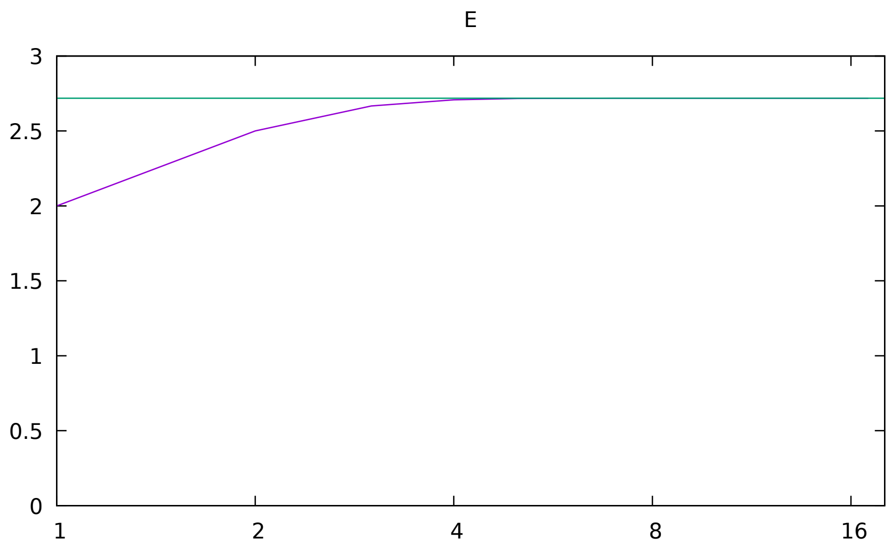
Assignment 2
Writeup
Prof Veenstra
Nguyen Vu

1. e.c

The difference between my code implementation (purple line) and the actual value (green line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Taylor series:

$$e = \sum_{k=0}^{\infty} \frac{1}{k!} = 1 + \frac{1}{1} + \frac{1}{2} + \frac{1}{6} + \frac{1}{24} + \frac{1}{120} + \frac{1}{720} + \frac{1}{5040} + \frac{1}{40320} + \frac{1}{362880} + \frac{1}{3628800} + \cdots$$

The difference at the first few x values is because the value at k= 1 starts at 2 then get closer and closer to the actual e value (2.7182) by by adding new terms indefinitely.



2. madhava.c

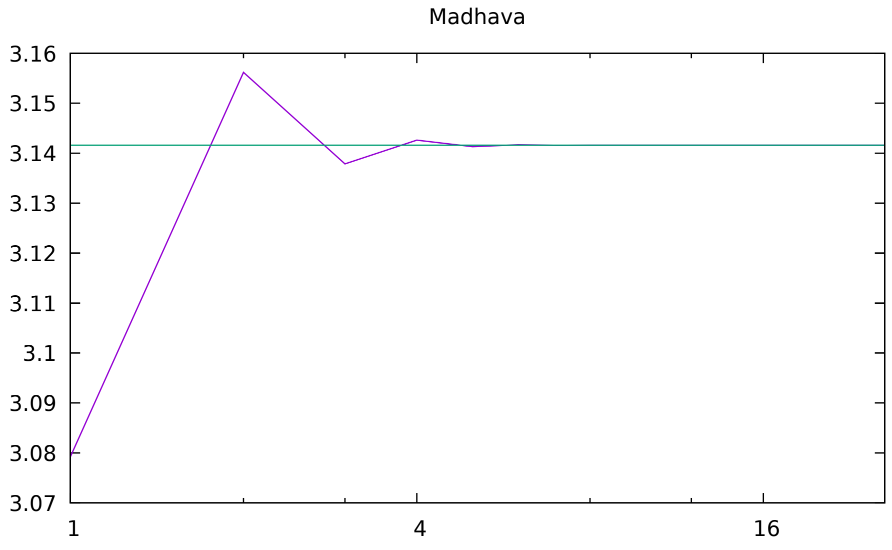
The difference between my code implementation (purple line) and the actual value (green line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Madhava Series.

$$\sqrt{12}\sum_{k=0}^{n}\frac{(-3)^{-k}}{2k+1}$$

The difference at the first few x values is because the value after the first term starts at

$$\sqrt{12}/(8/9)$$

which is approximately 3.0792 and then get closer and closer to the actual pi value (3.1415) by adding new terms indefinitely and multiply by square root of 12 at the end.



