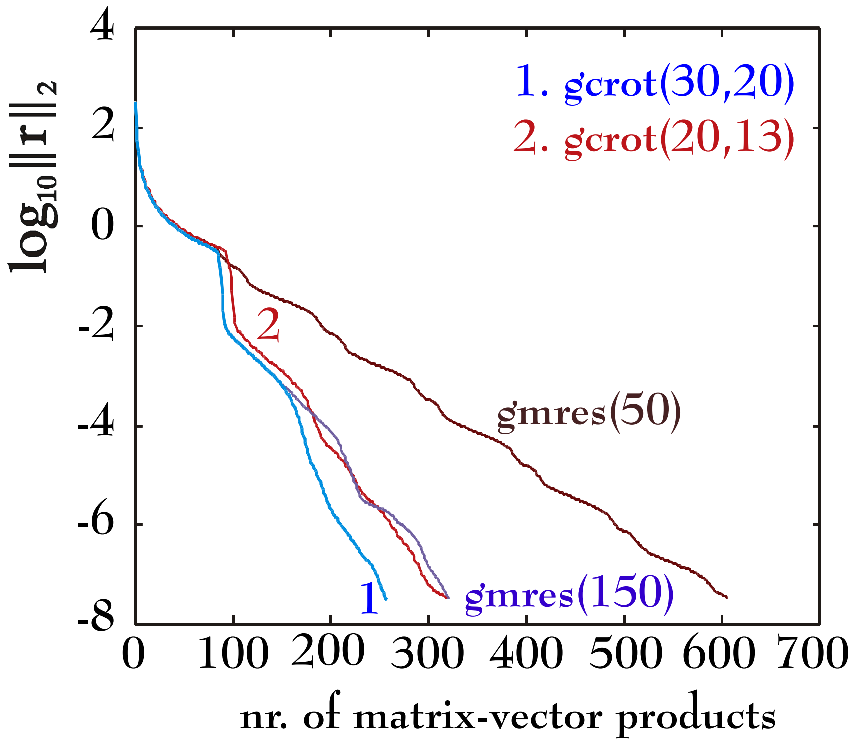
But search space no longer Krylov space – in return significant flexibility. This formed the basis for methods that *recycle Krylov subspaces* for sequences of slowly changing systems (2nd talk).

GCROT: Selective orthogonality

- Restarted GMRES versus GCROT, which maintains orthogonality against sequence of selected subspaces.
- Time-wise the advantage of GCROT is even larger as working with a smaller subspace is much faster.

Convection-diffusion problem with strong convection





Good reading

- GMRES: a generalized minimal residual algorithm for solving a nonsymmetric linear systems, Saad, Schultz, SIAM SSISC, 1986.
- A flexible inner-outer preconditioned GMRES algorithm, Saad, SIAM J. Sci. Statist. Comput., 1993
- GMRESR: a family of nested GMRES methods, van der Vorst, Vuik, Num. Lin. Alg. Appl., 1994
- BiCGStab: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, van der Vorst, SIAM SISC 1992
- Nested Krylov methods based on GCR, de Sturler, J. Comp. Appl. Math., '96 (GCRO)
- Truncation Strategies for optimal Krylov subspace methods, de Sturler, SIAM SISC, 1999 (GCROT)
- Recycling Krylov Subspaces for Sequences of Linear Systems, Parks, de Sturler, Mackey, Johnson, Maiti, SIAM SISC 28(5), 2006
- Recycling Subspace Information for Diffuse Optical Tomography, Kilmer, de Sturler, SIAM SISC 27(6), 2006
- Probing methods for saddle-point problems, Siefert, de Sturler, Elec. Trans. Numer. Anal. (ETNA), 2006

Good Reading

- Preconditioners for generalized saddle-point problems, PhD Thesis, Siefert, UIUC 2006
- Improved Scaling for Quantum Monte Carlo on Insulators, Ahuja, Clark, de Sturler, Ceperley, and Kim, SIAM SISC 33(4), 2011
- Recycling Krylov Subspaces and Preconditioners, PhD Thesis, Ahuja, Virginia Tech, 2011
- Large-scale topology optimization using preconditioned Krylov subspace methods with recycling, Wang, de Sturler, Paulino, IJNME 69, 2007
- www.math.vt.edu/people/sturler/index.html