Fixed-Point Iterations, Krylov Spaces, and Krylov Methods

Fixed-Point Iterations

Solve nonsingular linear system: \( Ax = b \)  
(solution \( \hat{x} = A^{-1}b \))
Solve an approximate, but simpler system: \( Mx_0 = b \rightarrow x_0 = M^{-1}b \)

Improve the solution using the residual: \( r_0 = b - Ax_0 \)  
( iterative refinement)
Error, \( e_0 = \hat{x} - x_0 \), satisfies \( Ae_0 = b - Ax_0 = r_0 \)
Don’t compute exact error, instead solve \( Mz_0 = r_0 \) and set \( x_1 = x_0 + z_0 \)

Iterate:
\[
\begin{align*}
  r_k &= b - Ax_k \\
  z_k &= M^{-1}r_k \\
  x_{k+1} &= x_k + z_k
\end{align*}
\]

Methods: Jacobi iteration (diagonal), Gauss-Seidel (upper triangular), many others such as (S)SOR, ...
Fixed-Point Iterations

Convergence of such iterations?
Let $A = M - N$ (matrix splitting).
Then $Mx_k = b - Ax_k \Leftrightarrow Mx_k = b - Mx_k + Nx_k \Leftrightarrow Mx_{k+1} = Nx_k + b$
(fixed-point iteration $x_{k+1} = M^{-1}Nx_k + M^{-1}b$)

Note that the fixed-point is the solution (proof?)

Error:

\[ e_k = \hat{x} - x_k = M^{-1}N\hat{x} + M^{-1}b - M^{-1}Nx_{k-1} - M^{-1}b \]
\[ = M^{-1}N(\hat{x} - x_{k-1}) = M^{-1}N\hat{x}_{k-1} \]
\[ = (M^{-1}N)^k e_0 \]

Residual: $r_k = (NM^{-1})^k r_0$ and $M^{-1}r_k = (M^{-1}N)^k (M^{-1}r_0)$ (proof?)

To analyze convergence we need to introduce/review a number of concepts

Rate of Convergence

Let $\hat{x}$ be the solution of $Ax = b$, and we have iterates $x_0, x_1, x_2, \ldots$

\{x_k\} converges (q-)linearly to $\hat{x}$ if there are $N \geq 0$ and $c \in [0,1)$ such that for $k \geq N: \|x_{k+1} - \hat{x}\| \leq c \|x_k - \hat{x}\|$

\{x_k\} converges (q-)superlinearly to $\hat{x}$ if there are $N \geq 0$ and a sequence $\{e_k\}$ that converges to 0 such that for $k \geq N: \|x_{k+1} - \hat{x}\| \leq e_k \|x_k - \hat{x}\|$

\{x_k\} converges to $\hat{x}$ with (q-)order at least $p$ if there are $p > 1$, $c \geq 0$, and $N \geq 0$ such that $k \geq N: \|x_{k+1} - \hat{x}\| \leq c \|x_k - \hat{x}\|^p$ (quadratic if $p = 2$, cubic if $p = 3$, and so on)

\{x_k\} converges to $\hat{x}$ with j-step (q-)order at least $p$ if there are a fixed integer $j \geq 1$, $p > 1$, $c \geq 0$, and $N \geq 0$, such that $k \geq N: \|x_{k+j} - \hat{x}\| \leq c \|x_k - \hat{x}\|^p$
Norms

A norm on a vector space $V$ is any function $f : V \rightarrow \mathbb{R}$ such that
1. $f(x) \geq 0$ and $f(x) = 0 \iff x = 0$,
2. $f(\alpha x) = |\alpha| f(x)$,
3. $f(x + y) \leq f(x) + f(y)$,
where $x \in V$ and $\alpha \in \mathbb{R}$.

Important vector spaces in this course: $\mathbb{R}^n$, $\mathbb{C}^n$, and $\mathbb{R}^{m \times n}$, $\mathbb{C}^{m \times n}$ (matrices).
Note that the set of all $m$-by-$n$ matrices (real or complex) is a vector space.

Many matrix norms possess the submultiplicative or consistency property:
$f(AB) \leq f(A)f(B)$ for all $A \in \mathbb{C}^{m \times k}$ and $B \in \mathbb{C}^{k \times n}$ (or real matrices).

Note that strictly speaking this is a property of a family of norms, because in general ‘each’ $f$ is defined on a different vector space.

Norms

We can define a matrix norm using a vector norm (an induced matrix norm):

$$
\| A \|_\alpha = \max_{\| x \|_\alpha = 1} \| Ax \|_\alpha = \max_{\| x \|_\alpha = 1} \| Ax \|_\alpha
$$

Induced norms are always consistent (satisfy consistency property).

Two norms $\| . \|_\alpha$ and $\| . \|_\beta$ are equivalent if there exist positive, real constants $a$ and $b$ such that

$$
\forall x : a \| x \|_\alpha \leq \| x \|_\beta \leq b \| x \|_\alpha
$$

The constants depend on the two norms but not on $x$.

All norms on a finite dimensional vector space are equivalent.
Norms

Some useful norms on $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{mxn}$, $\mathbb{C}^{mxn}$:

p-norms: $\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}$, especially $p = 1, 2, \infty$, where $\|x\|_\infty = \max_i |x_i|$.

Induced matrix p-norms are:

$\|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}|$ (max absolute column sum)
$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|$ (max absolute row sum)

Matrix Frobenius norm:

$\|A\|_F = \left(\sum_{i,j=1}^{n} |a_{ij}|^2\right)^{\frac{1}{2}}$ (similar to vector 2-norm for a matrix)

All these norms are consistent (satisfy the submultiplicative property)

Eigenvalues and Eigenvectors

Let $Ax = \lambda x$ and $\tilde{y}A = \tilde{\lambda} \tilde{y}$ (for same $\lambda$).

We call the column vector $x$ a (right) eigenvector, the row vector $\tilde{y}$ a left eigenvector, and $\lambda$ an eigenvalue, the triple together is called an eigentriple, and $(\lambda, x)$ and $(\tilde{\lambda}, \tilde{y})$ a (right) eigenpair or left eigenpair.

The set of all eigenvalues of $A$, $\Lambda(A)$, is called the spectrum of $A$.

If the matrix $A$ is diagonalizable (has a complete set of eigenvectors) we have

$A = V \Lambda V^{-1} \iff AV = VA$,

where $V$ is a matrix with the right eigenvectors as columns and $\Lambda$ is a diagonal matrix with the eigenvalues as coefficients.

A similar decomposition can be given for the left eigenvectors.
Spectral Radius

The spectral radius $\rho(A)$ is defined as $\rho(A) = \max \{ |\lambda| : \lambda \in \sigma(A) \}$.

Theorem:
For all $A$ and $\varepsilon > 0$ a consistent norm $\| \cdot \|$ exists such that $\| A \| \leq \rho(A) + \varepsilon$.

So, if $\rho(A) < 1$, then a consistent norm $\| \cdot \|$ exists such that $\| A \| < 1$.

Take $\varepsilon = \frac{1}{2} (1 - \rho(A))$ and apply theorem above.

Define $A^* = \overline{A^T}$ (complex conjugate transpose).

If $A$ is Hermitian ($A = A^*$), then $\rho(A) = \| A \|$.

If $A$ is normal ($AA^* = A^*A$), then $\rho(A) = \| A \|$.

Fixed-Point Iterations

Under what conditions does $e_k \to 0$ and $x_k \to \hat{x}$ (convergence) for arbitrary $e_0$?

Theorem: $e_k = \left( M^{-1}N \right)^k e_0 \to 0$ for arbitrary $e_0$ iff $\left( M^{-1}N \right)^k \to 0$.

Proof:
Let $G = M^{-1}N$ and the matrix norm $\| \cdot \|$ be induced by a vector norm.

1. Assume $G^k \to 0$.
   Then $G^k \to 0 \Rightarrow \| G^k \| \to 0$ and $\| G^k e_0 \| \leq \| G^k \| \| e_0 \| \to 0$ for any $e_0$.

2. Assume $G^k e_0 \to 0$ for all $e_0$.
   Consider the identity matrix $I = \begin{bmatrix} \eta_1 \eta_2 \cdots \eta_n \end{bmatrix}$.
   $G^k I = \begin{bmatrix} G^k \eta_1 \eta_2 \cdots \eta_n \end{bmatrix} \to \begin{bmatrix} 0 \cdots 0 \end{bmatrix}$; so $G^k \to 0$ since $G^k I = G^k$.
   Alternatively, consider $\| G^k \eta_1 \eta_2 \cdots \eta_n \| \to 0$ (note that $G^k \eta_i \to 0$).
Norms

Note that we can generalize the result for the one-norm to all norms by using the equivalence of norms on finite dimensional vector spaces.

Similarly, the results are readily generalized for inconsistent matrix norms (with \( \| AB \| > \| A \| \| B \| \) possible), by using the equivalence of norms on finite dimensional spaces.

Fixed-Point Iterations

Theorem: \( G^k \to 0 \) iff \( \rho(G) < 1 \).

Proof:
1. \( G^k \to 0 \Rightarrow \rho(G) < 1 \).

