Lecture 17: The Conjugate Gradient Method

COMP5930M Scientific Computation

Today

Outline

Gradient descent

Conjugate directions

Conjugate gradient method

Examples

Next

Reference

An Introduction to the Conjugate Gradient Method, Without the Agonizing Pain

Jonathan Shewchuk

http://www.cs.cmu.edu/∼jrs/jrspapers.html#cg

- A well-written, intuitive description
- This lecture (mostly) follows this paper

Outline

We can develop the algorithm as a series of enhancements, starting from a form of the Gradient Descent algorithm.

- The Gradient-Descent Algorithm
- The Conjugate Directions Algorithm
- The Conjugate Gradient Method

1. The problem

We want to compute the n-dimensional vector $\mathbf{x}$ satisfying $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is a large, sparse matrix

We will assume $\mathbf{A}$ is symmetric and *positive-definite* (SPD)

- SPD means that for any vector $\mathbf{p} \neq \mathbf{0}$, scalar $s = \mathbf{p}^T\mathbf{Ap} > 0$

Recall we can also define a set of n eigenvalues λ_i and eigenvectors $\mathbf{e}_i$ for symmetric $\mathbf{A}$

- For SPD matrices $\lambda_i > 0$ for all i

The quadratic form

Consider a related problem:

- Find $\mathbf{x}$ that minimises the scalar function $f(\mathbf{x})$

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

with some scalar constant c

- At a minimum of f we have the necessary condition

$$\frac{\partial f}{\partial \mathbf{x}} = \mathbf{A}\mathbf{x} - \mathbf{b} = 0$$

hence the minimum point $\mathbf{x}$ also solves $\mathbf{A}\mathbf{x} = \mathbf{b}$

Minimising $f(\mathbf{x})$

- Conjugate Gradient (CG) and related algorithms were designed as minimisation algorithms
- SPD matrices guarantee the function $f(\mathbf{x})$ is *strictly convex* and hence that the point where the gradient is zero is a minimum

Eigenvalues/vectors

- They are the primary *analysis* tool for these algorithms but we do not need to compute them (general eigendecomposition would require $\mathcal{O}(N^3)$)
- The shape of the space we search is related to the eigenvalues and eigenvectors: $\mathbf{A}\mathbf{u}_i = \lambda_i \mathbf{u}_i$
- For each pair λ_i, $\mathbf{u}_i$ the search space is stretched
  - by a factor $1/\lambda_i$
  - in the direction $\mathbf{u}_i$

History

- CG was designed in the 1960s as a *direct* solution algorithm
  - Formally it will terminate in n steps at the exact solution
  - It was discarded as inefficient compared to standard direct algorithms
- In the 1970s it was reinvented as an iterative algorithm
  - Due to the search process it will often be **close** to a solution in less than n steps

The residual

- Recall we defined the residual $\mathbf{r}_i$ for a given approximate solution $\mathbf{x}_i$

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i$$

- $\mathbf{r}_i$ defines a local search direction of steepest-descent. Why?

$$f\mathbf{x}(\mathbf{x}_i) = \mathbf{A}\mathbf{x}_i - \mathbf{b} = -\mathbf{r}_i$$

2. The Gradient-Descent Algorithm

Define an update

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{r}_i$$

for some scalar step length α_i

- We should compute the optimal distance α_i for that search direction
- This is the line-search problem, (considered in Lecture 6, but now for a linear system)
- We can exactly derive (see Ref p6)

$$\alpha_i = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_i^T \mathbf{A} \mathbf{r}_i}$$

Orthogonality 正交

- If we minimise the gradient in one direction, $\mathbf{r}_i$, the next step of steepest-descent must be orthogonal
- This is equivalent to observing that

$$\mathbf{r}_i^T \mathbf{r}_{i+1} = 0$$

- A series of steps defines a series of searches in consecutively orthogonal directions
- We cannot guarantee that all directions $\mathbf{r}_i$ will be orthogonal to all other search directions $\mathbf{r}_j \perp \mathbf{r}_i$ for all $i \neq j$.

3. Conjugate Directions Algorithm

Define an update

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$$

for some scalar step length α_i

- An improved search algorithm that guarantees the set of search directions, $\mathbf{d}_i$, are mutually orthogonal, for any $i \neq j$

$$\mathbf{d}_i^T \mathbf{d}_j = 0$$

- For example, in Cartesian space try $\mathbf{d}_i = \{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z, ...\}$ (called cyclic coordinate search)
  - Any point in space can be located by successively searching each coordinate direction

In practice?

To have orthogonal search directions $\mathbf{d}_i$ we must require that at each step the error has to be orthogonal to the search direction, $\mathbf{d}_i^T \mathbf{e}_{i+1}$.

