11

Continued Fractions

Why Study Continued Fractions?

- **They offer an alternative representations for numbers**. Every real number has a unique simple continued fraction expansion, that's often more revealing than decimal or binary.

- **They underlie efficient algorithms**. The Euclidean algorithm can be viewed as the computation of a continued fraction. This links continued fractions to number theory, modular arithmetic, and greatest common divisors.

- **They solve Pell-type and quadratic Diophantine equations**. Periodic continued fractions for $\sqrt{d}$ can be employed to solve Pell's equation $x^2 - dy^2 = 1$. This connects continued fractions to algebraic number theory and even to conics.

- **They give the best rational approximations to real numbers**. If you want to approximate a real number by a fraction with a small numerator and denominator, its continued fraction convergents are provably the best.

11.1 Introduction

If x is a real number (and not an integer) then we can write

$$x = a_0 + \frac{1}{x_1}, \quad a_0 = \lfloor x \rfloor, \quad x_1 = \frac{1}{x - a_0} > 1.$$

If x_1 is also not an integer, then we can apply the same transformation again

$$x_1 = a_1 + \frac{1}{x_2}, \quad a_1 = \lfloor x_1 \rfloor, \quad x_2 = \frac{1}{x_1 - a_1} > 1.$$

For rational numbers we'll eventually get

$$\begin{aligned} x &= a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ddots + \cfrac{1}{a_n}}}} \\ &= a_0 + 1/(a_1 + 1(a_2 + 1/(a_3 + \cdots + 1/a_n))\cdots) \end{aligned}$$

This expression is called a *simple continued fraction*, and often written as $[a_0; a_1, a_2, ..., a_n]$; the a_i's are *partial quotients*.

Example. Suppose we start with the rational number

$$\frac{57}{33}$$

Start converting this to a continued fraction:

$$\frac{57}{33} = 1 + \frac{24}{33}.$$

Now invert the remainder:

$$\frac{33}{24} = 1 + \frac{9}{24}.$$

Again:

$$\frac{24}{9} = 2 + \frac{6}{9},$$

and again:

$$\frac{9}{6} = 1 + \frac{3}{6},$$

and finally:

$$\frac{6}{3} = 2$$

Thus

$$\frac{57}{33} = [1; 1, 2, 1, 2]$$

This procedure is implemented by Listing 11.1 (cfLib.py).

```
def cf(numer, denom, maxTerms=10):
  # Return the terms of the continued fraction for
  # the numerator and denominator as a list
  terms = []
  quotient = numer//denom
  rem = numer - quotient*denom
  count = 0
```

```
  while (rem != 0) and (count < maxTerms):
    terms.append(quotient)
    count += 1
    prev_r = rem
    quotient = denom//prev_r
    rem = denom - quotient*prev_r
    denom = prev_r
  terms.append(quotient)
  return terms
```

Listing 11.1. From rational to continued fraction

Example:

```
>>> from cfLib import *
>>> cf(57,33)
[1, 1, 2, 1, 2]
```

Conversely, the continued fraction

$$[2;1,3,2] = 2 + \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{2}}}$$

represents the rational

$$\begin{aligned} 2 + \cfrac{1}{1 + \cfrac{2}{7}} &= 2 + \frac{7}{9} \\ &= \frac{25}{9} \end{aligned}$$

This conversion is implemented by Listing 11.2 (cfLib.py).

```
def cf2rat(cfList):
  # finite continued fraction to rational
  n = len(cfList)
  numer = 1
  denom = 0
  for i in range(n-1, -1, -1):
    n1 = numer*cfList[i] + denom
    denom = numer
    numer = n1
  return (numer, denom)
```

Listing 11.2. From continued fraction to rational

Example

```
>>> from cfLib import *
>>> cf2rat([2,1,3,2])
(25, 9)
```

When we evaluate $[a_0; a_1, a_2, ..., a_n]$, we may choose to stop at a_i, obtaining the i-th *convergent* $r_i = p_i/q_i$.