3. euler.c

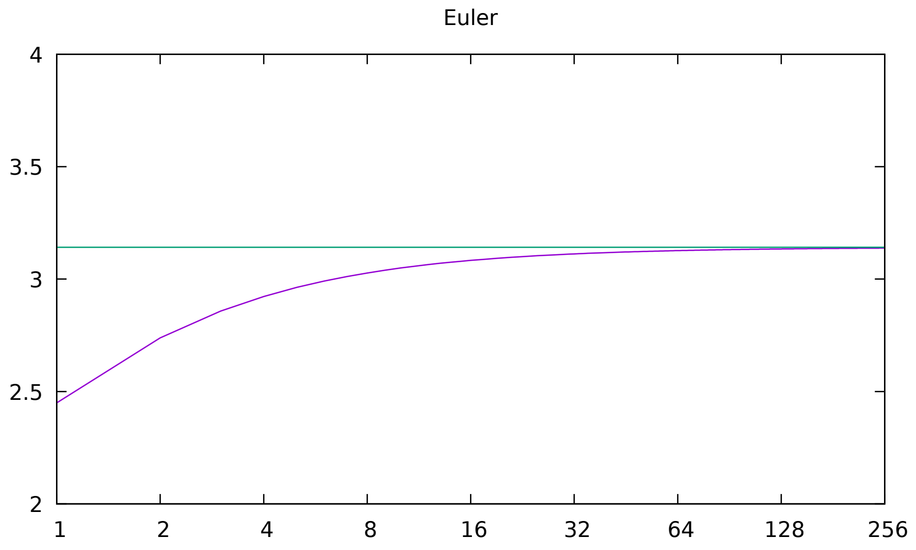
The difference between my code implementation (purple line) and the actual value (green line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Euler's Solution.

$$\sqrt{6\sum_{k=1}^{n}\frac{1}{k^2}}$$

The difference at the first few x values is because the value after the first term starts at

$$\sqrt{6}$$

which is approximately 2.4494 and then get closer and closer to the actual pi value (3.1415) by adding new terms indefinitely and multiply by 6 then square root at the end.

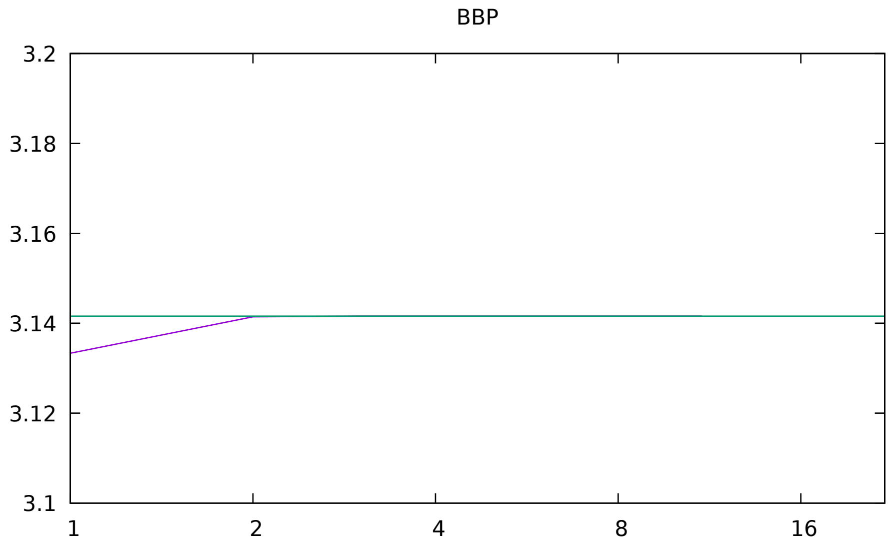


4. bbp.c

The difference between my code implementation (purple line) and the actual value (green line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Bailey-Borwein-Plouffe Formula.

$$\sum_{k=0}^{n} 16^{-k}\left(\frac{4}{8k+1}-\frac{2}{8k+4}-\frac{1}{8k+5}-\frac{1}{8k+6}\right).$$

The difference at the first few x values is because the value after the first term starts at approximately 3.1333 and then get closer and closer to the actual pi value (3.1415) by adding new terms indefinitely.