For each eigenvalue \( \lambda \) of \( G \) there exists at least one eigenvector \( v \) s.t. \( Gv = \lambda v \).

Then \( \| G^k v \| = \| \lambda^k v \| = |\lambda|^k \| v \| \) and \( G^k v \to 0 \). So, \( |\lambda|^k \| v \| \to 0 \Rightarrow |\lambda| < 1 \).

Since this holds for each eigenvalue, \( \rho(G) < 1 \) must hold.

2. \( \rho(G) < 1 \Rightarrow G^k \to 0 \).

There exists a consistent norm \( \| . \|_s \) s.t. \( \| G \|_s < 1 \).

Hence, \( \| G^k \|_s \leq \| G \|_s^k \to 0 \). Therefore \( \| G^k \|_s \to 0 \Rightarrow G^k \to 0 \).

So \( e_k \to 0 \) (\( x_k \to \hat{x} \)) for arbitrary \( e_0 \) iff \( \rho\left( M^{-1}N \right) < 1 \).
Krylov Spaces

Given $x_0$, set $r_0 = b - Ax_0$.

For $k = 0, 1, 2, \ldots$

$$z_k = M^{-1}r_k,$$

$$x_{k+1} = x_k + z_k,$$

$$r_{k+1} = b - Ax_{k+1} = r_k - Az_k.$$

Note that $x_{k+1} - x_k = z_k = M^{-1}r_k$ and hence $x_{k+1} - x_0 = z_0 + z_1 + \cdots + z_k$.

This implies $x_{k+1} - x_0 = M^{-1}r_0 + (M^{-1}N)M^{-1}r_0 + \cdots + (M^{-1}N)^kM^{-1}r_0$.

So, correction $x_{k+1} - x_0$ is given by polynomial $S_k(t) = 1 + t + t^2 + \cdots + t^k$:

$$x_{k+1} - x_0 = \sum_{i=0}^{k} (M^{-1}N)^i M^{-1}r_0 = S_k(M^{-1}N) \cdot M^{-1}r_0.$$

Note also $e_k = (M^{-1}N)^k e_0$ and $r_k = (NM^{-1})^k r_0 = M(M^{-1}N)^k M^{-1}r_0$.

Similarity Transformation

Let $A$ have eigenpairs $(\lambda_i, v_i); \, A v_i = \lambda_i v_i$

Define the similarity transformation: $BAB^{-1}$

The matrix $BAB^{-1}$ has the same eigenvalues, $\lambda_i$, as $A$ and eigenvectors $Bv_i$:

$$BAB^{-1}(Bv_i) = BA v_i = \lambda_i Bv_i$$

Show that $(M^{-1}N)^k$ has the same eigenvalues as $(NM^{-1})^k$. 
Polynomials and Spaces

Main computational cost is in the multiplication by $A$ and solving for $M$. So, we can try to generate better polynomials (faster convergence) at same cost.

Also correction $x_{k+1} - x_0 \in \text{span}\left\{ M^{-1}r_0, (M^{-1}N)M^{-1}r_0, \ldots, (M^{-1}N)^k M^{-1}r_0 \right\}$

We call a space $K_m(B, y) \equiv \text{span}\left\{ y, By, B^2y, \ldots, B^{m-1}y \right\}$ the Krylov (sub)space of dimension $m$ associated with $B$ and $y$.

So, $x_m - x_0 \in K_m(M^{-1}N, M^{-1}r_0)$

\[ e_m = (M^{-1}N)^m e_0 \in K_{m+1}(M^{-1}N, e_0) \text{ and } M^{-1}r_m = (M^{-1}N)^m M^{-1}r_0 \in K_{m+1}(M^{-1}N, M^{-1}r_0). \]

Therefore, alternatively we can compute better approximate solutions from the same space (faster convergence) at same cost.

Krylov Spaces

So, a Krylov space is a space of polynomials in a matrix times a vector.

These spaces inherit the many important approximation properties that polynomials on the real line or in the complex plane possess.

For simplicity let the matrix $B$ be diagonalizable, $B = VAV^{-1}$.

Then $B^2 = VAV^{-1}VAV^{-1} = VAV^{-1}$ and generally $B^k = VAV^{-1}$.

So, the polynomial $p_m(t) = \alpha_0 + \alpha_1 t + \cdots + \alpha_m t^m$ applied to $B$ gives

\[ p_m(B) = V \left( \alpha_0 I + \alpha_1 A + \alpha_2 A^2 + \cdots + \alpha_m A^m \right) V^{-1} \text{ and hence } \]

\[ p_m(B) = V p_m(A) V^{-1} = V \text{ diag} \left( p_m(\lambda_1), \ldots, p_m(\lambda_n) \right) V^{-1} \]

So, the polynomial is applied to the eigenvalues individually.

This allows us to approximate solutions to linear systems, eigenvalue problems, and more general problems using polynomial approximation.
Approximation by Matrix Polynomials

Let $B = VAV^{-1}$, let $\{B\} \subset \Omega \subset \mathbb{C}$.

If $p_m(t) \approx \frac{1}{t}$ for all $t \in \Omega$, then $p_m(B) \approx B^{-1}$.

Let $y = V\zeta$. Then $p_m(B)y = \sum_i v_i p_m(\lambda_i)\zeta_i \approx \sum_i v_i \frac{\zeta_i}{\lambda_i} = B^{-1}y$.

Furthermore, let $\varepsilon \approx 0$ and $|\lambda_i - \lambda_j| > \delta$ (for some eigenvalue $\lambda_i$). If $p_m(t) = \begin{cases} \varepsilon, & t \in \Omega \text{ and } |t - \lambda_i| > \delta, \\ 1, & t = \lambda_i, \end{cases}$ then $p_m(B)y \approx v_i\zeta_i$.

If we can construct such polynomials for modest $m$ we have an efficient linear solver or eigensolver.

Krylov Spaces

We have $M^{-1}N = M^{-1}(M - A) = I - M^{-1}A$.

So, $x_{k+1} - x_k = (I - M^{-1}A)^k M^{-1}r_0 \in K_{k+1}(M^{-1}A, M^{-1}r_0)$, and $x_{k+1} = M^{-1}Nx_k + M^{-1}b$, with fixed-point $\hat{x} = (I - M^{-1}A)\hat{x} + M^{-1}b \Leftrightarrow M^{-1}A\hat{x} = M^{-1}b$.

So, we solve the preconditioned problem $M^{-1}Ax = M^{-1}b$.

Preconditioning aims to improve the convergence of Richardson’s iteration $x_{k+1} = (I - A)x_k + b$.

However, with $\tilde{A} = M^{-1}A$, $\tilde{M} = I$, and $\tilde{b} = M^{-1}b$ our iteration becomes Richardson’s iteration, and we have Krylov spaces and polynomials based on $\tilde{A}$.

Hence, for simplicity we consider $A$ and $b$ as an explicitly preconditioned matrix and vector, and work with Krylov spaces in $A$ and $r_0$ (most of the time).
Approximations from Krylov Spaces

Richardson for $Ax = b$ seeks update $z_m \in K_m(\{I - A, r_0\}) = K_m(\{A, r_0\})$

How to define an iteration that finds better approximation in same space? Given $x_0$ and $r_0 = b - Ax_0$ find $z_m \in K_m(\{A, r_0\})$ and set $x_n = x_0 + z_m$.

There are several possibilities. Two particularly important ones are

1. Find $z_m \in K_m(\{A, r_0\})$ such that $\| r_m \| = \| r_0 - Az_m \|$ is minimal.
2. Find $z_m \in K_m(\{A, r_0\})$ such that $\| e_m \| = \| e - (x_0 + z_m) \|$ is minimal.

The second one seems hard, but is possible in practice for special norms.

Further possibilities for optimal solutions exist, and for non-Hermitian matrices certain non-optimal solutions turn out to have advantages as well.

Inner Products

Many methods to select $z_m$ from the Krylov space are related to projections.

We call $f : S \times S \to \mathbb{R}$ an inner product over the real vector space $S$, if for all vectors $x, y, z$ and scalars $\alpha$,

1. $f(x, x) \geq 0$ and $f(x, x) = 0 \iff x = 0$
2. $f(\alpha x, z) = \alpha f(x, z)$
3. $f(x + y, z) = f(x, z) + f(y, z)$
4. $f(x, z) = f(z, x)$

For a complex inner product, $f : S \times S \to \mathbb{C}$, over a complex vector space $S$ we have instead of property (4): $f(x, z) = f(z, x)$.

Inner products are often written as $\langle x, y \rangle$, $\langle x, y \rangle_\alpha$, or $\langle x, y \rangle_{\alpha}^\prime$, etc..

We say $x$ and $y$ are orthogonal (w.r.t $\alpha$-IP), $x \perp_{\alpha} y$ if $\langle x, y \rangle_\alpha = 0$. 
Inner products and Norms

Each inner product defines, or induces, a norm: $\|x\| = \sqrt{\langle x, x \rangle}$. (proof?)

