Iterative methods for Linear System

JASS 2009

Student: Rishi Patil
Advisor: Prof. Thomas Huckle
Outline

• Basics:
  - Matrices and their properties
  - Eigenvalues, Condition Number

• Iterative Methods
  - Direct and Indirect Methods

• Krylov Subspace Methods
  - Ritz Galerkin: CG
  - Minimum Residual Approach: GMRES/MINRES
  - Petrov-Galerkin Method: BiCG, QMR, CGS
Basics

• **Linear system of equations**
  \[ Ax = b \]

• **A Hermitian matrix** (or self-adjoint matrix) is a square matrix with complex entries which is equal to its own conjugate transpose, that is, the element in the \( i \)th row and \( j \)th column is equal to the complex conjugate of the element in the \( j \)th row and \( i \)th column, for all indices \( i \) and \( j \)

\[ A = \begin{bmatrix} 3 & 2 + i \\ 2 - i & 1 \end{bmatrix} \]

• **Symmetric** if \( a_{ij} = a_{ji} \)
• **Positive definite if, for every nonzero vector \( x \)**
  \[ X^T A x > 0 \]
• **Quadratic form:**
  \[ f(x) = \frac{1}{2} x^T A x - b^T x + c \]
• **Gradient of Quadratic form:**
  \[ f'(x) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(x) \\ \vdots \\ \frac{\partial}{\partial x_n} f(x) \end{bmatrix} = \frac{1}{2} A^T x + \frac{1}{2} A x - b \]
Various quadratic forms

(a) Positive-definite matrix

(b) Negative-definite matrix

(c) Singular positive-indefinite matrix

(d) Indefinite matrix
Various quadratic forms
Eigenvalues and Eigenvectors

For any $n \times n$ matrix $A$, a scalar $\lambda$ and a nonzero vector $v$ that satisfy the equation

$Av = \lambda v$

are said to be the eigenvalue and the eigenvector of $A$.

- If the matrix is **symmetric**, then the following properties hold:
  1. the eigenvalues of $A$ are real
  2. eigenvectors associated with distinct eigenvalues are orthogonal

- The matrix $A$ is **positive definite** (or positive semidefinite) if and only if all eigenvalues of $A$ are positive (or nonnegative).
Eigenvalues and Eigenvectors

Why should we care about the eigenvalues? *Iterative methods often depend on applying $A$ to a vector over and over again:*

(a) If $|\lambda| < 1$, then $A^i v = \lambda^i v$ vanishes as $i$ approaches infinity.

(b) If $|\lambda| > 1$, then $A^i v = \lambda^i v$ will grow to infinity.
Some more terms:

**Spectral radius** of a matrix is: $\rho(A) = \max |\lambda_i|$

**Condition number** is: $K = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$

**Error**: $e = x_{\text{exact}} - x_{\text{app}}$

**Residual**: $r = b - A \cdot x_{\text{app}}$
Preconditioning

Preconditioning is a technique for improving the condition number of a matrix. Suppose that $M$ is a symmetric, positive-definite matrix that approximates $A$, but is easier to invert. We can solve $Ax = b$ indirectly by solving

$$M^{-1}Ax = M^{-1}b$$

Type of preconditioners:

- **Perfect** preconditioner $M = A$
  
  Condition number $= 1$ $\rightarrow$ solution in one iteration
  
  but $Mx = b$ is not useful preconditioner

- **Diagonal** preconditioner, trivial to invert but mediocre

- **Incomplete Cholesky**: $A \rightarrow LL^T$
  
  • Not always stable
Stationary and non-stationary methods

Stationary methods for $Ax = b$:

$$x^{(k+1)} = Rx^{(k)} + c$$

neither $R$ or $c$ depend upon the iteration counter $k$.

• Splitting of $A$
  $A = M - K$ with nonsingular $M$
  $Ax = Mx - Kx = b$
  $x = M^{-1}Kx - M^{-1}b = Rx + c$

Examples:
  • Jacobi method
  • Gauss-Seidel
  • Successive Overrelaxation (SOR)
Jacobi Method

• Splitting for Jacobi Method, \( M=D \) and \( K=L + U \)

\[
\begin{align*}
    x^{(k+1)} &= D^{-1}((L+U)x^{(k)} + b) \\
    \text{solve for } x_i \text{ from equation } i, \text{ assuming other entries fixed}
\end{align*}
\]

\[
\begin{align*}
    \text{for } i = 1 \text{ to } n \\
    \text{for } j = 1 \text{ to } n \\
    u_{i,j}^{(k+1)} &= (u_{i-1,j}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i,j+1}^{(k)})/4
\end{align*}
\]
Gauss-Siedel Method and SOR(Successive-Over-Relaxation)

