MA 580; Iterative Methods for Linear Equations

C. T. Kelley
NC State University
tim_kelley@ncsu.edu

Version of October 10, 2016

Read Chapters 2 and 3 of the Red book.

NCSU, Fall 2016

Part VIb: Krylov Methods for Linear Equations: GMRES

Don't forget the notation

- We're solving $\mathbf{Ax} = \mathbf{b}$ for $\mathbf{x}^* = \mathbf{A}^{-1}\mathbf{b}$ in R^N.
- $\{\mathbf{x}_n\}_{n=0}^{\infty}$ is the sequence of iterates.
- $\mathbf{x}_0$ is the intial iterate (not guess).
- $\mathbf{e} = \mathbf{x}^* - \mathbf{x}$ is the error.
- $\mathbf{r} = \mathbf{b} - \mathbf{Ax} = \mathbf{Ae}$ is the residual.

What Krylov Methods Do

- Krylov iterative methods obtain $\mathbf{x}_n$ from the history of the iteration.
- The ones with theory do this by minimizing an error or residual function over the affine space

$$\mathbf{x}_0 + \mathcal{K}_k$$

- $\mathbf{x}_0$ is the initial iterate
- $\mathcal{K}_k$ is the kth Krylov subspace

$$\mathcal{K}_k = \text{span}(\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{k-1}\mathbf{r}_0)$$

for $k \geq 1$.

GMRES and Conjugate Gradient (CG)

These two methods can be expressed in terms of minimization principles

In GMRES (Generalized Minimum Residual), the kth iteration $\mathbf{x}_k$ minimizes the residual over $\mathbf{x}_0 + \mathcal{K}_k$

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}_k\| = \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_k} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|$$

for $\|\cdot\| = \|\cdot\|_2$. For CG, A must be spd and $\mathbf{x}_k$ minimizes the A-norm of the error

$$\|\mathbf{x}^* - \mathbf{x}\|_A = \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_k} \|\mathbf{x}^* - \mathbf{x}\|_A$$

where

$$\|\mathbf{v}\|_{\mathbf{A}}^2 = \mathbf{v}^T A \mathbf{v}.$$

Analysis of GMRES

If $\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_k$ then

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x} = \mathbf{b} - \mathbf{A}\mathbf{x}_0 - \sum_{j=1}^{k} \gamma_j \mathbf{A}^j \mathbf{r}_0 \equiv p(\mathbf{A})\mathbf{r}_0$$

where $p \in \mathcal{P}_k$, the set of k degree residual polynomials.

$$\mathcal{P}_k = \{p \,|\ \ p \text{ is a polynomial of degree } k \text{ and } p(0) = 1.\}$$

This simple observation is the key to analysis of Krylov methods.

GMRES and Residual Polynomials

Theorem: Let $\mathbf{A}$ be nonsingular and let $\mathbf{x}_k$ be the kth GMRES iteration. Then for all $\bar{p} \in \mathcal{P}_k$

$$\|\mathbf{r}_k\| = \min_{p \in \mathcal{P}_k} \|p(\mathbf{A})\mathbf{r}_0\| \leq \|\bar{p}(\mathbf{A})\mathbf{r}_0\|.$$

Proof of Theorem

Let $\mathbf{x}_k$ the the kth GMRES iteration. Then there is $p_k \in \mathcal{P}_k$ such that

$$\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k = p_k(\mathbf{A})\mathbf{r}_0$$

Since any $\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_k$ satisfies

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x} = \bar{p}(\mathbf{A})\mathbf{r}_0$$

for some $\bar{p} \in \mathcal{P}_k$, the minimization principle implies that

$$\|\mathbf{r}_k\|_2 = \min_{p \in \mathcal{P}_k} \|p(\mathbf{A})\mathbf{r}_0\| = \|\bar{p}_k(A)\mathbf{r}_0\| \leq \|\mathbf{b} - \mathbf{A}\mathbf{x}\| = \|\bar{p}(\mathbf{A})\mathbf{r}_0\|.$$