- We can derive the optimal step size α_i as

$$\alpha_i = -\frac{\mathbf{d}_i^T \mathbf{e}_i}{\mathbf{d}_i^T \mathbf{d}_i}$$

  for any orthogonal set $\mathbf{d}_i$

- This requires the error $\mathbf{e}_i = \mathbf{x} - \mathbf{x}_i$ and is not computable

A-orthogonality

- Note: when **A** is symmetric positive-definite, we can define the **A**-norm of a vector using the quadratic form

$$\|\mathbf{x}\|_A := \sqrt{\mathbf{x}^T \mathbf{A}\mathbf{x}}.$$

- **A**-orthogonality of directions $\mathbf{d}_i$ is defined by

$$\mathbf{d}_i^T \mathbf{A}\mathbf{d}_j = 0$$

for any $i \neq j$

- For this set we can derive a computable form

$$\alpha_i = \frac{\mathbf{d}_i^T \mathbf{A}\mathbf{e}_i}{\mathbf{d}_i^T \mathbf{A}\mathbf{d}_i} = \frac{\mathbf{d}_i^T \mathbf{r}_i}{\mathbf{d}_i^T \mathbf{A}\mathbf{d}_i}$$

- Note, that choosing $\mathbf{d}_i = \mathbf{r}_i$ would produce a modified gradient-descent algorithm

Computing $\mathbf{d}_i$

At step i, we use the residual $\mathbf{r}_i$ as a search direction, but enforce **A**-orthogonality by subtracting the previous search directions :

$$\mathbf{d}_i = \mathbf{r}_i + \sum_{k=0}^{i-1} \beta_{ik}\mathbf{d}_k = \mathbf{r}_i - \sum_{k=0}^{i-1} \mathrm{proj}_{\mathbf{d}_k}(\mathbf{r}_i)$$

where $\mathrm{proj}_{\mathbf{d}_k}(\mathbf{r}_i)$ is the projection of $\mathbf{r}_i$ to the direction $\mathbf{d}_k$ given by

$$\beta_{ik} = -\frac{\mathbf{r}_i^T \mathbf{A}\mathbf{d}_k}{\mathbf{d}_k^T \mathbf{A}\mathbf{d}_k}$$

- Although appearing complex we are simply subtracting out components not **A**-orthogonal to a previous search direction

Note

- This algorithm is generally called Gram-Schmidt
  - In general it has $\mathcal{O}(n^2)$ expense, as the work increases with i
- The set of search directions at each step i is called a Krylov subspace for A

4. The Conjugate Gradient Method

- The choice of basing the search direction $\mathbf{d}_i$ on the residual $\mathbf{r}_i$ allows a significant amount of algebraic simplification
- We can show
  - The residual $\mathbf{r}_i$ is $\mathbf{A}$-orthogonal to every previous search direction $\mathbf{d}_j$
  - $\beta_{ik} = 0$ unless $i = k + 1$
- The expense of CG is then $\mathcal{O}(nz)$

The CG algorithm

- $\mathbf{d}_0 = \mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
- $i = 0, 1, 2, \ldots$
  - $\alpha = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{d}_i^T \mathbf{A}\mathbf{d}_i}$
  - $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha \mathbf{d}_i$
  - $\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha \mathbf{A}\mathbf{d}_i$
  - $\beta = \frac{\mathbf{r}_{i+1}^T \mathbf{r}_{i+1}}{\mathbf{r}_i^T \mathbf{r}_i}$
  - $\mathbf{d}_{i+1} = \mathbf{r}_{i+1} + \beta \mathbf{d}_i$

Note: Only one matrix-multiply required per iteration

Efficiency

- For efficiency we require the approximate solution in a minimum number of iterations
  - we can relate the convergence to the distribution of eigenvalues of the matrix
  - the specific relationship depends on the iterative scheme
- The reference paper is more comprehensive and outlines the mathematical derivations
- The CG algorithm is often described more formally without *physical* insight

Examples

- Matlab code `runPCG.m` available on VLE
- Uses Matlab library `pcg.m` function
- Matrix generated with the Matlab `gallery()` function

Improving the Conjugate Gradient algorithm

- General convergence result for CG:

$$\|\mathbf{e}_k\|_A \leq 2\left(\frac{\sqrt{\kappa(\mathbf{A})}-1}{\sqrt{\kappa(\mathbf{A})}+1}\right)^k \|\mathbf{e}_0\|_A.$$

- We can improve the convergence,
  ie. reduce the number of iterations required,
  through preconditioning
  - We solve $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$
  - The matrix $\mathbf{M}^{-1}\mathbf{A}$ has an improved eigenvalue distribution
  - This implies a reduced Condition Number κ

Summary

- Conjugate gradient (CG) method efficient iterative solver for symmetric, positive-definite matrices
- Method based on construction of search directions $\mathbf{d_i}$ that are **A**-orthogonal (belong in ~~Krylov subspace~~), followed by descent method with exact line-search
- For non-symmetric problems we have other Krylov-methods:
  - Generalised minimal residual -method (GMRES)
  - Biconjugate gradient stabilised -method (BiCGSTAB)
- Number of iterations depends on condition number $\kappa(\mathbf{A})$