Splitting for Jacobi Method, $M=D-L$ and $K=U$

$$x^{(k+1)} = (D-L)^{-1}(U x^{(k)} + b)$$

While looping over the equations, use the most recent values $x_i$ for $i = 1$ to $n$

for $i = 1$ to $n$
  for $j = 1$ to $n$
    $$u_{i,j}^{(k+1)} = (u_{i-1,j}^{(k+1)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k+1)} + u_{i,j+1}^{(k)})/4$$

Splitting for SOR:

$$x^{(k+1)} = \omega x_i^{(k+1)} + (1-\omega) x_i^{(k)}$$

OR

$$x^{(k+1)} = (D-\omega L)^{-1}(\omega U + (1-\omega) D) x^{(k)} + \omega (D-\omega L)^{-1} b$$
Stationary and non-stationary methods

- Non-stationary methods:
  - The constant are computed by taking inner products of residual or other vectors arising from the iterative method
  - Examples:
    - Conjugate gradient (CG)
    - Minimum Residual (MINRES)
    - Generalized Minimal Residual Residual (GMRES)
    - BiConjugate Gradient (BiCG)
    - Quasi Minimal Residual (QMR)
    - Conjugate Gradient Squared (CGS)
Descent Algorithms

Fundamental underlying structure for almost all the descent algorithms:

- Start with an initial point
- Determine according to a fixed rule a direction of movement
- Move in that direction to a relative minimum of the objective function
- At the new point, a new direction is determined and the process is repeated.
- The difference between different algorithms depends upon the rule by which successive directions of movement are selected
The Method of Steepest Descent

• In the method of steepest descent, one starts with an arbitrary point \( x_{(0)} \) and takes a series of steps \( x_{(1)}, x_{(2)}, \ldots \) until we are satisfied that we are close enough to the solution.

• When taking the step, one chooses the direction in which \( f \) decreases most quickly, i.e.

  \[ -f'(x_{(i)}) = b \ - \ Ax_{(i)} \]

  error vector: \( e_{(i)} = x_{(i)} - x \)

  residual: \( r_{(i)} = b - Ax_{(i)} \)

• From \( Ax = b \), it follows that

  \( r_{(i)} = -Ae_{(i)} = -f'(x_{(i)}) \)

Residual is direction of Steepest Descent
The Method of Steepest Descent

Starting at (-2,-2) take steps in direction of steepest descent of $f$.

The parabola is the intersection of surfaces.

Find the point of intersection of these surfaces that minimizes $f$.

The gradient of the bottommost point is orthogonal to gradient of previous step.
The Method of Steepest Descent
The Method of Steepest Descent

• The algorithm

\[ r_{(i)} = b - Ax_{(i)} \]

\[ \alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T Ar_{(i)}} \]

\[ x_{(i+1)} = x_{(i)} + \alpha_{(i)} r_{(i)} \implies e_{(i+1)} = e_{(i)} + \alpha_{(i)} r_{(i)} \]

• To avoid one matrix-vector multiplication, one uses

\[ r_{(i+1)} = r_{(i)} - \alpha_{(i)} Ar_{(i)} \]

The disadvantage of using this recurrence is that the residual sequence is determined without any feedback from the value of \( x_{(i)} \), so that round-off errors may cause \( x_{(i)} \) to converge to some point near \( x \).
Steepest Descent Problem

• The gradient at the minimum of a line search is orthogonal to the direction of that search ⇒ the steepest descent algorithm tends to make right angle turns, taking many steps down a long narrow potential well. Too many steps to get to a simple minimum.
The Method of Conjugate Directions

Basic idea:
• Pick a set of orthogonal search directions $d_{(0)}$, $d_{(1)}$, … , $d_{(n-1)}$
• Take exactly one step in each search direction to line up with $x$
• Solution will be reached in $n$ steps

Mathematical formulation:
1. For each step we choose a point
   \[ x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \]
2. To find $\alpha_{(i)}$, we use the fact that $e_{(i+1)}$ is orthogonal to $d_{(i)}$
The Method of Conjugate Directions

