THE UNIVERSITY OF CHICAGO

Section 5

Conjugated Gradients

Mihai Anitescu

5.1 Matrix Free Methods for Parallel Computations

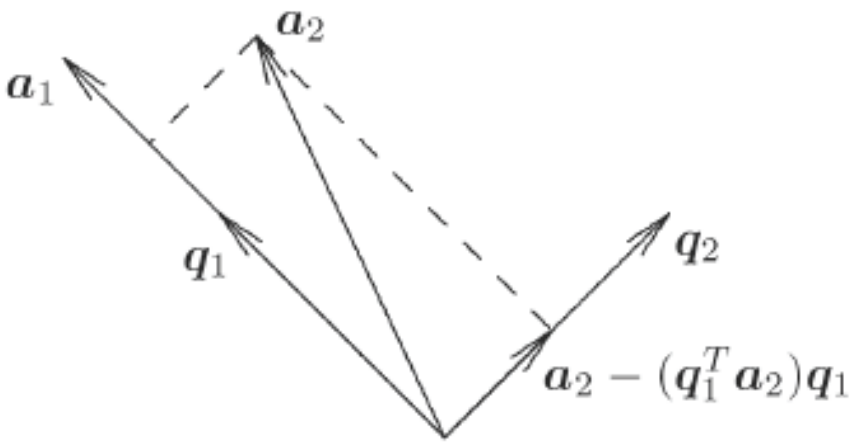
- Many problems have simple, sparse matrices associated with the linear algebra (e.g., example from Homework 2).
- Nevertheless, factorization is extremely expensive on parallel architectures due to processsor-to-processor data motion needs of pivoting, (as well as storage needs).
- Solution? Use methods (e.g., Krylov space or relaxation methods) that only multiply the matrix A times a vector x, code to calculate y=Ax can be written instead of storing the matrix A.
- This reduces the cost of the computer (which is mostly memory chips) and allows for vastly larger simulations.
- It also results in very fast code, if BLAS is optimized, which is a much easier proposition than optimizing factorization.

5.2 Conjugated Gradients-

- "The Conjugate Gradient Method is the most prominent iterative method for solving sparse systems of linear equations. Unfortunately, many textbook treatments of the topic are written with neither illustrations nor intuition, and their victims can be found to this day babbling senselessly in the corners of dusty libraries. For this reason, a deep, geometric understanding of the method has been reserved for the elite brilliant few who have painstakingly decoded the mumblings of their forebears." - Schewchuck: "*An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*".

Gram-Schmidt orthogonalization

- Given vectors a_1 and a_2, we seek orthonormal vectors q_1 and q_2 having same span
- This can be accomplished by subtracting from second vector its projection onto first vector and normalizing both resulting vectors, as shown in diagram



Gram-Schmidt algorithm

- Process can be extended to any number of vectors $a_1, \ldots, a_k$, orthogonalizing each successive vector against all preceding ones, giving *classical Gram-Schmidt* procedure

for $k = 1$ **to** n
  $q_k = a_k$
  for $j = 1$ **to** $k - 1$
    $r_{jk} = q_j^T a_k$
    $q_k = q_k - r_{jk} q_j$
  end
  $r_{kk} = \|q_k\|_2$
  $q_k = q_k / r_{kk}$
end

- Resulting q_k and r_{jk} form reduced QR factorization of A

Conjugated Search Directions

- *Conjugated=orthogonal*
- Positive definite A; linear system is equivalent to minimization.

$$A \succ 0 \Rightarrow Ax = b \Leftrightarrow \min_{x \in \mathbb{R}^n} \phi(x) = \frac{1}{2} x^T A x = b^T x$$

- IDEA 1: A-conjugate directions:

$$S_k = \{p_0, p_1, \ldots, p_{k-1}\} \quad p_i^T A p_j = 0 \quad 0 \le i \ne j \le k-1$$

$$x_k = \min_{x \in x_0 + S_k} \phi(x); \quad \phi(x) = \frac{1}{2} x^T A x - b^T x$$

Arithmetic of A-Conjugated Search Directions

- A-conjugated direction search methods: the subspace problems are much simpler:

$$S_k = \{p_0, p_1, \ldots, p_{k-1}\} \quad p_i^T A p_j = 0 \quad 0 \le i \ne j \le k-1$$

$$\Leftrightarrow \min_{\{\alpha_i\}_{i=0}^{k-1}} \frac{1}{2}\left(\sum_{i=0}^{k-1}\alpha_i p_i + x_0\right)^T A\left(\sum_{i=0}^{k-1}\alpha_i p_i + x_0\right) - b^T\left(\sum_{i=0}^{k-1}\alpha_i p_i + x_0\right) =$$

$$\min_{\{\alpha_i\}_{i=0}^{k-1}} \frac{1}{2}\left(\sum_{i=0}^{k-1}\alpha_i p_i\right)^T A\left(\sum_{i=0}^{k-1}\alpha_i p_i\right) + \left(Ax_0 - b\right)^T\left(\sum_{i=0}^{k-1}\alpha_i p_i\right) \overset{A-\text{Conjugated Directions}}{=}$$