How to Use This Theorem

- Connect properties of the matrix to a polynomial you understand.
- Manufacture a residual polynomial $\bar{p}$ from that
- Get an upper bound from

$$\|\mathbf{r}_k\| \leq \|\bar{p}(\mathbf{A})\mathbf{r}_0\| \leq \|\bar{p}(\mathbf{A})\|\|\mathbf{r}_0\|$$

Consequences of the Minimization Principle: I

Corollary: Let $\mathbf{A}$ be nonsingular. Then the GMRES algorithm will find the solution within N iterations.
Proof: The characteristic polynomial of $\mathbf{A}$ is $p(z) = det(\mathbf{A} - z\mathbf{I})$. p has degree N, $p(0) = det(\mathbf{A}) \neq 0$ since $\mathbf{A}$ is nonsingular, and so

$$\bar{p}_N(z) = p(z)/p(0) \in \mathcal{P}_N$$

is a residual polynomial. The Cayley-Hamilton theorem says that $\bar{p}_N(\mathbf{A}) = 0$, and so

$$\|\mathbf{r}_N\| \leq \|\bar{p}_N(\mathbf{A})\|\|\mathbf{r}_0\| = 0.$$

Consequences of the Minimization Principle: II

Corollary: If $\|\mathbf{I} - \mathbf{A}\| \leq \rho < 1$ then

$$\|\mathbf{r}_k\| \leq \rho^k \|\mathbf{r}_0\|_2.$$

Proof: Let $\bar{p}_k = (1 - z)^k$ and use the theorem.

Consequences of the Minimization Principle: III

Corollary: If $\mathbf{b}$ is a linear combination of k eigenvectors of $\mathbf{A}$, and let $\mathbf{x}_0 = 0$, then GMRES will converge it at most k iterations.
Proof: Let $\mathbf{b} = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i$ where $\mathbf{A}\mathbf{u}_i = \lambda_i \mathbf{u}_i$. Let $\bar{p} = \prod(\lambda_i - z)/\lambda_i$. Then $\bar{p}(A)\mathbf{r}_0 = \bar{p}(A)\mathbf{b} = 0$ (can you see why?)
We're done since

$$\|\mathbf{r}_k\| \leq \|\bar{p}(\mathbf{A})\mathbf{r}_0\| = 0.$$

Diagonalizable Matrices

A is diagonalizable if there is a nonsingular (**possibly complex!**) matrix **V** such that

$$\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}.$$

If **A** is diagonalizable and p is a polynomial then

$$p(\mathbf{A}) = \sum_{j=0}^{m} a_j\gamma_j\mathbf{A}^j = \sum_{j=0}^{m} a_j(\mathbf{V}\Lambda\mathbf{V}^{-1})^j = \mathbf{V}\sum_{j=0}^{m} a_j\Lambda^j\mathbf{V}^{-1} = Vp(\Lambda)\mathbf{V}^{-1}$$

So

$$\|p(\mathbf{A})\| \leq \|\mathbf{V}\|\|p(\Lambda)\|\|\mathbf{V}^{-1}\| = \kappa(\mathbf{V}) \max_{\lambda\in\sigma(\mathbf{A})} |p(\lambda)|$$

GMRES Convergence for Diagonalizable Matrices

We just proved . . .

Theorem: Let $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$ be a nonsingular diagonalizable matrix. Let $\mathbf{x}_k$ be the kth GMRES iterate. Then for all $\bar{p}_k \in \mathcal{P}_k$

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \kappa_2(\mathbf{V}) \max_{z \in \sigma(\mathbf{A})} |\bar{p}_k(z)|.$$

Easy Results for Diagonalizable **A**: I

If **A** has m distinct eigenvalues then GMRES will terminate in at most m iterations.

