A Survey of the Conjugate Gradient Method

Michael Lam
Math 221 Final Project

The conjugate gradient method (CG) was developed independently by Hestenes and Stiefel [1] back in the 1950s and has enjoyed widespread use as robust method for solving linear systems and eigenvalue problems when the associated matrix is symmetric and positive definite. It was originally described as a direct method for the solution of linear systems because the method gave the exact answer when using exact arithmetic, however, in practice the CG method never converged to the correct value in the predicted dim(N) iterations. Engeli [8] noted this and classified the CG method as an iterative method. Later, this apparent discrepancy was discovered to be due to numerical error accrued during floating point arithmetic. This paper will focus on how the CG method is able to solve linear systems, but for a good explanation of the history behind the development of CG, the reader is referred to [7].

An easy method for understanding the CG method, is to reinterpret the linear system as a different problem in disguise[2][3]. It can be shown that the extremum of the quadratic equation

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

occur when

$$f'(x) = \frac{1}{2}(A^T + A)x - b = 0 \tag{2}$$

implying that Equation 1 has an extremum when

$$\frac{1}{2}(A^T + A)x = b \tag{3}$$

which for a symmetric matrix A, Equation 3 reduces to

$$Ax = b$$

A necessary condition for an extremum to be a minimum, is for

$$x^T Ax > 0$$

Thus, if the matrix A is positive definite, the solution to Ax=b will be a minimum of the quadratic function f(x). So, for a symmetric, positive definite matrix, the linear system Ax=b has been transformed into a minimization problem of an N dimensional surface. Figure 1[*] shows an example of a 2D quadratic surface.

Given a starting value of x_0, a direction must be chosen to move in the direction of the minimum. The most obvious direction to choose is the negative of the gradient of f(x), or the direction of steepest decent.

$$-f'(x) = b - Ax = r \tag{4}$$

As shown above, this search direction corresponds to the residual of the linear system Ax=b. Stepping in this direction gives

$$x_{i+1} = x_i + \alpha_i r_i$$

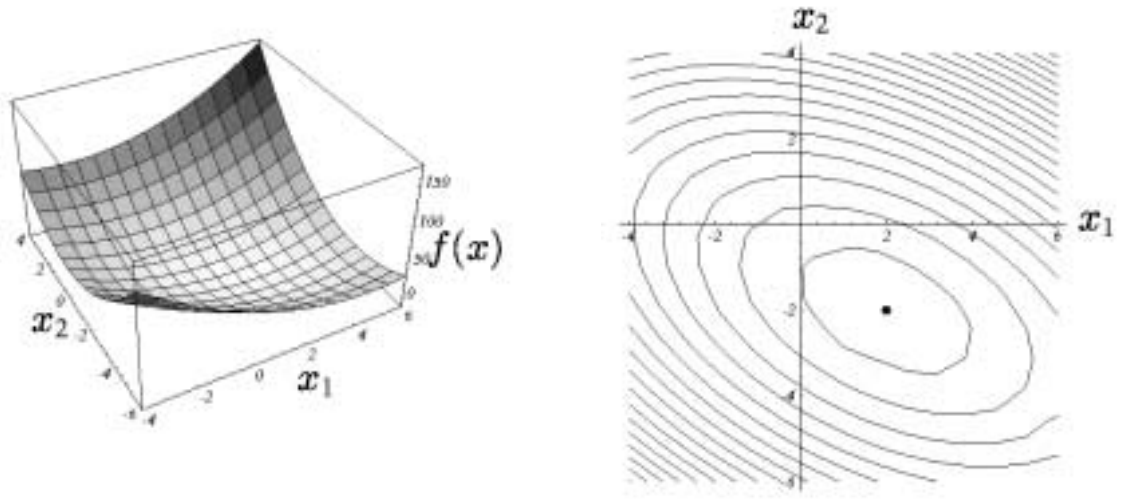
where α is chosen to minimize the value of f(x) along the search direction, corresponding to a line search. It is easily shown that f(x) is minimized along this direction when the directional derivative is zero.