Returning to $\frac{57}{33}$, the successive convergents for

$$\frac{57}{33} = [1; 1, 2, 1, 2]$$

are

$$\begin{aligned}
p_0/q_0 &= [1] = \tfrac{1}{1} \\
p_1/q_1 &= [1;1] = 1 + \tfrac{1}{1} = \tfrac{2}{1} \\
p_2/q_2 &= [1;1,2] = [1;3/2] = 1 + \tfrac{2}{3} = \tfrac{5}{3} \\
p_3/q_3 &= [1;1,2,1] = [1;1,3] = [1;4/3] = 1 + \tfrac{3}{4} = \tfrac{7}{4} \\
p_4/q_4 &= [1;1,2,1,2] = [1;1,2,3/2] = [1;1,8/3] = [1;11/8] = \frac{19}{11}
\end{aligned}$$

The production of convergents as a list is implemented by Listing 11.3 (cfLib.py).

```
def convergents(cfList, numRats=10):
  p0 = 1; q0 = 0
  p = cfList[0]; q = 1
  i = 1
  err = 1
  cs = [(1,1)]
  while i < len(cfList) and (i < numRats):
    a = cfList[i]
    if a < 1:
      break
    # update the numerator of the convergent
    p1 = a*p + p0
    p0 = p
    p = p1
    # update the denominator
    q1 = a*q + q0
    q0 = q
    q = q1
    cs.append((p,q))
    i += 1
  return cs
```

Listing 11.3. Generate the convergents for a continued fraction

Example:

```
>>> from cfLib import *
>>> convergents([1,1,2,1,2])
[(1, 1), (2, 1), (5, 3), (7, 4), (19, 11)]
```

We can define p_i and q_i (the numerator and denominator of the r_i convergent) recursively as:

$$
\begin{array}{lllllll}
p_0 & = & a_0, & q_0 & = & 1 & \\
p_1 & = & a_0 a_1 + 1, & q_1 & = & a_1 & \\
 & \cdots & & & \cdots & & \\
p_k & = & p_{k-1} a_k + p_{k-2} & q_k & = & q_{k-1} a_k + q_{k-2}, & k = 2, ..., n.
\end{array} \tag{11.1}
$$

Indeed,

$$
\begin{array}{lll}
p_0/q_0 & = & r_0, \\
p_1/q_1 & = & a_0 + 1/a_1 = r_1, \\
p_2/q_2 & = & r_2
\end{array}
$$

and induction shows that

$$p_k/q_k = r_k \text{ for } k = 0, 1, 2, ..., n.$$

11.1.1 From Rational to Irrational. Rational numbers are guaranteed to produce *finite length* continued fraction, but the conversion of irrationals (such as π, $\sqrt{2}$, and e) never terminates. For that reason, their conversion is implemented by cfList() using a generator function cfY() (see Listing 11.4; cfLib.py).

```
def cfList(x, maxTerms=10):
  cfs = []
  i = 0
  for val in cfY(x):
    # print(val)
    cfs.append(val)
    i += 1
    if i > maxTerms:
      break
  return cfs


def cfY(x):
  while x != int(x):
    a = int(x)
    x = 1/(x-a)
    yield a
```

Listing 11.4. From irrational to continued fraction

Examples:

```
>>> r2 = math.sqrt(2)
>>> cfList(r2)
[1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
>>> cfList(r2,30)
[1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
1, 1, 1, 3, 3, 1, 3, 1, 1, 2]

>>> cfList(math.pi)
[3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3]

>>> cfList(math.e, 18)
[2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, 1]

>>> phi = (math.sqrt(5)+1)/2
>>> cfList(phi, 40)
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1]
```

The problem with this approach is that rounding errors associated with Python's floats eventually start to overwhelm the calculations in cfY() and produce incorrect quotients. This can be seen in the long continued fractions for $\sqrt{2}$ and ϕ (the golden ratio), which should be $[1; \overline{2}]$ and $[1; \overline{1}]$ respectively (a bar over a digit (or digits) means that it is repeated indefinitely).

