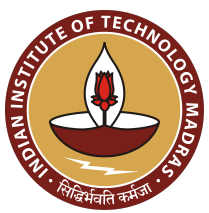
Conjugate Gradient Methods

Saurav Samantaray

Department of Mathematics

Indian Institute of Technology Madras

February 28, 2024



Conjugate Gradient Methods

- They are among the most useful techniques for solving large linear systems of equations.
- They can be adapted to solve non-linear optimisation problems.
- The linear conjugate gradient method is an alternative to Gaussian elimination that is well suited for solving large scale problems.
- Linear conjugate gradient method was proposed by Hestenes and Stiefel in 1950.
- A Key feature of these algorithms is, they require no matrix storage and are faster than the steepest descent method.

Linear Conjugate Gradient Method

The linear conjugate gradient method is an iterative method for solving linear system of equations

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

where A is an $n \times n$ symmetric positive definite matrix.

- The above problem of solving a linear system of equations can be equivalently stated as a minimisation problem:

$$\min_x \phi(x) := \frac{1}{2} x^T A x - b^T x \tag{2}$$

Remark

Both (1) and (2) have the same unique solution.

Linear Conjugate Gradient Method

- The equivalence of both the problems allows us to view conjugate gradient methods either as an algorithm for solving linear systems or as a technique for minimising convex quadratic functions.
- The residual r of the linear system (1) is defined as:

$$r(x) := Ax - b \tag{3}$$

- Note that the gradient of ϕ is:

$$\nabla\phi = r(x) \tag{4}$$

- In particular at $x = x_k$

$$r_k = r(x_k) = Ax_k - b$$

Conjugate Direction Methods

- Generates a set of vectors with a property known as conjugacy.
- The vectors are manufactured, in a very economical fashion.

Conjugacy

A set of non-zero vectors $\{p_0, p_1, \cdots, p_l\}$ is said to be conjugate with respect to the symmetric, positive definite matrix A if

$$p_i^T A p_j = 0 \qquad \text{for,} \quad i \neq j \tag{5}$$

- Any set of vectors satisfying this property is also linearly independent.

Conjugate Direction Methods

- The objective function $\phi(.)$ can be minimised in n steps by successively minimising it along the individual directions in a conjugate set.
- Let $x_0 \in \mathbb{R}^n$ and a set of conjugate directions $\{p_0, p_1, \cdots, p_{n-1}\}$, the sequence of iterates is generated as:

$$x_{k+1} = x_k + \alpha_k p_k \tag{6}$$

- Where α_k is the one-dimensional minimiser of the quadratic function $\phi(.)$ along $x_k + \alpha p_k$, and can be obtained explicitly as:

$$\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k} \tag{7}$$

Convergence of Conjugate Direction Methods

Theorem

For any $x_0 \in \mathbb{R}^n$ the sequence $\{x_k\}$ generated by the conjugate direction algorithm converges to the solution x^* of the linear system (1) in at most n steps.

Sketch of the Proof:

- Since the directions $\{p_i\}$ are linearly independent, they must span the whole space $\mathbb{R}^n$.
- Therefore, the difference between x_0 and the solution x^* can be written in the following way:

$$x^* - x_0 = \sigma_0 p_0 + \sigma_1 p_1 + \ldots + \sigma_{n-1} p_{n-1},$$

for some choice of scalars σ_k.

Convergence of Conjugate Direction Methods

- By premultiplying this expression by $p_k^T A$ and using the conjugacy property, we obtain:

$$\sigma_k = \frac{p_k^T A(x^* - x_0)}{p_k^T A p_k} \tag{8}$$

- We now establish the result by showing that these coefficients σ_k coincide with the step lengths α_k.
- If x_k is generated by the conjugate direction algorithm, we have

$$x_k = x_0 + \alpha_0 p_0 + \alpha_1 p_1 + \ldots + \alpha_{k-1} p_{k-1}.$$

- By premultiplying this expression by $p_k^T A$ and using the conjugacy property, we have that

$$p_k^T A(x_k - x_0) = 0,$$

Convergence of Conjugate Direction Methods

- Therefore,

$$p_k^T A(x^* - x_0) = p_k^T A(x^* - x_k) = p_k^T (b - Ax_k) = -p_k^T r_k$$

- By comparing the above relation with (7) and (8), we find that $\sigma_k = \alpha_k$, giving the result.

