Conjugate Gradient Method

Gabriel Ingesson

November 5, 2015

The Conjugate Gradient Method

- Solves large linear systems of equations.
  - Proposed by Hestenes and Stiefel in the 1950's as an alternative to Gaussian elimination for large problems with positive definite coefficient matrices.
- Adapted to solve nonlinear optimization problems.
  - Nonlinear conjugate gradient was introduced by Fletcher and Reeves in the 1960's. One of the first methods for solving large scale nonlinear optimization problems.
- Requires no matrix storage and are faster than the steepest descent method.

The Linear Conjugate Gradient Method

The (linear) CG is an iterative method for solving a linear system of equations

$$Ax = b \tag{1}$$

where A is a symmetric, positive definite matrix.

This is equivalent of solving the optimization problem

$$\min \ \phi(x) \equiv \frac{1}{2}x^T Ax - b^T x \tag{2}$$

$$\nabla\phi(x) = Ax - b \equiv r(x) \tag{3}$$

Problem 1 and 2 have the same unique solution.

Conjugacy

The CG method generates a set of conjugate vectors p_i w.r.t. A, these vectors are the step directions when minimizing $\phi(x)$.

Definition

Two vectors v and u are conjugate w.r.t. A if $u^T Av = 0$.

$u^T Av$ is an inner product since $A > 0$, n vectors fulfilling this property are linearly independent and are a base in $\mathbb{R}^n$.

Conjugate directions of A

Given any starting point x_0 and a set of conjugate directions $\{p_0, p_1, \ldots, p_{n-1}\}$ w.r.t. A. The sequence $\{x_k\}$ generated by

$$x_{k+1} = x_k + \alpha_k p_k$$

where α_k is the minimizer of ϕ along $x_k + \alpha_k p_k$, given by

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

will minimize ϕ in at most n steps.

Conjugate directions theorem.

Theorem

For any x_0, the sequence x_k generated by a set of conjugate directions $\{p_0, p_1, \ldots, p_{n-1}\}$ and the minimizing α_k will converge to the solution x^* in at most n steps.

Proof.

$x^* - x_0 = \sum_{i=0}^{n-1} \sigma_k p_k$ since p_k span $\mathbb{R}^n$, multiplying this expression by $p_k^T A$ from the left gives $\sigma_k = \frac{p_k^T A(x^* - x_0)}{p_k^T A p_k}$ which gives $\sigma_k = \alpha_k$ since $p_k^T A(x^* - x_0) = -p_k^T r_k$. $\square$

Geometrical Interpretation

If A is diagonal, ϕ has its level-curve ellipse axes aligned with the coordinate directions.

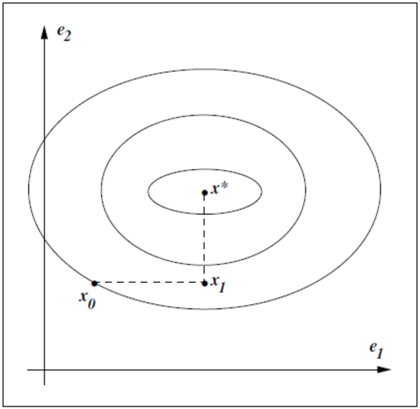


Figure 5.1 Successive minimizations along the coordinate directions find the minimizer of a quadratic with a diagonal Hessian in n iterations.

Geometrical Interpretation

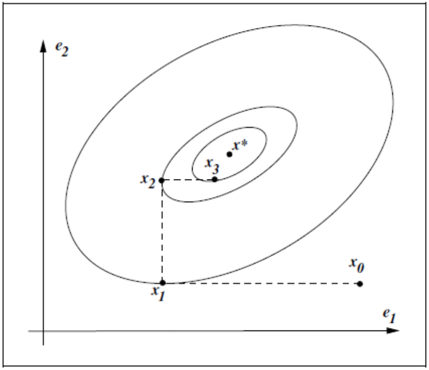


Figure 5.2 Successive minimization along coordinate axes does not find the solution in n iterations, for a general convex quadratic.

