Numerical Methods in Scientific Computing 2021
Answers Ex02

Jake Muff 29/01/21

Problem 1

Show that for a vector $\vec{x}$ of length n

$$\lim p \to \infty \Big[\sum_{i=1}^{n} |x_i|^p \Big]^{\frac{1}{p}} = \max_{i \leq i \leq n}(|x_i|)$$

To show this lets denote x_j to be the largest element in the vector so that

$$\lim p \to \infty \Big[\sum_{i=1}^{n} |x_i|^p \Big]^{\frac{1}{p}} = \lim_{p \to \infty} \Big[|x_1|^p + |x_2|^p + \ldots |x_j|^p + \ldots |x_n|^p \Big]^{\frac{1}{p}}$$

If we factor x_j out

$$= x_j \Big[\lim_{p \to \infty} \Big[\Big| \frac{x_1}{x_j} \Big|^p + \Big| \frac{x_2}{x_j} \Big|^p + \ldots 1 + \ldots \Big| \frac{x_n}{x_j} \Big|^p \Big]^{\frac{1}{p}} \Big]$$

Because x_j is defined as the largest vector,

$$\frac{x_i}{x_j} < 1$$

Thus

$$\Big| \frac{x_i}{x_j} \Big|^p \to 0 \text{ as } p \to \infty$$

This means that

$$\Big[\Big| \frac{x_1}{x_j} \Big|^p + \Big| \frac{x_2}{x_j} \Big|^p + \ldots 1 + \ldots \Big| \frac{x_n}{x_j} \Big|^p \Big]^{\frac{1}{p}} \to 0^0 \text{ as } p \to \infty \tag{1}$$

Mostly 0^0 is defined as 1 but it can be undefined/indeterminant and the limit can be found using L'Hôpital's rule, which uses the derivative to evaluate the limit. So, for example, we have

$$a = \Big(\frac{1}{b} \Big)^{\frac{1}{b}}$$

$$\lim_{b \to \infty} a = 0^0$$

To use L'Hôpital's rule we need it in the form $\frac{f(b)}{g(b)}$. Thus,

$$\ln a = \frac{1}{b} \ln \frac{1}{b} = -\frac{\ln b}{b}$$

$$\lim_{b\to\infty} \ln a = -\lim_{b\to\infty} \frac{\ln b}{b} = \frac{f(b)}{g(b)}$$

$$\frac{f'(b)}{g'(b)} = \frac{\frac{1}{b}}{1}$$

$$-\lim_{b\to\infty} \frac{\ln b}{b} = -\lim_{b\to\infty} \frac{\frac{1}{b}}{1}$$

Which can be evaluated

$$\lim_{b\to\infty} \ln a = 0$$

Such that

$$\lim_{b\to\infty} a = e^{\lim_{b\to\infty} \ln a} = 1$$

Thus

$$\lim_{b\to\infty} \left(\frac{1}{b}\right)^{\frac{1}{b}} = 1$$

Applying this to equation (1)

$$\lim_{p\to\infty} \left[\left|\frac{x_1}{x_j}\right|^p + \left|\frac{x_2}{x_j}\right|^p + \ldots 1 + \ldots \left|\frac{x_n}{x_j}\right|^p\right]^{\frac{1}{p}} = 1$$

Thus

$$\lim p \to \infty \Big[\sum_{i=1}^{n} |x_i|^p\Big]^{\frac{1}{p}} = x_j \cdot \Big[\lim_{p\to\infty} \left[\left|\frac{x_1}{x_j}\right|^p + \left|\frac{x_2}{x_j}\right|^p + \ldots 1 + \ldots \left|\frac{x_n}{x_j}\right|^p\right]^{\frac{1}{p}}\Big]$$

$$= x_j \cdot 1$$

$$= x_j$$

And x_j is defined as the max element, so,

$$= \max_{i\leq i\leq n}(|x_i|)$$

Problem 2

In this problem I had to implement the Kahan summation algorithm into the previous weeks work. The source file for this question is found under harmonc_kahan.cpp. The results are shown below. The function correctly returns the sum of the first N terms. Compared to the previous week it is interesting to note that at the same value (2097152) the function does not converge to a single finite value. Instead, as shown it prints finite values for numbers significantly higher than what was used in the previous week but still not infinite. This tells me that this algorithm is more precise and 'better' than the previous weeks but still not ideal as it converges to a finite value.