Proof: Use

$$p(z) = \prod_{i=1}^{m} \left(\frac{\lambda_i - \lambda}{\lambda_i} \right)$$

$p(0) = 1$ so $p \in \mathcal{P}_k$. Since $p(\lambda_i) = 0$ for all i, $\mathbf{r}_N = 0$.

This proof is (1) very easy and (2) typical of the way one thinks about Krylov methods.

Easy Results for Diagonalizable **A**: II

Let $\mathbf{x}_0 = 0$ (so $\mathbf{r}_0 = \mathbf{b}$) and assume that

- $\sigma(\mathbf{A}) \subset (9, 11)$
- $\kappa(\mathbf{V}) = 100$

Then if we let $\bar{p}_k(z) = (10 - z)^k / 10^k$ we see that

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{b}\|_2} \leq \kappa(\mathbf{V}) \|p_k(\Lambda)\| \leq (100) 10^{-k} = 10^{2-k}.$$

So $\|\mathbf{r}_k\| \leq \eta \|\mathbf{b}\|$ when

$$k > 2 - \log_{10}(\eta).$$

This tells us that an approximate inverse preconditioner could be useful.

Performance of GMRES

For this example, how many iterations do you need to reduce the residual by a factor of 10^6.
Use the formula with $\eta = 10^{-6}$ and get

$$k > 2 - \log_{10}(\eta) = 8.$$

More clustering stuff for diagonalizable **A**

Suppose

- $\kappa(\mathbf{V}) = 1$ (eg $\mathbf{V}$ is orthogonal)
- $\sigma(\mathbf{A}) \subset (9, 11) \cup (19, 21) \cup \{7\}$

We need a residual polynomial that takes care of both clusters and the isolated point 7. How about

$$\bar{p} = \frac{7 - z}{7} \left(\frac{10 - z}{10} \frac{20 - z}{20} \right)^k \in \mathcal{P}_{2k+1}.$$

Clearly $\bar{p}(7) = 0$.

What about the clusters?

If $z \in (9, 11)$, then

$$|\bar{p}(z)| \leq \frac{11-7}{7} \frac{1}{10^k} \frac{11^k}{20^k} = \frac{11^k}{200^k} \leq 1/9^k.$$

If $z \in (19, 21)$, then

$$|\bar{p}(z)| \leq \frac{21-7}{7} \frac{11^k}{10^k} \frac{1}{20^k} = 2\frac{11^k}{200^k} \leq 2/9^k.$$

Bottom line: $\bar{p} \in \mathcal{P}_{2k+1}$ and

$$\max_{z \in \sigma(\mathbf{A})} |\bar{p}(z)| \leq 2/9^k$$

Performance of GMRES

For this example, how many iterations do you need to reduce the residual by a factor of 10^6.
Answer: $2k+1$ where $2/9^k < 10^{-6}$.

- So $-k \log_{10} 9 + \log_{10} 2 < -6$
- and so $k > (6 + \log_{10} 2)/log_{10} 9 \approx 6.6$

So $k = 7$ works and the answer is $2k + 1 = 15$.

Normal Matrices

- $\mathbf{A}$ is normal if $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T$.
- $\mathbf{A}$ normal implies $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{\#}$.
  - Here $\mathbf{V}$ (complex!) is unitary (i.e. $\mathbf{V}^{-1} = \mathbf{V}^{\#}$)
  - $\mathbf{V}^{\#}$ is the complex conjugate transpose.
- $\mathbf{A}$ normal implies $\kappa(\mathbf{V}) = 1$

General Observations for diagonalizable matrices

- The heuristic is that if the eigenvalues are grouped into a few clusters the iteration will perform well.
- If $\kappa(\mathbf{V})$ is large, then $\sigma(\mathbf{A})$ does not tell the whole story.
- If the eigenvalues are clustered near 1, then GMRES is very happy and $\mathbf{A}$ is well-conditioned.

If $\mathbf{A}$ is **not diagonalizable** then clusters don't help you.