Geometrical Interpretation

Diagonalize A and minimize along the new coordinate directions

$$\hat{x} = S^{-1}x$$

where

$$S = \begin{bmatrix} p_0 & \dots & p_{n-1} \end{bmatrix},$$

$S^T AS$ is diagonal by the conjugacy property.

Expanding Subspace Minimization

The residual is orthogonal to the previous search directions.

Theorem

For any x_0, the sequence x_k generated by a set of conjugate directions method fulfils the properties

$$r_k^T p_i = 0, \text{ for } i = 0, 1, \ldots, k-1$$

and x_k is the minimizer of ϕ over the set

$$x0 + span\{p_0, p_1, \ldots, p_{k-1}\}$$

Conjugate Directions

How to generate the conjugate directions

- Take the eigenvectors v_k of A.
- Modified Gram-Schmidt orthogonalization.

These procedures are computationally heavy and requires storage.

The CG Method

The CG method computes p_k in an economical fashion that only requires the previous direction p_{k-1}. This requires little storage and computation. With the following update rule

$$p_k = -r_k + \beta_k p_{k-1}.$$

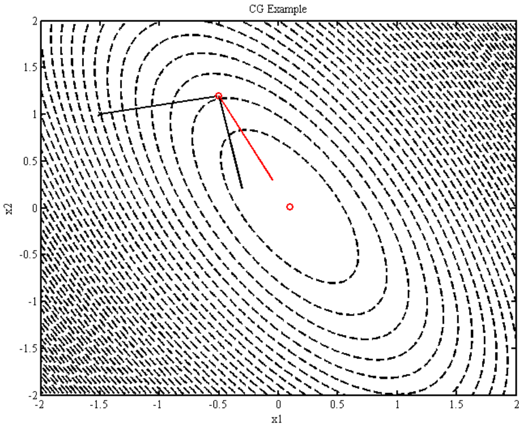
Conjugagy is fulfilled if

$$\beta_k = \frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}.$$

Algorithm 1 CG Algorithm

```
1:  $r_0 = Ax_0 - b$
2:  $p_0 = -r_0$
3:  $k = 0$
4:  while $r_{k+1} > \epsilon$ do
5:      $\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k}$
6:      $x_{k+1} = x_k + \alpha_k p_k$
7:      $r_{k+1} = Ax_{k+1} - b$
8:      $\beta_{k+1} = \frac{r_{k+1}^T A p_k}{p_k^T A p_k}$
9:      $p_{k+1} = -r_{k+1} + \beta_k p_k$
10:     $k = k + 1$
11: end while
```

Geometrical Interpretation



Theorem for Algorithm 1

Theorem

Suppose the iterate k is not the solution, then

- $r_k^T r_i = 0$ ***for*** $i = 0, 1, \ldots, k-1$
- $span\{r_0, \ldots, r_k\} = span\{r_0, \ldots, A^k r_0\}$
- $span\{p_0, \ldots, p_k\} = span\{r_0, \ldots, A^k r_0\}$
- $p_k^T A p_i = 0,$ *for* $i = 0, 1 \ldots, k-1$

Therefore the sequence $\{x_k\}$ converges in at most k steps.

Remark from the book:

"The conjugate gradient method should rather be called the conjugate (search) direction method since it is the search directions that are conjugate w.r.t. A and not the gradients."

The CG Method

Simplified updating formulas

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

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

gives a more economical CG algorithm which is also the standard implementation.

The major computational tasks is the vector matrix multiplication Ap_k, the inner products $p_k^T A p_k$, $r_{k+1}^T r_{k+1}$ and three vector sums.

Rate of Convergence

- We have seen that the method converges in at most n steps.
- If the eigenvalues of A are generously distributed, the method will converge in much fewer iterations.

