
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-Gaalerkin Method: BiCG, QMR, CGS

Basics

- **Linear system of equations**

$$\mathbf{Ax} = \mathbf{b}$$

- 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

$$\mathbf{A} = \begin{bmatrix} 3 & 2+i \\ 2-i & 1 \end{bmatrix}$$

- *Symmetric* if $a_{ij} = a_{ji}$
- *Positive definite* if, for every nonzero vector \mathbf{x}

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

- *Quadratic form:*

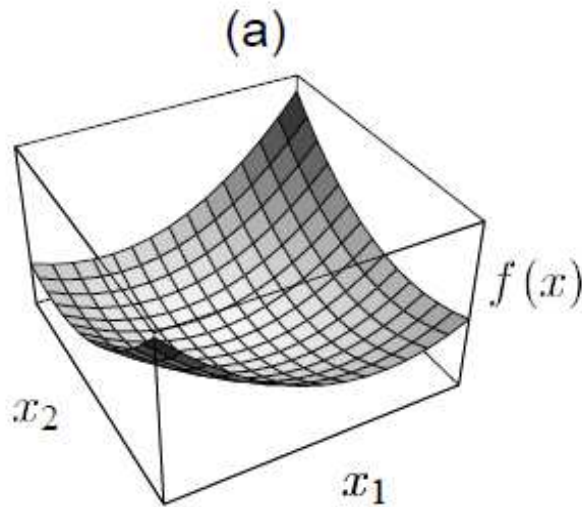
$$f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Ax} - \mathbf{b}^T \mathbf{x} + c$$

- *Gradient of Quadratic form:*

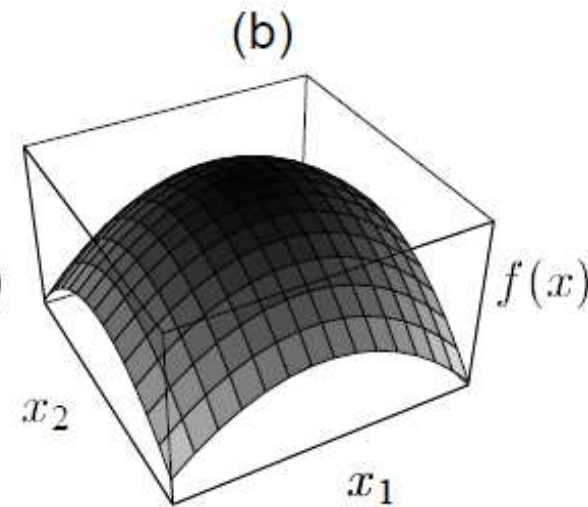
$$f'(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_n} f(\mathbf{x}) \end{bmatrix} = \frac{1}{2} \mathbf{A}^T \mathbf{x} + \frac{1}{2} \mathbf{Ax} - \mathbf{b}$$