Preconditioning

Preconditioning means to replace $\mathbf{A}x = b$ with

$$\mathbf{BAx} = \mathbf{Bb} \text{ (left)}$$

or

$$\mathbf{ABy} = \mathbf{b} \text{ (right), and then } \mathbf{x} = \mathbf{By}$$

and solve the preconditioned equation with GMRES. The preconditioner $\mathbf{B}$ should be

- very inexpensive matrix-vector products
- be a good approximate inverse of (part) of $\mathbf{A}$

Examples coming later.

Left Preconditioning

Solve

$$\mathbf{BAx} = \mathbf{Bb}$$

so

- solution to preconditioned equation is still $\mathbf{x}$
- preconditioned residual $\mathbf{Bb} - \mathbf{BAx} = \mathbf{Br}$ should be a better indicator of error

Right Preconditioning

Solve

$$\mathbf{ABz} = \mathbf{b}$$

for z. Then set $\mathbf{x} = \mathbf{Bz}$.

- The preconditioned residual is the same as the original residual because $\mathbf{b} - \mathbf{A}(\mathbf{Bz}) = \mathbf{b} - \mathbf{Ax}$.
- The solution of the preconditioned problem is different.
- The residual may not be a good indicator of the error **in x**.

Incomplete Factorizations

If you can store A as a sparse matrix then

- you can start a sparse factorization,
- and discard small elements in the factors,
- or enforce sparsity.

The MATLAB commands `ilu` and `ichol` create incomplete LU and Cholseky factorizations.

Integral Equations

- Many integral equations of the form

$$u(x) - \int k(x,y)u(y)\,dy = f(x) \text{ Fredholm } 2^{nd} \text{ kind}$$

  are well conditioned and GMRES does well.
- The transport equation is one example.
- Krylov method performance is mesh-independent.
- Preconditioning can still make a difference.

Elliptic PDEs I

Suppose you seek to solve an elliptic boundary value problem.

$$Lu = f$$

with Dirichlet/Neumann/mixed boundary conditions.
If you discretize the PDE to obtain

$$\mathbf{L}_h \mathbf{u}_h = \mathbf{f}_h$$

the resulting discrete problem is very poorly conditioned and Krylov methods will be slow.

Elliptic PDEs II

Split $L = L_1 + L_0$, where L_1 contains the high-order derivatives.
If you can find a fast solver for L_1 with the same type of boundary conditions, then L_1^{-1} is a mesh-independent preconditioner.
Why? $L_1^{-1}L$ is an integral operator. (Manteufel/Parter 1990)

Example of PDE preconditioning

- $-\nabla^2 u + c_1 u_x + c_2 u_y + c_0 u = f(x,y)$ for $0 < x, y < 1$
- $u(x,0) = u(0,y) = u(x,1) = u(1,y) = 0$
- $L_1 u = -\nabla^2 u$
- Apply fast Poisson solver $N \log(N)$ work or use an efficient sparse direct solver.

Scalability

The scenario:

- Continuous problem: $Lu = f$; Discrete problem: $\mathbf{L}_h\mathbf{u}_h = \mathbf{f}_h$.
- $h = 1/N$ spatial mesh width; N^2 number of mesh points.
- Second order accuracy: $\mathbf{u}_h - u^* = O(h^2)$
- Preconditioner $\mathbf{B}_h$ is "perfect", i.e. Krylovs needed to reduce error by factor of 10 is N_k for all h.
- Cost of $\mathbf{B}_h\mathbf{L}_h$ matvec is $O(N)$

Then, given h you can find $\mathbf{u}_h$ up to truncation error in $O(N)$ work!