$$\frac{d}{d\alpha} f(x_{i+1}) = f'(x_{i+1})^T \frac{d}{d\alpha} x_{i+1} = f'(x_{i+1})^T r_i = r_{i+1}^T r = 0$$

which corresponds to orthogonal residuals. Choosing α to satisfy orthogonal residuals gives the value

$$\alpha_i = \frac{r_i^T r_i}{r_i^T A r_i}$$

[*] All figures within this document borrowed from reference [2]



(a) (b)

Figure 1. (a) 3D plot of a typical quadratic function f(x), with a minimum at (2,-2). (b) A contour plot of the same quadratic function

When α is chosen, the new residual will be orthogonal to the old residual and the process can be repeated until the value of x that minimizes f(x) is found (corresponding to the solution of Ax=b).

Choosing the search direction as the negative of the residual is easy to implement, but this manner of searching can require a large number of iterations as the residuals zig zag towards the minimum value. For ill-conditioned matrices , as seen in Figure 2, the quadratic function can produce a trough-like topography that can require a large number of iterations to converge for a chosen x_0, since the same directions are traversed multiple times on the path towards the minimum. A better choice of search directions is to choose directions that are orthogonal to the error, such that the N dimensional surface could be minimized in n iterations. It turns out that choosing the appropriate α to achieve this requires knowledge of the answer, which is not helpful. However, conjugate search directions can be chosen to achieve the same result of dim(N) convergence.

Two vectors, x and y, are conjugate, or A-orthogonal, when

$$x^T A y = 0$$

Choosing the search directions as conjugate vectors d, we have the two equations

$$x_{i+1} = x_i + \alpha_i d_i \tag{5}$$

$$e_{i+1} = e_i + \alpha_i d_i \text{ (since } e_i = x_i - x) \tag{6}$$

and it is easy to show that a line search minimizes f(x) when

$$\frac{d}{d\alpha} f(x_{i+1}) = f'(x_{i+1})^T \frac{d}{d\alpha} x_{i+1} = f'(x_{i+1})^T d_i = r_{i+1}^T d_i = (Ae_{i+1})^T d_i = d_i^T A e_{i+1} = 0 \tag{7}$$

Equation 7 shows that the search directions should be conjugate to the error, rather than orthogonal. Solving for α

$$\alpha_i = \frac{d_i^T r_i}{d_i^T A d_i} \tag{8}$$

Conjugate vectors are orthogonal in a stretched space, rather than the current space and can thus be used as a basis of search directions to achieve convergence in N iterations. To prove this, imagine that the initial error in x_0 is a linear combination of these conjugate search directions

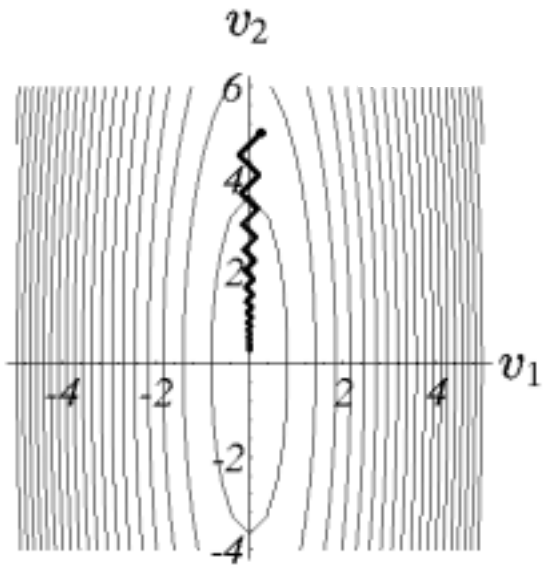


Figure 2. Residuals chosen as search directions (steepest decent method) can require a large number of iterations to converge.

$$e_0 = \sum_{j=0}^{N-1} \delta_j d_j \tag{9}$$

where δ_j are the coefficients of the conjugate search directions d_j. Multiplying on the left by $d^T_k A$ gives

$$d_k^T A e_0 = \sum_{j=0}^{N-1} \delta_j d_k^T A d_j = \delta_k d_k^T A d_k \quad \text{(from conjugacy of d vectors)}$$

implying that

$$\delta_k = \frac{d_k^T A e_0}{d_k^T A d_k} = \frac{d_k^T A(e_0 + \sum_{i=0}^{k-1} \alpha_i d_i)}{d_k^T A d_k} = \frac{d_k^T A e_k}{d_k^T A d_k} \quad \text{(since } d^T_k \text{ is conjugate to all } d_i\text{)} \tag{10}$$