Many norms are induced by inner products, but not all. Those norms that are have additional nice properties (that we’ll discuss soon).
An inner product and its induced norm satisfy: $|\langle x, y \rangle| \leq \|x\| \|y\|$ (CS ineq)

A norm induced by an inner product satisfies the parallelogram equality:

$$\|x + y\|^2 + \|x - y\|^2 = 2(\|x\|^2 + \|y\|^2)$$

In this case we can find the inner product from the norm as well:
Real case: $\langle x, y \rangle = \frac{1}{4}(\|x + y\|^2 - \|x - y\|^2)$
Complex case:

$$\text{Re}(x, y) = \frac{1}{4}(\|x + y\|^2 - \|x - y\|^2), \quad \text{Im}(x, y) = \frac{1}{4}(\|x + iy\|^2 - \|x - iy\|^2)$$

Minimum Residual Solutions

First, we consider minimizing the 2-norm of the residual:

Find $z_m \in K_m(A, r_0)$ such that $\|r_m\|_2 = \|r_0 - Az_m\|_2$ is minimal.

The vector 2-norm is induced by the Euclidean inner product $\langle y, x \rangle = \sum_{i=1}^n y_i x_i$.

It makes sense to minimize the residual, because the error is in general unknown, and we can only directly minimize special norms of the error (those that don’t require the error $\odot$). Moreover, $\|e_m\|_2 = \|A^{-1}r_m\|_2 \leq \|A^{-1}\|\|r_m\|_2$, so the norm of the error is bounded by a constant times the norm of the residual.

Finally, note that $\|r_m\|_2 = \|e_m\|_{A^{-1}} \equiv \|Ae\|_2$ (show this is a norm if $A$ regular)

Theorem: $z_m$ is the minimizer iff $r_m = b - A(x_0 + z_m) \perp K_m(A, r_0)$.

Note that $b - A(x_0 + z_m) = r_0 - Az_m$. 

Minimum Residual Solutions

Proof: Let \( f(z) = \| r_0 - A z \|_2 \). Then \( \hat{z} \in K_m(A, r_0) : \| r_0 - A \hat{z} \|_2^2 \min \) implies \( \hat{z} \) is a stationary point of \( f(z) \):

For any unit vector \( p \in K_m(A, r_0) \) we have \( \frac{\partial f(\hat{z})}{\partial p} = 0 \).

So, \( \lim_{\varepsilon \to 0} \frac{f(\hat{z} + \varepsilon p) - f(\hat{z})}{\varepsilon} = 0 \), which gives

\[
\| r_0 - A \hat{z} - \varepsilon A p \|_2^2 - \| r_0 - A \hat{z} \|_2^2 = \varepsilon p' A' (r_0 - A \hat{z}) - \varepsilon (r_0 - A \hat{z})' A p + \varepsilon^2 p' A' A p = 0 \quad \Leftrightarrow \quad p' A' (r_0 - A \hat{z}) - (r_0 - A \hat{z})' A p = 0 \quad \text{for any unit vector } p \in K_m(A, r_0).
\]

So, \( (r_0 - A \hat{z})' A p = 0 \Leftrightarrow (r_0 - A \hat{z}) \perp A p \) for any unit \( p \in K_m(A, r_0) \) (why?)

Since \( Ap \in K_m(A, Ar_0) \) we have \( (r_0 - A \hat{z}) \perp K_m(A, Ar_0) \equiv AK_m(A, r_0) \).

---

Minimum Residual Solutions

Minimizing a norm (cont. function) is in general complicated. However, the orthogonality conditions lead naturally to a linear system of equations.

Let \( \{w_1, w_2, \ldots, w_m\} \) form a basis for \( K_m(A, r_0) \) and \( \{A w_i\}_{i=1}^m \) for \( K_m(A, Ar_0) \).

Let \( W_m = [w_1, w_2 \ldots w_m] \). Then \( \hat{z}_m = W_m \zeta \) (for some unknown \( \zeta \)).

Now the orthogonality conditions \( A w_i' \perp (r_0 - A W_m \zeta) \) yield the linear equations \( \sum_{j=1}^m (w_j' A w_i) \zeta_j = w_i' A r_0 \).

In matrix form: \( W_m' A' A W_m \zeta = W_m' A' r_0 \) normal equations (accuracy problems)

LS problem: \( \min \| r_0 - K_m \zeta \|_2 \quad \text{where } K_m = AW_m \)

More accurate to solve LS problem using QR-decomposition.

Solving for \( \zeta \) requires only the solution of an \( m \times m \) system independent of \( n \).
Minimum Residual Solutions

We have seen that \[ \min_i \| r_i - K_m \zeta \| \Leftrightarrow K_m \perp (r_0 - K_m \zeta) \]

Compute \[ K_m = Q_m R_m \] (QR-decomposition) where
\[ Q_m \in \mathbb{C}^{n \times m}, \quad \left( R_m \right) \text{ s.t. } Q_m^* Q_m = I_m \text{ and } R_m \in \mathbb{C}^{m \times m} \text{ uppertriangular} \]

If \( \text{rank}(K_m) = m \), then \( R_m \) is nonsingular and \( \text{range}(Q_m) = \text{range}(K_m) \)

Now \( Q_m \perp (r_0 - K_m \zeta) \) gives \( Q_m^* r_0 - Q_m^* Q_m R_m \zeta = 0 \Leftrightarrow R_m \zeta = Q_m^* r_0 \) (easy solve)

Note that \( K_m \zeta = Q_m^* r_0 - r_0 - K_m \zeta = r_0 - Q_m^* r_0 \)
\( K_m \zeta \) is the orthogonal projection of \( r_0 \) onto \( \text{range}(K_m) = K_m (A, A r_0) \).
\( r_m = r_0 - K_m \zeta \) is the orthogonal projection of \( r_0 \) onto \( \left( K_m (A, A r_0) \right)^\perp \).

\( Q_m^* Q_m \) and \( I - Q_m^* Q_m \) are orthogonal projectors.
\( P \) is projector if \( P^2 = P \), orthogonal projector if \( R(P) \perp N(P) \Leftrightarrow P^* = P \).

Minimum Residual Solutions

Iteration-wise the problem is solved in four steps:

1. Extend the Krylov spaces \( K_m (A, r_0) \) and \( K_m (A, A r_0) \) by adding the respective next vectors \( A^m r_0 \) and \( A^{m+1} r_0 \) (only 1 matvec)
2. Update orthogonal basis for \( K_m (A, A r_0) \): QR-decomp. of \( K_m \)
3. Update projected matrix and projection of \( r_0 \) (orthog) onto \( K_m (A, A r_0) \)
4. Solve the projected problem, e.g. \( R \zeta = Q_m^* r_0 \). Note that this problem is only \( m \times m \) or \( (m+1) \times m \) irrespective of the size of the linear system.

These steps vary somewhat for different methods.
We would like to carry out these steps efficiently.

The GCR method (Generalized Conjugate Residuals) illustrates these steps well. (Eisenstat, Elman, and Schulz 1983)
Minimum Residual Solutions: GCR

GCR: \( Ax = b \)
Choose \( x_0 \) (e.g. \( x_0 = 0 \)) and tolerance \( \varepsilon \); set \( r_0 = b - Ax_0; \ i = 0 \)
while \( \| r_i \| \geq \varepsilon \) do
\begin{align*}
\quad i &= i + 1 \\
\quad u_i &= r_{i-1}; \quad c_i = Au_i \\
\quad &\text{for } j = 1, \ldots, i - 1 \text{ do} \\
\quad &\quad u_i = u_i - u_j c_j; \quad c_i = c_i - c_j c_j \\
\quad &\text{end do} \\
\quad &\text{Orthogonalize } c_i \text{ against previous } c_j \text{ and update } u_i \text{ such that } Au_i = c_i \text{ maintained} \\
\quad &\text{Normalize; (end QR decomposition)} \\
\quad &\text{Project new } c_i \text{ out of residual and update solution accordingly; note } r_i \perp c_j \text{ for } j \leq i \\
\quad &\text{end do}
\end{align*}

What happens if \( c_i \perp r_{i-1} \)

---

Minimum Residual Solutions: GCR

From the algorithm we see that if \( c_j r_{i-1} \neq 0 \) for \( j = 1 \ldots i \):

\begin{align*}
\quad &u_1 = \nu r_0 \in \text{span}\{r_0\} = K_1(A, r_0), \quad \nu \text{ is a normalization constant} \\
\quad &c_1 = \nu A r_0 \in \text{span}\{A r_0\} \in K_1(A, A r_0). \\
\quad &r_1 = r_0 - \alpha c_1 = r_0 - \nu \alpha A r_0 \in K_2(A, r_0) \quad \text{also } A r_0 \in \text{span}\{r_0, r_1\} \\
\quad &\text{By induction (and the statements above) we can show} \\
\quad &u_i = r_{i-1} - \sum_{j<i} \beta_j u_j \in \text{span}\{r_0, \ldots, r_{i-1}\} = K_i(A, r_0) \\
\quad &c_i = A r_{i-1} - \sum j \beta_j c_j \in \text{span}\{A r_0, \ldots, A r_{i-1}\} = K_i(A, A r_0) \quad \text{and also that} \\
\quad &c_i = A u_i, \quad c_i \perp c_1, c_2, \ldots, c_{i-1}, \text{ and } c_i^T c_i = 1 \text{ (last by construction)} \\
\quad &r_i = r_{i-1} - \alpha c_i \in \text{span}\{r_0, A r_0, A r_1, \ldots, A r_{i-1}\} = K_{i+1}(A, r_0) \\
\quad &\text{These relations hold even if } \{r_0, A r_0, \ldots, A^m r_0\} \text{ are dependent for some } m. \\
\quad &\text{In that case all the spaces have a maximum dimension of } m + 1.
\end{align*}
Minimum Residual Solutions: GCR

Theorem:
Let \( c_i^* r_{i-1} \neq 0 \) for \( i = 1, 2, \ldots, m \). Then \( \|b - Ax\|_2 \) for GCR is minimal.

