Conjugate Gradient Method

This note is mainly based on 'Numerical Analysis' by Burden and Faires.
It was first developed as direct method to solve a system of linear equations $Ax = b$ but was found to be inferior to Gauss elimination. However, it can can be used as an iterative method for a sparse system with predictable pattern.

Here the matrix A is assumed to be symmetric and positive definite. For two n-vectors u and v, we define the standard inner product $(u, v) = u^t v = \sum_{i=1}^n u_i v_i$. For a positive definite matrix A, we define A-inner product of two vectors u and v by $(u, v)_A = (u, Av) = u^t Av = (Au)^t v = (Au, v)$. As with inner product, there is an associated norm $||u||_A = \sqrt{(u, Au)}$.

Theorem: The vector x^* is a solution to the symmetric positive definite linear system $Ax = b$ if and only if x^* minimizes the value of $f(x) = \frac{1}{2}(x, x)_A - (x, b)$.

Proof: For u and any $v \neq 0$, consider

$$f(u + tv) = \frac{1}{2}(u + tv, u + tv)_A - (u + tv, b) = f(u) + t((u, v)_A - (v, b)) + \frac{t^2}{2}(v, v)_A$$

The above can be written as $f(u+tv) \equiv h(t) = \alpha + \beta t + \frac{1}{2}\gamma t^2$ with $\gamma > 0$. Hence it's minimum occurs at $t = \hat{t}$ given by $\hat{t} = -\beta/\gamma$ and $h(\hat{t}) = \alpha - \beta^2/2\gamma$. Hence $h(\hat{t}) \leq \alpha$ and $h(\hat{t}) = \alpha$ only when $\beta = 0$. Note that $(u, v)_A - (v, b) = (Au - b, v) \implies \hat{t} = (b - Au, v)/(v, Av)$ and $f(u + \hat{t}v) \leq f(u)$ for all $v \neq 0$ unless $(b - Au, v) = 0$ for which $f(u + \hat{t}v) = f(u)$.

Let x^* satisfies $Ax = b$ and then $(b - Ax^*, v) = 0$ for any vector v. Now $f(x^* + tv) \geq f(x^* + \hat{t}v)$ for any t since minimum attained at $\hat{t}$. But $f(x + \hat{t}v) = f(x^*)$ and hence $f(x^* + tv) \geq f(x^*)$. Now $x = x^* + tv$ is an arbitrary vector and hence $f(x) \geq f(x^*)$. Consequently $f(x)$ cannot be made smaller than $f(x^*)$ and thus x^* minimizes $f(x)$.
Conversely, let x^* minimizes $f(x)$. Then $f(x^*) \leq f(x^* + \hat{t}v)$ for any nonzero vector v. But $f(x^* + \hat{t}v) \leq f(x^*)$ which implies that $f(x^*) = f(x^* + \hat{t}v) \implies (b - Ax^*, v) = 0$ for any nonzero vector v. If $b - Ax^* \neq 0$, we take $v = b - Ax^*$, which gives $||b - Ax^*||^2 = 0 \implies Ax^* = b$.

The above theorem shows the way to proceed. We choose an $x^{(0)}$ which is an initial approximation to x^*. If $b - Ax^{(0)} \neq 0$, we chose a nonzero $v^{(1)}$ (search direction) and $t_1 = (b - Ax^{(0)}, v^{(1)})/(v^{(1)}, v^{(1)})_A$ so that $x^{(1)} = x^{(0)} + t_1 v^{(1)}$ is a better approximation. This suggests the following algorithm:

Algorithm

Choose an initial $x^{(0)}$ and $v^{(1)} \neq 0$.
For $k = 1, 2, 3, \cdots,$
Calculate $t_k = (b - Ax^{(k-1)}, v^{(k)})/(v^{(k)}, v^{(k)})_A = (r^{(k-1)}, v^{(k)})/(v^{(k)}, v^{(k)})_A$
$x^{(k)} = x^{(k-1)} + t_k v^{(k)}$.
Search the new direction $v^{(k+1)}$.

Finding search directions:

Note that $f(x) = \frac{1}{2}(x, x)_A - (x, b)$ and solution x^* of $Ax = b$ minimizes $f(x)$. Note that ∇f is the direction of fastest increase of f at a point and hence $-\nabla f = b - Ax = r$ where r is the residual vector gives the direction of fastest decrease. Hence one way to choose search direction is by $v^{(k+1)} = r^{(k)} = b - Ax^{(k)}$. Choosing this way is called method of steepest descent which is quite slow for linear system.

An alternative approach is to choose a set of nonzero direction vectors (called conjugate directions) $v^{(1)}, v^{(2)}, \cdots, v^{(n)}$ which satisfy $(v^{(i)}, v^{(j)})_A = 0$ for $i \neq j$. This is called A-orthogonality conditions.

Theorem: With the choice of conjugate directions $v^{(1)}, v^{(2)}, \cdots, v^{(n)}$, it can be proved that $Ax^{(n)} = b$. (Here $x^{(n)}$ is obtained using algorithm given in first page)

Remark: This theorem implies that if exact arithmetic is used with conjugate directions, then the algorithm converges in n-steps. In that sense, it is comparable to direct methods. However, the number of iterations becomes large when n is large. Hence, we terminate the iteration when suitable accuracy is achieved.

