Computational methods-Lecture 10

The Conjugate Gradient Method for solving positive definite systems

Today, we introduce another technique, the Conjugate Gradient (CG) Method, for solveing the linear systems when the matrix is positive definite:

$$Ax = b.$$

If the matrix is positive semi-definite, some modification of this method can be applied, which we do not consider in our course.

The CG in principle is a direct method since it eliminates the errors step by step and finds the exact solution in n steps. However, it can also be viewed as an iterative method.

1 Reducing it to optimization problem and line search

As we shall later, solving system of equations can be formulated into an optimization problem. Finding the roots is equivalent to finding the minimizers. For our problem, we claim

Theorem 1. *Let A be positive definite. Then, x^* is a solution of $Ax = b$ if and only if it is the global minimizer of*

$$g(x) = \frac{1}{2}x^T Ax - x^T b.$$

Proof. "$\Leftarrow$" This is straightforward, because any critical point satisfies

$$\nabla g = 0.$$

However,

$$\nabla g = Ax - b.$$

The result follows.

"$\Rightarrow$" In fact, since A is positive definite, for any $M > 0$, there exists $R > 0$ such that $g(x) > M$ when $|x| > R$. Hence, there is a global minimizer of g. Hence, it must be a critical point and thus satisfies $Ax = b$. However, the solution of $Ax = b$ is unique, since A is invertible. This then justifies the claim.

□

For the direction "⇒", a more natural way is to consider

$$g(x)-g(x^*) = \langle x-x^*, Ax^*-b\rangle+\frac{1}{2}\langle x-x^*, A(x-x^*)\rangle = \frac{1}{2}\langle x-x^*, A(x-x^*)\rangle \geq 0.$$

As soon as we have the optimization setup, we can try to find the minimizer as follows using the **line search strategy**:

- Find a search direction p and form a function
$$h(t) = g(x + tp).$$
- Minimize the function $h(t)$:
$$p \cdot \nabla h = 0 \Rightarrow p \cdot (A(x + \hat{t}p) - b) = 0.$$
Hence
$$\hat{t} = \frac{p^T(b - Ax)}{p^T Ap}.$$
- The new position is
$$x \leftarrow x + \hat{t}p.$$

For the convenience, we denote

$$r := b - Ax,$$

which is the residual vector.

1.1 Method of steepest descent

The first natural way is to adopt a greedy strategy: the negative gradient points the fastest direction locally. Hence, a naturaly way is to choose

$$p^{(k)} := r^{(k)} = -\nabla g(x^{(k)}) = b - Ax^{(k)}.$$

Then,

$$x^{(k+1)} = x^{(k)} + \frac{|r^{(k)}|^2}{(r^{(k)})^T Ar^{(k)}} r^{(k)}.$$

Remark 1. *This is different from the so-called gradient descent, where*

$$x^{(k+1)} = x^{(k)} - \eta_k \nabla g(x^{(k)})$$

where η_k is a small parameter and it is no a quantity that tries to minimize the function most.

Unfortunately, the steepest descent does not perform well in practice, and see Figure 1.

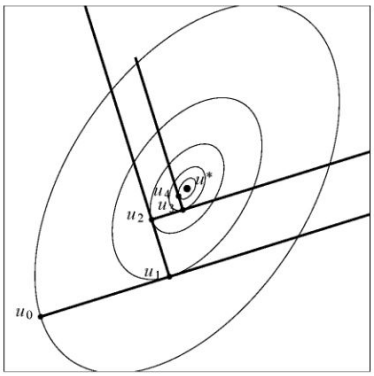

Figure 1: Illustration of steepest descent iterations. Figure cut from the book of Randall Leveque.

1.2 The A-orthogonal directions

The steepest descent is not good in general. However, there is one case when it is ideal: when the contours are spheres (circles in 2D). In this case, the method will find the minimum in n steps.