Various quadratic forms

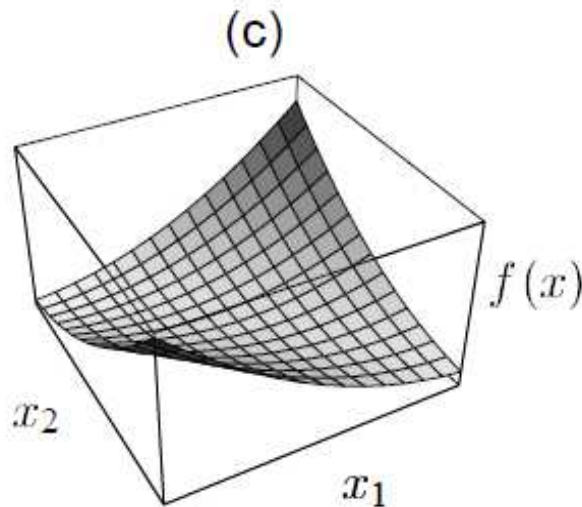
*Positive-definite
matrix*



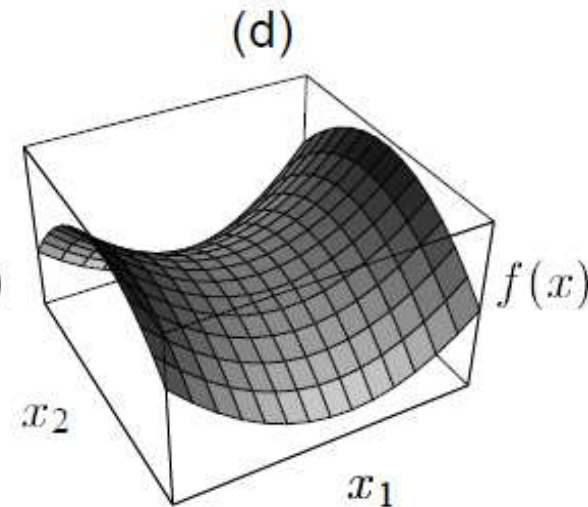
*Negative-definite
matrix*



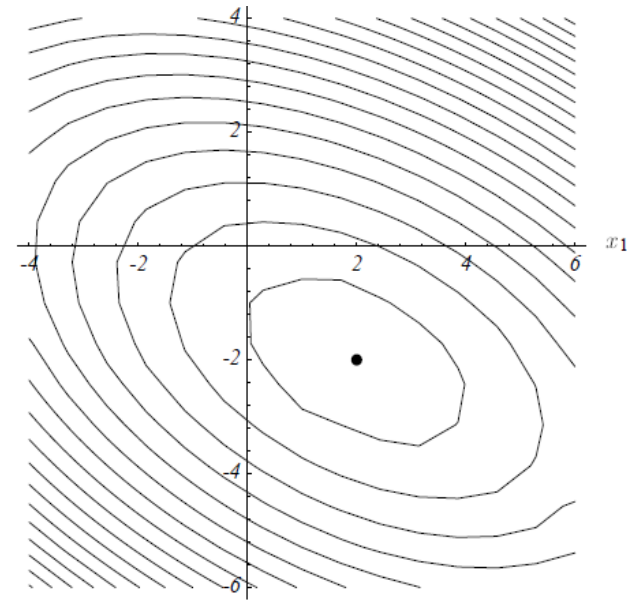
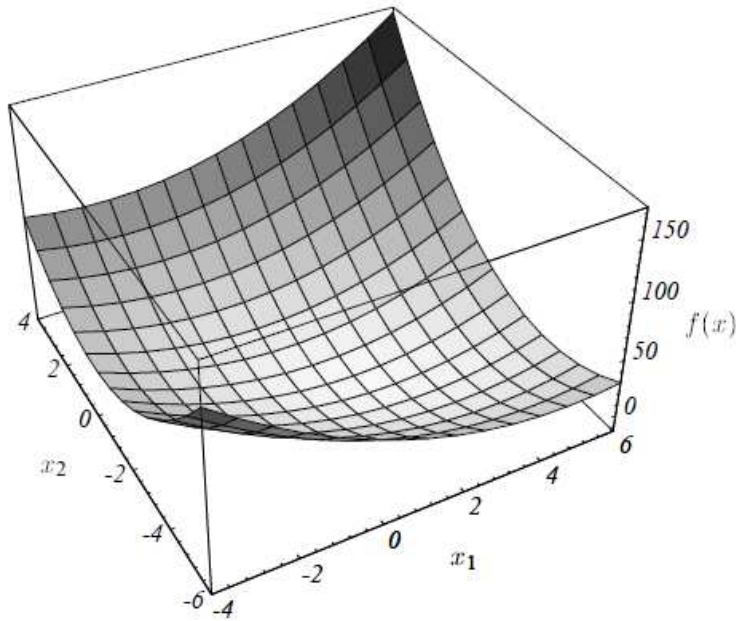
*Singular
positive-indefinite
matrix*



*Indefinite
matrix*



Various quadratic forms



Eigenvalues and Eigenvectors

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

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

are said to be the eigenvalue and the eigenvector of \mathbf{A} .

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

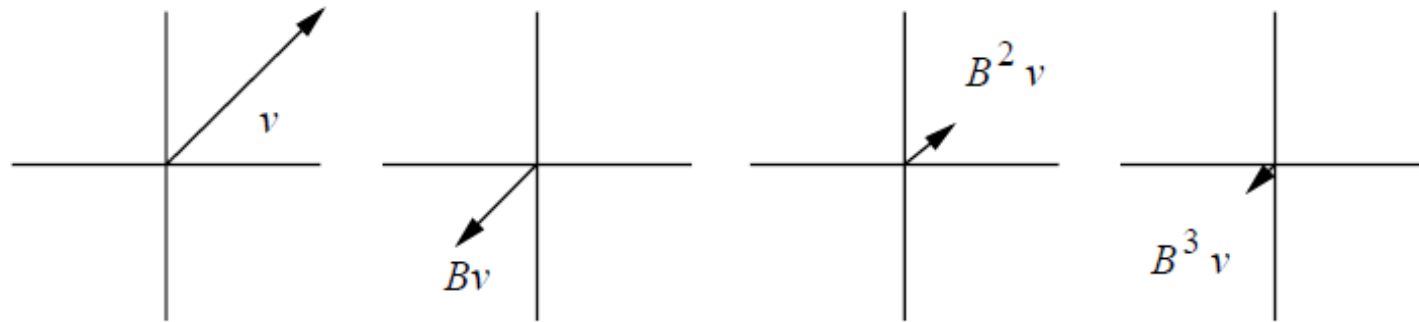
(a) the eigenvalues of \mathbf{A} are real