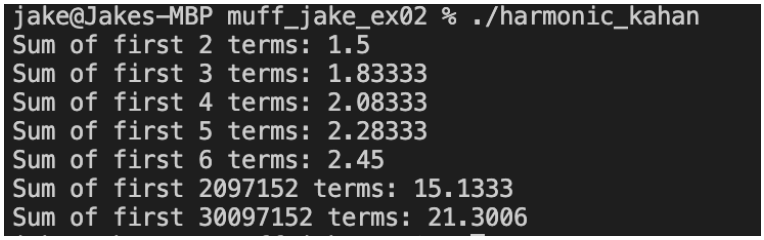

Figure 1: Output from harmonic_kahan.cpp

Problem 3

Despite being a seemingly simple problem, it took a surprising amount of time due to trying to get LAPACK to work on my computer. For this I used LAPACK on OSX Catalina 31/1/21 Since OSX Mavericks there is a system install version of LAPACK and BLAS that is optimized for use on OSX devices but often requires use of XCODE or fiddling about with default path locations. Another method is to use Homebrew with

brew install lapack (or Lapack)

Which installs LAPACK as well as BLAS and dependencies. To use this in compilation requires addition of −L/Path/To/Directory which is usually in /usr/local/lib In the end I used A.Kuronen version of LAPACK from SC II example Progs page which requires compiling using gfortran after you have read the README file attached

gfortran −o ex2_p3 ex2_p3.c −llapack

NOTE: There is a implicit declaration warning due to calling the LAPACK function but not declaring in a header file as requires in C99. Can still rule the executable. This method just outdated imo but still works.
NOTE: The lineardouble and eigendouble files in the .tar from Example Progs Page are terribly commented and coded for use of Matrices and general linear algebra, therefore, I edited them by adding in input from file and printf statements

For Part B, "edit matrix6 in such a way that it becomes singular" I did not fully understand. A singular matrix has determinant 0 and is therefore non-invertible. To make matrix6 singular, the easiest way would just to make matrix6 entirely 0's, however I do not think this is what you mean in this question. I am not quite sure by what method you wish us to make it singular. The closest idea I came up with was to REF the matrix and change the diagonal value to a 0 thereby making the determinant 0.

```
jake@Jakes-MBP muff_jake_ex02 % ./ex2_p3
The rank is = 6
Reading the matrix A...
Matrix A read..
Reading Matrix b..
Matrix b read..
A
  0.61611086   0.7106747   0.32931846   0.51194823   0.25304845   0.54417247
  0.03563731   0.08815959   0.30034593   0.14795998   0.46325326   0.27647871
  0.28538886   0.71970248   0.50220811   0.61753166   0.96837115   0.87984169
  0.37600851   0.74768132   0.44811806   0.17185755   0.64607608   0.11992062
  0.32336015   0.29026943   0.00099506   0.19949169    0.2208458    0.9813996
  0.97817445   0.31981957   0.94624835   0.63531208   0.53851491   0.72695738

b
  0.66863006   0.94118035    0.7253744   0.54309487   0.21732061    0.4329184
x
  -22.413899    12.990382    29.555568   -9.3848441   -21.079137    10.385575
```

