

(1) Let's consider vector $\bar{x} = [x_1, x_2, \dots, x_n]$ with single biggest element x_j . It is clear that if $x_j > x_i$ for every $i \neq j$ and $i \leq n$ then $\lim_{p \rightarrow \infty} x_j^p \gg \lim_{p \rightarrow \infty} x_i^p$ and therefore $\lim_{p \rightarrow \infty} \sum_{i=1}^n x_i^p \approx \lim_{p \rightarrow \infty} x_j^p$ and $\lim_{p \rightarrow \infty} (\sum_{i=1}^n |x_i|^p)^{1/p} \approx \lim_{p \rightarrow \infty} (|x_j|^p)^{1/p} = |x_j| = \max(|x_i|)$.

If there happened to be $n \leq n$ elements with value x_j so that $x_j \geq x_i$ for every $i \leq n$. In that case $\lim_{p \rightarrow \infty} \sum_{i=1}^n x_i^p \approx n \cdot \lim_{p \rightarrow \infty} x_j^p$. Now taking the root yields $\lim_{p \rightarrow \infty} (\sum_{i=1}^n |x_i|^p)^{1/p} \approx \lim_{p \rightarrow \infty} (n |x_j|^p)^{1/p}$ where for large p $n^{1/p} \approx 1$ and therefore $\lim_{p \rightarrow \infty} (\sum_{i=1}^n |x_i|^p)^{1/p} \approx \lim_{p \rightarrow \infty} (|x_j|^p)^{1/p} = |x_j| = \max(|x_i|)$.

(2) (5) When setting $N = \text{INT_MAX}$ the value returned by harmonic_when is ≈ 0.647793 which is more than either of week 1 functions returned. This is due to the algorithm reducing the effect of numerical error when adding floats.

Ex. 3

I compiled LAPACK with lines 19-24 changed to following in make.inc because running tests caused segfault with -frecursive:

```
FORTRAN = gfortran
OPTS     = -O2
DRVOPTS  = $(OPTS)
NOOPT    = -O0
LOADER   = gfortran
LOADOPTS =
```

Then I compiled fortran version of the given program with "gfortran -o ex2_p3 ex2_p3.f90 -llapack -lblas" (flags to link lapack) and ran it first with "./ex2_p3 < matrix6" which yielded following x:

```
-22.413899      12.990382      29.555568      -9.3848441      -21.079137
10.385575
```

After that I gave it matrix100, vector x for which was following:

```
  7.2144883      11.163655      3.0358445      1.0799059      1.8689894
4.7119332      -8.2343102      5.5900076      -4.1903471      -3.5101576
10.645714      -4.0814602      -8.6241006      -0.63710851      2.7890658
8.7764311      -4.1028957      -4.7785748      -4.4614943      -11.498610
 -11.963623      11.075400      5.6697070      3.5490224
0.58301116E-01 -5.0599416      -1.9754759      -1.2538670      -3.8985944
8.2744529      -5.6839025      -5.6691980      0.37960791      -4.1624764
10.597685      -0.34778509      -1.6239494      -3.8423147      -4.8322450
0.62900360
 -2.6086740      6.8657889      -8.9785345      4.6703364      0.54891920
-8.3940974      -5.2656037      5.1105622      2.7500810      7.5109883
5.4796523      9.1413935      2.0645985      -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      0.23276902E-01  4.8850823      -0.69939738      3.1103510
0.94131700      0.10284713      4.3865474      5.2776985      4.2953560
-2.3233743      15.556085      -4.8151092      6.9185871      -8.0440978
 -0.75193463      2.3261727      -1.4131884      -1.7420666      3.6335352
-13.360496      6.8480197      -7.3610078      -2.3170964      3.3144431
-1.6072032      -3.5258292      4.2756271      -6.1851535      -5.3759474
-8.5609332      -3.9181126      -0.66337326      8.2351565      8.6265645
```

Ex. 4

Residuals:

m	matrix6	matrix100
0	1.952974e-07	1.003843e-06
1	5.412062e-07	2.496027e-05
2	2.742264e-07	3.031577e-06

I used dgemm for matrix multiplication. Makefile for compiling is attached. I did not manage to use LAPACK with C-standard newer than C89, is that how it just is or did I fail at googling?