(b) eigenvectors associated with distinct eigenvalues are orthogonal

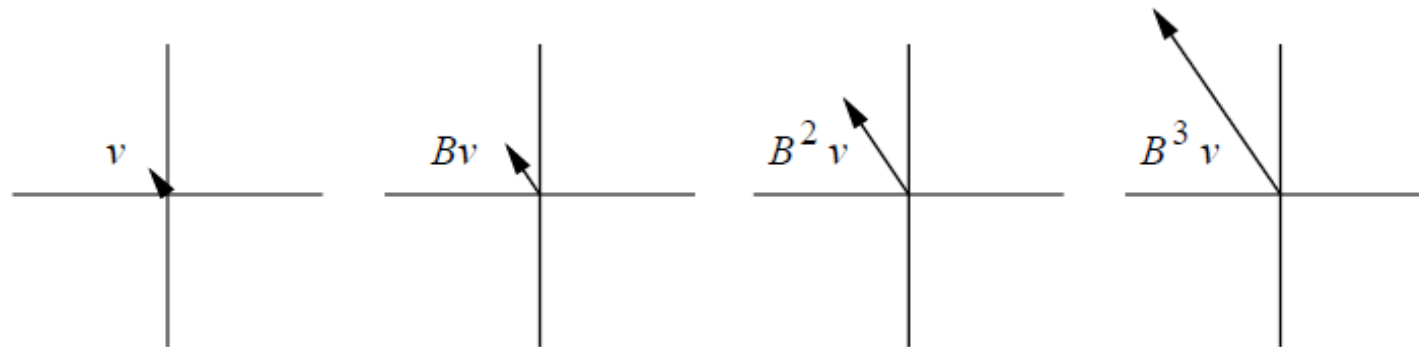
• The matrix \mathbf{A} is *positive definite* (or positive semidefinite) if and only if all eigenvalues of \mathbf{A} are positive (or nonnegative).

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 $\mathbf{A}^i \mathbf{v} = \lambda^i \mathbf{v}$ 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 is: $\rho(\mathbf{A}) = \max|\lambda_i|$

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

Error: $\mathbf{e} = \mathbf{x}_{exact} - \mathbf{x}_{app}$

Residual: $\mathbf{r} = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}_{app}$

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 $\mathbf{Ax} = \mathbf{b}$ indirectly by solving

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

¹Type 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 $K=L+U$

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

solve for x_i from equation i , assuming other entries fixed

for $i = 1$ **to** n

for $j = 1$ **to** n

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

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 $j = 1$ **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:

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

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 (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 $\mathbf{x}_{(0)}$ 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.

- Definitions: $-\mathbf{f}'(\mathbf{x}_{(i)}) = \mathbf{b} - \mathbf{A}\mathbf{x}_{(i)}$

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

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

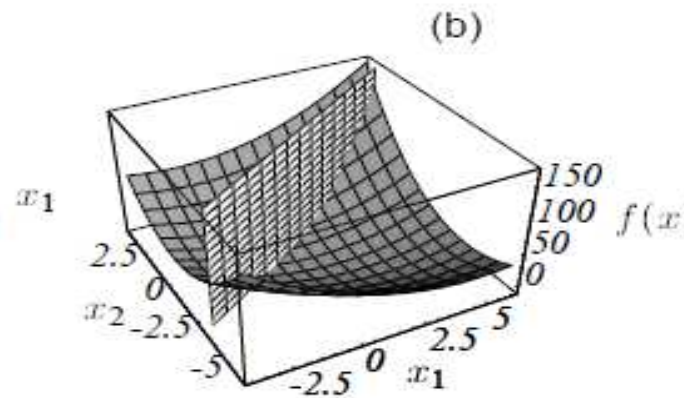
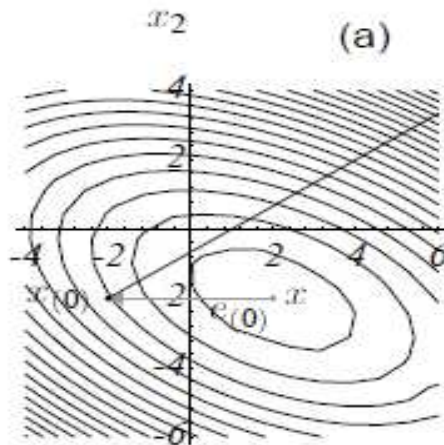
- From $\mathbf{A}\mathbf{x} = \mathbf{b}$, it follows that