Remark

If the matrix A is diagonal, the contours of the function $\phi(.)$ are ellipses whose axes are aligned with the co-ordinate directions $e_1, e_2, \cdots, e_n$.

Convergence of Conjugate Direction Methods

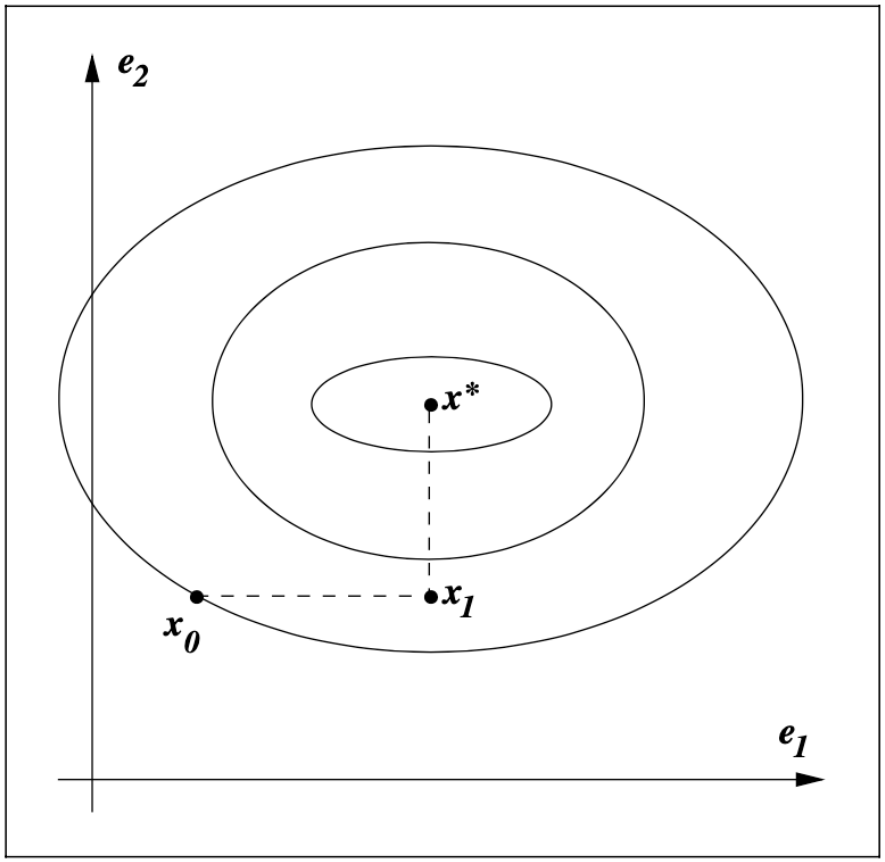
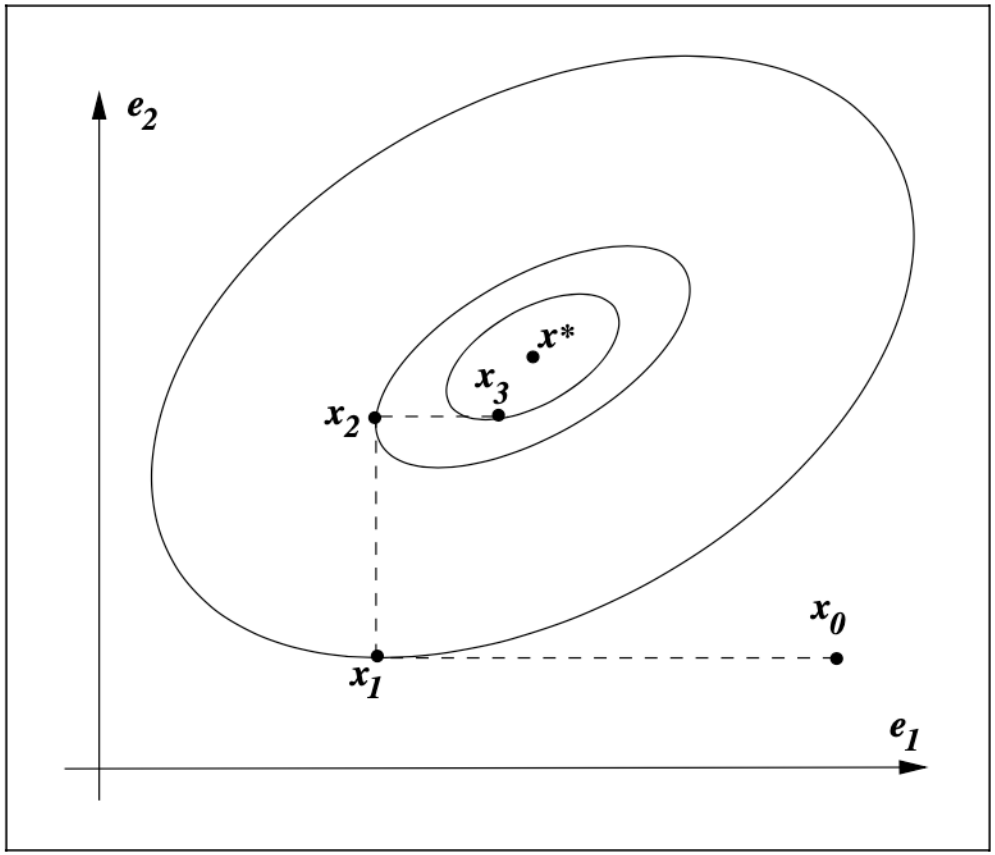