Fast Solvers

Pick $h_0 = 2^p h$ so that $L_{h_0} u_{h_0} = f_{h_0}$ is easy to solve.
Solve $\mathbf{L}_{h_0} \mathbf{u}_0 = \mathbf{f}_{h_0}$
for l=1:p **do**
  $h_l = h_{l-1}/2$; $\mathbf{u}_l^0 = \mathbf{u}_{l-1}$
  Apply GMRES to $\mathbf{L}_{h_l} \mathbf{u}_l = \mathbf{f}_{h_l}$ with $\mathbf{u}_l^0$ as the start.
  Terminate when residual is reduced by factor of 10.
  Accept $\mathbf{u}_l$
end for

Cost Analysis

- A matvec for $h_l = 2^l h$ costs $O((2^{-l}N)^2)$ operations
- We do at most N_k matvecs at each level
- So ...

$$\text{Cost} \quad \leq \sum_{l=0}^{p} N_k (2^{-l}N)^2 \leq \sum_{l=0}^{\infty} N_k (2^{-l}N)^2$$

$$= N_k N \sum_{l=0}^{\infty} 4^{-l} = 4 N_k N^2/3.$$

GMRES Implementation

The kth GMRES iteration is the solution of the linear least squares problem

$$\min \|\mathbf{Ax} - \mathbf{b}\|$$

where $\mathbf{x} = \sum_{j=0}^{k-1} \gamma_j \mathbf{A}^j \mathbf{r}_0$

The key to a successsful implementation is to solve this in an efficient and stable way.

A Questionable GMRES Implementation

How about this?

- As the iteration progresses store $\mathbf{A}^j \mathbf{r}_0$.
- Let $\mathbf{B}_k = (\mathbf{r}_0, A\mathbf{r}_0, \ldots \mathbf{A}^{k-1}\mathbf{r}_0)$
- Compute the QR factorization of $\mathbf{B}_k = \mathbf{Q}_k \mathbf{R}_k$
- The $\mathbf{x}_k = \mathbf{R}_k^{-1} \mathbf{Q}_k^T b$

What could go wrong?

What could go wrong?

- Accumulating $\mathbf{A}^j\mathbf{r}_0$ can be unstable
  Example $\mathbf{A} = diag(1, 2, ..., N)$
- The cost of $\mathbf{B}_k = \mathbf{Q}_k\mathbf{R}_k$ is $O(Nk^2)$.
- You have to start over with each k and are not reusing the old columns.

Arnoldi Factorization is Better

Suppose one had an orthogonal projector $\mathbf{V}_k$ onto $\mathcal{K}_k$.
Then any $z \in \mathcal{K}_k$ can be written as

$$z = \sum_{l=1}^{k} \mathbf{y}_l v_l^k$$

where v_l^k is the lth column of $\mathbf{V}_k$.
So we can convert the problem for $\mathbf{x}_k$ to a problem in R^k.
Begin by writing any $\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_k$ as

$$x = \mathbf{x}_0 + \mathbf{V}_k y,$$

where y is the vector of coefficients of $\mathbf{x} - \mathbf{x}_0$ using the columns of $\mathbf{V}_k$ as the basis for $\mathcal{K}_k$.

Arnoldi Part II

So if $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{V}_k\mathbf{y}_k$ then

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}_k\| = \|\mathbf{b} - A(\mathbf{x}_0 + \mathbf{V}_k\mathbf{y}_k)\|_2 = \|\mathbf{r}_0 - A\mathbf{V}_k\mathbf{y}_k\|_2.$$

So the least squares problem for $\mathbf{y}_k$ is

$$\min \|\mathbf{r}_0 - A\mathbf{V}_k\mathbf{y}\|$$

If we can build $\mathbf{V}_k$ in a stable way, we have solved the stability problem (but that is not completely simple).
Can we do it efficiently?

Arnoldi Part III

The Gram-Schmidt process will

- build $\mathbf{V}_k$ incrementally, so $\mathbf{V}_k = (\mathbf{V}_{k-1}, \mathbf{v}_k)$,
- enable a fast QR factorization of $\mathbf{AV}_k$, and
- be stable (if done correctly).

Orthogonalization is the central part of the Arnoldi method.