$$\min_{\{\alpha_i\}_{i=0}^{k-1}} \sum_{i=0}^{k-1}\frac{1}{2}\alpha_i^2 p_i^T A p_i + \sum_{i=0}^{k-1}\alpha_i r_0^T p_i = \sum_{i=0}^{k-1}\left[\frac{1}{2}\alpha_i^2 p_i^T A p_i + \alpha_i r_0^T p_i\right];$$

$$\text{where } r_0 = \left(Ax_0 - b\right) = \nabla_x \phi\left(x_0\right)$$

Arithmetic of A-Conjugated Search Directions-Consequences

The minimum can be computed incrementally

$$\min_{\{\alpha_i\}_{i=0}^{k-1}} \sum_{i=0}^{k-1} \left[\frac{1}{2}\alpha_i^2 p_i^T A p_i + \alpha_i r_0^T p_i \right]; \quad \Rightarrow \alpha_i = -\frac{r_0^T p_i}{p_i^T A p_i}; \Rightarrow$$

$$x_k = x_0 + \sum_{i=0}^{k-1} \left[-\frac{r_0^T p_i}{p_i^T A p_i} p_i \right] = x_{k-1} + \alpha_{k-1} p_{k-1} (C1)$$

$$\nabla\phi(x_k) = Ax_k - b = r_k = r_{k-1} + \alpha_{k-1} A p_{k-1} (C2)$$

$$r_k^T p_i = \nabla\phi(x_k) p_i = \left. \frac{\partial\phi}{\partial\alpha_i} \right|_{x=x_k} = 0; \quad 0 = 1, 2, \ldots k-1 (C3)$$

Optimality conditions for k-th incremental problem

$$\phi(x_k) = \min_\alpha \phi(x_{k-1} + \alpha p_{k-1}) \quad (C4)$$

Conjugated Directions Methods

- Are truly minimization- line search methods but their quadratic nature makes the problem "simple".
- The recursion in the variable is very simple.
- The residuals (the gradients) are orthogonal to the search direction space.
- *But, how do I find the A-conjugate search directions?*
- IDEA 2: The space of the search directions should coincide with the space of the residuals.
- In principle, achievable with Gram-Schmidt using the A inner product, but maybe it is simpler ….

Obtaining the A-conjugated Search Directions

- IDEA 2: The space of the search directions should coincide with the space of the residuals.

$$S_k = \{p_0, p_1, \ldots, p_{k-1}\} = \{r_0, r_1, \ldots, r_{k-1}\}$$

- Consequence K1: the gradients (residuals themselves) are orthogonal, that is they are *conjugated*.

$$r_k \perp S_k (C3) \Rightarrow r_k^T r_i = 0 (K1); \quad 0 \leq i \neq k$$

- This fact gives the methods its name: The method of conjugated gradients.

Obtaining the A-conjugated Search Directions

- Consequence K2: the search space = the gradient space from a Krylov space (by induction)

$$S_k = \{p_0, p_1, \ldots, p_{k-1}\} = \{r_0, r_1, \ldots, r_{k-1}\}$$

$$r_k = r_{k-1} + \alpha_{k-1} A p_{k-1} (C2) \Rightarrow S_{k+1} \subset S_k + AS_k$$

By induction and (K1) $S_k = \{r_0, Ar_0, \ldots, A^{k-1} r_0\} = K_k(A, r_0) \quad (K2)$

- Consequence K3 (the subtlest): computation of the next search direction needs only two terms !!

$$(K_1) \Rightarrow r_k \perp S_k \overset{(K2)}{=} K_k, \text{ BUT from (K1): } AS_{k-1} \subset S_k \Rightarrow \boxed{r_k \perp AS_{k-1}}$$

$$\text{Idea 1} \Rightarrow p_{k-1} \perp AS_{k-1} \Rightarrow d_k := -r_k + \beta_k p_{k-1}; \boxed{d_k \perp AS_{k-1} \forall \beta_k}$$

β_k so $d_k \perp Ap_{k-1}$ gets $p_k = d_k$ (since A-orthogonal on p_i, $0 \leq i \leq k-1$)!

$$p_{k-1}^T A d_k = 0 \rightarrow \beta_k = \frac{p_{k-1}^T A r_k}{p_{k-1}^T A p_{k-1}}; \; p_k = -r_k + \beta_k p_{k-1}$$

METHOD OF CONJUGATED GRADIENTS VERSION 1

Algorithm 5.1 (CG–Preliminary Version).

Given x_0;

Set $r_0 \leftarrow Ax_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$;

while $r_k \neq 0$

$$\alpha_k \leftarrow -\frac{r_k^T p_k}{p_k^T A p_k};$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$

$$r_{k+1} \leftarrow Ax_{k+1} - b;$$

$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T A p_k}{p_k^T A p_k};$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k;$$

$$k \leftarrow k + 1;$$

end (while)

Simplifications

$$\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}.$$

$$\beta_{k+1} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}.$$

CG: PRACTICAL VERSION (MINIMAL STORAGE)

Algorithm 5.2 (CG).