Figure: Successive minimizations along the coordinate directions find the minimizer of a quadratic with a diagonal Hessian in n iterations.

Convergence of Conjugate Direction Methods

- Find the minimiser of this function by performing one-dimensional minimisations along the coordinate directions $e_1, e_2, \cdots, e_n$ in turn.
- When A is not diagonal, its contours are still elliptical, but they are usually no longer aligned with the coordinate directions.
- Successive minimization along these directions in turn no longer leads to the solution in n iterations.

Convergence of Conjugate Direction Methods



Convergence of Conjugate Direction Methods

- The nice behaviour of Figure 1 can be obtained if we transform the problem to make A diagonal and then minimize along the coordinate directions.
- We transform the problem by defining new variables $\hat{x}$ as:

$$\hat{x} = S^{-1}x \tag{9}$$

- S is the $n \times n$ matrix defined by

$$S = [p_0, p_1, \ldots, p_{n-1}]$$

- The quadratic ϕ defined by (2) now becomes:

$$\hat{\phi}(\hat{x}) := \phi(S\hat{x}) = \frac{1}{2}\hat{x}^T(S^TAS)\hat{x} - (S^Tb)^T\hat{x}. \tag{10}$$

Convergence of Conjugate Direction Methods

- By conjugacy property (5), the matrix $S^T AS$ is diagonal.
- The minimising value of $\hat{\phi}$ can be found by performing n one-dimensional minimisations along the coordinate directions of $\hat{x}$.
- The coordinate search strategy applied to $\hat{\phi}$ is equivalent to the conjugate direction algorithm (6)-(7).
- The conjugate direction algorithm terminates in at most n steps.

Convergence of Conjugate Direction Methods

- When the Hessian matrix is diagonal, each coordinate minimisation correctly determines one of the components of the solution x^* .
- After k one-dimensional minimisations, the quadratic has been minimized on the subspace spanned by $e_1, e_2, \ldots, e_k$.
- The following theorem proves this result for the general case in which the Hessian of the quadratic is not necessarily diagonal.

