Scientific Computing - Homework 1

蔡子若 018071910011

1 Page 45, 1.7 (Scientific Computing: An Introductory Survey)

(a) The accurate value of sec(1) is approximately 3.42551.

For $h = 10^{-k}, k = 0, \cdots, 16$, the absolute error of the finite difference method first decreases as k increases. After the absolute error achieves its minimum when $k = 8$, it increases as k increases.

This agrees with $\sqrt{\epsilon_{mach}} = 1.49012 \times 10^{-8}$.

(b) For central difference method, the curve of error is similar but achieves its minimum when $k = 7$, a little earlier than the finite difference method.

The following figure shows the variation of log(absolute error) and log(h).

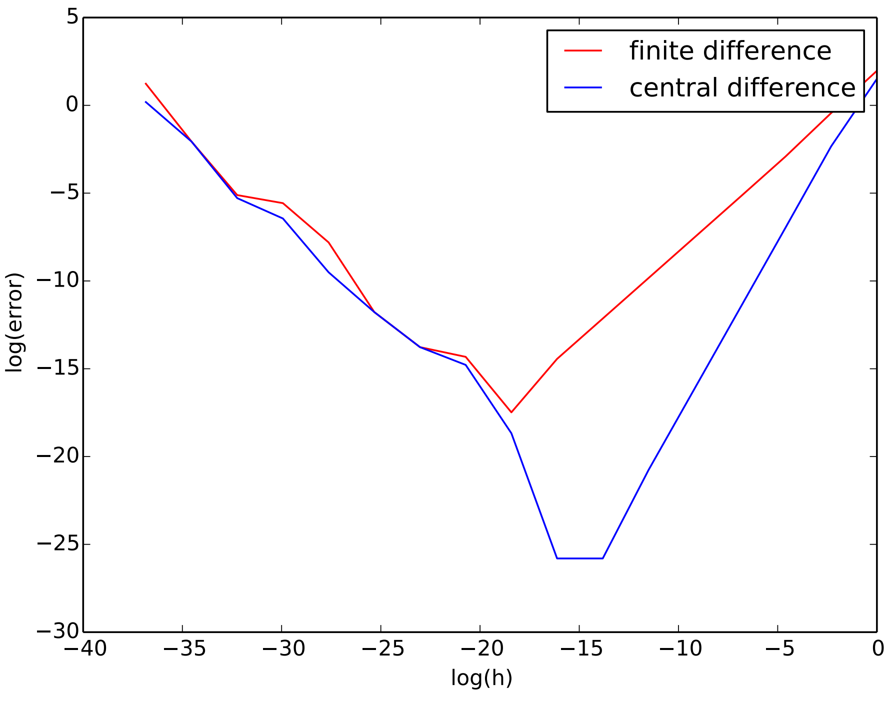

Figure 1: Finite Difference Error and Central Difference Error

2 Page 46, 1.9 (Scientific Computing: An Introductory Survey)

(b) Since the taylor expansion has the formulation $e^x = \sum_{i=0}^{+\infty} \frac{x^i}{i!}$,

I set up the stopping criterion as $\left|\frac{x^i}{i!}\right| < \epsilon$, where $\epsilon = 10^{-9}$.

(c) For $x = 1,\ 5,\ 10,\ 15,\ 20$, the taylor expansion results are similar to the built-in function $exp(x)$.

Both the absolute error and the relative error are satisfyingly small if I set ϵ small enough.

For $x = -1, -5, -10, -15, -20$, the relative error is quite bigger than above.

When $x = -20$, the relative error is always bigger than 1 even when $\epsilon < 10^{-20}$. In this case summing in the natural order is no longer useful.

(d) For $x < 0$, I modified the program as $e^{-x} = \dfrac{1}{e^x}$ and find out when $x = -1, -5, -10, -15, -20$, the absolute error and relative error are both satisfyingly small (smaller than 10^{-10}).

(e) I think not. However I rearrange the terms, the results will be still closed to 0 when $x \ll 0$.

In this case rearranging the terms can not alleviate the rounding error.

3 Matlab test

For $n = 500, 1000, 2000, 4000, 8000$, generate an $n * n$ random matrix, B, and an $n * 1$ vector, b. Find the symmetric matrix $A = B' * B$.

1) Using function 'eig' to test the time cost for eigen decomposition;

2) Using $x = A/b$ and $x = A^{-1} * b$ to test the time cost in finding the solution of $Ax = b$;

3) Plot all the time costs as a function of n and the power law for the time costs.

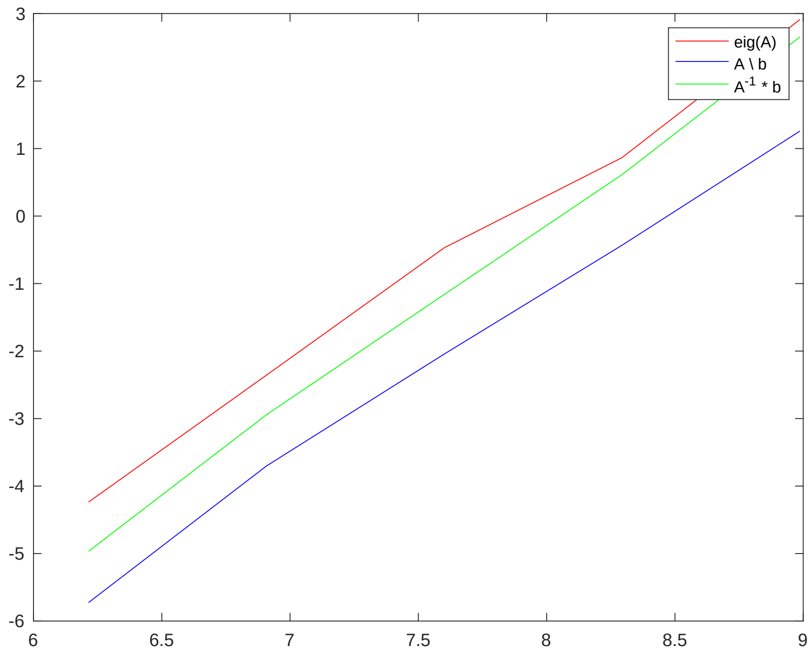

Figure 2: Matlab test

To solve the equation $Ax = b$, computing $x = A^{-1} * b$ is much slower than computing $x = A\backslash b$.

In the above figure, the slopes of the three lines are all approximately 3, which means the time cost of $eig(A)$, $A\backslash b$ and $A^{-1} * b$ are all proportional to n^3.