We use our earlier theorem on the minimum residual. We have (previous slide)
\[
x_m = x_0 + \sum_{i=1}^{m} u_i \beta_i = x_0 + z_m \quad \text{and} \quad z_m \in K_m(A, r_0)
\]
as required.

Now we only need to show that \( r_m \perp K_m(A, r_0) \).
Note that our assumption implies \( r_{i-1} \neq 0 \) for \( i = 1, 2, \ldots, m \).

By construction we have \( r_j = r_0 - c_i^* r_i \Rightarrow c_i^* r_j = c_i^* r_0 - c_i^* c_i r_i = 0 \)
Assume that for \( j = 1 \ldots i-1 \) we have \( r_{j-1} \perp c_1, c_2, \ldots, c_{j-1} \) (induc. hypo.).
From \( r_j = r_{j-1} - c_j^* c_j r_{j-1} \) we have \( r_j \perp c_j \).
For \( i < j \), \( c_j r_j = c_j^* r_{j-1} = 0 \) (from orthogonality \( c_i \) and induction hypothesis).

This proves the required orthogonality result since the \( c_i \) span \( K_m(A, r_0) \).

Recapitulation of GCR after \( m \) it.s:
\[
\|r_m\|_2 = \min \{ \|r_0 - Az_m\|_2 \mid z_m = U_m \zeta \}
\]
\( u_i \in K_m(A, r_i) \), \( c_i \in K_m(A, A r_0) \), \( r_i \in K_{m+1}(A, r_0) \) for \( i = 1, \ldots, m \) \& \( i = 0 \) for \( r_i \).
This implies that \( K_{i+1}(A, r_0) = \text{span}(r_0, \ldots, r_i) \) for \( i = 0, \ldots, m \) (if \( c_i^* r_{i-1} \neq 0 \)).

Let \( U_m = [u_1 \ u_2 \ldots \ u_m] \) and \( C_m = [c_1 \ c_2 \cdots \ c_m] \)
Then \( A U_m = C_m \), \( C_m^* C_m = I_m \), and \( \text{range}(U_m) = K_m(A, r_0) \).

For GCR the projected system is the matrix for the normal equations (but computed implicitly):
\[
U_m^* A^* A U_m = (A U_m)^* (A U_m) = C_m^* C_m = I_m.
\]
The projected right hand side is \( C_m^* r_0 \), and so we have \( \zeta = C_m^* r_0 \).

\( z_m = U_m \zeta \) is given by condition \( C_m^* (r_0 - A U_m \zeta) = 0 \), which gives \( \zeta = C_m^* r_0 \).
This gives \( z_m = U_m C_m^* r_0 \), \( x_m = x_0 + z_m \), and \( r_m = r_0 - C_m C_m^* r_0 \).
Minimum Residual Solutions: GCR

Note that $C_m^*$ and $I - C_m^*$ are both orthogonal projectors.

$C_m^*$ is a projection since $C_m^* C_m^* C_m^* = C_m^* C_m^* C_m^*$.

$C_m^*$ is an orthogonal projection since $C_m^* C_m^*$ is Hermitian.

$I - C_m^*$ is a projection since

$(I - C_m^*)(I - C_m^*) = I - C_m^* - C_m^* C_m^* + C_m^* C_m^* = I - C_m^* C_m^*$.

$I - C_m^*$ is an orthogonal projection since $I - C_m^* C_m^*$ is Hermitian.

Model Problems

Discretize $- \left( pu_x \right)_x - \left( qu_y \right)_y + ru_x + su_y + tu = f$.

Integrate equality over box $V$. Use Gauss' divergence theorem to get

$$\int_V \left( pu_x \right)_x + \left( qu_y \right)_y \, dx dy = \int_{\partial V} \left( pu_x \right)_x \cdot n \, ds$$

And approximate the line integral numerically.
Model Problems

Now we approximate the boundary integral \( \int_{\partial V} \left( \frac{p_{u}}{q_{u}} \right) \cdot n \, ds \).

We approximate the integrals over each side of box \( V \) using the midpoint rule and we approximate the derivatives using central differences.

\[
\int_{y}^{y+dy} p_{u} n_{i} \, dy \approx \frac{\Delta y}{\Delta x} p_{i+1/2,j}(U_{i+1,j} - U_{i,j}) \quad \text{and so on for the other sides}
\]

We approximate the integrals over \( ru_x, su_y, tu \), and \( f \) using the area of the box and the value at the midpoint of the box, where we use central differences for derivatives. So, \( u_x \approx \frac{1}{2}(U_{i+1,j} - U_{i-1,j})/(2\Delta x) \), and so on.

For various examples we will also do this while strong convection relative to the mesh size makes central differences a poor choice (as it gives interesting systems).

---

Model problems

This gives the discrete equations

\[
- \frac{\Delta y}{\Delta x} \left[ p_{i+1/2,j} \left( U_{i+1,j} - U_{i,j} \right) - p_{i-1/2,j} \left( U_{i,j} - U_{i-1,j} \right) \right] \\
- \frac{\Delta x}{\Delta y} \left[ q_{i,j+1/2} \left( U_{i,j+1} - U_{i,j} \right) - q_{i,j-1/2} \left( U_{i,j} - U_{i-1,j} \right) \right] \\
+ \left( \frac{\Delta y}{2} \right) r_{i,j} \left( U_{i+1,j} - U_{i-1,j} \right) + \left( \frac{\Delta x}{2} \right) s_{i,j} \left( U_{i,j+1} - U_{i,j-1} \right) \\
+ \Delta x \Delta y t_{i,j} U_{i,j} = \Delta x \Delta y f_{i,j}
\]

Often we divide this result again by \( \Delta x \Delta y \).
Experiments with GCR

Solve on unit square with Dirichlet boundary conditions.

Note the modest deterioration of convergence rate as discretization becomes finer (so cost slightly worse than linear).
Experiments with GCR

Eigenvalues of the discrete Laplacian for three discretization sizes

For this problem, the smallest eigenvalues and the number of small eigenvalues matter most (as we shall consider later in more detail). For this problem, the smallest eigenvalues and the number of small eigenvalues of the discrete Laplacian for three discretization sizes.

All eigenvalues for 32 x 32 mesh

Eigenvalues close to 0 for all three meshes

Experiments with GCR
Experiments with GCR

- Convergence for 322 x 322 problem using GCR restarting every 200 it.s
- Restarting saves memory and time (from excessive orthogonalizations)
- No preconditioning
- Runtime: ~15 minutes on Laptop (Intel Core 2 Duo P8700 @ 2.53 GHz)

Experiments with GCR
Experiments with GCR

- Convergence for 322 x 322 problem using preconditioned GCR
- Restarting saves memory and time (from excessive orthogonalizations)
- Preconditioner: Incomplete ILU without fill-in
- Runtimes: GCR(50) ~ 2 min., GCR (without restart) ~ 4 min.

Experiments with GCR
Minimum Residual Solutions: GMRES

An alternative is to generate iteration-wise an orthogonal basis for $K_{m+1}(A, r_0)$. The Arnoldi algorithm (iteration) goes as follows:

Let $v_i = r / \| r \|_2$;

for $k = 1 \ldots m$,

\[ \tilde{v}_{k+1} = A v_k; \]

for $j = 1 \ldots k$,

\[ h_{j,k} = v_j^T \tilde{v}_{k+1}; \quad \tilde{v}_{k+1} = \tilde{v}_{k+1} - h_{j,k} v_j; \]

end

\[ h_{k+1,k} = \| \tilde{v}_{k+1} \|_2; \quad v_{k+1} = \tilde{v}_{k+1} / h_{k+1,k}; \]

end

Note/show the following results:

\[ AV_m = V_{m+1} H_m \] (Arnoldi recurrence)

\[ V_{m+1} V_{m+1} = I_{m+1} \] (orthogonal),

\[ H_m = V_{m+1}^T A V_m \] (upper Hessenberg)

Using $AV_m = V_{m+1} H_m$, we solve $\min \{ \| r_0 - A z \|_2 \mid z \in K_m(A, r_0) \}$ as follows.

Let $z = V_m \zeta$, and minimize $\| r_0 - A V_m \zeta \|_2$ over all $m$-vectors $\zeta$.

Note that this is an $m \times m$ least squares problem (as before).