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

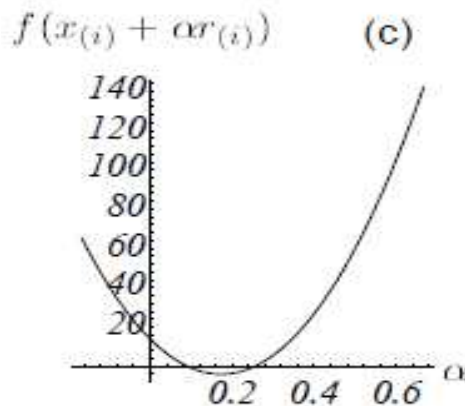
Residual is direction of Steepest Descent

The Method of Steepest Descent

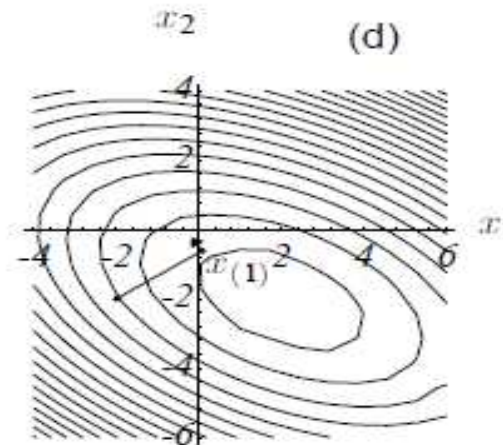
Starting at $(-2, -2)$ take steps in direction of steepest descent of f



