MAS212 Assignment #1 (2018): Rational approximations

Dr. Sam Dolan (s.dolan@sheffield.ac.uk)

In this assignment you will use Python to find rational approximations to real numbers such as π, $\sqrt{2}$ and the golden ratio $\varphi = \frac{1}{2}(1+\sqrt{5})$. Your completed assignment will consist of **one** `.py` script file. Your file should address the tasks in Part 1 and Part 2, below. Your files should be submitted at `http://somas-uploads.shef.ac.uk/mas212`. The submission deadline is found on the course website. Please refer to Appendix A for the preferred format for the file, and to Appendix B for the sample output of your script.

Part 0: Introduction

(a) Mathematical background. The **Farey sequence** F_n is the set of all irreducible rational numbers $\frac{a}{b}$ in the unit interval with denominator b less than or equal to n, arranged in increasing order. For example,

$$F_5 = \{\tfrac{0}{1}, \tfrac{1}{5}, \tfrac{1}{4}, \tfrac{1}{3}, \tfrac{2}{5}, \tfrac{1}{2}, \tfrac{3}{5}, \tfrac{2}{3}, \tfrac{3}{4}, \tfrac{4}{5}, \tfrac{1}{1}\}. \tag{1}$$

(*Irreducible* means that the numerator and denominator are coprime, that is, they have no prime factors in common.)

The Farey sum $\oplus$ is a function $\mathbb{Q}\times\mathbb{Q} \to \mathbb{Q}$ that gives the **mediant** of a pair of rational numbers:

$$\frac{a}{b} \oplus \frac{c}{d} = \frac{a+c}{b+d}, \qquad a,b,c,d \in \mathbb{N}. \tag{2}$$

(Note that $\frac{a}{b} \oplus \frac{c}{d} \neq \frac{a}{b} + \frac{c}{d} = \frac{ad+bc}{db}$!). A pair of fractions which are adjacent in some Farey sequence are called **Farey neighbours**. A pair of Farey neighbours $\frac{a}{b} < \frac{c}{d}$ have the following properties:

1. $ad - bc = -1$
2. The mediant lies between the neighbours: $\frac{a}{b} < (\frac{a}{b} \oplus \frac{c}{d}) < \frac{c}{d}$.
3. The mediant $\frac{a}{b} \oplus \frac{c}{d}$ is the Farey neighbour of both $\frac{a}{b}$ and $\frac{c}{d}$, in the Farey sequence F_{b+d}.

For example, let $a/b = 1/3$ and $c/d = 1/2$. These are neighbours in F_3. Their mediant is $2/5$. Then (1) $ad - bc = 1\cdot 2 - 3\cdot 1 = -1$; (2) $1/3 < 2/5 < 1/2$; (3) 1/3 & 2/5, and 2/5 & 1/2, are Farey neighbours in F_5, above.

> **Exercise** *(not assessed & **not** part of your submission)*: Prove statements 1, 2 and 3.

(b) The mediant bracketing algorithm.

Below is an algorithm for finding a rational approximation $\frac{a}{b}$ to a real number x in the interval $[0,1)$.

1. Start with the interval $\left[\frac{0}{1}, \frac{1}{1}\right)$.
2. Let $\frac{p_1}{q_1}$ and $\frac{p_2}{q_2}$ be the upper and lower bounds of this interval, respectively.
   Compute the mediant $\frac{p_3}{q_3} = \frac{p_1}{q_1} \oplus \frac{p_2}{q_2}$.

3. Determine whether x is in the interval $[\frac{p_1}{q_1}, \frac{p_3}{q_3})$ or the interval $[\frac{p_3}{q_3}, \frac{p_2}{q_2})$. If the former, change the upper bound to $\frac{p_3}{q_3}$. If the latter, change the lower bound to $\frac{p_3}{q_3}$.

4. Repeat steps 2 & 3 until a **stopping condition** is reached. Either:
   (i) $|x - \frac{p_3}{q_3}| < \epsilon$, where ϵ is some tolerance, or,
   (ii) the maximum number of iterations `nmax` is exceeded.

The algorithm can be used to generate a converging sequence of rational approximants to the real number x. The algorithm can also be used to generate a certain binary representation of a real number, $b_1 b_2 b_3 \ldots$, where b_i denotes the 'decision' at step i, with $b_i = 0$ if x was in the lower interval and $b_i = 1$ if x was in the upper interval. We shall call this the **Farey representation**.

Part 1: Implementing the algorithm

Look first at Appendix A, showing the suggested file format, and Appendix B, showing part of the expected output of your code. Following these templates is likely to lead to a higher mark.

1(a) Write a function `encode(r, eps=1e-12, nmax=100)` to implement the algorithm. The function should calculate the Farey representation of the non-integer part of a real number r. It should return a string consisting of 0s and 1s only. The optional arguments `eps` and `nmax` determine the stopping condition.

Add a docstring and comments to your function. See Appendix A for guidance.

Test your function by calculating the Farey representation of:

(i) $\sqrt{2}$;

(ii) the golden ratio φ ;

(iii) π.

Print the Farey representations to screen. See Appendix B for example output.

1(b) Write a function `decode(s)` to convert the Farey representation (a string `s` consisting of 0s and 1s) into a sequence of mediants. The function should return a list of tuples, where each tuple is a pair of integers specifying the numerator and denominator of the corresponding mediant.