Now substitute $r_0 = V_{m+1} \eta$ and $AV_m = V_{m+1} H_m$. This gives

\[ \| V_{m+1} \eta \|_2 - V_{m+1} H_m \zeta \|_2 = \| V_{m+1} \left( \eta - H_m \zeta \right) \|_2 = \| \eta \|_2 - H_m \zeta \|_2. \]

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 $H_m$ and the least squares problem to
1. do this efficiently,
2. compute the residual norm without computing the residual.
By construction $\mathbf{H}_m$ has the following structure:

$$
\mathbf{H}_m = 
\begin{bmatrix}
  h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,m-1} & h_{1,m} \\
  h_{2,1} & h_{2,2} & h_{2,3} & \cdots & h_{2,m-1} & h_{2,m} \\
  h_{3,1} & h_{3,2} & h_{3,3} & \cdots & h_{3,m-1} & h_{3,m} \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
  h_{m-1,1} & h_{m-1,2} & h_{m-1,3} & \cdots & h_{m-1,m} & h_{m-1,m} \\
  h_{m,1} & h_{m,2} & h_{m,3} & \cdots & h_{m,m} & h_{m,m} \\
  h_{m+1,1} & h_{m+1,2} & h_{m+1,3} & \cdots & h_{m+1,m} & h_{m+1,m} \\
\end{bmatrix}
$$

(Upper Hessenberg)

Cheapest QR decomp. is by Givens rotations to zero lower diagonal.

$$
\mathbf{G}_1^H \mathbf{H}_m = 
\begin{bmatrix}
  c_1 & \hat{s}_1 \\
  -s_1 & \hat{c}_1 \\
\end{bmatrix}
\begin{bmatrix}
  * & * & \cdots & * \\
  0 & * & \cdots & * \\
  h_{3,2} & \cdots & h_{3,m} \\
  \vdots & \ddots & \vdots & \vdots \\
\end{bmatrix}
$$

Next step we compute:

$$
\mathbf{G}_2^H \mathbf{G}_1^H \mathbf{H}_m = 
\begin{bmatrix}
  1 & c_2 & \hat{s}_2 \\
  -s_2 & \hat{c}_2 & 0 \\
\end{bmatrix}
\begin{bmatrix}
  * & * & \cdots & * \\
  0 & * & \cdots & * \\
  h_{3,2} & h_{3,3} & \cdots & h_{3,m} \\
  h_{4,3} & \cdots & h_{4,m} \\
  \vdots & \ddots & \vdots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
  * & * & \cdots & * \\
  0 & * & \cdots & * \\
  h_{4,3} & \cdots & h_{m,3} \\
  \vdots & \ddots & \vdots & \vdots \\
\end{bmatrix}
$$

After $m$ Givens rotations:

$$
\mathbf{G}_m^H \cdots \mathbf{G}_1^H \mathbf{H}_m = \mathbf{Q}_m^H \mathbf{H}_m = 
\begin{bmatrix}
  r_{1,1} & \cdots & r_{1,m} \\
  0 & r_{2,2} & \cdots \\
  \vdots & \vdots & \ddots \\
  0 & \cdots & r_{m,m} \\
\end{bmatrix}
= \mathbf{R}_m
$$
Theorem: An unreduced \((m + 1) \times m\) Hessenberg matrix is nonsingular. (unreduced means no zeros on subdiagonal)

Proof: ?

So the least squares problem

\[
y_m = \arg \min \left\{ \| e_1 \| r_0 \|_2 - H_m y \|_2 : y \in \mathbb{C}^m \right\}
\]

can be solved by multiplying \(H_m y \approx e_1 \| r_0 \|_2\) from left by \(R_m^{-1} Q_m^H\):

\[
y_m = R_m^{-1} Q_m^H e_1 \| r_0 \|_2
\]

In practice:
Stepwise compute \(G_i^H(G_{i-1}^H \cdots G_1^H H_i)\) and \(G_i^H(G_{i-1}^H \cdots G_1^H e_1 \| r_0 \|_2)\)
In Arnoldi step, update \(H_{i-1}\) with new column; then carry out previous Givens rotations on new column.
Compute new Givens rotation and update \(H_i\) and right hand side (of small least squares problem): \(G_i^H(G_{i-1}^H \cdots G_1^H e_1 \| r_0 \|_2)\)
The least squares system now looks like $R_i y_i = Q_{i+1}^H e_1 r_0 \|_2$.

We may assume $R_i$ has no zeros on diagonal (see later)

Since bottom row of $R_i$ is zero we can only solve for $(Q_{i+1}^H e_1 \| r_0 \|_2)_{1...i}$ (first $i$ coeff.s)

This is exactly what we do by solving $R_i y_i = Q_i^H e_1 r_0 \|_2$

Note LS residual norm equals the norm of the actual residual:

$\|r_i\|_2 = \|\tilde{q}_{i+1}^H e_1 \| r_0 \|_2$ (since it changes with $i$):

$\|r_0 - AV_m y\|_2 = \|V_{m+1} e_1 \| r_0 \|_2 - V_{m+1} H_m y\|_2 = \|e_1 \| r_0 \|_2 - H_m y\|_2$

This way we can monitor convergence without actually computing updates to solution and residual (cheap).

---

**GMRES**

The least squares system now looks like $R_i y_i = Q_{i+1}^H e_1 r_0 \|_2$.

We may assume $R_i$ has no zeros on diagonal (see later)

Since bottom row of $R_i$ is zero we can only solve for $(Q_{i+1}^H e_1 \| r_0 \|_2)_{1...i}$ (first $i$ coeff.s)

This is exactly what we do by solving $R_i y_i = Q_i^H e_1 r_0 \|_2$

Note LS residual norm equals the norm of the actual residual:

$\|r_i\|_2 = \|\tilde{q}_{i+1}^H e_1 \| r_0 \|_2$ (since it changes with $i$):

$\|r_0 - AV_m y\|_2 = \|V_{m+1} e_1 \| r_0 \|_2 - V_{m+1} H_m y\|_2 = \|e_1 \| r_0 \|_2 - H_m y\|_2$

This way we can monitor convergence without actually computing updates to solution and residual (cheap).

---

**GMRES**

GMRES: $Ax = b$

CHOOSE $x_0$ (e.g., $x_0 = 0$) AND $tol$

$r_0 = b - Ax_0$; $k = 0$; $v_1 = r_0 / \| r_0 \|_2$;

WHILE $\| r_k \|_2 > tol$

$k = k + 1$;

$v_{k+1} = Av_k$;

FOR $j = 1 : k$,

$h_{j,k} = v_j^H v_{k+1}$; $v_{k+1} = v_{k+1} - h_{j,k} v_k$;

END

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

UPDATE QR-DECOMP.: $H_k = Q_{k+1}^H R_k$

$\| r_k \|_2 = \| q_{k+1}^H e_1 \| r_0 \|_2$

END

$y_k = R_k^{-1} Q_k^H e_1 \| r_0 \|_2$; $x_k = x_0 + V_k y_k$;

$r_k = r_0 - V_{k+1} H_k y_k = V_{k+1} (I - Q_k Q_k^H) e_1 \| r_0 \|_2$; (or simply $r_k = b - Ax_k$)
So we have generated the Krylov subspace (step 1), and we have an orthogonal basis for it (step 2, more or less). However, we do not have an orthogonal basis for the Krylov subspace (step 1), and we do not have an orthogonal basis for the Krylov subspace (step 2, more or less). Hence, we want to orthogonalize the Krylov subspace, but we don’t want to orthogonalize the right-hand side vector.

We can drop last row of \( R \) and last column of \( \bar{D} \) giving:

\[
\begin{align*}
\bar{D}^{\perp} &= \bar{R}^{1+w} \bar{D} = \bar{H} \\
\end{align*}
\]

where \( \bar{R} \) is upper triangular and has last row entirely zero. CPR-decomposition, (m Givens rotations),

\[
\bar{R}^{1+w} \bar{D} = \bar{H} \equiv \bar{H}
\]

Obviously we don’t want to orthogonalize the approximate solution from the residual. Step 3 is the orthogonal projection of the residual on \( \text{range}(\bar{C}) \).

\[
\begin{align*}
\text{range}(\bar{C}) &= \text{range}(\bar{A}^\perp \bar{A}) \\
\text{range}(\bar{C}) &= \text{range}(\bar{A}^\perp \bar{A})
\end{align*}
\]

Why not? Have we generated the Krylov subspace?
Using the QR-decomposition we have a QR-decomposition of $A$:

$AV_m = V_m + 1H_m = V_m + 1QR_m$

where $Q$ is unitary and $R$ is upper triangular.

So for the cost of $m$ Givens rotations we get the orthogonal basis for $V_m$ implicitly, since $\Lambda V_m = \Lambda V$.