Arnoldi Part IV

The algorithm orthogonalizes each $\mathbf{A}\mathbf{v}_i$ against the columns of $\mathbf{V}_k$ to construct $\mathbf{v}_{k+1}$

$V = \texttt{arnoldi}(\mathbf{x}_0, b, A, k)$

$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$; $\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|$

for $i = 1 : k$ **do**

$w = \mathbf{A}\mathbf{v}_i$

for $j = 1 : i$ **do**

$h_{ji} = w^T\mathbf{v}_j (= (\mathbf{A}\mathbf{v}_i)^T\mathbf{v}_j)$; $w = w - h_{ji}\mathbf{v}_j$

end for

$h_{ki} = \|w\|$; $\mathbf{v}_{i+1} = w/h_{ki}$

end for

Examine the Arnoldi Loops

What if you divide by zero in

$$\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\| \text{ or } \mathbf{v}_{i+1} = w/\|w\|?$$

- If $\mathbf{r}_0 = 0$, then $\mathbf{x}_0$ is the solution and the GMRES iteration would terminate.
- If $w = 0$, then you have a happy breakdown of the Arnoldi process. This implies that you found the solution as $\mathbf{x}_{k-1}$.
- A well-designed implementation would stop before division by zero.

Results from Arnoldi

- $\mathbf{V}_{k+1}$: orthonormal basis for $\mathcal{K}_k$.
- $\mathbf{H}_k$: $(k+1) \times k$ **upper Hessenberg** matrix

and the **Arnoldi factorization**

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_{k+1}\mathbf{H}_k$$

So, setting $\beta = \|\mathbf{r}_0\|_2$.

$$\begin{aligned} \|\mathbf{r}_0 - A\mathbf{V}_k\mathbf{y}\|_2 &= \|\mathbf{r}_0 - \mathbf{A}\mathbf{V}_k\mathbf{y}\|_2 = \|\mathbf{r}_0 - \mathbf{V}_{k+1}\mathbf{H}_k\mathbf{y}\| \\ &= \|\mathbf{V}_{k+1}(\beta\mathbf{e}_1 - \mathbf{H}_k\mathbf{y})\|_2 \\ = \|\beta\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2 & \end{aligned}$$

Why Arnoldi is good

We have converted the $N \times k$ least squares problem for the GMRES iterate $\mathbf{x}_k$ to

$$\min \|\beta \mathbf{e}_1 - \mathbf{H}_k \mathbf{y}\|_2$$

which is $(k+1) \times k$ and $\mathbf{H}_k$ is upper Hessenberg.

- So we can solve it with Givens rotations.
- When done cleverly, it's $O(k^2)$ work/iteration to keep the QR factorization of $\mathbf{A}$ up-to-date and compute $\mathbf{y}$.

Details: See Ch 3 of the red book

- I did the orthogonalization with CGS. That's not very good.
  - Use CGS2 and make sure that $\mathbf{v}_{k+1}$ and the kth column of $\mathbf{H}$ come out correctly.
- Do the Givens rotations astutely.
- We never compute $\|\mathbf{r}\|$ directly.

A Framework for GMRES Implementation

$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$, $\mathbf{v}_1 = \mathbf{r}/\|\mathbf{r}\|_2$, $\rho = \|\mathbf{r}\|_2$, $\beta = \rho$, $k = 0$
while $\rho > \epsilon\|\mathbf{b}\|_2$ and $k < kmax$ **do**
  $k = k + 1$
  Apply Arnoldi (orthogonalization!)
  to obtain $\mathbf{H}_k$ and $\mathbf{V}_{k+1}$ from $\mathbf{V}_k$ and $\mathbf{H}_{k-1}$
  $\mathbf{e}_1 = (1, 0, \ldots, 0)^T \in R^{k+1}$
  Solve $\min \|\beta\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}_k\|_{R^{k+1}}$ for $\mathbf{y}_k \in R^k$.
  $\rho = \|\beta\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}_k\|_{R^{k+1}}$.
end while
$\mathbf{x}_k = \mathbf{x}_0 + \mathbf{V}_k\mathbf{y}_k$.

