H3

January 31, 2018

1 HW01

1.1 H3 problem

```
In [1]: import numpy as np
```

1.1.1 a) solution

Write a program to calculate Sup and Sdn in single precision

```
In [33]: ini_up = np.float32(0.0)

         result_dn = []
         result_up = []

         for p in range(2,9):
             up_lim = 10**p
             print("p is:", p)
                 ##    calculate the sum^{1}_{N} 1/n
             ini_dn = np.float32(0.0)
             for i in range(up_lim, 0, -1):
                 ini_dn = ini_dn + np.float32(1/i)
             result_dn.append(ini_dn)

                 ##    calculate the sum^{N}_{1} 1/n
             ini_up = np.float32(0.0)
             for i in range(1, up_lim+1):
                 ini_up = ini_up + np.float32(1/i)
             result_up.append(ini_up)

             print("ini_up is:", ini_up)
             print("ini_dn is:", ini_dn)
             print("--------------------")
         result_up_s = np.array(result_up)
         result_dn_s = np.array(result_dn)
         print("result_up", result_up_s)
         print("result_dn", result_dn_s)
         print('|S_up - S_dn| is:', np.abs(result_dn_s - result_up_s))
```

```
p is: 2
ini_up is: 5.18738
ini_dn is: 5.18738
--------------------
p is: 3
ini_up is: 7.48548
ini_dn is: 7.48547
--------------------
p is: 4
ini_up is: 9.78761
ini_dn is: 9.7876
--------------------
p is: 5
ini_up is: 12.0909
ini_dn is: 12.0902
--------------------
p is: 6
ini_up is: 14.3574
ini_dn is: 14.3927
--------------------
p is: 7
ini_up is: 15.4037
ini_dn is: 16.686
--------------------
p is: 8
ini_up is: 15.4037
ini_dn is: 18.8079
--------------------
result_up [  5.18737793   7.4854784    9.78761292  12.09085083  14.35735798
  15.40368271  15.40368271]
result_dn [  5.18737698   7.48547173   9.78760433  12.09015274  14.39265156
  16.68603134  18.80791855]
|S_up - S_dn| is: [  9.53674316e-07   6.67572021e-06   8.58306885e-06   6.98089600e-04
   3.52935791e-02   1.28234863e+00   3.40423584e+00]
```

1.1.2 b)

Write a program to calculate Sup and Sdn in double precision for N = 10p with p = 2, 3, ...8

```
In [34]: ini_up = np.float64(0.0)

         result_dn = []
         result_up = []

         for p in range(2,9):
             up_lim = 10**p
             print("p is:", p)
```

```python
            ##    calculate the sum^{1}_{N} 1/n
        ini_dn = np.float64(0.0)
        for i in range(up_lim, 0, -1):
            ini_dn = ini_dn + np.float64(1/i)
        result_dn.append(ini_dn)

            ##    calculate the sum^{N}_{1} 1/n
        ini_up = np.float64(0.0)
        for i in range(1, up_lim+1):
            ini_up = ini_up + np.float64(1/i)
        result_up.append(ini_up)

        print("ini_up is:", ini_up)
        print("ini_dn is:", ini_dn)
        print("--------------------")
    result_up_d = np.array(result_up)
    result_dn_d = np.array(result_dn)
    print("result_up", result_up_d)
    print("result_dn", result_dn_d)
    np.abs(result_dn_d - result_up_d)
```

```
p is: 2
ini_up is: 5.18737751764
ini_dn is: 5.18737751764
--------------------
p is: 3
ini_up is: 7.48547086055
ini_dn is: 7.48547086055
--------------------
p is: 4
ini_up is: 9.78760603604
ini_dn is: 9.78760603604
--------------------
p is: 5
ini_up is: 12.0901461299
ini_dn is: 12.0901461299
--------------------
p is: 6
ini_up is: 14.3927267229
ini_dn is: 14.3927267229
--------------------
p is: 7
ini_up is: 16.6953113659
ini_dn is: 16.6953113659
--------------------
p is: 8
ini_up is: 18.9978964139
ini_dn is: 18.9978964139
```

```
--------------------
result_up [  5.18737752   7.48547086   9.78760604  12.09014613  14.39272672
  16.69531137  18.99789641]
result_dn [  5.18737752   7.48547086   9.78760604  12.09014613  14.39272672
  16.69531137  18.99789641]
```

```
Out[34]: array([  8.88178420e-16,   2.66453526e-15,   3.73034936e-14,
                  7.28306304e-14,   7.83373366e-13,   2.69295697e-12,
                  8.91731133e-13])
```