New residual and approximate solution:

$\bar{r}_m = I - \left(V_m + 1Q_m\right)^{-1}\left(V_m + 1Q_m\right)Hr_0 = r_0 - V_m + 1Q_mQ_mHr_0 = r_0 - e_1\bar{r}_0\bar{e}_2$

and

$x_m = x_0 + A^{-1}(r_m - r_0) = x_0 + V_mR_m - 1Q_mH e_1\bar{r}_0\bar{e}_2$

Using this QR-decomposition we have a QR-decomposition of $A$.  

$$GMRES$$
Comparing with GCR, we see that apart from possibly scaling each column with a unit scalar:

\[ \Lambda = w \Lambda \]

\[ \Lambda = w \Lambda \]

The least squares solution: \( f_{H} \bar{\Omega}^{-1} R \approx \Lambda w \bar{H} \)

So, multiplying an equation \( f \) from the left by \( \bar{H} \) will give:

\[ \bar{H} \bar{\Omega}^{-1} R \]

Note that \( \bar{H} \bar{\Omega}^{-1} R \) is the left inverse of \( \bar{H} \).

\[ \| f R \| = 0 \| e_{H} \bar{\Omega} = 0 \| 1 + w \Lambda H \]

The solution to the least squares problem (in GCR) is given by

\[ \Lambda = w \Lambda \]

\[ \Lambda = w \Lambda \]

Column with a unit scalar:

Comparing with GCR, we see that apart from possibly scaling each column with a unit scalar:
Minimum Residual Solutions: GMRES

GMRES: \( Ax = b \)
Choose \( x_0 \), tolerance \( \varepsilon \); set \( r_0 = b - Ax_0 \); \( v_1 = r_0 / \| r_0 \|, k = 0 \).

while \( \| r_k \| \geq \varepsilon \) do

\[ k = k + 1 \]
\[ \tilde{v}_{k+1} = A v_k ; \]
for \( j = 1 \ldots k \),

\[ h_{j,k} = v_j^T \tilde{v}_{k+1} ; \tilde{v}_{k+1} = \tilde{v}_{k+1} - h_{j,k} v_j ; \]
end

\[ h_{k+1,k} = \| \tilde{v}_{k+1} \| ; \; v_{k+1} = \tilde{v}_{k+1} / h_{k+1,k} ; \]

Solve LS \( \min \{ \| r_k - H_k \zeta \| \} \) by construction

(actually we update the solution rather than solve from scratch – see later)

end

\[ x_k = x_0 + V_k \zeta_k ; \]
\[ r_k = r_0 - V_k H_k \zeta_k = V_k \left( \eta_1 \| r_0 \| - H_k \zeta_k \right) \text{ or simply } r_k = b - Ax_k \]

Convergence Restarted GCR

Test problem on unit square: \( 202 \times 202 \) grid points

Interior: \( -\nabla \cdot (\nabla u) = 0 \)
Boundary \( u = 1 \) for \( x = 0 \) and \( y = 1 \)
\( u = 0 \) elsewhere
Convergence restarted GMRES

Test problem on unit square: $202 \times 202$ grid points

Interior: $-\nabla \cdot (\nabla u) = 0$

Boundary $u = 1$ for $x = 0$ and $y = 1$

$u = 0$ elsewhere

GMRES vs GCR

<table>
<thead>
<tr>
<th>GMRES(m)</th>
<th>200 x 200 unknowns</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>time (s)</td>
</tr>
<tr>
<td>full</td>
<td>72.888</td>
</tr>
<tr>
<td>100</td>
<td>40.256</td>
</tr>
<tr>
<td>50</td>
<td>41.087</td>
</tr>
<tr>
<td>20</td>
<td>63.604</td>
</tr>
<tr>
<td>10</td>
<td>111.26</td>
</tr>
<tr>
<td>5</td>
<td>199.42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>rGCR(m)</th>
<th>200 x 200 unknowns</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>time (s)</td>
</tr>
<tr>
<td>full</td>
<td>215.87</td>
</tr>
<tr>
<td>100</td>
<td>114.04</td>
</tr>
<tr>
<td>50</td>
<td>97.89</td>
</tr>
<tr>
<td>20</td>
<td>103.56</td>
</tr>
<tr>
<td>10</td>
<td>131.69</td>
</tr>
<tr>
<td>5</td>
<td>180.88</td>
</tr>
</tbody>
</table>
4. Optimal Krylov Subspace Methods with short recurrences

Multigrid and Iterative Methods
Consider again how \( \text{GMRES} \) builds an orthogonal basis for \( \mathcal{R}(A, r_0) \), and \( \mathcal{R}(A, r_0) \) is contained in \( \mathcal{R}(A, r_0) \).

Verify that the (Arnoldi) algorithm generates the following recurrence:

\[
 v_1 = \frac{r_0}{\|r_0\|^2}; \quad v_k = \frac{Av_k}{\|Av_k\|^2}; \quad H V^{m+1} = V^{m} H + H V^{m+1};
\]

What does \( \Lambda H^I \Lambda^{m+1} \) look like?

\[
\Lambda = \Lambda^{m+1} H = H^{m+1} \Lambda
\]

Prove \( \Lambda^{m+1} \) is orthogonal.

Note \( \Lambda^{m+1} \) is orthogonal.

Verify that the (Arnoldi) algorithm generates the following recurrence:

\[
 v_1 = \frac{r_0}{\|r_0\|^2}; \quad v_k = \frac{Av_k}{\|Av_k\|^2}; \quad H V^{m+1} = V^{m} H + H V^{m+1};
\]

end
Now consider $A$ being Hermitian:

\[ A = A^H \]

Another way to write the recurrence relation from Arnoldi:

\[ AV_m = V_m + H_{m+1}v_m + e_m^T h_m + 1, \]

where $H_m$ is the upper part of $H$.

So, $H_m^* = H_m^T$.

Another way to write the recurrence relation from Arnoldi:

\[ AV = V A_H \]

This has some important consequences...

So, $H$ must be Hermitian as well.

\[ H = H^T \]

Thus, $A$ must be Hermitian as well.

\[ A = A^H \]

Now consider $A$ being Hermitian: $A = A^H$. (2)
A Hermitian upper Hessenberg matrix is tridiagonal. This means that (in exact arithmetic) we need to orthogonalize each new vector against the vectors $v^i$ and $v^{i-1}$.

What is the computational cost of $m$ iterations of GMRES?

We could solve the least squares problem in the same way as for MINRES (3). Except that we save on orthogonalizations (inner products), GMRES, except that we save on orthogonalizations.

Proof:

$\text{span}\{A^j v^1, v^2, \ldots, v^j\}$, and so $A^j v^1, v^2, \ldots, v^j$ span $K^j(A, v^1)$. Then $A v^1, v^2, \ldots, v^j$, $A v^1, v^2, \ldots, v^j$ be the vectors generated by the Arnoldi algorithm (so they span $K^m(A, v^1)$).

Theorem: Let $A$ be Hermitian and let $v^1, v^2, \ldots, v^m$ be the vectors generated by the Arnoldi algorithm (so they span $K^m(A, v^1)$). Then

$A v^i$ is orthogonal to $v^1, v^2, \ldots, v^{i-2}$, $A v^i$ and so $A v^i$ span $A v^1, v^2, \ldots, v^j$.

Proof:

$\text{span}\{v^1, v^2, \ldots, v^j\}$, and so $A v^1, v^2, \ldots, v^j$ span $K^j(A, v^1)$. Then $A v^1, v^2, \ldots, v^j$, $A v^1, v^2, \ldots, v^j$ be the vectors generated by the Arnoldi algorithm (so they span $K^m(A, v^1)$).
Proof:

Consider \( \lambda \in \mathcal{A}(A, v) \).

We know \( \lambda \in \text{span} \{A, v\}_1, \ldots, \{A, v\}_n \} \).

So if \( j + 1 > i \) then \( A^j v_1 \) and \( A^j v_1 = 0 \).

Since \( \lambda \in \mathcal{K}_j(A, v_1) \), we have \( A^j \mathcal{K}_{i+1}(A, v_1) \).

\[ \lambda A^j v_1 = \lambda H^j v_1 = \lambda H^j v_1 = \lambda A^j v_1 \]

(4) MINRES
The algorithm now proceeds as follows:

Lanczos recurrence: \( T \) for tridiagonal.

\[
\mathbf{A}^{w_{i+1}} - t_0 \mathbf{r}_0 = w \Lambda + 0 \mathbf{x} = w \mathbf{x}
\]

At each step we only orthogonalize on previous \( 2 \) vectors.

At end we update \( \Lambda \) and solve

\[
\mathbf{R}^i \mathbf{y} = \mathbf{Q}^i \mathbf{H} \mathbf{r}_0
\]

Every step we update the QR-decomposition of \( \mathbf{H} \) and solve

\[
\mathbf{w}^i \mathbf{w} \Lambda^i + 0 \mathbf{x} = w \Lambda \mathbf{A}
\]

We have

\[
\mathbf{H} \mathbf{O}^i = w \Lambda
\]

Solve just as in GMRES:

\[
\mathbf{A}^w \mathbf{b} \mathbf{r} = w \Lambda \mathbf{A}
\]

What would seem an obvious improvement. Can we do that here?

Note that each step we only orthogonalize on previous \( 2 \) vectors.

MINRES (5)
We need a trick.

completely. So we need all previous \( v_1 \).

The problem is that \( R^m \) changes and hence \( w \) changes (in general).

Can we update every step and discard the vectors \( w \)?

However, we need them for the update at the end.

Like to discard the other vectors.

Since we only orthogonalize on the previous two vectors, we would
A cunning plan:

Since changes completely every step, apply a change of variables $y^m$.