Figure 2: Output from ex2_p3.c, solving matrix6 for the vector x.

```
x
   7.2144883    11.163655    3.0358445    1.0799059    1.8689894    4.7119332   -8.2343102    5.5900076   -4.19034
71   -3.5101576    10.645714   -4.0814602   -8.6241006  -0.63710851    2.7890658    8.7764311   -4.1028957   -4.77
85748   -4.4614943   -11.49861   -11.963623      11.0754     5.669707    3.5490224  0.058301116   -5.0599416   -1
.9754759   -1.253867   -3.8985944    8.2744529   -5.6839025   -5.669198   0.37960791   -4.1624764    10.597685
-0.34778509   -1.6239494   -3.8423147   -4.832245    0.6290036   -2.608674    6.8657889   -8.9785345    4.670336
4    0.5489192   -8.3940974   -5.2656037    5.1105622     2.750081    7.5109883    5.4796523    9.1413935    2.064
5985   -3.7819297   -3.8121316   -1.5216914    10.472063    10.756644   -12.808315   -3.2649371    4.2395374   -8.
4943304   -3.9817579   -5.1799615    2.5102437   -1.9911264  0.023276902    4.8850823  -0.69939738     3.110351
  0.941317   0.10284713    4.3865474    5.2776985     4.295356   -2.3233743    15.556085   -4.8151092    6.9185871
   -8.0440978  -0.75193463    2.3261727   -1.4131884   -1.7420666    3.6335352   -13.360496    6.8480197   -7.3610
078   -2.3170964    3.3144431   -1.6072032   -3.5258292    4.2756271   -6.1851535   -5.3759474   -8.5609332   -3.9
181126  -0.66337326    8.2351565    8.6265645
```

Figure 3: Output from ex2_p3.c, solving matrix100 for the vector x. Obviously due to the size of the matrix, the output is messy but the solution should be correct.

Problem 4

For this problem I first copied over the problem 3 code into the function to get the vector $\vec{x}$. Then I used the LAPACK subroutine dgemm to multiple the Matrix A and the vector $\vec{x}$, the subroutine also allows for $\vec{b}$ to be taken away. The only problem arose when doing this was the fact that because the subroutine is written in fortran it only takes pointers as inputs. I then calculated the norm below it by using an IF statement as well as FOR loops.
I implemented

$$\Big(\sum_{i=1}^{n} |x_i|^m\Big)^{\frac{1}{m}}$$

For $m = 1, 2 \ldots$. For $m = 0$ in the loop, the IF statement calls it and calculates the max absolute value in the vector according to the infinity norm.

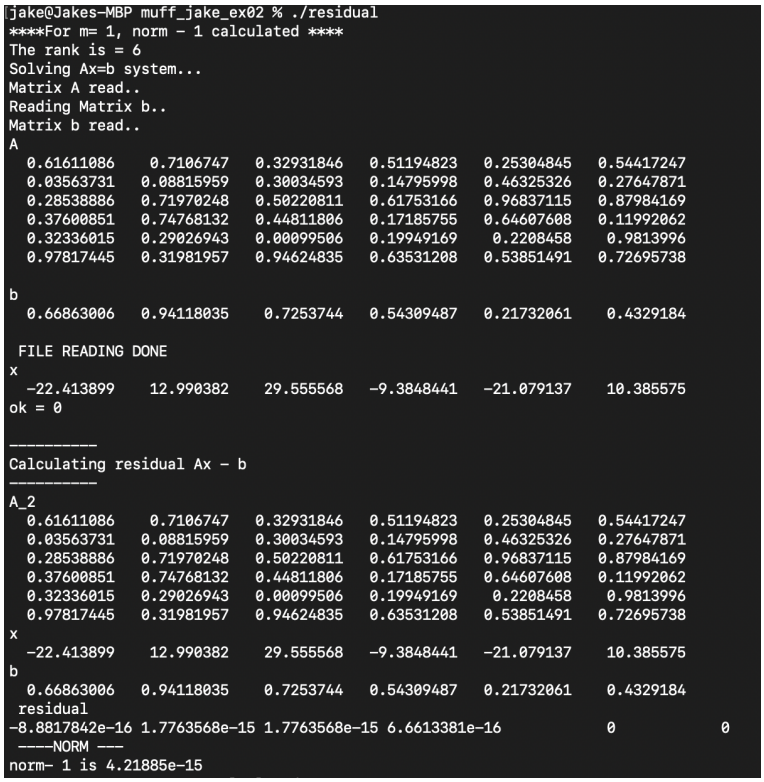

Figure 4: Output from residual.c for $m = 1$.