1.1.3 c)

Assuming that the double-precision result x is exact, show that the single-precision Sup is less accurate than the single-precision Sdn, i.e., plot |Sup x| and |Sdn x| as a function of p

```
In [36]: print(np.abs(result_up_s - result_up_d))
         print(np.abs(result_dn_s - result_up_d))
```

```
[  4.12047879e-07   7.54063374e-06   6.87899471e-06   7.04700215e-04
   3.53687440e-02   1.29162866e+00   3.59421371e+00]
[  5.41626437e-07   8.64913524e-07   1.70407413e-06   6.61061518e-06
   7.51649426e-05   9.28002430e-03   1.89977865e-01]
```

1.1.4 d)

Plot |S ln(2)| vs p with p up to 9 for both single and double precision

```
In [39]: result_up_d = []
         result_up_s = []

         for p in range(2,9):
             up_lim = 10**p
             print("p is:", p)
                 ##    calculate the sum^{N}_{1} {(-1)^(n+1)_/n}  Double precision
             ini_up_d = np.float64(0.0)
             for i in range(1, up_lim+1):
                 ini_up_d = ini_up_d + np.float64((-1)**(i+1)/i)
             result_up_d.append(ini_up_d)

                 ##    calculate the sum^{N}_{1} {(-1)^(n+1)_/n} single precesion
             ini_up_s = np.float32(0.0)
             for i in range(1, up_lim+1):
                 ini_up_s = ini_up_s + np.float32((-1)**(i+1)/i)
             result_up_s.append(ini_up_s)

             print("ini_up_s is:", ini_up_s)
             print("ini_up_d is:", ini_up_d)
```

```python
            print("--------------------")
        result_up_d = np.array(result_up_d)
        result_up_s = np.array(result_up_s)
        print("result_up_d", result_up_d)
        print("result_up_s", result_up_s)
        print("Error of Double precesion :", np.abs(result_up_d - np.log(2)))
        print("Error of Single precesion :", np.abs(result_up_s - np.log(2)))
```

```
p is: 2
ini_up_s is: 0.688172
ini_up_d is: 0.68817217931
--------------------
p is: 3
ini_up_s is: 0.692646
ini_up_d is: 0.69264743056
--------------------
p is: 4
ini_up_s is: 0.693092
ini_up_d is: 0.69309718306
--------------------
p is: 5
ini_up_s is: 0.693134
ini_up_d is: 0.693142180585
--------------------
p is: 6
ini_up_s is: 0.693137
ini_up_d is: 0.69314668056
--------------------
p is: 7
ini_up_s is: 0.693137
ini_up_d is: 0.69314713056
--------------------
p is: 8
ini_up_s is: 0.693138
ini_up_d is: 0.69314717556
--------------------
result_up_d [ 0.68817218  0.69264743  0.69309718  0.69314218  0.69314668  0.69314713
  0.69314718]
result_up_s [ 0.68817192  0.69264585  0.69309169  0.69313407  0.69313729  0.69313747
  0.69313753]
Error of Double precesion : [  4.97500125e-03   4.99750000e-04   4.99975000e-05   4.99997496e-06
   4.99999693e-07   4.99998389e-08   4.99952613e-09]
Error of Single precesion : [  4.97525930e-03   5.01334667e-04   5.54919243e-05   1.31130219e-05
   9.89437103e-06   9.71555710e-06   9.65595245e-06]
```