Find the point of intersection of these surfaces that minimizes f

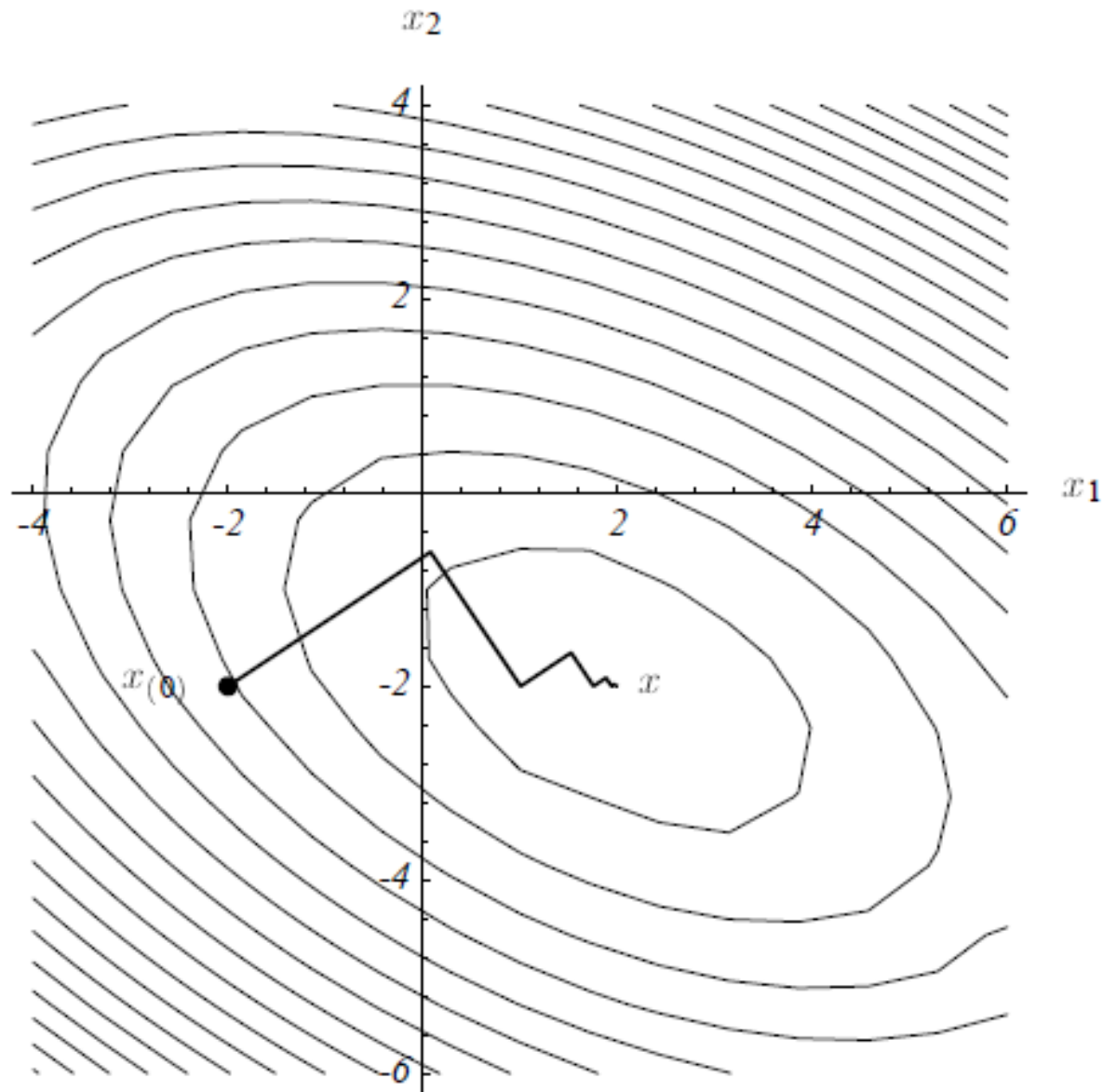


The parabola is the intersection of surfaces



The gradient of the bottommost point is orthogonal to gradient of previous step

The Method of Steepest Descent



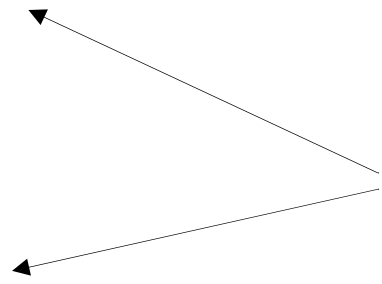
The Method of Steepest Descent

- The algorithm

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

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

Two matrix-vector multiplications are required.



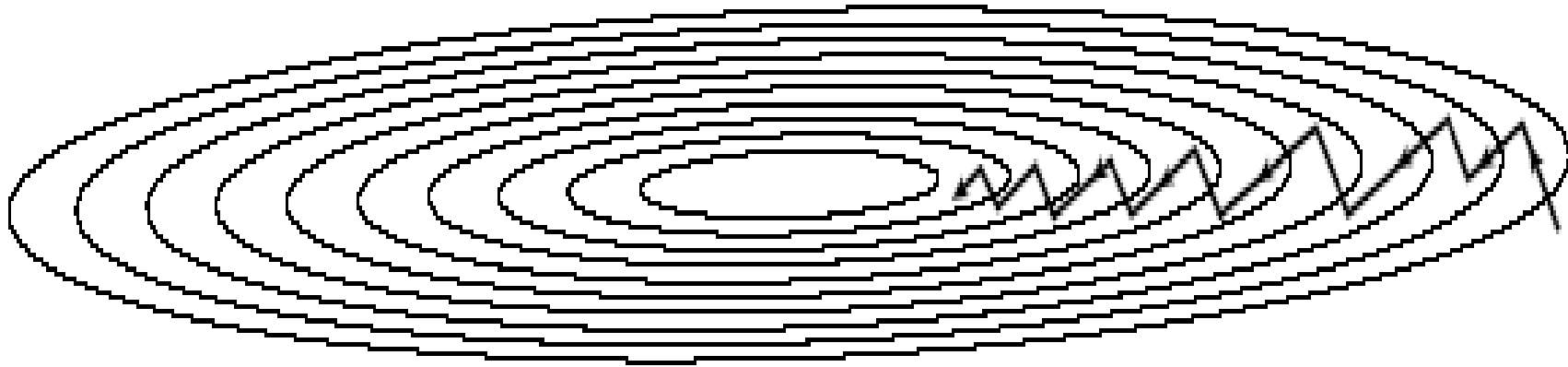
$$\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \alpha_{(i)}\mathbf{r}_{(i)} \Rightarrow \mathbf{e}_{(i+1)} = \mathbf{e}_{(i)} + \alpha_{(i)}\mathbf{r}_{(i)}$$

- To avoid one matrix-vector multiplication, one uses

$$\mathbf{r}_{(i+1)} = \mathbf{r}_{(i)} - \alpha_{(i)}\mathbf{A}\mathbf{r}_{(i)}$$

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} .

Steepest Descent Problem



- The gradient at the minimum of a line search is orthogonal to the direction of that search \Rightarrow 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 $\mathbf{d}_{(0)}, \mathbf{d}_{(1)}, \dots, \mathbf{d}_{(n-1)}$
- Take exactly one step in each search direction to line up with \mathbf{x}
- Solution will be reached in n steps

- Mathematical formulation:

