Assignment 3

Haozhe Su

12/06/2018

1 Conjugate Gradient

1.1 Introduction

In mathematics, the conjugate gradient method is an algorithm for the numerical solution of particular systems of linear equations, namely those whose matrix is symmetric and positive-definite. The conjugate gradient method is often implemented as an iterative algorithm, applicable to sparse systems that are too large to be handled by a direct implementation or other direct methods such as the Cholesky decomposition. Large sparse systems often arise when numerically solving partial differential equations or optimization problems.

The conjugate gradient method can also be used to solve unconstrained optimization problems such as energy minimization. It was mainly developed by Magnus Hestenes and Eduard Stiefel.

The biconjugate gradient method provides a generalization to non-symmetric matrices. Various nonlinear conjugate gradient methods seek minima of nonlinear equations.

1.2 Algorithm

Algorithms can be included using the commands as shown in algorithm 1.

Algorithm 1 Conjugate Gradient

```
1:  $\mathbf{r_0} := \mathbf{b} - \mathbf{Ax_0}$
2:  $\mathbf{z_0} := \mathbf{M^{-1}r_0}$
3:  $\mathbf{p_0} = \mathbf{z_0}$
4:  $k := 0$
5:  repeat
6:      $\alpha_k := \frac{\mathbf{r_k^T}}{\mathbf{p_k^T Ap_k}}$
7:      $\mathbf{x_{k+1}} := \mathbf{x_k} + \alpha_k \mathbf{p_k}$
8:      $\mathbf{r_{k+1}} :=: \mathbf{r_k} - \alpha_k \mathbf{Ap_k}$
9:      if $\mathbf{r_{k+1}}$ then exit loop
10:     $\mathbf{z_{k+1}} := \mathbf{M^{-1}r_{k+1}}$
11:     $\beta := \frac{\mathbf{z_{k+1}^T r_{k+1}}}{\mathbf{z_k^T r_k}}$
12:     $\mathbf{p_{k+1}} := \mathbf{z_{k+1}} + \beta_k \mathbf{p_k}$
13:     $k := k + 1$
14: until exit
15: the result is $\mathbf{x_{k+1}}$
```

1.3 Result

We run tests on different grids($16 \times 16, 32 \times 32, 64 \times 64, 128 \times 128, 256 \times 256$), and set the tolerance to be 10^{-6} and get the result shown below:

grid	iterations	r ($\times 10^{-7}$)	error$_x$($\times 10^{-3}$)
16×16	10	1.30×10^{-8}	9.43
32×32	24	3.90	6.82
64×64	49	5.92	3.33
128×128	95	8.23	2.05
256×256	193	6.54	1.12

1.4 Conclusion

We can see from the result:

$$\begin{gathered} \frac{9.43}{6.82} = 1.38 \\ \frac{6.82}{3.33} = 2.05 \\ \frac{3.33}{2.05} = 1,62 \\ \frac{2.05}{1.12} = 1.83 \end{gathered} \tag{1}$$

As we double the size of the grid, the error will decrease by roughly 50%, which is predicted by the algorithm.