Theorem: It can be proved that the residual vectors $r^{(k)} = b - Ax^{(k)}$ for $k = 1, 2, \cdots, n$ of the conjugate direction method satisfy $(r^{(k)}, v^{(j)}) = 0$ for $j = 1, 2, \cdots, k$.

Conjugate gradient method:

The conjugate gradient method of Hestenes and Stiefel chooses the search directions $v^{(k)}$ during the iterative process so that the residual vectors $r^{(k)}$ are mutually orthogonal. To construct the direction vectors $v^{(1)}, v^{(2)}, \cdots, v^{(n)}$ and the approximations $x^{(1)}, x^{(2)}, \cdots, x^{(n)}$, we proceed as follows.

We start with an initial approximation $x^{(0)}$ and if $x^{(0)}$ is not a solution, then we use the steepest descent direction $r^{(0)} = b - Ax^{(0)}$ as $v^{(1)}$.

Assume that conjugate directions $v^{(1)}, v^{(2)}, \cdots, v^{(k)}$ and approximate solutions $x^{(1)}, x^{(2)}, \cdots, x^{(k)}$ have already been computed, where

$x^{(k)} = x^{(k-1)} + t_k v^{(k)}$ and $(v^{(i)}, v^{(j)})_A = 0, \quad (r^{(i)}, r^{(j)}) = 0$ for $i \neq j$.

If $x^{(k)}$ is the solution of $Ax = b$, then nothing to do. Otherwise,

$r^{(k)} = b - Ax^{(k)} \neq 0$ and $(r^{(k)}, v^{(j)}) = 0$ for $j = 1, 2, \cdots, k$.

We generate $v^{(k+1)}$ from $r^{(k)}$ by setting $v^{(k+1)} = r^{(k)} + s_k v^{(k)}$ and choose s_k so that

$(v^{(k)}, v^{(k+1)})_A = 0 \implies s_k = -(v^{(k)}, r^{(k)})_A/(v^{(k)}, v^{(k)})_A. \cdots\cdots$ (*)

It can be also shown that $(v^{(j)}, v^{(k+1)})_A = 0$ for $j = 1, 2, \cdots, k$ and hence $v^{(1)}, v^{(2)}, \cdots, v^{(k+1)}$ is an A-orthogonal set. Having chosen $v^{(k+1)}$, we compute

$t_{k+1} = (v^{(k+1)}, r^{(k)})/(v^{(k+1)}, v^{(k+1)})_A = (r^{(k)}, r^{(k)})/(v^{(k+1)}, v^{(k+1)})_A + s_k(v^{(k)}, r^{(k)})/(v^{(k+1)}, v^{(k+1)})_A.$

Since $(v^{(k)}, r^{(k)}) = 0$ (by the theorem stated above), we have

$t_{k+1} = (r^{(k)}, r^{(k)})/(v^{(k+1)}, v^{(k+1)})_A. \cdots\cdots$ (**)

Thus $x^{(k+1)} = x^{(k)} + t_{k+1} v^{(k+1)}$ is obtained.

To compute $r^{(k)} = b - Ax^{(k)}$, we have

$r^{(k)} = b - Ax^{(k)} = b - A\{x^{(k-1)} + t_k A v^{(k)}\} \implies r^{(k)} = r^{(k-1)} - t_k A v^{(k)}$. This gives

$(r^{(k)}, r^{(k)}) = (r^{(k-1)}, r^{(k)}) - t_k(Av^{(k)}, r^{(k)}) = -t_k(r^{(k)}, Av^{(k)})$ (since residual vectors are orthogonal). Now from (**), we have $t_k = (r^{(k-1)}, r^{(k-1)})/(v^{(k)}, v^{(k)})_A$ and hence

$$(r^{(k)}, r^{(k)}) = -\frac{(r^{(k-1)}, r^{(k-1)})}{(v^{(k)}, v^{(k)})_A}(r^{(k)}, v^{(k)})_A$$

From (*)

$$s_k = -\frac{(v^{(k)}, r^{(k)})_A}{(v^{(k)}, v^{(k)})_A} = \frac{(r^{(k)}, r^{(k)})}{(r^{(k-1)}, r^{(k-1)})}$$

In summary, the algorithm is as follows

Algorithm

Choose an initial $x^{(0)}$ and compute $r^{(0)} = b - Ax^{(0)}$ and $v^{(1)} = r^{(0)}$.

For $k = 1, 2, 3, \cdots$ until given accuracy achieved

$$\begin{aligned} t_k &= (r^{(k-1)}, r^{(k-1)})/(v^{(k)}, Av^{(k)}) \\ x^{(k)} &= x^{(k-1)} + t_k v^{(k)} \\ r^{(k)} &= r^{(k-1)} - t_k A v^{(k)} \\ s_k &= \frac{(r^{(k)}, r^{(k)})}{(r^{(k-1)}, r^{(k-1)})} \\ v^{(k+1)} &= r^{(k)} + s_k v^{(k)} \end{aligned}$$

Note that we have one matrix-vector multiplication $Av^{(k)}$, three inner products $(r^{(k-1)}, r^{(k-1)})$,$(v^{(k)}, Av^{(k)})$, $(r^{(k)}, r^{(k)})$ and three scalar vector multiplications in the computation of $x^{(k)}$, $r^{(k)}$, $v^{(k+1)}$.

Remark:

If the matrix A is not well-conditioned, then we usually use pre-conditioned conjugate gradient method.