1. For each step we choose a point

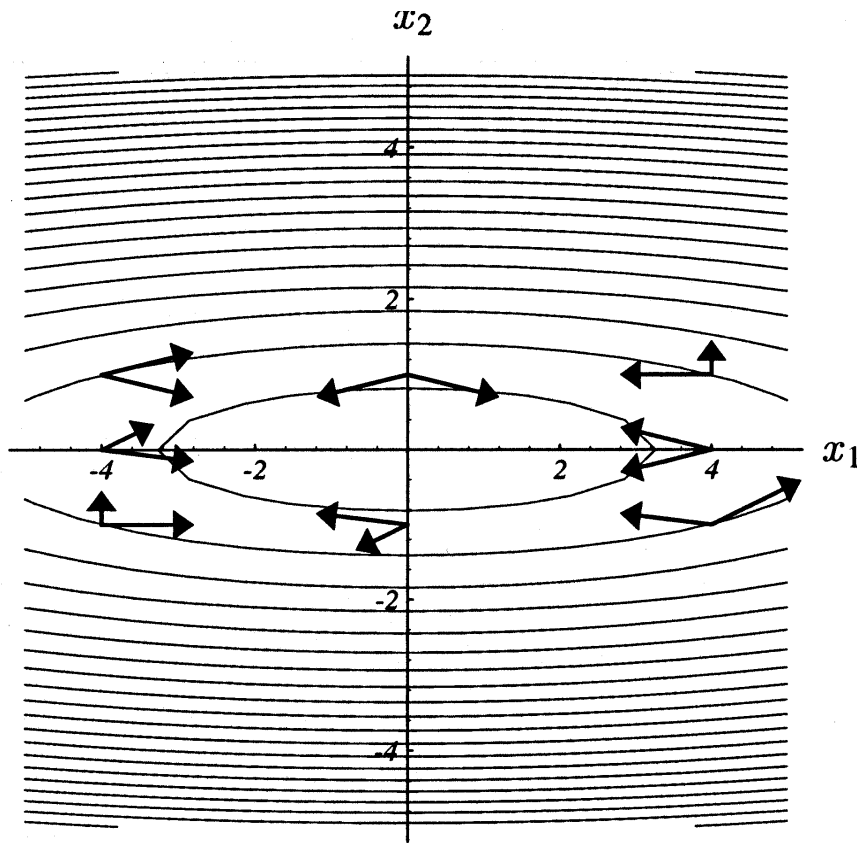
$$\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \alpha_{(i)} \mathbf{d}_{(i)}$$

2. To find $\alpha_{(i)}$, we use the fact that $\mathbf{e}_{(i+1)}$ is orthogonal to $\mathbf{d}_{(i)}$

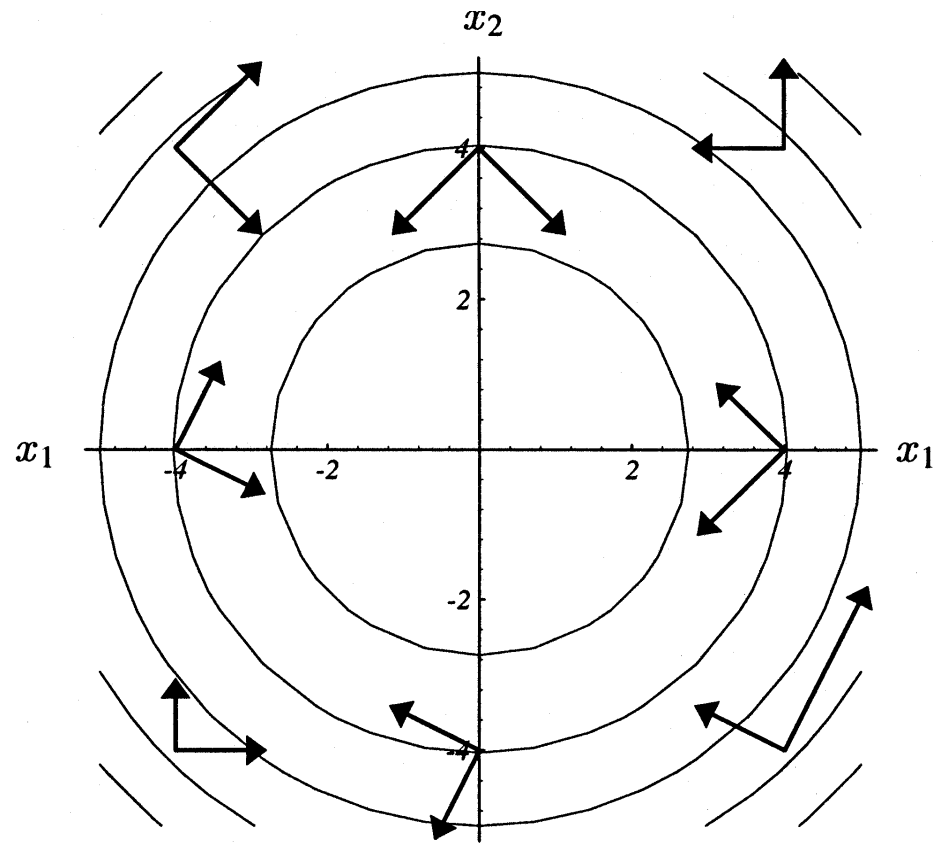
The Method of Conjugate Directions

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

$$\mathbf{d}_{(i)}^T \mathbf{A} \mathbf{d}_{(j)} = 0$$



(a)