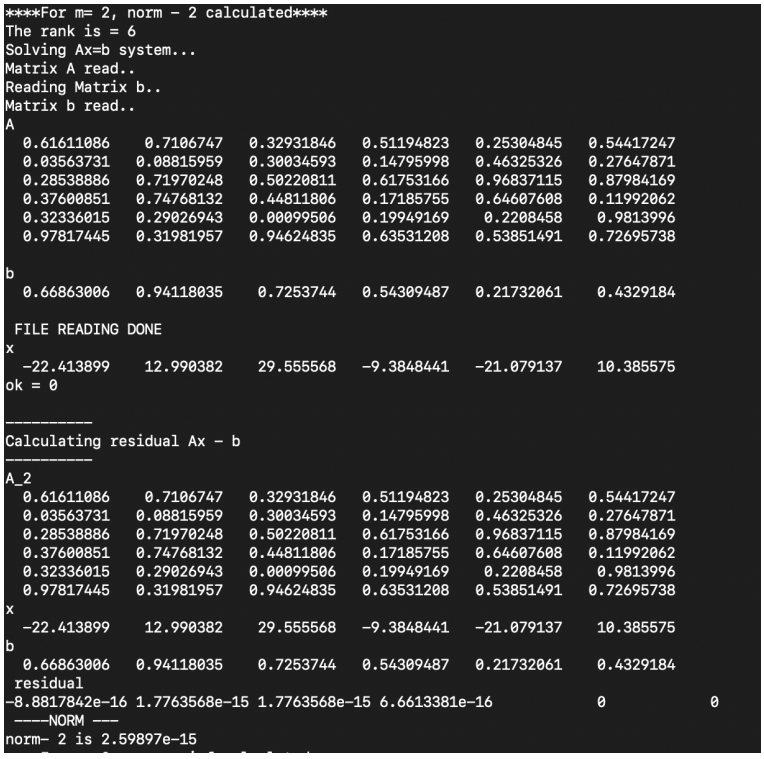

Figure 5: Output from residual.c for $m = 2$.

```
****For m= 0, norm - inf calculated****
The rank is = 6
Solving Ax=b system...
Matrix A read..
Reading Matrix b..
Matrix b read..
A
  0.61611086    0.7106747   0.32931846   0.51194823   0.25304845   0.54417247
  0.03563731   0.08815959   0.30034593   0.14795998   0.46325326   0.27647871
  0.28538886   0.71970248   0.50220811   0.61753166   0.96837115   0.87984169
  0.37600851   0.74768132   0.44811806   0.17185755   0.64607608   0.11992062
  0.32336015   0.29026943   0.00099506   0.19949169    0.2208458    0.9813996
  0.97817445   0.31981957   0.94624835   0.63531208   0.53851491   0.72695738

b
  0.66863006   0.94118035    0.7253744   0.54309487   0.21732061    0.4329184

 FILE READING DONE
x
  -22.413899    12.990382    29.555568   -9.3848441   -21.079137    10.385575
ok = 0

----------
Calculating residual Ax - b
----------
A_2
  0.61611086    0.7106747   0.32931846   0.51194823   0.25304845   0.54417247
  0.03563731   0.08815959   0.30034593   0.14795998   0.46325326   0.27647871
  0.28538886   0.71970248   0.50220811   0.61753166   0.96837115   0.87984169
  0.37600851   0.74768132   0.44811806   0.17185755   0.64607608   0.11992062
  0.32336015   0.29026943   0.00099506   0.19949169    0.2208458    0.9813996
  0.97817445   0.31981957   0.94624835   0.63531208   0.53851491   0.72695738
x
  -22.413899    12.990382    29.555568   -9.3848441   -21.079137    10.385575
b
  0.66863006   0.94118035    0.7253744   0.54309487   0.21732061    0.4329184
 residual
-8.8817842e-16 1.7763568e-15 1.7763568e-15 6.6613381e-16           0           0
 ----NORM ---
Max element is 1.77636e-15
```

Figure 6: Output from residual.c for $m = \infty$.