Equation 8 and 10 show that $\delta = -\alpha$, meaning that stepping by α in the conjugate search directions is equivalent to removing all the error associated with that direction in the basis. This is more clearly understood by expressing the *i*th error using the recurrence relation from Equation 6:

$$\begin{aligned} e_i &= e_0 + \sum_{j=0}^{i-1} \alpha_j d_j \\ &= e_0 - \sum_{j=0}^{i-1} \delta_j d_j \qquad \text{(since } \alpha = -\delta\text{)} \\ &= \sum_{j=0}^{N} \delta_j d_j - \sum_{j=0}^{i-1} \delta_j d_j \quad \text{(from Equation 9)} \\ &= \sum_{j=i}^{N} \delta_j d_j \end{aligned} \tag{11}$$

This shows that the error in the *i*th iteration is a linear combination of unsearched directions. After stepping in N conjugate search directions, all of the error has been removed and the exact answer to Ax=b has been

found. Thus, choosing conjugate directions will cause the method to cnverge to the exact solution in N iterations.

An additional property of choosing conjugate directions, is that the energy norm is minimized

$$\left\|e_i\right\|_A = e_i A e_i = \sum_{j=i}^{N-1}\sum_{k=i}^{N-1} \delta_j \delta_k d_j^T A d_k = \sum_{j=i}^{N-1} \delta_j^2 d_j^T A d_j \quad \text{(from conjugacy)} \qquad (12)$$

This expression for the energy norm shows that the *i*th error is associated with directions not yet searched, showing that the energy norm has been optimally minimized based on the already searched directions.

Although conjugate search directions have been shown to converge in dim(N) iterations, the question remains how to obtain conjugate directions. Given a basis of vectors U_N, the conjugate Gram-Schmidt process can be used to make the basis conjugate. The Gram-Schmidt process uses the first vector in U_N as the first conjugate direction. Each succeeding vector is conjugated to the all the previous search directions by subtracting out linear combinations of the previously conjugated search directions. The process is shown below without proof:

$$d_0 = u_0$$

$$\beta_{ij} = -\frac{u_i^T A d_j}{d_j^T A d_j} \qquad (13)$$

$$d_i = u_i + \sum_{j=0}^{i-1} \beta_{ij} d_j \qquad (14)$$

The result of repeatedly applying this process is to form a set of search vectors D_N that are mutually conjugate.

The method of conjugate gradients can now be formed by using the residuals not as search directions, as in the steepest decent method, but instead by using the residuals as a basis to form conjugate search directions . In this manner, the conjugated gradients (residuals) form a basis of search directions to minimize the quadratic function f(x). By choosing the residuals as the basis to conjugate, a few important property can be shown. First, all previous search directions will be orthogonal to newly calculated residuals. Multiply Equation 11 on the left by $-d_k^T A$:

$$-d_k^T A e_i = -\sum_{j=i}^{N-1} \delta_j d_k^T A d_j = 0 \qquad \text{(for k<j, from conjugacy)}$$

$$d_k^T r_i = 0 \qquad \text{(since } -Ae = r\text{)} \qquad (15)$$

Another important property is that the residuals are conjugate to every other residual. This can be shown by simply taking the inner product of Equation 14 with any residual:

$$d_i^T r_k = r_i^T r_k + \sum_{j=0}^{i-1} \beta_{ij} d_j^T r_k$$

which from Equation 15 reduces to:

$$r_i^T r_k = 0 \qquad \text{(for } i \neq k\text{)} \qquad (16)$$

Finally, one of the most important properties of the conjugate gradient method can be shown. Multiplying Equation 6 on the left by $-r_j^T A$:

$$r_j^T r_{i+1} = r_j^T r_i - \alpha_i r_j^T A d_i$$

which can be rearranged to:

$$r_j^T A d_i = (r_j^T r_i - r_j^T r_{i+1})(1/\alpha_i)$$

The left hand side can be identified as a portion of Equation 13 (since the residuals are the the vectors to conjugate), and with the knowledge from orthogonal residuals in Equation 16, the above equation can be reduced to:

$$\beta_{ij} = (\frac{r_i^T r_i}{d_{i-1}^T A d_{i-1}})(1/\alpha_{i-1}) \quad \text{(for i = j+1)} \quad (17)$$

$$\beta_{ij} = 0 \quad \text{(for i > j+1)} \quad (18)$$

This says that the coefficients used to conjugate new residuals for the creation of new search directions are all zero, except for the very last search direction. In other words, the residuals are already conjugate to every past search direction except for the immediate previous search direction. This is a fantastic result that allows the conjugate gradient method to enforce conjugacy of new search directions by only storing the previous search direction.

The CG algorithm can be formulated as follows:

$d_0 = r_0 = b - Ax_0$ (choose residual as starting direction)

$\alpha_i = \frac{r_i^T r_i}{d_i^T A d_i}$ (perform line search in d_0 direction)

$x_{i+1} = x_i + \alpha_i d_i$ (obtain new x)

$r_{i+1} = r_i - \alpha_i A d_i$ (calculate new residual)

$\beta_{i+1} = \frac{r_{i+1}^T r_{i+1}}{r_i^T r_i}$ (calculate coefficient to conjugate residual)

$d_{i+1} = r_{i+1} + \beta_{i+1} d_i$ (obtain new conjugated search direction)

The conjugate gradient method can also be understood by investigating the Lanczos Algorithm. The Lanczos Algorithm reduces a symmetric, positive, definite matrix A to a tridiagonal matrix T by producing an orthogonal similarity of A.

$$T = Q^T A Q$$

The column q_{k+1} of Q, is a vector that is parallel to the residual r_k. For a proof of this, the reader is referred to [4]. In essence, the calculation of the *k*th residual in the conjugate gradient method, is actually calculating the q_{k+1} column of the matrix Q in the Lanczos Algorithm. More detailed information about the Lanczos Algorithm can be found in [3] and [4].

In principle, the CG algorithm converges to the exact solution in dim(N) steps, however, due to round off errors in the calculation, the residual can become inaccurate and the search directions will lose conjugacy. When dimensions are large, it is important to understand the convergence of the CG method since taking N steps can cause a large lose of conjugacy in the search directions. Since the search directions are all conjugated residuals, they span a Krylov sub-space:

$$D_i = span\{r_0, Ar_0, A^2 r_0, ..., A^{i-1} r_0\} = span\{Ae_0, A^2 e_0, A^3 e_0, ..., A^i e_0\}$$

The error in the *i*th step of the CG method can be expressed as a linear combination of the conjugated vectors:

$$e_i = e_o + \sum_{j=1}^{i} \psi_j A^j e_o = (I + \sum_{j=1}^{i} \psi_j A^j) e_o = P_i(A) e_o \quad (19)$$

Equation 9 shows this linear combination can be expressed as a polynomial of the matrix A multiplied by the initial error, assuming that $P_i(0)=1$. Furthermore, the initial error can be expressed as a linear combination of the eigenvectors of A.

$$e_0 = \sum_{j=1}^{N} \xi_j v_j$$

$$e_i = P_i(A)[\sum_{j=1}^{N} \xi_j v_j] = \sum_{j=1}^{N} \xi_j P_i(A) v_j = \sum_{j=1}^{N} \xi_j P_i(\lambda_j) v_j$$

From Equation 12, it is known that the CG method minimizes the energy norm of the *i*th error term, meaning that

$$\left\|e_i\right\|_A^2 = (e_i A e_i)^2 = \sum_j \xi_j^2 [P_i(\lambda_j)]^2 \lambda_j \le \min_{P_i} \max_{\lambda \in \Lambda(A)} [P_i(\lambda)]^2 \left\|e_0\right\|_A^2 \tag{20}$$

Equation 20 has shown that the CG method minimizes this energy norm, which corresponds to minimizing the polynomial $P_i(\lambda)$. A polynomial of degree N, constrained to have $P_i(0)=1$, can minimize N separate eigenvalues. This fact reinforces the knowledge the CG method will converge to the exact solution in N steps, since after N steps an N degree polynomial can be calculated to minimize the energy norm.