(b)

The Method of Conjugate Directions

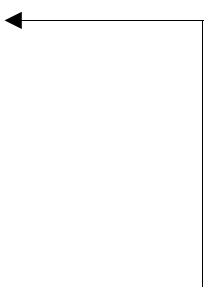
- The new requirement is now that $\mathbf{e}_{(i+1)}$ is A-orthogonal to $\mathbf{d}_{(i)}$

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

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

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

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

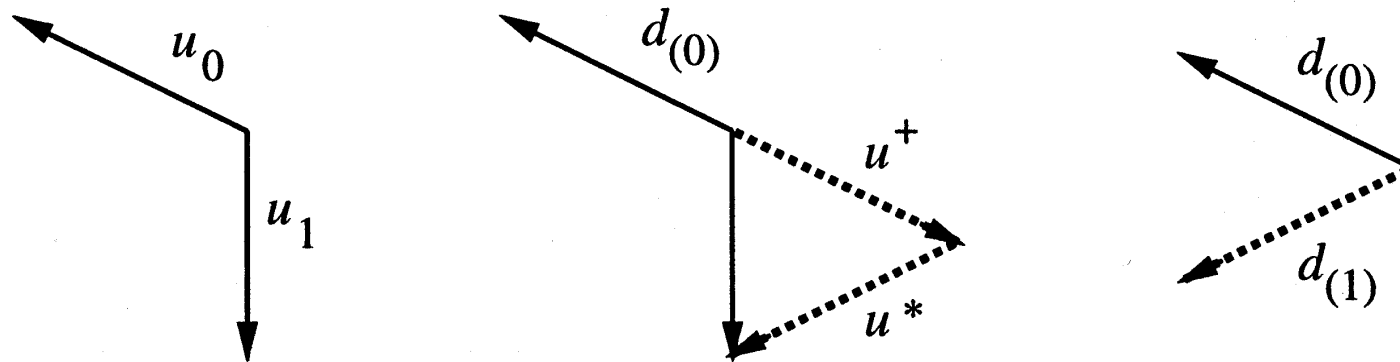
$$\alpha_{(i)} = \frac{\mathbf{d}_{(i)}^T \mathbf{r}_{(i)}}{\mathbf{d}_{(i)}^T \mathbf{A} \mathbf{d}_{(i)}}$$


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 $\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{n-1}$
 2. Assume that $\mathbf{d}_{(0)} = \mathbf{u}_0$
 3. For $i > 0$, take an \mathbf{u}_i and subtracts all the components from it that are not A-orthogonal to the previous search directions

$$\mathbf{d}_{(i)} = \mathbf{u}_{(i)} + \sum_{j=0}^{i-1} \beta_{ij} \mathbf{d}_{(j)}, \quad \beta_{ij} = -\frac{\mathbf{u}_{(i)}^T \mathbf{A} \mathbf{d}_{(j)}}{\mathbf{d}_{(j)}^T \mathbf{A} \mathbf{d}_{(j)}}$$



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. $\mathbf{u}_i = \mathbf{r}_{(i)}$
- This allows us to simplify the calculation of the new search direction because

$$\beta_{ij} = \begin{cases} \frac{1}{\alpha_{(i-1)}} \frac{\mathbf{r}_{(i)}^T \mathbf{r}_{(i)}}{\mathbf{d}_{(i-1)}^T \mathbf{A} \mathbf{d}_{(i-1)}} = \frac{\mathbf{r}_{(i)}^T \mathbf{r}_{(i)}}{\mathbf{r}_{(i-1)}^T \mathbf{r}_{(i-1)}} & i = j + 1 \\ 0 & i > j + 1 \end{cases}$$

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

$$\mathbf{d}_{(i+1)} = \mathbf{r}_{(i+1)} + \beta_i \mathbf{d}_{(i)}$$

The Method of Conjugate Directions

$$\mathbf{x}_0 = \mathbf{0}, \quad \mathbf{r}_0 = \mathbf{b}, \quad \mathbf{d}_0 = \mathbf{r}_0$$

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

$$\alpha_k = (\mathbf{r}_{k-1}^T \mathbf{r}_{k-1}) / (\mathbf{d}_{k-1}^T \mathbf{A} \mathbf{d}_{k-1}) \quad \text{step length}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_k \mathbf{d}_{k-1} \quad \text{approx solution}$$

$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k \mathbf{A} \mathbf{d}_{k-1} \quad \text{residual}$$

$$\beta_k = (\mathbf{r}_k^T \mathbf{r}_k) / (\mathbf{r}_{k-1}^T \mathbf{r}_{k-1}) \quad \text{improvement}$$

$$\mathbf{d}_k = \mathbf{r}_k + \beta_k \mathbf{d}_{k-1} \quad \text{search direction}$$

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

Krylov subspace

Krylov subspace \mathbf{K}_j is the linear combinations of $b, Ab, \dots, A^{j-1}b$.

