Iterative methods for Linear System of Equations

Joint Advanced Student School (JASS-2009)

Course #2: Numerical Simulation - from Models to Software

Introduction

In numerical simulation, Partial Differential Equations (PDE) are solved to model some physical phenomenon. The PDEs are discretized by methods like Finite Differences, Finite Element Method, etc to obtain a linear system of equations of the following form:

$$\mathbf{Ax = b}$$

where A is a matrix (n*n)

b is the known vector (n)

x is the unknown vector to be solved (n)

$$\begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & & A_{2n} \\ \vdots & & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Such a system can be solved by direct methods like Gauss-Elimination if the system is small. For a system with large number of unknowns, the direct methods are not efficient due to increasing number of operations and the increasing memory requirements. Hence iterative methods are used to solve the linear system of equations.

Notations

A matrix is *symmetric positive definite* if for every non-zero vector *x*

$$\mathbf{x}^T\mathbf{Ax} > 0$$

Quadratic form is defined as follows:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x} + c$$

The gradient of quadratic form is :

$$f'(x) = \frac{1}{2}A^T x + \frac{1}{2}Ax - b$$

If A is symmetric then this quadratic form reduces to:

$$f'(x) = Ax - b$$

Eigenvalues and Eigenvectors

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

$\mathbf{Av}=\lambda\mathbf{v}$

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

If the matrix is *symmetric*, then the following properties hold:

(a) the eigenvalues of **A** are real

(b) 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).

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 $\mathbf{A}^i\mathbf{v}=\lambda^i\mathbf{v}$ vanishes as i approaches infinity

(b) If $|\lambda|>1$, then $\mathbf{A}^i\mathbf{v}=\lambda^i\mathbf{v}$ will grow to infinity.

Some more terms:

Spectral radius of a matrix **A** is defined as the maximum value of eigenvalue.

$$\rho(A) = \max|\lambda_i|$$

Condition number is the ratio of the maximum and minimum eigenvalues. $K = \frac{\lambda_{max}}{\lambda_{min}}$

Error is defined as the difference between the exact value and the calculated approximate value.

$$e = x_{exact} - x_{app}$$

Typically the exact value is not known and we try to get an approximation for this unknown by solving

the linear system in iterative manner. Hence the residual is calculated. The residual indicates how far we are from the correct value of ***b*** and is defined as:

$$\mathbf{r = b - A.x_{app}}$$

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

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

There are different type of preconditioners:

- **Perfect preconditioner**

  $M = A$

  In this case condition number is one and the solution is reached in one iteration, however it is not a very useful technique as calculating the inverse is a tedious task.

- **Diagonal preconditioner**

  Using a diagonal as a preconditioner is a trivial however mediocre technique.

- **Incomplete Cholesky**

  The Choleksy factorization of a positive definite matrix A is $A=LL^T$ where L is a lower triangular matrix. However this is not always stable

Stationary and non-stationary Iterative Methods

There are two types of iterative methods. Stationary methods for $Ax = b$:

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

neither R or c depend upon the iteration counter k.. These methods are older, simpler to understand and usually not so effective. Examples of such methods are Jacobi, Gauss-Seidel, SOR, etc.

Matrix A can be split as follows: $A = M - K$ with nonsingular M

$Ax = Mx - Kx = b$

$x = M^{-1}Kx - M^{-1}b = Rx + c$

Non-stationary methods:

Nonstationary methods differ from stationary methods in that the computations involve information that changes at each iteration. Typically, constants are computed by taking inner products of residuals or other vectors arising from the iterative method

Examples:

Conjugate gradient (CG)

Minimum Residual (MINRES)

Generalized Minimal Residual (GMRES)

BiConjugate Gradient (BiCG)

Quasi Minimal Residual (QMR)

Conjugate Gradient Squared (CGS)

Krylov subspace

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

Krylov matrix

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

The methods to construct a basis q_j for K_j are *Arnoldi's* method and *Lanczos* method.

Krylov subspace methods fall in three different classes:

- *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 $K_j(A^T)$ (Biconjugate Gradient)

Arnoldi's Method

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

The algorithm is as follows:

```
q_1 = b / ||b||          % Normalize b to ||q_1|| = 1
 for j = 1,...,n_1       % Start computation of q_{j+1}
   t = A q_j             % one matrix multiplication
  for i = 1,...,j        % t is in the space K_{j+1}
```

$h_{ij} = q^T_i$ % $h_{ij}q^T_i$ =projection of t on q_i

$t = t - h_{ij}q_i$ % Subtract that projection

end; % t is orthogonal to $q_1,...,q_j$

$h_{j+1,j} = ||t||$ % Compute the length of t

$q_{j+1} = t / h_{j+1,j}$ % Normalize t to $||q_{j+1}||=1$

end %$q_1,... q_n$ are orthnormal

$AQ_{n-1} = Q_n H_{n,n-1}$ $H_{n,n-1}$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= arbitrary, $q_1=b / ||b||$

for i = 1,...,n_1

$v = Aq_j$

$\alpha_i = q^T_i v$

$v = v - B_{i-1} q_{i-1} - \alpha_i q$

$B_i = ||v||$

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

end

Conjugate Gradient (CG)

Conjugate gradient methods are based upon steepest descent method. 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

In the method of steepest descent, one starts with an arbitrary point $\mathbf{x}_{(0)}$ and takes a series of steps $\mathbf{x}_{(1)}$, $\mathbf{x}_{(2)}$, … 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. $f'(\mathbf{x}_{(i)}) = b-\mathbf{A}\mathbf{x}_{(i)}$

The error and residual are defined as :

error vector: $\mathbf{e}_{(i)}=\mathbf{x}_{(i)}-\mathbf{x}$

residual: $\mathbf{r}_{(i)}=\mathbf{b}-\mathbf{A}\mathbf{x}_{(i)}$

From $\mathbf{Ax}=\mathbf{b}$, it follows that

$$\mathbf{r}_{(i)}=-\mathbf{A}\mathbf{e}_{(i)}=-f'(\mathbf{x}_{(i)})$$

Residual is direction of Steepest Descent.