What you need to remember.

- Do not write your own GMRES code! Use tools!
- Costs are dominated by
  - The matrix-vector products $\mathbf{Av}$
  - The orthogonalization of $\mathbf{v}_{k+1}$ against columns of $\mathbf{V}_k$ Work and storage increase with k.
- Storage increases with k. This can be very bad.
- Poor orthogonalization means
  - $\rho = \|\beta\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}_k\|$ is a poor approximation to $\mathbf{r}_k$
  - So you'll terminate with incorrect results.

Bad things happen with loss of orthogonality.

You knew this, but I will tell you again. Take this problem, please.

$$\mathbf{A} = \begin{pmatrix} .001 & 0 & 0 \\ 0 & .0011 & 0 \\ 0 & 0 & 10^4 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \text{ and } \mathbf{x}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

We'll solve $\mathbf{Ax} = \mathbf{B}$ with GMRES. We should converge in 3 iterations.

Trying different orthogonalizations: $\|\mathbf{r}_k\|/\|\mathbf{r}_0\|$

k	CGS	MGS	MGS2	CGS2
0	1.00e+00	1.00e+00	1.00e+00	1.00e+00
1	8.16e-01	8.16e-01	8.16e-01	8.16e-01
2	3.88e-02	3.88e-02	3.88e-02	3.88e-02
3	6.69e-05	6.42e-08	6.42e-08	6.34e-34
4	4.74e-05	3.70e-08	5.04e-24	
5	3.87e-05	3.04e-18		
6	3.35e-05			
7	3.00e-05			
8	2.74e-05			
9	2.53e-05			
10	2.37e-05			

An Evil Matrix for GMRES

$$\mathbf{J} = \begin{pmatrix} 1 & 1 & 0 & & & \\ 0 & 1 & 1 & 0 & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 1 & 0 \\ & & & 0 & 1 & 1 \\ & & & & 0 & 1 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}, \mathbf{x}_0 = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{pmatrix},$$

Solution: $\mathbf{x}^* = \mathbf{b}$ is an eigenvector of $\mathbf{A}$.
$\lambda = 1$ is the only eigenvalue. Nice clustering, but ...

Solution of $\mathbf{Jx} = \mathbf{b}$ with GMRES: $N = 4$

What happened?

k	$\|\mathbf{r}_k\|$
0	3.16e+00
1	8.62e-01
2	2.35e-01
3	1.19e-01
4	7.94e-32

J is the world's most Krylov-hostile matrix.

- It's a Jordan block, which has nothing to do with the NBA.
- $N \times N$ Jordan blocks have eigenvalues with
  - algebraic multiplicity N,
  - geometric multiplicity 1.
- You can't get more non-diagonalizable than that.

So the theorems are true, but Jordan blocks can bite you.

Bottom line on GMRES: theory

- GMRES is happy if
  - **A** is diagonalizable,
  - $\kappa(\mathbf{V})$ is not too large,
  - and the eigenvalues are clustered.
- GMRES can be very unhappy if
  - **A** is not diagonalizable,
  - $\kappa(\mathbf{V})$ is large,
  - $\kappa(\mathbf{A})$ is large.
- You can fix large $\kappa(\mathbf{A})$ with preconditioning.

Bottom line on GMRES: ma580 applications

- Boundary value problems need preconditioning.
- Integral equations generally don't.
- Neither of these lead to Jordan blocks that cause problems, if you do your preconditioning job right.

Bottom line on GMRES: implementation

- Mat-vecs and orthogonalization are the expensive parts.
- CGS2 is a sensible way to go on machines with two or more cores.
- MGS2 is ok for single core machines (phones, pagers, . . .)
- Storage can be a serious problem. More on this later.