For each of the cases (i)–(iii), `print` to screen: the last tuple in the list; the decimal representation of the fraction represented by the tuple; and the non-integer part of the real number `r`. This is illustrated in Appendix B.

Exercise *(not assessed & **not** part of your submission)*: Prove that the golden ratio has a recurring Farey representation $101010101010101\ldots$.

Part 2: Experiments and conjectures.

The tasks in Part 2 are more open-ended, giving some scope for you to apply to your investigative skills. Do not start part 2 until you have finished part 1. Do not spend an excessive amount of time on part 2.

2(a) You may have noticed that the rational approximant to π found in part 1(a)(iii) is less accurate than in the cases (i) and (ii). This is related to the observation that the algorithm finds two "unexpectedly good" low-order rational approximations to π. The first is $22/7$.

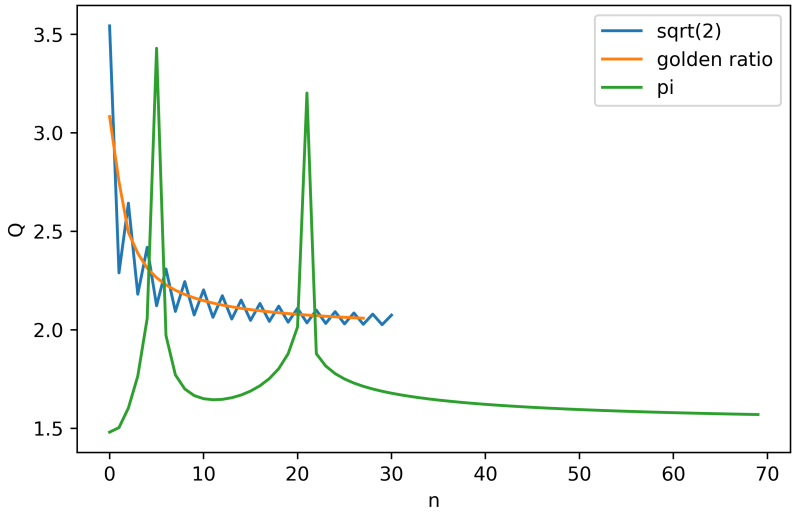
(i) Write code to find the second "unexpectedly good" rational approximation to π, after $22/7$. `print()` this to screen.

(ii) The **quality** Q of a rational approximant p/q to a real number $0 < x \leq 1$ is defined by

$$Q = -\frac{\ln\left|\frac{p}{q} - x\right|}{\ln q}.$$

The plot below shows Q for each mediant in the sequence generated by the algorithm, for $\sqrt{2}$, φ and π. The two "unexpectedly good" approximations to π correspond to the pair of peaks.

Write a function `quality(r, s)`, which returns a list `Qs`, where each element of `Qs` is the quality of the corresponding approximant, as sequentially generated from the Farey representation `s`. Use `matplotlib.pyplot()` to recreate the plot below. Save the plot in a file labelled by your registration number (e.g. `fig_2a_17015394.png`).



2(b) A student conjectures that non-rational **algebraic numbers**, such as $\sqrt{n}$ and φ, have recurring Farey representations; whereas **trancendental numbers**, such as π, have non-recurring Farey representations.

Write code to investigate the student's conjecture. Describe your findings in one paragraph (using `print`). If you find a counter-example, state it. If you do not, give **three** pieces of evidence in support of the conjecture, beyond the examples in Part 1.

Appendix A: File format. Include a docstring at the top of the script file with your name and registration number. Next, import any modules you will use. Next, define all the functions used. Finally, call the functions to produce human-readable output for each of the tasks. Please make the output easy to read for the marker running the script (for example, indicate which task is being completed). See the example below.

```
"""
Sam Dolan. Registration number: xxxxxxxxx
"""
#####
# (1) Import modules
import math
import numpy as np

#####
# (2) Define functions

def encode(r, eps=1e-12, nmax=100):
    """Write a docstring here to describe the purpose of the function,
    its parameters, and its return values."""
    x = r - math.floor(r)  # the non-integer part between 0 and 1
    p1,q1 = 0,1
    p2,q2 = 1,1
    s = ""
    for i in range(nmax):
        # your code here:
        # - compute the mediant
        # - determine which interval x is in
        # - update s and the interval bounds (p1,q1) or (p2,q2)
        # - add a stopping condition
    return s

def decode(s):
    """Converts the Farey representation (a string of 0s and 1s)
        into a list of tuples representing a converging sequence of fractions."""

#####
# (3) Tasks:
print("Task 1(a):")
# ... add code here ...
```

Appendix B: Example output of your script

```
# Note: This output was changed on 18/10/18 to include one more digit on the sequences.
# Output matching either the old version or the new version will be marked as correct.

Task 1(a):
  sqrt(2)      : 0110011001100110011001100110011
  golden ratio : 1010101010101010101010101010
  pi           : 0000001111111111111110111111111111111111111111111111111111111111111111111111111111111111111111111111

Task 1(b):
sqrt(2)      : 470832 / 1136689
   =  0.4142135623728214
      0.414213562373
golden ratio : 514229 / 832040
   =  0.6180339887505408
      0.61803398875
pi           : 1263 / 8920
   =  0.14159192825112107
      0.14159265358979312

Task 2(a)(i):
The second unexpectedly good rational approximant to pi is:  xxx / xxx

Task 2(a)(ii):
The quality Q of the rational approximants is shown below:
<your plot here>

etc.
```