Krylov matrix $\mathbf{K}_j = [b \quad Ab \quad A^2b \quad \dots \quad A^{j-1}b]$.

Methods to construct a basis for \mathbf{K}_j :

Arnoldi's method and Lanczos method

Approaches to choosing a good \mathbf{x}_j in \mathbf{K}_j :

- **Ritz-Galerkin approach:** $r_j = b - Ax_j$ is orthogonal to \mathbf{K}_j
(Conjugate Gradient)
- **Minimum Residual approach** r_j has minimum norm for x_j in \mathbf{K}_j
(GMRES and MINRES)
- **Petrov-Galerkin approach:** r_j is orthogonal to a different space $\mathbf{K}_j(\mathbf{A}^T)$ (Biconjugate Gradient)

Arnoldi's Method

The best basis q_1, \dots, 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, \dots, q_j that are already chosen. The iteration to compute these orthonormal q 's is Arnoldi's method.

```
q1 = b / ||b||           % Normalize b to ||q1|| = 1
for j = 1, ..., n-1       % Start computation of qj+1
    t = Aqj              % one matrix multiplication
    for i = 1, ..., j      % t is in the space Kj+1
        hij = qiT t      % hij qiT = projection of t on qi
        t = t - hij qi    % Subtract that projection
    end;                  % t is orthogonal to q1, ..., qj
    hj+1,j = ||t||        % Compute the length of t
    qj+1 = t / hj+1,j    % Normalize t to ||qj+1|| = 1
end                       % q1, ..., qn are orthonormal
```

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

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

Lanczos Method

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

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

$H_{n-1,n-1}$ 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 $i = 1, \dots, n-1$

$$v = Aq_j$$

$$a_i = q_i^T v$$

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

$$B_i = \|v\|$$

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

end

Minimum Residual Methods

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

Solution: Minimum Residual Methods.

Choose \mathbf{x}_j in the Krylov subspace \mathbf{K}_j so that $\|\mathbf{b} - \mathbf{Ax}_j\|$ is minimal

The first orthonormal vectors q_1, \dots, q_j go in the columns \mathbf{Q}_j so $\mathbf{Q}_j^T \mathbf{Q}_j = \mathbf{I}$

Setting $\mathbf{x}_j = \mathbf{Q}_j \mathbf{y}$

$$\|r_j\| = \|\mathbf{b} - \mathbf{Ax}_j\| = \|\mathbf{b} - \mathbf{AQ}_j \mathbf{y}\| = \|\mathbf{b} - \mathbf{Q}_{j+1} \mathbf{H}_{j+1,j} \mathbf{y}\|$$

Using first j columns of Arnoldi's formula $\mathbf{AQ} = \mathbf{QH}$

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

Minimum Residual Methods

The problem becomes:

Choose \mathbf{y} to minimize

$$\|r_j\| = \|Q_{j+1}^T \mathbf{b} - H_{j+1,j} \mathbf{y}\|$$

This is least squares problem.

Using zeros in \mathbf{H} and $Q_{j+1}^T \mathbf{b}$ to find a fast algorithm that computes \mathbf{y} .

GMRES (Generalised Minimal Residual Approach)

\mathbf{A} is *not symmetric* and the upper triangular part of \mathbf{H} can be full.

All previously computed vectors have to be stored.

MINRES: (Minimal Residual Approach)

\mathbf{A} is *symmetric* (likely indefinite) and \mathbf{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 \mathbf{H} .

This is done by *Givens rotations*

GMRES

Algorithm: GMRES

$$q_1 = b / \|b\|$$

for $j = 1, 2, 3, \dots$

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 $\mathbf{K}_j(\mathbf{A}^T)$
- **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 $\beta_1 = \delta_1 \equiv 0, w_0 = v_0 \equiv 0$

For $j = 1, 2, \dots, m$ 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 $j = 0, 1, \dots$, **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.

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

For $m = 1, 2, \dots$, **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:**

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

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

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

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

EndDo

Conjugate Gradient Squared (CGS)

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

Set $p_0 := u_0 := r_0$.

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

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

$$q_j = u_j - \alpha_j Ap_j$$

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

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

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

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

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

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 !