Radu Mihai-Emilian, Group B1, Year 3
Faculty of Computer Science Iasi
Conf. Dr. Ignat Anca, Numerical Calculus

Solving Ill Conditioned Linear Systems of Equations

1. Introduction

An ill-conditioned linear system of equations is a system in which a small change in the independent variable has a big impact on the dependent variable. In the field of numerical analysis this propriety is defined by the condition number function. This function analyses how big the impact of a change in the input affects the output. Ill-conditioned systems are identified by having a high condition number, while well-conditioned systems are characterised by a value closer to 1.

To illustrate how drastic a small change of the input matrix results into a big change of the output I am going to consider the following system of linear equations:

$$\begin{bmatrix} 400 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}.$$

This system can be solved by $x_1 = -100$ and $x_2 = -200$, but changing A_{11} from 400 to 401 results in the following result:

$$\begin{bmatrix} 401 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}.$$

This time the solution is $x_1 = 40000$ and $x_2 = 79800$.

2. Condition number

The condition number is an application of the derivate and is frequently applied to question linear algebra. This number is paired with a number of algorithms to find the solution of the system but as the condition number increases the correct answer becomes hard to calculate. As a rule, if the condition number is $\kappa(A) = 10^k$, the error in the expected result is going to have k-lost digits of accuracy on top of the precision of the arithmetic methods. The condition number is calculated by the formula:

$$\kappa(A) = ||A|| \times ||A^{-1}||.$$

3. Preconditioning

Preconditioning is the application of a transformation, called the preconditioner, that conditions a given problem into a form that is more suitable for numerical solving problems.

Solving an ill-conditioned system is not an easy task most algorithms requiring a lot of pre-work in order to find a correct solution. There is not a fixed number of algorithms when talking about solving these kind of systems but the general thought is to go with preconditioning to make the system well-conditioned, thereby reducing the condition number of the matrix, then apply the conjugate gradient algorithm. It is also a good practice to find what kind of ill-condition system it is - as in if it is rank-deficient or ill-posed before going head-on into solving the problem.

There are a few choices when talking about preconditioners like:

- Jacobi (or diagonal) precontioner - in which the preconditioner is chosen to be the diagonal of the matrix
- Symmetric successive over-relaxation preconditioner defined as $M = (D + L)D^{-1}(D + L)$, where D is the diagonal, L is the lower triangular and L^{-1} is the upper triangular.

The unfortunate aspect is that there are books on preconditioning alone and finding a good preconditioner for the system we are trying to solve can become overwhelming because of how large the solution space is. For this research I am going to make an assumption that the matrix we are trying to solve isn't ill-posed or rank-deficient and can be solved by preconditioning.

4. Solving algorithm

The **conjugate gradient method** is the algorithm used for symmetric and positive-definite ill-conditioned systems of equations. It can be used as a direct method or an iterative algorithm, but it is often implemented as the latter, applicable to sparse systems that are too large to be handled by direct implementation.

The algorithm is computationally expensive because it requires searching directions of all the previous iterations and residue vectors, as well as matrix-vector multiplications. In most cases, preconditiong is necessary to ensure fast convergence of the conjugate gradient method. The algorithm takes the following form:

```
$r_0 = b - Ax_0$
$z_0 = M^{-1}r_0$
$p_0 = z_0$
$k = 0$
**repeat**
        $\alpha_k = \frac{r_k^T z_k}{p_k^T A p_k}$
        $x_{k+1} = x_k + \alpha_k p_k$
        $r_{k+1} = r_k - \alpha_k A p_k$
        **if** $r_{k+1}$ is sufficiently small **then** exit loop **end if**
        $z_{k+1} = M^{-1} r_{k+1}$
        $\beta_k = \frac{r_{k+1}^T z_{k+1}}{r_k^T z_k}$
        $p_{k+1} = z_{k+1} + \beta_k p_k$
        $k = k + 1$
**end repeat**
```

- x_{k+1} is the result
- x_0 is 0.
- M has to be symmetric, positive-definite and fixed.

Conclusions

Using the preconditioned conjugate gradient method works only in specific cases (the matrix has to be symmetric and positive) and that shows how complex the field of solving ill-conditioned linear systems can be. There are a lot of alternative proposed algorithms that are more complex or don't require as much compute power, but there does not exist a solve-all solution in mathematics.

Bibliography

(1) Yuka Kobayashi and Takeshi Ogita, ***Accurate and efficient algorithm for solving ill-conditioned linear systems by preconditioning methods***, July 1, 2016 [https://www.jstage.jst.go.jp/article/nolta/7/3/7_374/_pdf]

(2) Fazlollah Soleymani, ***A new method for solving ill-conditioned linear systems***, 13.03.2012 [https://core.ac.uk/download/pdf/26121748.pdf]

(3) Jonathan Richard Shewchuck, ***An Introduction to the Conjugate Gradient Method Without the Agonizing Pain,*** August 4, 1994 [https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf]

(4) University of Saskatchewan, ***Ill-conditioned Systems*** [http://engrwww.usask.ca/classes/EE/840/notes/ILL_Conditioned%20Systems.pdf]