Expanding Subspace Minimization

$$r_{k+1} = r_k + \alpha_k A p_k \tag{11}$$

Theorem(Expanding Subspace Minimization)

Let $x_0 \in \mathbb{R}^n$ be any starting point and suppose that the sequence $\{x_k\}$ is generated by the conjugate direction algorithm (6)-(7). Then

$$r_k^T p_i = 0, \qquad \text{for} \quad i = 0, 1, \ldots, k-1, \tag{12}$$

and x_k is the minimiser of $\phi(x) = \frac{1}{2} x^T A x - b^T x$ over the set

$$\{x | x = x_0 + \text{span}\{p_0, p_1, \cdots, p_{k-1}\}\} \tag{13}$$

- That is, the method minimizes ϕ piece-wise, one direction at a time.
- The current residual r_k is orthogonal to all previous search direction.

How to obtain conjugate directions??

- The discussion applies to a conjugate direction method (6)-(7) based on any choice of the conjugate direction set $\{p_0, p_1, \ldots, p_{n-1}\}$.
- There are many ways to choose the set of conjugate directions.
- The eigenvectors $\{v_1, v_2, \ldots, v_n\}$ of A are mutually orthogonal as well as conjugate with respect to A.
- For large-scale applications computation of the complete set of eigenvectors requires an excessive amount of computation.
- One could modify the Gram–Schmidt orthogonalisation process to produce a set of conjugate directions rather than a set of orthogonal directions.
- The Gram–Schmidt approach is also expensive, since it requires us to store the entire direction set.

Conjugate Gradient Method

- The conjugate gradient method is a conjugate direction method with a very special property.
- In generating its set of conjugate vectors, it can compute a new vector p_k by using only the previous vector p_{k-1}.
- Does not need to know all the previous elements $p_0, p_1, \ldots, p_{k-2}$ of the conjugate set, p_k is automatically conjugate to these vectors.
- Requires little storage and computation.

Conjugate Gradient Method

- The direction p_k is chosen to be a linear combination of the negative residual $-r_k$ and the previous direction p_{k-1}:

$$p_k = -r_k + \beta_k p_{k-1} \tag{14}$$

- The scalar β_k is to be determined by the requirement that p_{k-1} and p_k must be conjugate with respect to A.
- Note that we want to impose $p_{k-1}^T A p_K = 0$ (the conjugacy condition).
- By pre-multiplying (14) by $p_{k-1}^T A$ and using the above imposition, we have:

$$\beta_k = \frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}} \tag{15}$$

Conjugate Gradient Method

- We choose the first search direction p_0 to be the steepest descent direction at the initial point x_0.
- We perform successive one-dimensional minimisations along each of the search directions generated.

Algorithm (CG–Preliminary Version)

Given x_0;
Set $r_0 \longleftarrow Ax_0 - b (= \nabla\phi(x_0))$, $p_0 \longleftarrow -r_0$, $k \longleftarrow 0$;
while $(r_k \neq 0)$:

$\alpha_k \longleftarrow -\frac{r_k^T p_k}{p_k^T A p_K}$;
$x_{k+1} \longleftarrow x_k + \alpha_k p_k$;
$r_{k+1} \longleftarrow Ax_{k+1} - b$;
$\beta_{k+1} \longleftarrow \frac{r_{k+1}^T A p_k}{p_k^T A p_k}$;
$p_{k+1} \longleftarrow -r_{k+1} + \beta_{k+1} p_k$;
$k \longleftarrow k + 1$

end(while)

The algorithm Works

- We present a more efficient version later.