**Alternative for update:**

$$W^m y^m = V^m R^m - I$$

Take $w^m$ only known:

$$w^m y^m = R^m y^m = R^m R^m - I$$

Then $w^m$ columns are found by solving each iteration.

$$w^m \Lambda = w^m R^m$$

Columns are upper triangular with 2 upper diagonals.

From the Givens QR decomposition of a tridiagonal matrix is $R^m$.

$w^m$ changes. So we can update $w^m \Lambda$ without keeping all $w^m$.

$$w^m \Lambda = w^m R^m$$

$$w^m \Lambda = w^m R^m$$

Take $w^m$ and each component only the last component of $w^m \Lambda = w^m R^m$.

Since changes completely every step, apply a change of variables.

A cunning plan:
What would be an update formula for $y_m$?

Do we need to continue the iteration?

How many vectors do we need to keep (independent of # iterations)?

$$y_m = y_0 + \sum_{i=1}^{m} v_i x$$

$$x = x_0 + \sum_{i=1}^{m} \lambda_i v_i$$

Since $y_m$, contrary to $v_m$, changes only in its last position we can do

$$y_m = y_{m-1} + v_m$$

The update iteration-wise:

Update solution: $x = x_0 + \sum_{i=1}^{m} \Lambda_i$
$$A\mathbf{x} = \mathbf{b}$$

Choose and set $\mathbf{x}_0$, $\mathbf{d}_0 = \mathbf{b} - A\mathbf{x}_0$, $\mathbf{t}_0 = 0$.

While $\mathbf{d}_k > \mathbf{tol}$ do

$$\mathbf{v}_{k+1} = \mathbf{d}_k / \| \mathbf{d}_k \|$$

Update $\mathbf{R}$:

$$Q_{k+1} = Q_k G_k; \quad R_{k+1} = G_k H(Q_k H^T k_{k+1})$$

$$y_{k+1} = q_k H e \frac{1}{\| \mathbf{d}_0 \|}$$

$$w_k = r_{k+1} - (\mathbf{v}_k - w_{k-1} r_{k+1} - w_{k-2} r_{k+2})$$

$$x_k = x_{k-1} + w_k y_{k+1}$$

Choose $x_0$, $q_0 = A x_0$ and $\mathbf{t}_0 = 0$.

Set $k = 0$.

$$q = A x$$

END
Conjugate Gradients (1)

Hermitian matrices: Error minimization in the A-norm

We are solving $Ax = b$ with initial guess $x_0 \rightarrow r_0 = b - Ax_0$ and $\hat{x}$ is the solution to $Ax = b$.
The error at iteration $i$ is $e_i = \hat{x} - (x_0 + z_i)$, where $z_i \in K^i(A, r_0)$ is the $i$th update to the initial guess.

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

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

Conjugate Gradients (2)

Proof:
$z_i = \arg \min \{ \| \hat{x} - (x_0 + z) \|_A : z \in K^i(A, r_0) \} \iff (\hat{x} - x_0) - z_i \perp_A K^i(A, r_0)$

We know $K^i(A, r_0) = \text{span} \{ r_0, r_1, \ldots, r_{i-1} \}$.

This gives $r_k \perp_A (\hat{x} - x_0 - z_i)$ for $k = 0, \ldots, i - 1$ $\iff$

$\langle A(\hat{x} - x_0 - z_i), r_k \rangle$ for $k = 0, \ldots, i - 1$ $\iff$

$\langle b - Ax_0 - Az_i, r_k \rangle$ for $k = 0, \ldots, i - 1$ $\iff$

$\langle r_0 - Az_i, r_k \rangle$ for $k = 0, \ldots, i - 1$ $\iff$

$\langle r_i, r_k \rangle$ for $k = 0, \ldots, i - 1$ $\iff$

$r_i \perp K^i(A, r_0)$
Conjugate Gradients (3)

So we can minimize the error by choosing the update such that the new residual is orthogonal to all previous residuals. Hence, the name orthogonal residual methods.

(note the comparison between Orthomin(1) and Steepest Descent)

We can generate an orthogonal basis for the Krylov subspace using the Lanczos iteration, the 3-term recurrence version of the Arnoldi-iteration.

Lanczos iteration:

\[ q_1; \beta_0 = 0; q_0 = 0; \]

\text{FOR } i = 1, 2, \ldots \text{ DO}

\[ \begin{aligned} 
\tilde{q}_{i+1} &= Aq_i; \\
\alpha_i &= \langle Aq_i, q_i \rangle; \\
\beta_i &= \| \tilde{q}_{i+1} \|_2; \\
q_{i+1} &= \tilde{q}_{i+1} / \beta_i; \\
q_{i+1} &= q_{i+1} - \beta_{i-1} q_{i-1}; \\
Aq_i &= Q_i T_i + \beta_i q_{i+1} e_i^T, \text{ where } Q_i = [q_1 \ q_2 \ \cdots \ q_i], \ T_i = \begin{bmatrix} a_1 & \beta_1 & 0 & \cdots \\ \beta_1 & a_2 & \beta_2 & \cdots \\ 0 & \beta_2 & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots \\ \end{bmatrix}. 
\end{aligned} \]

Conjugate Gradients (4)

Show \( \tilde{q}_{i+1} = q_{i+1} - \beta_{i-1} q_{i-1} \) sets \( \tilde{q}_{i+1} \perp q_{i-1} \). (one argument is the symmetry of the Hessenberg matrix for Arnoldi, give another)

This algorithm generates the recurrence relation:
**Conjugate Gradients (5)**

Use Lanczos orthonormal basis for minimizing A-norm of error.

\[ z_i = \arg \min \{ \| \hat{x} - (x_0 + z) \|_A : z \in K_i(A, r_0) \} \]

iff \( r_i = r_0 - Az_i \) satisfies \( r_i \perp K_i(A, r_0) \).

\[ q_1 = r_0/\| r_0 \|_2; \]

Lanczos method: \( AQ_i = Q_iT_i + \beta_i q_{i+1} e_i^T \)

Solve \( r_0 - AQ_iy_i = 0 \) \( \iff \) \( Q_i^H (\| r_0 \|_2 q_1 - AQ_iy_i) = 0 \) \( \iff \) \( Q_i^H (\| r_0 \|_2 q_1 - AQ_iy_i) = 0 \iff \| r_0 \|_2 e_1 - Q_i^H AQ_iy_i = 0 \)

Notice \( \text{range}(Q_i) = \text{span}\{r_0, r_1, \ldots, r_{i-1}\} \).

\( AQ_i = Q_iT_i + \beta_i q_{i+1} e_i^T \Rightarrow Q_i^H AQ_i = T_i \)

So we reduced our problem to solving \( \| r_0 \|_2 e_1 - T_i y_i = 0 \):

\[ y_i = T_i^{-1} e_1 \| r_0 \|_2 \]

**Conjugate Gradients (6)**

In order to update step-by-step we use same trick as in MINRES:

Let \( T_i = L_i D_i L_i^H \); then \( y_i = L_i^{-1} D_i^{-1} L_i^{-1} e_1 \| r_0 \|_2 \), where \( L_i \) is unit lower bi-diagonal with lower diagonal coeff.s, \( l_1, l_2, \ldots, l_{i-2} \) (index gives the column)

Change of variables: \( P_i = Q_i L_i^{-H} \) and \( \hat{y}_i = D_i^{-1} L_i^{-1} e_1 \| r_0 \|_2 \):

\[ Q_i y_i = P_i \hat{y}_i \]

Each iteration only the last component of \( \hat{y}_i \) changes. From \( P_i L_i^H = Q_i \) we get a recurrence for \( p_i \): \( p_i + l_{i-1} p_{i-1} = q_i \) \( (p_1 = q_1) \)

So every new step we compute a new \( q_{i+1} \), we update the decomposition of \( T_i \) and from that \( \hat{y}_{i+1} \) and \( p_{i+1} \).

\[ x_i = x_{i-1} + p_i \hat{y}_{ii} \]

\[ r_i = r_{i-1} - Ap_i \hat{y}_{ii} = q_{i+1} \beta_i \hat{y}_{ii} \] (where \( \hat{y}_{ii} \) is \( i^{th} \) comp of vector \( \hat{y}_i \))
Conjugate Gradients (7)

This leads to the coupled two-term recurrence form of CG

CG algorithm: $Ax = b$

Choose $x_0 \rightarrow r_0 = b - Ax_0$;

$p_1 = r_0; i = 0$

While $\|r_i\|_2 > tol$ do

\[ i = i + 1; \]

\[ a_i = \frac{(r_i^T r_{i-1})}{(p_{i-1}^T A p_{i-1})}; \]

\[ x_i = x_i + a_i p_i; \]

\[ r_i = r_{i-1} - a_i A p_i; \]

\[ \beta_i = \frac{(r_i^T r_i)}{(r_{i-1}^T r_{i-1})}; \]

\[ p_i = r_i - \beta_i p_{i-1}; \]

End
A SPD, solve $Ax = b$ by iterative method $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$. Let $\hat{x}$ denote the solution ($A\hat{x} = b$).