An alternative to minimizing the polynomial at specific λ_j, one can minimize the polynomial over the entire range of eigenvalues, $[\lambda_{min}, \lambda_{max}]$. This can be accomplished via Chebychev Polynomials:

$$T_i(\omega) = \frac{1}{2}[(\omega + \sqrt{\omega^2 - 1})^i + (\omega - \sqrt{\omega^2 - 1})^i] \tag{21}$$

Chebychev Polynomials have $\left|T_i(\omega)\right| \le 1$ on the interval $\omega \in [-1,1]$, and rise quickly to $\left|T_i(\omega)\right| = \infty$ when outside the interval. The polynomial that minimizes Equation 12 over the interval $[\lambda_{min}, \lambda_{max}]$ is

$$P_i(\lambda) = \frac{T_i(\frac{\lambda_{\max} + \lambda_{\min} - 2\lambda}{\lambda_{\max} - \lambda_{\min}})}{T_i(\frac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}})} \tag{22}$$

To prove this, assume there exists another polynomial of degree *i*, Q_i, that is better at minimizing Equation 12 on the appropriate interval $[\lambda_{min}, \lambda_{max}]$, such that $Q_i(0)=1$.

$$Q_i(\lambda) < T_i(\frac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}})^{-1}$$

The polynomial P_i-Q_i, must have a zero at $\lambda=0$ and at the *i* zeros of the polynomials, meaning this polynomial must have *i*+1 zeros, which is a contradiction. Therefore, P_i must be the minimizing polynomial on the interval $[\lambda_{min},\lambda_{max}]$.

Plugging the polynomial T_i into Equation 12, we have

$$\left\|e_i\right\|_A \le T_i(\frac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}})^{-1} \left\|e_0\right\|_A$$

$$= T_i(\frac{\kappa + 1}{\kappa - 1})^{-1} \left\|e_0\right\|_A = 2[(\frac{\sqrt{\kappa} + 1}{\sqrt{\kappa} - 1})^i + (\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1})^i]^{-1} \left\|e_0\right\|_A = \omega \left\|e_0\right\|_A \tag{23}$$

For large dimensional systems, *i* is very large and Equation 11 can be reduced to

$$\left\|e_i\right\|_A \le 2(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1})^i \left\|e_0\right\|_A \tag{24}$$

Equation 24 shows that the convergence of the CG method depends heavily on the condition number of the matrix, where $\kappa = \lambda_{max} / \lambda_{min}$ is the condition number for the matrix A. Figure 3 plots the convergence of the CG method (ω multiplying $\left\|e_0\right\|_A$) through condition number.

From the above convergence analysis, it can be shown that the CG method can converge in a quicker time than dim(N) iterations. If the matrix A has duplicated eigenvalues, the CG method will converge quicker to the exact solution because there will be less search directions involved.

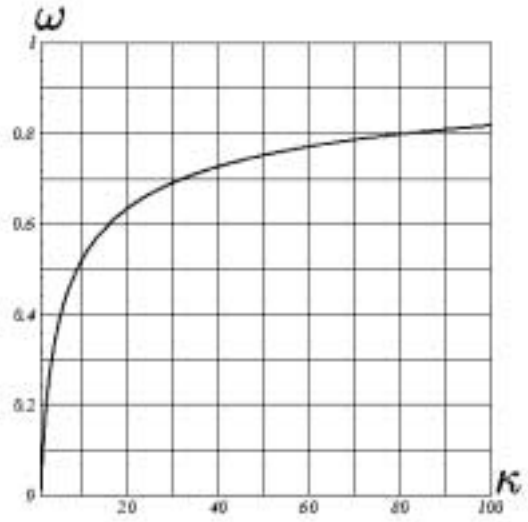


Figure 3. Convergence of the CG method as a function of the matrix condition number.

