1 Problem 1 Solution

Example done in `double` precision.

Approximate	Absolute Error	Relative Error
3	0.14159265359	0.0450703414486
3.14	0.00159265358979	0.000506957382897
22/7	0.00126448926735	0.000402499434771

1.1 Common Mistakes

1.1.1 Misapplication of Formula

The formula for relative error is:

$$e = \frac{|q - \bar{q}|}{|q|}$$

where q is the exact value and $\bar{q}$ is the approximate value. The results will be similar to the correct numerical values listed above, but not exactly the same if you divide by the approximation $\bar{q}$ instead of the exact value q. Incorrect application of the formula produces:

Approximate	Relative Error
3	0.0471975511966
3.14	0.000507214519042
22/7	0.000402337494157

1.1.2 Integral Division

In some Python interpreters 22/7 produces 3 (integer division), while in others it produces 3.1428571429. For those students who didn't catch the integral issue, absolute error and relative error will be the same for 22/7 as it is for 3. Looking at the answer listed above, it can be seen that 22/7 is a much better approximate than just 3.

1.1.3 Not Taking Absolute Value

Neither the absolute error or the relative error should be reported as a negative number. The equation specifically states to take the absolute value of both the numerator and denominator, which will never yield a negative number.

2 Problem 2 Solution

Yes. The value of machine epsilon ε is as follows:

- Single Precision : 2^{-23} for truncation, 2^{-24} for rounding
- Double Precision : 2^{-52} for truncation, 2^{-53} for rounding

All the above numbers are representable, and the proof that they are the smallest numbers satisfying $1 + \varepsilon \neq 1$ is in the class notes.

2.1 Common Mistakes

2.1.1 Answer is no

The answer is yes.

2.1.2 Not mention the value of machine epsilon

At least one of the values 2^{-23}, 2^{-24}, 2^{-52}, or 2^{-54} should be mentioned with explanation of `single` or `double` precision.

3 Problem 3 Solution

Single Precision:

n	Absolute Error	Relative Error
1	0.077863	0.0844375
2	0.0809957	0.0422071
3	0.16379	0.0280645
4	0.493824	0.0210083
5	1.98083	0.016784
6	9.92181	0.0139728
7	59.604	0.0119677
8	417.605	0.0104657
9	3343.12	0.00929842
10	30104.5	0.00836539

Double Precision:

n	Absolute Error	Relative Error
1	0.0778629911042	0.0844375514192
2	0.080995648511	0.0422071208167
3	0.163790408654	0.0280645179188
4	0.493824867107	0.0210083037463
5	1.98083204241	0.0167839858278
6	9.92181535782	0.0139728491487
7	59.6041683875	0.0119677572632
8	417.604547343	0.0104656510619
9	3343.12715805	0.00929842642182
10	30104.381259	0.0083653591324

3.1 Common Mistakes

Similar mistakes were made here as to Problem 1. See the list of common mistakes there.

3.1.1 Not Comparing Single and Double Precision

The question specifically states that code should be written to produce both `single` *and* `double` precision values, then compare them. Points were deducted when there was only a single or double precision table and no comparison was made.

4 Problem 4 Solution

1. $c||x||_\infty \leq ||x||_2$

   By definition of the infinity norm,

   $$||x||_\infty = \max_{1\leq i\leq n} |x_i|$$

   Let j be the index that gives the maximum value of $|x_i|$, among $1 \leq i \leq n$. Then,

   $$||x||_\infty^2 = |x_j|^2 \leq |x_j|^2 + \sum_{\substack{i=1 \\ i\neq j}}^{n} |x_i|^2 = \sum_{i=1}^{n} |x_i|^2 = ||x||_2^2$$

2. $||x||_2 \leq d||x||_\infty$

$$||x||_2^2 = \sum_{i=1}^{n} x_i^2 \leq \sum_{i=1}^{n} \left(\max_{1\leq i\leq n} |x_i| \right)^2 = n \left(\max_{1\leq i\leq n} |x_i| \right)^2 = n||x||_\infty^2$$

Therefore, $||x||_\infty \leq ||x||_2 \leq \sqrt{n}||x||_\infty$

4.1 Common Mistakes

4.1.1 Equality instead of Inequality.

In the middle of the proof, improper equality is used instead of inequality, which is not helpful to prove the statement. Equality can be used if it is necessary to go on in the proof. However, if there is no key inequality part in the proof, it cannot be regarded to prove the statement at all.

4.1.2 Just giving example not proving the statement.

Example is not enough.

4.1.3 Trivial by definition of the vector norm.

Especially for $c||x||_\infty \leq ||x||_2$, some students just mentioned the definition of those two norms and did not expand it as a formula to explain it. All students are required to show every step regardless of how trivial it may be. Justifying each step in a proof is critical.

5 Problem 5 Solution

The matrix initialization code should be changed to:

```
for i in range(1,m-1):
    for j in range(1,n-1):
        A[n*i+j,n*i+j] = 8./16.
        A[n*i+j,n*(i-1)+j] = 1./16.
        A[n*i+j,n*(i-1)+(j-1)] = 1./16.
```

```
A[n*i+j,n*(i-1)+(j+1)] = 1./16.
A[n*i+j,n*(i+1)+j] = 1./16.
A[n*i+j,n*(i+1)+(j-1)] = 1./16.
A[n*i+j,n*(i+1)+(j+1)] = 1./16.
A[n*i+j,n*i+j-1] = 1./16.
A[n*i+j,n*i+j+1] = 1./16.
```

5.1 Common Mistakes

5.1.1 Updating Image Directly

The question *specifically* asks you to manipulate the matrix I, to see if you can compute a sparse matrix-vector product. Direct manipulation of the image may have lead to similar results, but it is not the procedure that was asked for. This is a class on Numerical Methods, not Computer Vision. Those students who went this route had two weeks to ask if this was acceptable, but they never did.