Conjugate Gradients (CG)

$A$ defines inner product $\langle x, y \rangle_A = y^*Ax$ and $\|x\|_A = (x^*Ax)^{1/2}$

Often matrices from discretized PDE's are very sparse, so matrix-vector product very cheap: $O(n)$ work. However, std direct method, Gaussian elim. takes $O(n^3)$ work. That's where iterative methods come in.

At iteration $m$, method picks approx. solution $x_m \in \text{Ker}_m(A, b) = \text{span}\{ b, Ab, A^2b, \ldots, A^{m-1}b \}$
such that $\|\hat{x} - x_m\|_A$ minimal. (in this sense, the method is optimal)

Theo: $x_m = \arg\min_{x \in \text{Ker}_m} \|\hat{x} - x_m\|_A \iff \hat{x} - x_m \perp_A \text{Ker}_m(A, b)$

We first consider a general approach to implement this method, and then derive a very efficient algorithm. We follow the recipe in lemma 8.9.

Pick basis $p_0, p_1, \ldots, p_{m-1}$ for $\text{Ker}_m(A, b)$ and set $x_m = \sum_{j=0}^{m-1} p_j x_j$, such that $\hat{x} - x_m \perp_A \text{Ker}_m(A, b)$
Since \( \{p_0, ..., p_{m-1}\} \) basis for \( \mathcal{K}_m(A, b) \), the orthog. relation gives
\[
\langle \hat{x} - \sum_{j=0}^{m-1} p_j \xi_j, p_i \rangle_A = 0 \quad \text{for} \quad i = 0..m-1
\]
Form Gram matrix and solve the system
\[
p_i^* A (\hat{x} - \sum_{j=0}^{m-1} p_j \xi_j) = 0 \iff p_i^* b - \sum_{j=0}^{m-1} p_i^* A p_j \xi_j = 0
\]
for \( i = 0..m-1 \)

Note that writing out these equations, we get
\[
G : G_{ij} = p_i^* A p_j = \langle p_j, p_i \rangle_A \quad \text{and} \quad \hat{p} : \hat{p}_i = p_i^* A \hat{x} = p_i^* b = \langle \hat{x}, p_i \rangle_A
\]

Solve for the unknown expansion coefficients \( \xi_j \).
\[
G \xi = \hat{p} \iff \xi = G^{-1} \hat{p}
\]
\[
x_m = \sum_{j=0}^{m-1} p_j (G^{-1} \hat{p})_j \quad (\text{optimal approx. from } \mathcal{K}_m(A, b))
\]

This also provides some other useful/interesting props.
We cannot monitor the error \( \hat{x} - x_m \), so we monitor the residual
\[
r_m = b - A x_m = A (\hat{x} - x_m) = A e_m \quad (\text{em error})
\]
\[
\langle \hat{x} - x_m, p \rangle_A = 0 \quad \text{for all} \quad p \in \mathcal{K}_m(A, b) \iff
p^* A (\hat{x} - x_m) = p^* r_m = 0 \quad \text{for all} \quad p \in \mathcal{K}_m(A, b)
\]
So,
\[
r_m \perp \mathcal{K}_m(A, b) \quad (\text{in std. inner product})
\]
Next, we consider an efficient algorithm.
At each step, we extend the search space \( \mathcal{K}_m(A, b) \) to \( \mathcal{K}_{m+1}(A, b) \), pick additional \( p_m \) s.t. \( \{p_0, p_1, ..., p_m\} \)
is a basis for $\mathbb{K}_{m+1}(A, b)$.

However, we need to update and solve the Gram matrix system at each step. In general, $\xi$ changes in each component, so we must form $x_{m+1}$ from scratch, and store all vectors $p_0, p_1, \ldots$. We can do better.

If we pick the vectors $p_j$ $A$-orthogonal (conjugate), that is, $\langle p_j, p_i \rangle_A = p_j^* A p_i = 0$, for $i \neq j$, $G$ is diagonal. Solving for $G$ is then cheap and solving

\[
\begin{pmatrix}
    g_{11} & \cdots & g_{1m+1} \\
    \vdots & \ddots & \vdots \\
    g_{m1} & \cdots & g_{mm+1}
\end{pmatrix}
\begin{pmatrix}
    \xi_0 \\
    \xi_{m-1} \\
    \xi_m
\end{pmatrix}
= \begin{pmatrix}
    p_0^* b \\
    p_{m-1}^* b \\
    p_m^* b
\end{pmatrix}
\]

leaves $\xi_0, \ldots, \xi_{m-1}$ the same. This suggests that we can update $x_m$ as we go: $x_{m+1} = x_m + p_m \xi_m$ and discard the $p_j$ vectors. However, we need to orthogonalize the $p$ vectors. We also need to extend the Krylov space.

We saw that $r_m \perp \mathbb{K}_m(A, b)$. In addition $x_m \in \mathbb{K}_m(A, b)$ and therefore $r_m = b - A x_m \in \mathbb{K}_{m+1}(A, b)$. Since $r_m \perp \mathbb{K}_m(A, b), r_m \notin \mathbb{K}_m(A, b)$

so $\mathbb{K}_{m+1}(A, b) = \text{span}\{r_m\} \oplus \mathbb{K}_m(A, b)$

Note the direction $\text{span}\{r_m\}$ in $\mathbb{K}_{m+1}(A, b)$ and orthog. to $\mathbb{K}_m(A, b)$ is unique.
This suggests we pick \( p_m = r_m + \sum_{j=0}^{m-1} p_j y_j \) s.t.
\[ p_j^* A p_m = 0 \quad \text{for } j = 0 \ldots m-1. \]

It turns out we get this mostly for free:
\[ p_i^* A r_m + \sum_{j=0}^{m-1} p_i^* A p_j y_j = 0 \iff p_i^* A r_m + p_i^* A p_i y_i = 0 \]
\[ \implies p_i^* A p_j = 0 \quad i \neq j \]

So, \( y_i = -\frac{p_i^* A r_m}{p_i^* A p_i} \)

Note that \( p_i^* A r_m = r_m^* A p_i \). Since \( p_i \in K_{i+1}(A, b) \), \( A p_i \in K_{i+2}(A, b) \) and \( r_m \perp K_m(A, b) \).

So, \( r_m \perp A p_i \) for \( m \geq i+2 \iff i \leq m-2 \).

So, only need to orthog. against \( p_{m-1} : p_m = r_m + r_{m-1} p_{m-1} \)
\[ \iff p_m = r_m - \frac{p_{m-1}^* A r_m}{p_{m-1}^* A p_{m-1}} \]
\[ x_{m+1} = x_m + p_m \xi_m = x_m + p_m \frac{p_m^* b}{p_m^* A p_m} \]
\[ r_{m+1} = r_m - A p_m \xi_m \]

Few more simplifications:
\[ r_{m-1} = r_{m-2} - A p_{m-2} \xi_{m-2} = r_{m-3} - A p_{m-3} \xi_{m-3} - A p_{m-2} \xi_{m-2} \]
\[ \implies b = r_{m-1} + \sum_{i=0}^{m-2} A p_i \xi_i \]
\[ P_{m-1}^* b = P_{m-1}^* r_{m-1} + \sum_{i=0}^{m-2} P_{m-1}^* A P_i \xi_i \]

Moreover, \( P_{m-1} = r_{m-1} + P_{m-2} y_{m-2} \Rightarrow \)
\[
  r_{m-1}^* P_{m-1} = r_{m-1}^* r_{m-1} \quad (= P_{m-1}^* b) 
\]

Analogously \( P_{m-1}^* A r_m = r_{m}^* A P_{m-1} \) and \( r_{m}^* r_{m} = r_{m}^* r_{m-1} - r_{m}^* A P_{m-1} \xi_{m-1} \Leftrightarrow \)
\[
  r_{m}^* A P_{m-1} = \frac{r_{m}^* r_{m}}{\xi_{m-1}} = r_{m}^* r_{m} \cdot \frac{P_{m-1}^* A P_{m-1}}{P_{m-1}^* b} 
\]
\[
  = r_{m}^* r_{m} \cdot \frac{P_{m-1}^* A P_{m-1}}{r_{m-1}^* r_{m-1}} 
\]

So, \( \frac{P_{m-1}^* A r_m}{P_{m-1}^* A P_{m-1}} = \frac{r_{m}^* r_{m}}{r_{m-1}^* r_{m-1}} \cdot \frac{P_{m-1}^* A P_{m-1}}{P_{m-1}^* b} \) (is real)

\( x_0 = 0, \ r_0 = b, \ p_0 = r_0 \)
\[
  x_m = x_{m-1} + P_{m-1}^* \frac{r_{m-1}^* r_{m-1}}{P_{m-1}^* A P_{m-1}} = x_{m-1} + \alpha_{m-1} P_{m-1} 
\]
\[
  r_m = r_{m-1} - A P_{m-1} \cdot \frac{r_{m}^* r_{m}}{P_{m-1}^* A P_{m-1}} = r_{m-1} - \alpha_{m-1} A P_{m-1} 
\]
\[
  P_m = r_m - P_{m-1} \frac{r_{m}^* r_{m}}{r_{m-1}^* r_{m-1}} = r_m - \beta_{m-1} P_{m-1} 
\]

Very efficient algorithm. It keeps only a few vectors, and per iteration does 1 matvec, a few dot products, and a few vector additions.