Additionally, if the eigenvalues are grouped close together, the polynomial chosen to minimize the energy norm could in fact satisfy the minimization requirements with a lower order polynomial. Also, the starting value of x_0 could be a vector that is already conjugate to some of the search directions, resulting in reduced search directions needed to find the minimum. Each of these instances can cause the CG method to converge in a quicker time than the typical dim(N) convergence. Preconditioning schemes have been created to take advantage of the faster convergence properties by mapping the eigenvalues of A to either a smaller spread in their max and min values (smaller κ) or by grouping some of the eigenvalues closer together such that a lower order polynomial can minimize the grouped eigenvalues. Combinations of the above instances will further improve convergence.

As already mentioned, the CG method would converge to the exact solution in dim(N) steps in the absence of round off errors. Due to round off errors, the CG method cannot give the exact answer and is therefore classified as an iterative method, where some tolerance is specified. The biggest threat to the CG method is a loss of conjugacy in the search vectors. Each new residual is conjugated with respect to the last search direction, and assumed to be conjugate to the previous search directions. This can reintroduce error along already searched directions, resulting in an answer that does not converge to the solution after dim(N) steps. One fix to this, is to reothogonalize the search directions after a few directions have been searched, enforcing conjugacy explicitly among the previously searched directions. This, however, is a poor fix since the computational cost of directly enforcing conjugacy would wipe out any gains made by using CG. Another fix is to periodically restart the CG method, allowing new search directions to be generated from a new residual. This allows the error accumulated from the previous starts, to be conjugate to the new vectors, allowing the removal of such errors as the method proceeds. Finally, performing the CG method in extra precision will help to ensure that the rounding errors generated in the calculation remain at a level that is inconsequential to your desired precision, without complicating the application of the method.

Despite the predicted convergence rate of approximately dim(N) iterations, in general the conjugate gradient method converges much quicker than any of these analyses indicate. Kaniel [5] showed that the convergence of CG was determined solely by the eigenvalue spectrum, regardless of dimension of the matrix. In essence, the condition number of the matrix determined the rate of convergence and not its dimension. An important implication of Kaniel's work is that the conjugate gradient method can be used on systems of equations that are so large, that even taking the dim(N) iterations would be much too long. This can be understood on the basis of the Chebychev polynomials discussed earlier. A Chebychev polynomial of certain order can minimize the energy norm based on the specific interval given ($[\lambda_{min},\lambda_{max}]$), and adding more eigenvalues within the interval will generally not change the minimization properties. So the convergence properties of different dimensional matrices will be nearly the same if their condition numbers are all roughly equal. A review of Kaniel's work can be seen in [6].

The CG method can also be applied to non-linear problems, but with much less success since the non-linear functions have multiple minimums. The CG method will indeed find a minimum of such a non-linear function, but it is in no way guaranteed to be a global minimum, or the minimum that is desired.

The conjugate gradient method is great iterative method for solving large, sparse linear systems with a symmetric, positive, definite matrix. Floating point errors that accrue in its implementation drastically change its properties from those predicted in exact arithmetic, but it remains a powerful and robust iterative tool to solve linear systems.

References

[1] M. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems", J. Res. Nat. Bur. Stand., 49 (1952), pp. 409-436.

[2] J.R. Shewchuck, "An Introduction to the Conjugate Gradient Method Without the Agonizing Pain," http://www.cs.berkeley.edu/~jrs/

[3] G. Golub and C. Van Loan. *Matrix Computations*, Johns Hopkins University Press, Baltimore, pp. 322-337, pp. 352-377, 1983.

[4] J.W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, pp. 299-319, 1997.

[5] S. Kaniel. "Estimates for Some Computational Techniques in Linear Algebra," Math. Comp. 20, 369-78.

[6] D. Scott. "Analysis of the Symmetric Lanczos Process," Electronic Research Laboratory Technical Report UCB/ERL M78/40, University of California.

[7] Y. Saad, H. Van der Vorst, "Iterative solution of linear systems in the 20th century," J. Comput. Appl. Math., 123 (2000), 1-33.

[8] M. Engeli, T. Ginsburg, H. Rutishauser, and E. Stiefel. "Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems," pp. 79-107, Basel-Stuttgart, 1959. Institute of Applied Mathematics, Zurich, Birkhauser Verlag.