- To solve the problem of not knowing $e^{(i)}$, one makes the search directions to be $A$-orthogonal rather than orthogonal to each other, i.e.:

$$d^{T}_{(i)}Ad_{(j)} = 0$$
The Method of Conjugate Directions

• The new requirement is now that $\mathbf{e}_{(i+1)}$ is A-orthogonal to $\mathbf{d}_{(i)}$

$$
\frac{d}{d\alpha} f(\mathbf{x}_{(i+1)}) = f'(\mathbf{x}_{(i+1)})^T \frac{d\mathbf{x}_{(i+1)}}{d\alpha} = 0
$$

$$
\mathbf{r}_{(i+1)}^T \mathbf{d}_{(i)} = 0
$$

$$
\mathbf{d}_{(i)}^T \mathbf{A} \mathbf{e}_{(i+1)} = 0
$$

$$
\mathbf{d}_{(i)}^T \mathbf{A} \left( \mathbf{e}_{(i)} + \alpha_{(i)} \mathbf{d}_{(i)} \right) = 0
$$

$$
\alpha_{(i)} = \frac{\mathbf{d}_{(i)}^T \mathbf{r}_{(i)}}{\mathbf{d}_{(i)}^T \mathbf{A} \mathbf{d}_{(i)}}
$$

If the search vectors were the residuals, this formula would be identical to the method of steepest descent.
The Method of Conjugate Directions

- Calculation of the A-orthogonal search directions by a **conjugate Gram-Schmidt process**
  1. Take a set of linearly independent vectors $u_0, u_1, \ldots, u_{n-1}$
  2. Assume that $d_{(0)} = u_0$
  3. For $i > 0$, take an $u_i$ and subtracts all the components from it that are not A-orthogonal to the previous search directions

$$d_{(i)} = u_{(i)} + \sum_{j=0}^{i-1} \beta_{ij} d_{(j)}, \quad \beta_{ij} = -\frac{u_{(i)}^T A d_{(j)}}{d_{(j)}^T A d_{(j)}}$$

![Diagram](image)
The Method of Conjugate Directions

• The method of Conjugate Gradients is simply the method of conjugate directions where the search directions are constructed by conjugation of the residuals, i.e. \( u_i = r_i \)

• This allows us to simplify the calculation of the new search direction because

\[
\beta_{ij} = \begin{cases} 
  \frac{1}{\alpha_{(i-1)}} \frac{r_{(i)}^T r_{(i)}}{d_{(i-1)}^T A d_{(i-1)}} = \frac{r_{(i)}^T r_{(i)}}{r_{(i-1)}^T r_{(i-1)}} & i = j + 1 \\
  0 & i > j + 1 
\end{cases} 
\]

• The new search direction is determined as a linear combination of the previous search direction and the new residual

\[
d_{(i+1)} = r_{(i+1)} + \beta_i d_{(i)}
\]
The Method of Conjugate Directions

\[ x_0 = 0, \quad r_0 = b, \quad d_0 = r_0 \]

for \( k = 1, 2, 3, \ldots \)

\[ \alpha_k = \frac{(r_{k-1}^T r_{k-1})}{(d_{k-1}^T A d_{k-1})} \] step length

\[ x_k = x_{k-1} + \alpha_k d_{k-1} \] approx solution

\[ r_k = r_{k-1} - \alpha_k A d_{k-1} \] residual

\[ \beta_k = \frac{(r_k^T r_k)}{(r_{k-1}^T r_{k-1})} \] improvement

\[ d_k = r_k + \beta_k d_{k-1} \] search direction

- One matrix-vector multiplication per iteration
- Two vector dot products per iteration
- Four n-vectors of working storage
Krylov subspace

Krylov subspace $K_j$ is the linear combinations of $b, Ab, ..., A^{j-1}b$.

Krylov matrix $K_j = [b \ Ab \ A^2b \ ... \ A^{j-1}b]$.

Methods to construct a basis for $K_j$:

Arnoldi’s method and Lanczos method

Approaches to choosing a good $x_j$ in $K_j$:

- **Ritz-Galerkin approach**: $r_j = b - Ax_j$ is orthogonal to $K_j$ (Conjugate Gradient)
- **Minimum Residual approach**: $r_j$ has minimum norm for $x_j$ in $K_j$ (GMRES and MINRES)
- **Petrov-Galerkin approach**: $r_j$ is orthogonal to a different space $K_j(AT)$ (Biconjugate Gradient)
Arnoldi’s Method

The best basis $q_1, \ldots, q_j$ for the Krylov subspace $K_j$ is orthonormal. Each new $q_j$ comes from orthogonalizing $t = Aq_{j-1}$ to the basis vectors $q_1, \ldots, q_j$ that are already chosen. The iteration to compute these orthonormal q’s is Arnoldi’s method.

$$q_1 = b / \|b\|$$
for $j = 1, \ldots, n-1$
$$t = Aq_j$$
for $i = 1, \ldots, j$
$$h_{ij} = q_i^T$$
$$t = t - h_{ij}q_i$$
end;
$$h_{j+1,j} = \|t\|$$
$$q_{j+1} = t / h_{j+1,j}$$
end

$AQ_{n-1} = Q_n H_{n,n-1}$

$H_{n,n-1}$ is upper Hessenberg matrix
Lanczos Method

Lanczos method is specialized Arnoldi iteration, if $A$ is symmetric (real)

$$H_{n-1,n-1} = Q^T_{n-1} A Q_{n-1}$$

$H_{n-1,n-1}$ is tridiagonal and this means that in the orthogonalization process, each new vector has to be orthogonalized with respect to the previous two vectors only, since the inner products vanish.