For general A, the contours (level sets) are ellipsoids. However, if we do change of variables

$$y = \sqrt{A}x,$$

then the contours will be spheres in y variables. In this sense, we should need the directions to be perpendicular in the new variables. In other words, we need

$$(p^{(i)})^T A p^{(j)} = 0.$$

Vectors satisfying this conditions are said to be **A-conjugate**, or **A-orthogonal**.

The intuition above suggests that using A-orthogonal search directions can achieve zero residual in n directions. In fact, this is true.

Theorem 2. *For A-orthogonal search directions, using the line search strategy can yield*

$$Ax^{(n)} = b.$$

Proof. Recall

$$x^{(k+1)} = x^{(k)} + t_k p^{(k)}.$$

Consequently,

$$Ax^{(k)} = Ax^{(0)} + \sum_{j=0}^{k-1} t_j A p^{(j)}.$$

Using A-orthogonality,

$$p^{(k)} \cdot (Ax^{(k)} - b) = p^{(k)} \cdot (Ax^{(0)} - b).$$

Now, consider the residual

$$-r_n = Ax^{(n)} - b = (Ax^{(0)} - b) + \sum_{j=0}^{k-1} t_j A p^{(j)}$$

One finds

$$\langle -r_n, p^{(j)} \rangle = \langle Ax^{(0)} - b, p^{(j)} \rangle + t_j \langle Ap^{(j)}, p^{(j)} \rangle = \langle Ax^{(0)} - b, p^{(j)} \rangle + p^{(j)}(b - Ax^{(j)}).$$

Using the relation we just obtained, this equals zero.

Hence, r_n is perpendicular to $p^{(j)}$ for all $j = 0, \cdots, n-1$. However, since A is positive definite, they are indpendent. Hence, $r_n = 0$. □

How do we understand? In fact, besides the sphere picture mentioned above, we can also understand this in another way: in the kth iteration, the algorithm achieves the minimum in the hyper-plane

$$x + span\{p^{(0)}, \cdots, p^{(k-1)}\}$$

Each time, we buid in a direction, and the new result is optimal in the new hyperplane, instead of just in the direction of the new line, due to the A-orthogonality.

One important property that similar proof can give

Proposition 1. *The residual vector at iteration k is perpendicular to all previous search direction.*

2 The conjugate gradient descent

We have been convinced that using A-orthogonal search directions can be beneficial. How do we construct such directions? One strategy is as follows:

- Given $x^{(0)}$, the initial search direction is the negative gradient direction:
$$p^{(0)} := r^{(0)} = b - Ax^{(0)}.$$

- Suppose that $x^{(k)}$ $(k \geq 1)$ is found and

$$r^{(k)} := b - Ax^{(k)}$$

is nonzero. We now construct $p^{(k)}$ that should be A-orthogonal to previous search directions. The CG algorithm does the following:

$$p^{(k)} = r^{(k)} + sp^{(k-1)}.$$

To make $\langle p^{(k)}, Ap^{(k-1)} \rangle = 0$, we need

$$s_{k-1} = -\frac{\langle p^{(k-1)}, Ar^{(k)} \rangle}{\langle p^{(k-1)}, Ap^{(k-1)} \rangle}.$$

The above provides a way to construct the search directions. However, we must verify that $\{p^{(k)}\}'s$ are A-orthogonal.

Proposition 2. *The space span*$\{p^{(0)}, \cdots, p^{(m-1)}\} = span\{r^{(0)}, Ar^{(0)}, \cdots, A^{m-1}r^{(0)}\}$*;* $p^{(m)}$ *is A-orthogonal to the previous directions for all* $m \geq 1$*.*

Proof. The proof can be done by induction. If $m = 1$, $p^{(0)} = r^{(0)}$ and the first claim is clear. For the second, claim the construction of $p^{(1)}$ gurantess this by choosing s.

Suppose for $m = k \geq 1$, the claim holds. Since

$$r^{(k)} = r^{(0)} - \sum_{j=0}^{k-1} t_j Ap^{(j)}$$

Hence,

$$p^{(k)} = r^{(0)} - \sum_{j=0}^{k-1} t_j Ap^{(j)} + sp^{(k-1)}.$$

By the induction assumption. We write $p^{(j)}$ as linear combination of $A^q r^{(0)}$ with $q \leq k-1$. Then, when $m = k+1$, the left hand side is contained in the right hand side.

