CS3430 S26: Scientific Computing
HW 5, Problem 1 – Continued Fractions for π and e

Vladimir Kulyukin

Overview

In this problem, I implemented generator-based continued fraction approximations for π and e using two numerical representations: `mpmath` (arbitrary-precision binary arithmetic) and `decimal` (arbitrary-precision base-10 arithmetic).

I validated both implementations against exact rational values computed with `Fraction`. All validation was performed using the provided unit tests.

Structural Verification

I first verified that all required generator functions were present and callable. The unit tests confirmed the existence of:

- `pi_cf_mp`
- `e_cf_mp`
- `pi_cf_dec`
- `e_cf_dec`

The test output explicitly stated:

```
[STRUCTURE TEST] Checking that required generator functions
exist:
- pi_cf_mp:  FOUND and callable
- e_cf_mp:  FOUND and callable
- pi_cf_dec:  FOUND and callable
- e_cf_dec:  FOUND and callable
```

I then instantiated each generator with fixed precision (`dps = 80` for `mpmath` and `prec = 80` for `decimal`) and consumed the first three values. The test output showed:

```
 First three approximations produced:
pi_cf_mp:  [3.166666..., 3.133333..., 3.145238...]
e_cf_mp:  [3.0, 2.666666..., 2.75]
pi_cf_dec:  [3.166666..., 3.133333..., 3.145238...]
e_cf_dec:  [3, 2.666666..., 2.75]
```

All generators produced multiple values and returned results of the correct numerical type, confirming that they were implemented as true generators.

Numerical Correctness for e

To verify numerical correctness, I compared the first eight convergents of e against exact rational values computed using `Fraction`.

Using `mpmath` with `mp.dps = 90`, every convergent matched the exact value with zero absolute difference. For example:

```
 convergent 5:
approx = 2.71875
exact = 2.71875
|diff| = 0.0
```

Using `decimal` with context precision 90, the results also matched the exact oracle up to extremely small differences on the order of 10^{-89}:

```
 convergent 7:
approx = 2.7183098591549...95774
exact = 2.7183098591549...95775
|diff| = 1E-89
```

These differences are far below machine precision and confirm that the implementation is numerically correct.

Numerical Correctness for π

I performed a similar comparison for π, testing the first six convergents.

With `mpmath` at `mp.dps = 90`, all convergents matched the exact oracle with zero absolute difference. For example:

```
 convergent 3:
approx = 3.145238095238095...
exact = 3.145238095238095...
|diff| = 0.0
```

Using decimal with precision 90 produced exact agreement up to 10^{-89}:

```
 convergent 5:
approx = 3.142712842712842...
exact = 3.142712842712842...
|diff| = 0E-89
```

This confirms that the inside-out evaluation of the continued fraction was implemented correctly for both representations.

Conclusions

All four generators correctly compute continued fraction convergents for π and e. The results demonstrate that:

- Continued fractions are naturally expressed using generators.
- Inside-out evaluation is essential for correctness.
- Numerical convergence eventually becomes limited by representation precision rather than mathematics.

The agreement with exact rational oracles confirms the correctness of both the `mpmath` and `decimal` implementations.

I did not notice any wall clock difference betweeen `mpmath` and `decimal`.

Appendix: My Condensed Write-Up for Problem 1

Below is a condensed version (CS/Math journal style) of the Problem 1 report. You should insert a version like this as a multi-line comment at the beginning of your `cs3430_s26_hw_5_prob_1.py`.

```
HW 5  Problem 1: Continued Fractions for pi and e

In this problem, I implemented generator-based continued fraction
approximations for pi and e using two numerical representations:
mpmath (arbitrary-precision binary arithmetic) and decimal
(arbitrary-precision base-10 arithmetic).

I verified the structure of my solution by confirming that all
required generator functions existed, were callable, and produced
multiple successive values. Using fixed precision (dps = 80 for
mpmath and prec = 80 for decimal), each generator yielded valid
```

approximations of the correct numerical type.

To validate numerical correctness, I compared the first several convergents of pi and e against exact rational values computed using Fraction. With high working precision (90 digits), both the mpmath and decimal implementations matched the exact oracles with absolute errors at or below 1e-70.

For e, the first eight convergents matched the exact values, with mpmath showing zero difference and decimal differing only at the 1e-89 level in later convergents. For pi, the first six convergents matched the exact oracle to the same level of accuracy.

These results confirm that the continued fractions were evaluated correctly using inside-out recurrence, and that observable convergence is ultimately limited by numerical representation rather than mathematics.