$B_0 = 0$, $q_0 = 0$, $b = \text{arbitrary}$, $q_1 = b / \|b\|$

for $i = 1, \ldots, n-1$

$$v = Aq_i$$

$$a_i = q^T_i v$$

$$v = v - B_{i-1} q_{i-1} - a_i q$$

$$B_i = \|v\|$$

$$q_{i+1} = v / B_i$$

end
Minimum Residual Methods

**Problem:** If $A$ is **not symmetric positive definite**, CG is not guaranteed to solve $Ax=b$.

**Solution:** Minimum Residual Methods.
Choose $x_j$ in the Krylov subspace $K_j$ so that $\|b - Ax_j\|$ is minimal.

The first orthonormal vectors $q_1, \ldots, q_j$ go in the columns $Q_j$ so $Q_j^TQ_j = I$.
Setting $x_j = Q_j y$

$\| r_j \| = \|b - Ax_j\| = \|b - AQ_j y\| = \|b - Q_{j+1} H_{j+1,j} y\|$

Using first $j$ columns of Arnoldi’s formula $AQ = QH$

First $j$ columns of $QH = \begin{bmatrix} q_1 & \cdots & q_{j+1} \end{bmatrix} \begin{bmatrix} h_{11} & \cdots & h_{1j} \\ h_{12} & \ddots & \vdots \\ \vdots & \ddots & h_{jj} \\ h_{j+1,1} & \cdots & h_{j+1,j} \end{bmatrix}$
Minimum Residual Methods

The problem becomes:
Choose \( y \) to minimize
\[
\| r_j \| = \| Q_{j+1}^T b - H_{j+1,j} y \|
\]
This is least squares problem.
Using zeros in \( H \) and \( Q_{j+1}^T b \) to find a fast algorithm that computes \( y \).

**GMRES** (Generalised Minimal Residual Approach)

*\( A \) is *not symmetric* and the upper triangular part of \( H \) can be full.
All previously computed vectors have to be stored.

**MINRES**:(Minimal Residual Approach)

*\( A \) is symmetric* (likely indefinite) and \( H \) is tridiagonal.
Avoids storage of all basis vectors for the Krylov subspace

Aim: to clear out the non-zero diagonal below the main diagonal of \( H \).
This is done by *Givens rotations*
**GMRES**

**Algorithm: GMRES**

\[ q_1 = \frac{b}{\|b\|} \]

**for** \( j = 1, 2, 3... \)

*step j of Arnoldi iteration*

**Find y to minimize** \( \| r_j \| = \| Q_{j+1}^T b - H_{j+1,j} y \| \)

\[ x_j = Q_j y \]

**Full-GMRES**

The upper triangle in \( H \) can be full and step \( j \) becomes expensive and possibly it is inaccurate as \( j \) increases.

**GMRES\((m)\)**

Restarts the GMRES algorithm every \( m \) steps However tricky to choose \( m \).
Petrov-Galerkin approach

- $r_j$ is orthogonal to a different space $\text{K}_j(\text{A}^T)$

- BiCG (Bi-Conjugate Gradient)
- QMR (Quasi Minimum Residual)
- CGS (Conjugate Gradient Squared)
Lanczos Bi-Orthogonalization Procedure

- Extension of the symmetric Lanczos algorithm
- Builds a pair of bi-orthogonal bases for the two subspaces $K_m(A, v_1)$ and $K_m(A^T, w_1)$

Choose two vectors $v_1, w_1$ such that $(v_1, w_1) = 1$

Set $\beta_1 = \delta_1 = 0$, $w_0 = v_0 = 0$

For $j = 1, 2, \ldots, m$ Do:

\[ \alpha_j = (Av_j, w_j) \]
\[ \hat{v}_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1} \]
\[ \hat{w}_{j+1} = A^T w_j - \alpha_j w_j - \delta_j w_{j-1} \]
\[ \delta_{j+1} = |(\hat{v}_{j+1}, \hat{w}_{j+1})|^{1/2}. \text{ If } \delta_{j+1} = 0 \text{ Stop} \]
\[ \beta_{j+1} = (\hat{v}_{j+1}, \hat{w}_{j+1})/\delta_{j+1} \]
\[ w_{j+1} = \hat{w}_{j+1}/\beta_{j+1} \]
\[ v_{j+1} = \hat{v}_{j+1}/\delta_{j+1} \]

EndDo
Bi-Conjugate Gradient (BiCG)

Compute $r_0 := b - Ax_0$. Choose $r_0^*$ such that $(r_0, r_0^*) \neq 0$.