Krylow Subspace

Definition

The order-r Krylov subspace generated by an n-by-n matrix A and a vector b of dimension n is the linear subspace spanned by the images of b under the first r powers of A (starting from $A^0 = I$), that is,

$$K_r(A, b) = \text{span}\{b, Ab, A^2 b, \dots, A^{r-1} b\}.$$

Rate of Convergence

From the previous results we have that

$$x_{k+1} = x_0 + \alpha_0 p_0 + \ldots + \alpha_k p_k$$

$$x_{k+1} = x_0 + \gamma_0 r_0 + \ldots + \gamma_k A^k r_k$$

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

where P is a k:th order polynomial.

Since x_{k+1} minimizes

$$\|x^* - x_{k+1}\|_A = \phi(x^*) - \phi(x_{k+1})$$

over the Krylow subspace

$$x_0 + \text{span}\{r_0, \ldots, A^k r_0\}$$

$P_k^*(A)$ solves

$$\min \|x_0 + P_k^*(A) r_0 - x^*\|_A$$

Rate of Convergence

From this formulation the following inequality can be derived

$$\|x_{k+1} - x^*\|_A^2 \leq \min \max (1 + \lambda_i P_k(\lambda_i)^2)^2 \|x_0 - x^*\|_A^2$$

Convergence Theorem I

The following theorem is then proven by the construction of P_k.

Theorem

If A has r distinct eigenvalues, then, the CG method will converge in at most r steps.

Convergence Theorem II

The following theorem is given without proof.

Theorem

If A has n eigenvalues $\lambda_1 < \ldots < \lambda_n$, then

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

This implies...

Convergence

That if we A have m large eigenvalues and $n-m$ small such that $\epsilon > \lambda_{n-m} - \lambda_1$, then

$$\|x_{m+1} - x^*\|_A \approx \epsilon \|x_0 - x^*\|_A.$$

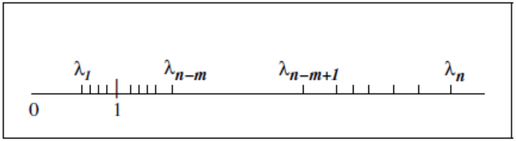


Figure 5.3 Two clusters of eigenvalues.

Convergence

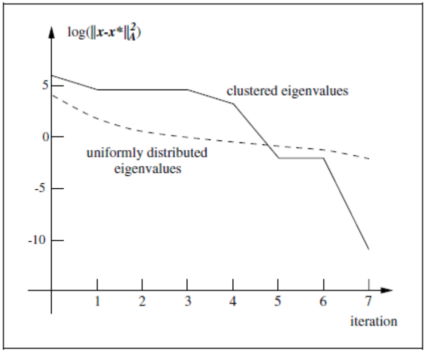


Figure 5.4 Performance of the conjugate gradient method on (a) a problem in which five of the eigenvalues are large and the remainder are clustered near 1, and (b) a matrix with uniformly distributed eigenvalues.

Convergence

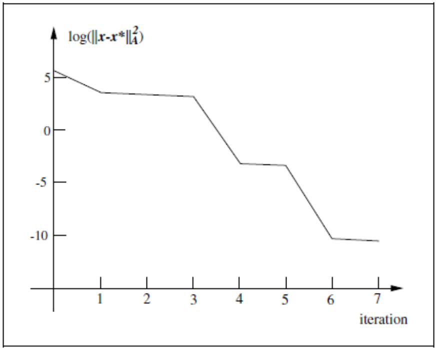


Figure 5.5 Performance of the conjugate gradient method on a matrix in which the eigenvalues occur in four distinct clusters.

Convergence

"It is generally true that if the eigenvalues occur in r distinct clusters, the CG iterates will approximately solve the problem in about r steps."

Another quite conservative bound on convergence is given by

$$\|x_k - x^*\|_A \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^2 \|x_0 - x^*\|_A.$$

Preconditioning

The convergence can be accelerated by the use of preconditioning, which is a rescaling of the variables on the form

$$\hat{x} = Cx$$

Now we instead solve the system

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

or equivalently minimize

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

The convergence now depend on the eigenvalues of $C^{-T}AC^{-1}$ rather than A.