Given x_0;

Set $r_0 \leftarrow Ax_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$;

while $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T A p_k};$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$

$$r_{k+1} \leftarrow r_k + \alpha_k A p_k;$$

$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k};$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k;$$

$$k \leftarrow k + 1;$$

end (while)

NEEDS ONLY 1 MATRIX-VECTOR MULTIPLICATION PER STEP.
AX NEVER FORMED AS BEFORE.

5.3 CG convergence

- Iteration in Krylov Space

$$x_{k+1} = x_0 + \alpha_0 p_0 + \cdots + \alpha_k p_k$$
$$= x_0 + \gamma_0 r_0 + \gamma_1 A r_0 + \cdots + \gamma_k A^k r_0,$$

- Matrix Polynomial

$$P_k^*(A) = \gamma_0 I + \gamma_1 A + \cdots + \gamma_k A^k,$$

- Iteration as a matrix Polynomial

$$x_{k+1} = x_0 + P_k^*(A) r_0.$$

- Fun problem: Solve Ax=b using Cayley-Hamilton

Error in A-space

- Error in A-norm (recall also the theorem on steepest descent rate of convergence)

$$\|z\|_A^2 = z^T A z. \qquad \tfrac{1}{2}\|x - x^*\|_A^2 = \tfrac{1}{2}(x - x^*)^T A(x - x^*) = \phi(x) - \phi(x^*).$$

- So what is the conjugate gradient method computing?

- Rewrite slightly the minimization problem as a function of a polynomial

$$x_{k+1} = x_0 + P_k^*(A) r_0. \qquad \min_{P_k} \|x_0 + P_k(A) r_0 - x^*\|_A.$$

$$x_{k+1} - x^* = x_0 + P_k^*(A) r_0 - x^* = [I + P_k^*(A) A](x_0 - x^*).$$

The calculation in eigenvalue space

Let $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of A, and let $v_1, v_2, \ldots, v_n$ be the corresponding orthonormal eigenvectors, so that

$$A = \sum_{i=1}^{n} \lambda_i v_i v_i^T.$$

$$\|x_{k+1} - x^*\|_A^2 = \min_{P_k} \sum_{i=1}^{n} \lambda_i [1 + \lambda_i P_k(\lambda_i)]^2 \xi_i^2.$$

$$\begin{aligned} \|x_{k+1} - x^*\|_A^2 &\leq \min_{P_k} \max_{1 \leq i \leq n} [1 + \lambda_i P_k(\lambda_i)]^2 \left(\sum_{j=1}^{n} \lambda_j \xi_j^2 \right) \\ &= \min_{P_k} \max_{1 \leq i \leq n} [1 + \lambda_i P_k(\lambda_i)]^2 \|x_0 - x^*\|_A^2, \end{aligned}$$

Consequences for Convergence

- Linear Convergence Rate Estimate: $\min_{P_k} \max_{1 \le i \le n} [1 + \lambda_i P_k(\lambda_i)]^2.$
- Consequences:

Theorem 5.4.

If A has only r distinct eigenvalues, then the CG iteration will terminate at the solution in at most r iterations.

Theorem 5.5.

If A has eigenvalues $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$, *we have that*

$$\|x_{k+1} - x^*\|_A^2 \le \left(\frac{\lambda_{n-k} - \lambda_1}{\lambda_{n-k} + \lambda_1} \right)^2 \|x_0 - x^*\|_A^2.$$

- Note: finite termination in n steps.

5.4 Preconditioning

- Rescaling of the problem

$$\hat{x} = Cx$$

- The modified objective function

$$\hat{\phi}(\hat{x}) = \frac{1}{2}\hat{x}^T \left(C^{-T} A C^{-1}\right) \hat{x} - (C^{-T} b)^T \hat{x}$$

- Equivalent linear system.

$$C^{-T} A C^{-1} \hat{x} = C^{-T} b$$

How to find a preconditioner?

- Idea (from Theorem 5.5). Compute a C such that the eigenvalues are "clustered", then convergence is fast. For example

$$C^{-T}AC \approx I \quad \text{or } C \approx L^T; A = LL^T$$

- Preconditioners must be easy to factorize or invert.
- Example preconditioners:
  - Incomplete Cholesky (use sparsity pattern of A)
  - Symmetric Successive overrelaxation
  - Multigrid (Order (1) for PDEs)
- The Holy Grail (of iterative/preconditioned methods): Condition number is O(1).

Preconditioned conjugate gradient

Algorithm 5.3 (Preconditioned CG).

Given x_0, preconditioner M;

Set $r_0 \leftarrow Ax_0 - b$;

Solve $My_0 = r_0$ for y_0;

Set $p_0 = -y_0$, $k \leftarrow 0$;

while $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^T y_k}{p_k^T A p_k};$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$

$$r_{k+1} \leftarrow r_k + \alpha_k A p_k;$$

Solve $My_{k+1} = r_{k+1}$;

$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T y_{k+1}}{r_k^T y_k};$$

$$p_{k+1} \leftarrow -y_{k+1} + \beta_{k+1} p_k;$$

$$k \leftarrow k + 1;$$

end (while)

Preconditioner action

$$M = C^T C$$