Nicholas Garrett

Dr. Zhu

MATH 4441

12/15/2021

Final Exam

1.a

If $a \approx b$, then the difficulty would be in trying to calculate for $a + b \approx 2a = 1 = 0$, which is impossible to solve for. Also, as $a \approx b$ we can get that $x = \frac{a}{a^2-b^2}$ and $y = \frac{-b}{a^2-b^2}$. These two approximations lead to $a^2 - b^2$ being close to zero, which leads to x and y overflowing.

1.b

A numerically stable formula for calculating this would be, since from the matrix we know that $ax + by = 1$ and $bx + ay = 0$, we can rewrite these equations to state: $z = \frac{a}{(a-b)(a+b)} + \frac{b}{(b-a)(a+b)}$, and through rewriting this fraction and simplifying, we get $z = \frac{1}{a+b}$.

So, $z = \frac{1}{a+b}$ is a numerically stable way of caluclating z = x+y.

2.a

The code can be found in "Problem2a.m"

The return made by my code was:

```
I(5) = 1.455329e-01
I(10) = 8.387707e-02
I(20) = -3.019239e+01
I(40) = -1.014081e+31
I(80) = -8.895192e+101
I(100) = -1.159929e+141
```

Obviously, these approximations are far from the expected trend of $I \Rightarrow 0$ as $n \Rightarrow \infty$, so I checked and rechecked my code. I couldn't find a bug by my fifth pass through the code, so I am hoping that I did it correctly and this result is expected. I would assume that this error is caused by roundoff error between iterations.

2.b

A better way to do this would be: to remove the sutraction–which removes the cancellation error.

This code can be found in the file: "Problem2b.m".

3.a.i

For these values, the result will be $x = \frac{10^6 \pm \sqrt{10^{12}-4}}{2}$. The problem is that 10^6 and $\sqrt{10^{12}-4}$ are very close to the same value.

In this instance, we can fix the cancellation error by mulitiplying the problematic case, where $x = \frac{-b-\sqrt{b^2-4ac}}{2a}$, and multiplying it by $\frac{-b+sqrtb^2-4ac}{-b+sqrtb^2-4ac}$, yielding: $\frac{4ac}{2ab+2a\sqrt{b^2-4ac}}$, which removes the cancellation error.

So, it would then be x = $\frac{-4c}{2b+2\sqrt{b^2-4ac}}$ or $\frac{-b+\sqrt{b^2-4ac}}{2a}$

3.a.ii

For these values, the resulting formula is $x = \frac{10^{20} \pm \sqrt{10^{40}-4*10^{-20}*10^{20}}}{2*10^{-20}}$. One of the problems for this is that a and c are equal, so -4ac = $-4 * 10^{-20} * 10^{20}$= -4, by a catastrophic cancellation. Further, because of this cancellation, we have a cancellation error, namely that $10^{20} \approx \sqrt{10^{40}-4}$.

Then, because we are left with a number devided by another number close to zero, the result is very large.

Unfortunately, we cannot use the same technique as in 3.a.i to fix this cancellation error, as even factoring out the a, there still remains a very small number (b) in the denominator.

3.b

My code can be found in "Problem3b.m" and is preset to run the values described further in this problem. For randomly generated values for a, b, and c, my algorithm returns the same result as MATLAB's roots() function.

I have it running a few instances for initial values:

Once for a, b, c being randomly generated. In this sistuation, my algorithm returns the same, or close to the same thing as MATLAB's roots()

Again for a=0 and b,c being randomly generated, my algorithm similarly returns the same or close to the same things as MATLAB's roots() function. Though for the x2, it only returns -inf. In this example, the quadratic formula would only return +- inf.

Again for a = 1, b = -10^6, and c = 1, my algorithm returns what the roots() function returns. With the roots() function returning $1 * 10^6$, and my algorithm returning 9.999924e+0.

And again for $a = 10^-20$, $b = -10^{20}$, and $c = 10^{20}$. In this instance, my algorithm fails and returns a value completely off from MATLAB's roots() function. I belive this is due to overflow from the large b in the denominator and cancelation between 2b and $2 * sqrtb^2 - 4ac$

4

The code I used can be found in "Problem4.m"

Included in the script is a call for initual guesses x0 = 1, and x0 = 0.1.

Unfortunately, this script does not return the correct values for the approximation because of the iteration function.

5

$||x||_A = \sqrt{x^T Ax}$ can be proved to be a vector norm by:

i. Because A is a symmetric positive definite matrix, all the values in the matrix are positive. Therefore, because the matrix is right multiplied by x and left multiplied by x^T, regardless of what vector x is, $||x||_A >= 0$. Further, because there are no additions being made to the norm and the matrix is multiplied by x, when x = 0, $||x||_A = 0$.

ii. The norm takes the square root of $x^T * A * x$. If we take x = αx, then we would get $||\alpha x|| = \sqrt{(\alpha x)^T A \alpha x}$, which is $\sqrt{\alpha * x^T * A * \alpha * x}$, which by factoring the α's together, we get: $\sqrt{\alpha^2 * x^T * A * x}$. By taking the square root of α^2 from in the square root, we get: $|\alpha| * \sqrt{x^T Ax}$.
So, $||\alpha x||_A = |\alpha| * ||x||_A$.

iii. If we set x = (x + y), then $||x||_A = \sqrt{(x+y)^T A(x+y)}$

6.a

The code for this problem can be found in "Problem6.m"

6.b

The flop count for this algorithm is: (n-1 * (n/2 * (1 + (n/2 * (4))))) $= n^2/2 + (2/)n^3 - n/2 - n^2$ $= \frac{2}{3}n^3 - \frac{n^2}{2} - \frac{n}{2}$

The flop count to calculate Ax=b is simply the flop count for the algorithm + the flop count for reverse elimination ($2n^2 - n$), so: $\frac{2}{3}n^3 + \frac{3n^2}{2} - \frac{3n}{2}$

7.a

For the Jacobi iteration method to converge, or any iteration method for that matter, $\rho(B) < 1$. As spectral radius is the maximum of the set of eigenvalues, $\rho(B) = max|1 - \alpha\lambda|$. And the eigenvalues are $\lambda : det(A - \lambda I) = 0$. Additionally, there must be no zeros along diagonal, though that is independant of a,b in this example.

7.b spectral radius ¡ 1, and diagonally dominant, non-zero diagonals.

8.a.
If A_0 is symmetric and positive definite, then the Cholesky factorization, which yields both a lower and lower-transpose matrix. Because it is an iterative funciton, and all the terms are similar to one another in the sequence, each term in the iterative sequence is symmetric and positive definite.

8.b

9.a

$$A = \begin{bmatrix} X & 7/10 & 4/9 & 3/9 & 2/5 \\ 3/10 & X & 4/6 & 3/6 & 4/7 \\ 5/9 & 2/6 & X & 6/9 & 4/8 \\ 6/9 & 3/6 & 3/9 & X & 2/6 \\ 3/5 & 3/7 & 4/8 & 4/6 & X \end{bmatrix}$$

9.b
The code can be found in file "Problem9b.m"
Return made by my code:

```
largest eigenvalue: 1.993257e+00
 it's eigenvector:
x =

    0.4275
    0.4576
    0.4544
    0.4140
    0.4795
```

.

9.c
The ranking for this team then, taking the values and running them through the summation, is 0.3225.