Preconditioned CG

Given x_0 and M, $M = C^T C$.

-
-
- $p_0 = -y_0$
- $k = 0$
- while$(r_{k+1} > \epsilon)$
- $\alpha_k = -\frac{r_k^T y_k}{p_k^T A p_k}$
- $x_{k+1} = x_k + \alpha_k p_k$
- $r_{k+1} = r_k + \alpha_k A p_k$
- solve $M y_{k+1} = r_{k+1}$ for y_{k+1}
- $\beta_{k+1} = \frac{r_{k+1}^T y_{k+1}}{r_k^T y_k}$
- $p_{k+1} = -y_{k+1} + \beta_k p_k$
- $k = k + 1$
- end while

Algorithm 2 Preconditioned CG Algorithm

```
1:  $r_0 = Ax_0 - b$
2:  solve $My_0 = r_0$ for $y_0$
3:  $p_0 = -y_0$
4:  $k = 0$
5:  while $r_{k+1} > \epsilon$ do
6:      $\alpha_k = -\frac{r_k^T y_k}{p_k^T A p_k}$
7:      $x_{k+1} = x_k + \alpha_k p_k$
8:      $r_{k+1} = r_k + \alpha_k A p_k$
9:      solve $My_{k+1} = r_{k+1}$ for $y_{k+1}$
10:     $\beta_{k+1} = \frac{r_{k+1}^T y_{k+1}}{r_k^T y_k}$
11:     $p_{k+1} = -y_{k+1} + \beta_k p_k$
12:     $k = k + 1$
13: end while
```

Practical Preconditioners

The choice of M is a trade off between the effectiveness of M, storage and the computational cost of solving $My = r$.

A common choice is the incomplete cholesky factorization

$$A = LL^T$$

where an $\tilde{L}$ that is sparser than L is computed, then $M = \tilde{L}\tilde{L}^T$, now $\tilde{L}$ is stored rather than M.

When should CG be used?

The CG method should be used for large problems, otherwise Gaussian elimination or other factorization algorithms are less sensitive to rounding errors. The CG method also does not produce fill in the arrays holding the matrix and has fast convergence.

Nonlinear CG

Fletcher and Reeves showed how to the CG method to nonlinear functions by making changes:

- α_k is computed using line search.
- The gradient of the nonlinear objective has to be used.

Algorithm 3 FR

```
1: Given $x_0$
2: Evaluate $f_0 = f(x_0)$, $\nabla f_0 = \nabla f(x_0)$
3: $p_0 = -\nabla f_0$
4: $k = 0$
5: while $\nabla f_k \neq 0$ do
6:     Compute $\alpha_k$ and set $x_{k+1} = x_k + \alpha_k p_k$.
7:     Evaluate $\nabla f_{k+1}$
8:     $\beta_{k+1}^{FR} = \frac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}$
9:     $p_{k+1} = -\nabla f_{k+1} + \beta_{k+1} p_k$
10:    $k = k + 1$
11: end while
```

Reduces to the linear CG if f is strongly convex quadratic.

Nonlinear CG

We have that

$$\nabla f_k^T p_k = -\|\nabla f_k\|^2 + \beta_k^{FR} \nabla f_k^T p_{k-1}$$

so if α_k is exact, then p_k is a descent direction.

The step lengths α_k could be chosen to fulfill the strong Wolfe conditions

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k \nabla f_k^T p_k,$$

$$\|\nabla f(x_k + \alpha_k p_k)^T p_k\| \le -c_2 \nabla f_k^T p_k.$$

This ensures a descent direction.

The Polak-Ribiére Method

The Polak-Ribiére Method which alters the β_k update accordingly

$$\beta_{k+1}^{PR} = \frac{\nabla f_{k+1}^T(\nabla f_{k+1} - \nabla f_k)}{\|\nabla f_k\|^2}, \; \left(\beta_{k+1}^{FR} = \frac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}\right)$$

has been shown to be more robust and efficient when applied to general nonlinear functions, it is however the same when f is a strongly convex quadratic function.