Now, since the previous search directions are A-orthogonal, we find $r^{(k)}$ is perpendicular to the all previous search directions. Hence, $r^{(k)}$ is independent of the previous subspace. Hence, $p^{(k)}$ is linear independent to the previous subspace. By this independence, the left hand side has dimension m while the right hand side has dimension at most m. Then, they must be equal.

Now, we verify the A-orthogonality holds for $m = k + 1$.

$$p^{(k+1)} = r^{(k+1)} + sp^{(k)}.$$

For $j < k$, we have

$$\langle p^{(k+1)}, Ap^{(j)} \rangle = \langle r^{(k+1)}, Ap^{(j)} \rangle$$

By what has been proved. $Ap^{(j)}$ is a linear combination of $A^q r^{(0)}$ for $q \leq k$, thus a linear combination of $p^{(q)}$ for $q \leq k$. Hence, by the induction hypothesis, $r^{k+1} \cdot p^{(q)} = 0$. This means

$$\langle p^{(k+1)}, Ap^{(j)} \rangle = \langle r^{(k+1)}, Ap^{(j)} \rangle = 0.$$

The A-orthogonality between $p^{(k+1)}$ and $p^{(k)}$ is ensured by the construction. □

By the proof above, we in fact have the following

Corollary 1.
- $r^{k+1} \cdot p^{(j)} = 0$ *for* $j \leq k$ *and* $\langle r^{(k+1)}, Ap^{(j)} \rangle = 0$ *for* $j < k$.
- $r^{(i)} \cdot r^{(j)} = 0$ *for* $i \neq j$.

Proof. The first part has been proved. For the second part, let us assume $i > j$. Then, $r^{(j)} = p^{(j)} - s_j p^{(j-1)}$. Using the claim just proved, the orthogonality follows easily. □

Note that since $r^{(k+1)} = r^{(k)} - t_k Ap^{(k)}$, $\langle r^{(k+1)}, Ap^{(k)} \rangle \neq 0$.
Here are some observation to reduce the computational cost:

$$-r^{(k+1)} = -r^{(k)} + t_k Ap^{(k)}.$$

Hence, we do not compute $r^{(k+1)} = b - Ax^{(k+1)}$ using the definiton, but with the above formula. Moreover,

$$(p^{(k)})^T r^{(k)} = |r^{(k)}|^2.$$

Moreover,

$$s_k = -\frac{p^{(k)} A r^{(k+1)}}{p^{(k)} A p^{(k)}} = -\frac{r^{(k+1)} \cdot (r^{(k)} - r^{(k+1)})}{t_k p^{(k)} A p^{(k)}} = \frac{|r^{(k+1)}|^2}{|r^{(k)}|^2}.$$

1. Choose x_0; $x \leftarrow x_0$.

2. $r \leftarrow b - Ax$, $p \leftarrow r$

3. For $k = 1, \cdots$

   $w \leftarrow Ap$;

   $t \leftarrow r^T r/(p^T w)$;

   $x \leftarrow x + t * p$;

   $r_o \leftarrow r$;

   $r \leftarrow r - tw$;

   if stopping criterion is not achieved,

   $s \leftarrow (r^T r)/r_o^T r_o$;

   $p \leftarrow r + s * p$;

   Endfor

3 Convergence of the algorithm

Even though CG can achieve the accurate solution in n steps, we can often stop far before finishing n steps. In fact, the error is like

$$C(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1})^k$$

See section 4.3.4 of the book "Finite Difference Methods for ordinary and partial differential equations" by Randall J. LeVeque.

For practical systems like the ones constructed from finite difference, one often has the condition number to be $\kappa \approx n$. Hence, one often needs $k = \sqrt{n}$ to achive desired accuracy. This is much smaller than the required n steps. Hence, it can save time compared with Gauss elimination.

Moreover, when A is sparse, the matrix-vector multiplication can be cheap.