Theorem

Suppose that the kth iterate generated by the conjugate gradient method is not the solution point x^*. The following four properties hold:

$$r_k^T r_i = 0, \qquad \text{, for } i = 0, 1, \ldots, k-1, \tag{16}$$

$$\text{span}\{r_0, r_1, \ldots, r_k\} = \text{span}\{r_0, Ar_0, \ldots, A^k r_0\}, \tag{17}$$

$$\text{span}\{p_0, p_1, \ldots, p_k\} = \text{span}\{r_0, Ar_0, \ldots, A^k r_0\}, \tag{18}$$

$$p_k^T A p_i = 0, \qquad \text{for } i = 0, 1, \ldots, k-1. \tag{19}$$

Therefore the sequence $\{x_k\}$ converges to x^* in at most n steps.

A More Efficient Form of Conjugate Gradient Method

- A slightly more economical version of the CG method can be derived using the results of the previous theorems.
- First we can use the definition of p_{k+1} i.e.

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

and the orthogonality of the residual with the (previous) conjugate directions

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

- Now consider α_k as

$$\begin{aligned} \alpha_k &= -\frac{r_k^T p_k}{p_k^T A p_k} \\ &= -\frac{r_k^T(-r_k + \beta_k p_{k-1})}{p_k^T A p_k} \end{aligned}$$

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

A More Efficient Form of Conjugate Gradient Method

- Second from $r_{k+1} = r_k + \alpha_k A p_k$ we get

$$A p_k = \frac{1}{\alpha_k}(r_{k+1} - r_k)$$

- Note that β_{k+1} is given by

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

- Concentrate on the denominator

$$\begin{aligned} p_k^T A p_k &= \frac{p_k^T}{\alpha_k}(r_{k+1} - r_k) \\ &= -\frac{1}{\alpha_k} p_k^T r_k \qquad (p_k^T r_{k+1} = 0) \\ &= -\frac{1}{\alpha_k}(-r_k^T + \beta_k p_{k_1}^T) r_k \quad (p_k = -r_k + \beta_k p_{k-1}) \\ &= -\frac{1}{\alpha_k} r_k^T r_k \qquad (p_{k-1}^T r_k = 0) \end{aligned}$$

A More Efficient Form of Conjugate Gradient Method

- Now the numerator of β_{k+1}

$$
\begin{aligned}
r_{k+1}^T A p_k &= r_{k+1}^T \left(\frac{1}{\alpha_k}(r_{k+1} - r_k)\right) \\
&= \frac{r_{k+1}^T r_{k+1}}{\alpha_k} - \frac{1}{\alpha_k} r_{k+1}^T r_k \\
&= \frac{r_{k+1}^T r_{k+1}}{\alpha_k} - \frac{1}{\alpha_k} r_{k+1}^T (-p_k + \beta_k p_{k-1}) \\
&= \frac{r_{k+1}^T r_{k+1}}{\alpha_k} \qquad (r_{k+1}^T p_k = 0),\ (r_{k+1}^T p_{k-1} = 0)
\end{aligned}
$$

- Therefore finally we have:

$$
\beta_{k+1} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} \tag{20}
$$

Updated CG algorithm

Algorithm (Refined Version)

Given x_0;
Set $r_0 \longleftarrow Ax_0 - b(= \nabla\phi(x_0))$, $p_0 \longleftarrow -r_0$, $k \longleftarrow 0$;
while $(r_k \neq 0)$:

$$\alpha_k \longleftarrow \frac{r_k^T r_k}{p_k^T A p_K};$$
$$x_{k+1} \longleftarrow x_k + \alpha_k p_k;$$
$$r_{k+1} \longleftarrow r_k + \alpha_k A p_k;$$
$$\beta_{k+1} \longleftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k};$$
$$p_{k+1} \longleftarrow -r_{k+1} + \beta_{k+1} p_k;$$
$$k \longleftarrow k + 1$$

end(while)

Computation in CG Method

The major computational tasks to be performed at each step are:

- computation of the matrix–vector product Ap_k,
- calculation of the inner products $p_k^T(Ap^k)$ and,
- $r_{k+1}^T r_{k+1}$,
- and calculation of three vector sums.

Remark

The CG method is recommended only for large problems; otherwise, Gaussian elimination or other factorization algorithms such as the singular value decomposition are to be preferred, since they are less sensitive to rounding errors.