5. pi_viete.c

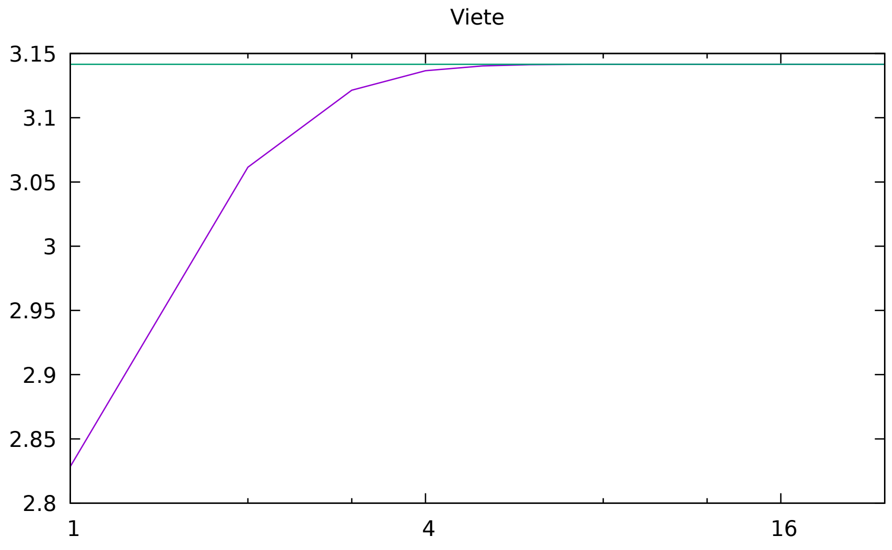
The difference between my code implementation (purple line) and the actual value (green line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Viete's Formula.

$$\frac{2}{\pi} = \prod_{k=1}^{\infty} \frac{a_k}{2}$$

The difference at the first few x values is because the value after the first term starts at

$$\frac{4}{\sqrt{2}}$$

which is approximately 2.8284 and then get closer and closer to the actual pi value (3.1415) by multiplying new terms indefinitely and the get divided by 2 at the end.



6. newton.c

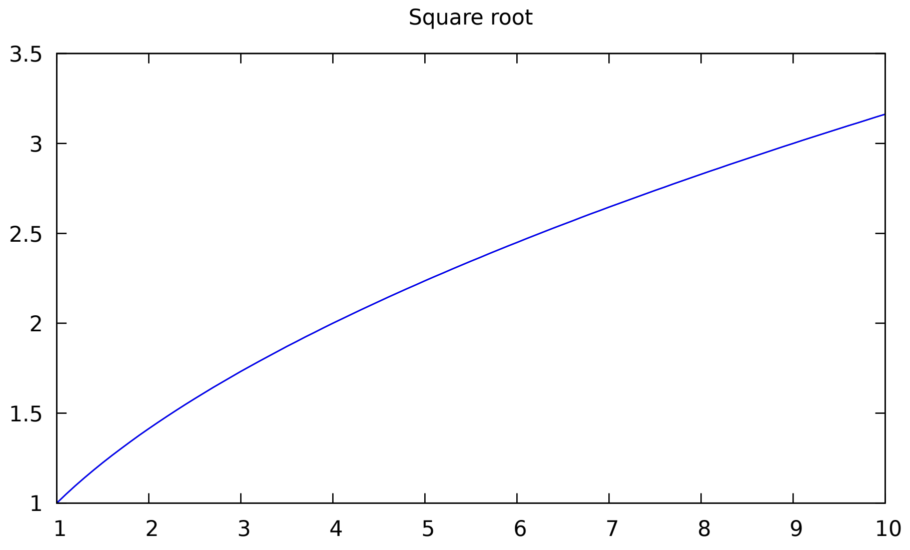
The difference between my code implementation (blue line) and the actual value (black line) up to 15 decimal points accuracy of the required function is shown below. The formula used in my code is the Newton Raphson Method with

$$f(x_0) = x_0^2 - (givenvalue)$$

$$f^{'}(x_0) = 2x$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

There are no visible difference between my value and the actual value is because I only put the final value of each test case and the difference between them and the actual square root value of each test case is smaller than EPSILON (1e-14) so it is impossible to see with human eyes with the scale needed for the plot to include all test cases.


Square root
3.5
3
2.5
2
1.5
1
1
2
3
4
5
6
7
8
9
10