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

The conjugate gradient method overcomes the disadvantage of steepest descent method by picking up a set of A-orthogonal search directions and taking exactly one step in each search direction. Hence the solution will be reached in *n* steps.

The conjugate gradient algorith is as follows:

$x_0 = 0, \quad r_0 = b, \quad d_0 = r_0$

for $k = 1, 2, 3, \ldots$

$\alpha_k = (r^T_{k-1} r_{k-1}) / (d^T_{k-1} A d_{k-1})$	%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
$\beta_k = (r^T_k r_k) / (r^T_{k-1} r_{k-1})$	%improvement
$d_k = r_k + \beta_k d_{k-1}$	%search direction

Minimum Residual Approaches (MINRES and GMRES)

If A is not symmetric positive definite, then CG is not suitable to solve Ax=b. Hence Minimum Residual Methods like MINRES (Minimum Residual approach) and GMRES (Generalized Minimum Residual Approaches).

Choose x_j in the Krylov subspace K_j so that $||b - Ax_j||$ is minimal. The first orthonormal vectors

$q_1,...,q_j$ go in the columns Q_j so $Q_j^T Q_j = I$

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 = $\begin{bmatrix} q_1 & \cdots & q_{j+1} \end{bmatrix} \begin{bmatrix} h_{11} & \cdots & h_{1j} \\ h_{12} & \ddots & \vdots \\ & \ddots & h_{jj} \\ & & h_{j+1,j} \end{bmatrix}$

The problem becomes to choose $\boldsymbol{y}$ to minimize

$$|| r_j ||= ||Q^T_{j+1} b - H_{j+,1,j} y||$$

which is a least squares problem.

Using zeros in $\boldsymbol{H}$ and $\boldsymbol{Q^t_{j+1} b}$ to find a fast algorithm that computes $\boldsymbol{y}$.

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

GMRES algorithm is as follows:

$q_1 = b / ||b||$

for j = 1, 2, 3...

step j of Arnoldi iteration

Find y to minimize $\boldsymbol{|| r_j ||= ||Q^T_{j+1} b - H_{j+,1,j} y||}$

$x_j = Q_j\, y$

There are two vairants of GMRES: Full-GMRES and GMRES(m)

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 Methods

The Petrov-Galerkin methods are the third category of Krylov subspace methods, which are based on finding $\mathbf{x}_{(k)}$ such that the residual is orthogonal to some other suitable k-dimensioned subspace..

Lanczos Bi-orthogonalisation process which is an extension of symmetric Lanczos algorithm is used to obtain a pair of bi-orthogonal subspaces.

BiCG (Bi-Conjugate Gradient)

Bi-CG finds two mutually orthogonal sequences r_0 and r_0^*

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:,***

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

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

$r_{j+1} := r_j - \alpha_j A p_j$

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

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

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

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

EndDo

QMR (Quasi Minimal Residual)

BiCG often has irregular convergence behaviour. The implicit LU decomposition of reduced tridiagonal system may not exist, resulting in breakdown of BiCG. QMR solves the reduced tridiagonal system in least square sense. 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.

Compute $r_0 = b - Ax_0$ ***and*** $\gamma_0 := \|r_0\|_2$, $w_1 := v_1 := r_0/\gamma_1$

For $m = 1, 2, \ldots,$ ***until convergence Do:***

Compute α_m, δ_{m+1} ***and*** v_{m+1}, w_{m+1} ***as in Lanczos Algor.***

Update the QR factorization of $\bar{T}_m$***, i.e.,***

Apply Ω_i, $i = m-2, m-1$ ***to the*** m***-th column of*** $\bar{T}_m$

Compute the rotation coefficients c_m, s_m

Apply rotation Ω_m, ***to*** $\bar{T}_m$ ***and*** $\bar{g}_m$, ***i.e., compute:***

$\gamma_{m+1} := -s_m\gamma_m$; $\gamma_m := c_m\gamma_m$; ***and*** $\alpha_m := c_m\alpha_m + s_m\delta_{m+1}$

$p_m = \left(v_m - \sum_{i=m-2}^{m-1} t_{im}p_i\right)/t_{mm}$

$x_m = x_{m-1} + \gamma_m p_m$

If $|\gamma_{m+1}|$ ***is small enough Stop***

EndDo

CGS (Conjugate Gradient Squared)

CGS is another variatn of Petrov-Galerkin approach and is often faster compared to BiCG however it has a irregular convergence behaviour.

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:***

$\alpha_j = (r_j, r_0^*)/(Ap_j, r_0^*)$

$q_j = u_j - \alpha_j A p_j$

$x_{j+1} = x_j + \alpha_j(u_j + q_j)$

$r_{j+1} = r_j - \alpha_j A(u_j + q_j)$

$\beta_j = (r_{j+1}, r_0^*)/(r_j, r_0^*)$

$u_{j+1} = r_{j+1} + \beta_j q_j$

$p_{j+1} = u_{j+1} + \beta_j(q_j + \beta_j p_j)$

EndDo

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 Stran*

- *Templates for the Solution of Linear Systems:* Building Blocks for Iterative Methods1 - *Henk Van der Vorst etal.*