Project : Steepest Descent for solving linear equations

This project is devoted to an idea for iteratively solving linear systems, i.e., solving equations of the form

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

where A is an $n \times n$ matrix and b is a vector in R^n. Henceforth we shall assume that A is a positive definite matrix. Recall that this means that for all non-zero vectors $x \in R^n$

$$x \cdot Ax > 0 \ .$$

This means, in particular, that the kernel of the matrix A consists of the zero vector only and hence the matrix is invertible. The equation (1) has always a unique solution.

A first thought is to formulate the problem as a minimization problem. To this end we consider the function

$$F(x) = \frac{1}{2} x \cdot Ax - b \cdot x \ .$$

At the minimum, which we call y_0, the gradient has to vanish which means that $Ay_0 = b$ and hence y_0 is the solution.

Example

Lets take A be the 2×2 diagonal matrix

$$\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

and

$$b = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \ .$$

In this case the function F is a function of two variables and is given by

$$F(u, v) = \frac{1}{2}(u^2 + 2v^2) - 3u - 4v$$

which, by completing the squares, can be brought into the form

$$F(x, y) = \frac{1}{2}(u - 3)^2 + (v - 2)^2 - \frac{17}{2} \ .$$

We see right away that the level curves of F are ellipses. The gradient of F is given by

$$\nabla F(u, v) = (u - 3, 2v - 4)$$

and it vanishes precisely at the point $(3, 2)$ which is the solution of the system $Ax = b$.

The completion of the square can be carried out quite generally. Expanding the expression

$$\frac{1}{2}(x - A^{-1}b) \cdot A(x - A^{-1}b) = \frac{1}{2} x \cdot Ax - b \cdot x + \frac{1}{2} b \cdot A^{-1} b$$

yields

$$F(x) = \frac{1}{2}(x - A^{-1}b) \cdot A(x - A^{-1}b) - \frac{1}{2} b \cdot A^{-1} b \ .$$

From this we see right away that the minimal value of F is attained at the unique solution y_0 and the the value of the function F at this point is $-\frac{1}{2} b \cdot A^{-1} b$. We also see that the level surfaces of the function F are ellipsoids who have y_0 as the common center.

So far so good, but the real question we want to address is how to find y_0 efficiently and we would like to acquaint you with an idea on how this can be achieved. Note that the issue at hand is not to work this out for 2×2 or 3×3 matrices, but with large matrices, where computers are needed.

The method, which is in a way the simplest one, is the steepest descent method. Start at a point x_0 and think of skiing as fast as possible towards the lowest point. Let us assume that we are not good skiers and cannot turn in a continuous fashion, i.e., we ski first in a straight line, then stop and turn and then again ski in a straight line. You would start in the direction of steepest descent which is opposite to the gradient direction, i.e., you would ski first in the direction of $-\nabla F(x_0)$. How far would you ski? First you will go down and at some point you will go up again. Clearly you want to stop once you have reached the lowest point in the valley along that direction. Call this point x_1. Then turn and ski in the direction $-\nabla F(x_1)$ until you hit the bottom of you straight trajectory in this new direction and then stop. Repeating this gets you obviously closer and closer to absolute bottom of the valley.

Now we formalize this picture. You starting trajectory is

$$x_0 + t d_0$$

where $d_0 = -\nabla F(x_0)$. Now, by a calculation using general properties of the dot product,

$$F(x_0 + t d_0) = F(x_0) + t d_0 \cdot (A x_0 - b) + \frac{t^2}{2} d_0 \cdot A d_0 \ .$$

Note that $d_0 = -(A x_0 - b)$ and hence

$$F(x_0 + t d_0) = F(x_0) - t |d_0|^2 + \frac{t^2}{2} d_0 \cdot A d_0 \ .$$

According to our plan, we have to find the minimum of this function with respect to the variable t which is attained at

$$t_0 = \frac{|d_0|^2}{d_0 \cdot A d_0} \ .$$

Thus the first stopping point is at

$$x_1 = x_0 + \frac{|d_0|^2}{d_0 \cdot A d_0} d_0 \ .$$

Repeating the same argument we get the next stopping point at

$$x_2 = x_1 + \frac{|d_1|^2}{d_1 \cdot A d_1} d_1$$

where

$$d_1 = -\nabla F(x_1) = -(Ax_1 - b) \ .$$

After the k-th step we have

$$x_k = x_{k-1} + \frac{|d_{k-1}|^2}{d_{k-1} \cdot Ad_{k-1}} d_{k-1} \tag{$SD1$}$$

where

$$d_{k-1} = -(Ax_{k-1} - b) \ . \tag{$SD2$}$$

With every step we get closer to the minimum of the function F.

Notice that this procedure is not exact in the sense that it does not stop after a finite number of steps. The reason is that in an ellipsoid the gradient direction does not in general pass through the center of the ellipse. Nevertheless, it is clear that we keep 'descending' towards the minimum although we may never may reach it. Thus, we have to specify a stopping rule, namely at what accuracy should we stop.

The smallness of $|d_k|$ is certainly a measure for how accurately the vector x_k is a solution of the equation. It measures how well x_k solves the equation.

To get an estimate how far away x_k is from the actual solution we note that

$$|d_k| = |(Ay_0 - b) - (Ax_k - b)| = |A(y_0 - x_k)| \geq \min_k \lambda_k |y - x_k|$$

where λ_k are the eigenvalues of A. Hence

$$|y_0 - x_k| \leq \frac{|d_k|}{\min_k \lambda_k} \ .$$

In order to profit from this estimate one has to know, approximately, the smallest eigenvalue, which might be not so easy to get. In prac

Nevertheless, we have now a method for computing the solution of this system of linear equations via an iterative method which can be summarized as follows:

Steepest descent algorithm:

Prescribe an accuracy ε. Start with any vector x_0. Compute

$$x_k = x_{k-1} + \frac{|d_{k-1}|^2}{d_{k-1} \cdot Ad_{k-1}} d_{k-1} \tag{$SD1$}$$

where

$$d_{k-1} = -(Ax_{k-1} - b) \ . \tag{$SD2$}$$

If $|d_k| \geq \varepsilon$ repeat again (SD1) and (SD2). If $|d_k| < \varepsilon$ stop.

Nice, as it looks, there are some limitations to this method. If the level surfaces of the function $F(x)$, which are ellipsoids, are very elongated, the method can be very slow. Recall that an elongated ellipsoid means that the ratio of the largest and the smallest eigenvalue is very large. Thus, it may happen that when starting on the shallow end of the ellipsoid, one makes always very small steps. Thus it may take a fairly long time until you come close to the bottom of the valley. For this algorithm to work we need, generally speaking a good condition number, i.e., the ratio of the largest to the smallest eigenvalue should not be too big.

Problems

1: Implement (i.e., write a program) the steepest descent algorithm and apply it first to solve simple problems such as

$$\begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1.001 & -0.999 \\ -0.999 & 1.001 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Use an accuracy of 10^{-5}. Draw a qualitative picture of the level curves of the corresponding function F. Based on that, use various starting points x_0 and describe what you observe. List the number of steps it takes for the various starting points.

2: Pick randomly five 10×10 positive definite matrices A and vectors b with integer coefficients and solve the equation

$$A\mathbf{x} = \mathbf{b}$$

Use an accuracy of 10^{-5}. Check your answers.

To generate these matrices proceed as follows. Pick randomly a 10×10 matrix B with integer coefficients and compute $A = B^T B$. This matrix is very likely to be positive definite. Randomly chosen matrices are unlikely to have nontrivial kernel.