Set, $p_0 := r_0$, $p_0^* := r_0^*$

For $j = 0, 1, \ldots$, until convergence Do:

\[
\alpha_j := (r_j, r_j^*) / (Ap_j, p_j^*)
\]

\[
x_{j+1} := x_j + \alpha_j p_j
\]

\[
r_{j+1} := r_j - \alpha_j Ap_j
\]

\[
r_{j+1}^* := r_j^* - \alpha_j A^T p_j^*
\]

\[
\beta_j := (r_{j+1}, r_{j+1}^*) / (r_j, r_j^*)
\]

\[
p_{j+1} := r_{j+1} + \beta_j p_j
\]

\[
p_{j+1}^* := r_{j+1}^* + \beta_j p_j^*
\]

EndDo
Quasi Minimum Residual (QMR)

- QMR uses unsymmetric Lanczos algorithm to generate a basis for the Krylov subspaces
- The lookahead technique avoids breakdowns during Lanczos process and makes QMR robust.

\[ r_0 = b - Ax_0 \text{ and } \gamma_0 := \|r_0\|_2, \quad w_1 := v_1 := r_0 / \gamma_1 \]

For \( m = 1, 2, \ldots \), until convergence Do:

- Compute \( \alpha_m, \delta_{m+1} \) and \( \nu_{m+1}, \omega_{m+1} \) as in Lanczos Algor.

Update the QR factorization of \( \tilde{T}_m \), i.e.,

- Apply \( \Omega_i, i = m - 2, m - 1 \) to the \( m \)-th column of \( \tilde{T}_m \)

- Compute the rotation coefficients \( c_m, s_m \)

Apply rotation \( \Omega_m \), to \( \tilde{T}_m \) and \( \tilde{g}_m \), i.e., compute:

\[ \gamma_{m+1} := -s_m \gamma_m; \quad \gamma_m := c_m \gamma_m; \text{ and } \alpha_m := c_m \alpha_m + s_m \delta_{m+1} \]
\[ \rho_m = \left( v_m - \sum_{i=m-2}^{m-1} t_{im} p_i \right) / t_{mm} \]
\[ x_m = x_{m-1} + \gamma_m \rho_m \]

If \( |\gamma_{m+1}| \) is small enough Stop

EndDo
Conjugate Gradient Squared (CGS)

**Compute** $r_0 := b - Ax_0$; $r_0^*$ arbitrary.

**Set** $p_0 := u_0 := r_0$.

For $j = 0, 1, 2 \ldots$, until convergence Do:

\[
\begin{align*}
\alpha_j &= (r_j, r_0^*) / (Ap_j, r_0^*) \\
q_j &= u_j - \alpha_j Ap_j \\
x_{j+1} &= x_j + \alpha_j (u_j + q_j) \\
r_{j+1} &= r_j - \alpha_j A (u_j + q_j) \\
\beta_j &= (r_{j+1}, r_0^*) / (r_j, r_0^*) \\
u_{j+1} &= r_{j+1} + \beta_j q_j \\
p_{j+1} &= u_{j+1} + \beta_j (q_j + \beta_j p_j)
\end{align*}
\]

EndDo
Summary

• Stationary Iterative Solvers:
  • Jacobi, Gauss-Seidel, SOR

• Non-Stationary Solvers:
  • Krylov subspace methods
    • Conjugate Gradient
      • Symmetric positive definite systems
  • GMRES and MINRES
    • Non-symmetric matrices, but expensive
  • Bi-CG
    • Non-symmetric, two matrix-vector product
  • QMR
    • Non-symmetric, avoids irregular convergence of BiCG
  • CGS
    • Non-symmetric, faster than BICG, does not require transpose
References

• An Introduction to the Conjugate Gradient Method Without the Agonizing Pain  *Jonathan Richard Shewchuk*  August 4, 1994
• Closer to the solution: Iterative linear solvers – *Gene h. Golub*, *Henk A. Van der Vorst*
• Krylov subspaces and Conjugate Gradient-  *Gilbert Strang*
Thank You!