Now the strong Wolfe conditions does not guarantee that p_k is a descent direction. However the modification

$$\beta_{k+1}^{+} = \max(0, \beta_{k+1}^{PR})$$

fixes this problem.

The FR-PR

Global convergence can be guaranteed if $|\beta| \leq |\beta_{FR}|$.

$$\beta_k = \begin{cases} \beta_k^{PR}, \text{ if, } |\beta_k^{PR}| \leq |\beta_k^{FR}| \\ \beta_k^{FR}, \text{ if, } |\beta_k^{PR}| \geq |\beta_k^{FR}| \end{cases}$$

There are some alternative choices for β presented in the book that are not covered here.

Modifications

- Quadratic interpolation along the search direction in the line search gives an exact α_k for strongly convex quadratic functions.
- Restarting by setting $\beta_k = 0$ after every n iterations to refresh the algorithm.

Descent Lemma

Lemma

Suppose that FR is implemented with the strong Wolfe conditions line search with $0 < c_2 < 1/2$ then the method generates descent directions p_k satisfying

$$-\frac{1}{1-c_2} \leq \frac{\nabla f_k^T p_k}{\|\nabla f_k\|^2} \leq \frac{2c_2-1}{1-c_2}$$

FR Direction

A problem with the FR method is that if $\cos(\theta_k)$ is small, a long sequence of unproductive steps will follow, i.e. $p_{k+1} \approx p_k$.

The PR method on the other hand produces a steepest descent step if $\cos(\theta_k)$ is small.

"In general FR should not be implemented w/o a restart strategy".

Global Convergence

Theorem

If the level set $L = \{x | f(x) \leq f(x_0)\}$ *is bounded and that in some open neighbourhood* N *of* L, f *is Lipschitz continuously differentiable, then,*

$$\liminf \|\nabla f_k\| = 0.$$

Global Convergence

Theorem

Consider the PR method with an ideal line search. There exists a twice continously differentiable function f and a starting point x_0 such that the sequence $\{\|\nabla f_k\|\}$ is bounded away from zero.

Numerical Comparison

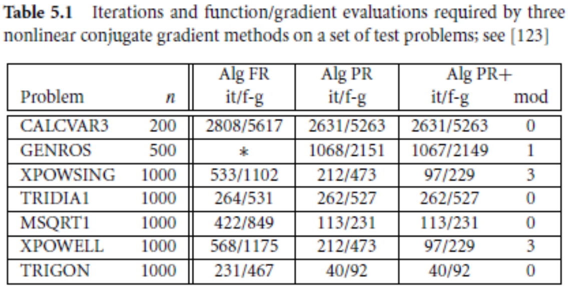


Table 5.1 Iterations and function/gradient evaluations required by three nonlinear conjugate gradient methods on a set of test problems; see [123]

Problem	n	Alg FR it/f-g	Alg PR it/f-g	Alg PR+ it/f-g	mod
CALCVAR3	200	2808/5617	2631/5263	2631/5263	0
GENROS	500	*	1068/2151	1067/2149	1
XPOWSING	1000	533/1102	212/473	97/229	3
TRIDIA1	1000	264/531	262/527	262/527	0
MSQRT1	1000	422/849	113/231	113/231	0
XPOWELL	1000	568/1175	212/473	97/229	3
TRIGON	1000	231/467	40/92	40/92	0

Exercises

1. Show that conjugate directions of $A > 0$ are linearly independent.
2. Show that $\|x_0 - x^*\|_A^2 = \phi(x_0) - \phi(x^*)$.
3. Verify formula (5.7) in the book.
4. Construct matrices with various eigenvalue distributions (clustered and non-clustered) and apply the CG. Comment on whether the behavior can be explained from Theorem 5.5.
5. Prove Theorem 5.2
6. Prove Theorem 5.3

The last two are easily found in the book but I think it is a good exercise to be able to prove these on the whiteboard.