One partial solution is to switch to the Python decimal version of cfY() shown in Listing 11.5 (cfLib.py).

```
def cfDecList(x, maxTerms=20):
  cfs = []
  i = 0
  for val in cfDecY(x):
    # print(val)
    cfs.append(val)
    i += 1
    if i > maxTerms:
      break
  return cfs


def cfDecY(x):
  decimal.getcontext().prec = 100
  d1 = D(1)
  dx = D(x)
  while dx != math.trunc(dx):
    a = math.trunc(dx)
    dx = d1/(dx-a)
    yield a
```

Listing 11.5. From irrational to continued fraction using decimals

Examples:

```
>>> phi = (math.sqrt(5)+1)/2
>>> cfDecList(phi, 40)
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 7, 1]

>>> cfDecList('1.6180339887498948482045868343656381177203091798057628621
354486227052604628189 0', 100)
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1]
```

One issue illustrated by the first example above is that supplying an irrational encoded as a float may not actually improve the continued fraction result. This is due to the inherent inaccuracy of the float, and the way that the decimal module converts a float to a decimal. It's much better to supply the irrational as a string (as in the second example), making sure to include enough digits for the continued fraction's calculation.

Many surprising patterns are revealed in the irrationals when they're converted to continued fractions. For example, the continued fraction expansion of e is:

$$e = [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, 1, 1, 14, 1, 1, 16, \cdots]$$

But the continued fraction of π doesn't appear to have any regularity:

$$\pi = [3; 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, 84, 2, 1, 1, 15, \\ 3, 13, 1, 4, 2, 6, 6, 99, 1, 2, 2, 6, 3, 5, 1, 1, 6, \cdots]$$

One objection to continued fractions is that they seem to make arithmetic more difficult. For example, it's not at all obvious how to add or multiply two continued fractions. This has been addressed by Bill Gosper who devised algorithms for computing with continued fractions term-by-term. His unpublished paper can be found at `https://web.archive.org/web/20030202011209/http://www.tweedledum.com/rwg/cfup.htm`, and is very well explained in a online article by Ross Dempsey at `https://srossd.com/posts/2020-09-18-gosper-1/`.

By far the best beginners book on continued fractions is by Olds [**Old63**], which we recommend highly.

11.2 Continued Fractions and Euclid's Algorithm

We apply Euclid's GCD algorithm (see 2.1.1) to two positive integers n_0 and n_1, where $n_0 \geq n_1 > 0$:

$$\begin{array}{rcl} n_0 & = & q_1 n_1 + n_2 \\ n_1 & = & q_2 n_2 + n_3 \\ & \cdots & \\ n_{k-2} & = & q_{k-1} n_{k-1} + n_k \\ n_{k-1} & = & q_k n_k \end{array}$$

From these equations we can obtain

$$\frac{n_0}{n_1} = [q_1; q_2, \cdots, q_k].$$

Example:

$$\begin{array}{rcl rcl} 271828 & = & 2 * 100000 + 71828 & 1504 & = & 1 * 1292 + 212 \\ 100000 & = & 1 * 71828 + 28172 & 1292 & = & 6 * 212 + 20 \\ 71828 & = & 2 * 28172 + 15484 & 212 & = & 10 * 20 + 12 \\ 28172 & = & 1 * 15484 + 12688 & 20 & = & 1 * 12 + 8 \\ 15484 & = & 1 * 12688 + 2796 & 12 & = & 1 * 8 + 4 \\ 12688 & = & 4 * 2796 + 1504 & 8 & = & 2 * 4 \\ 2796 & = & 1 * 1504 + 1292 & & & \end{array}$$

$$\text{And so } \frac{271828}{100000} = [2; 1, 2, 1, 1, 4, 1, 1, 6, 10, 1, 1, 2].$$

We can double check by using cf() (Listing 11.1):

```
>>> cf(271828, 100000, 20)
[2, 1, 2, 1, 1, 4, 1, 1, 6, 10, 1, 1, 2]
```

11.3 Continued Fractions for Square Roots

A continued fraction is eventually periodic if and only if it represents a *quadratic* irrational number (i.e. a solution to a quadratic equation with rational coefficients). The proof is a bit technical, but the idea is to establish a recurrence among the terms of the continued fraction. A simple example is $\sqrt{2}$ whose continued fraction is

$$[1; 2, 2, 2, \cdots]$$

We know that $\sqrt{2}$ is a solution to $x^2 = 2$, which we can write as $x^2 - 1 = 1$, or $(x-1)(x+1) = 1$, so

$$x = 1 + \frac{1}{1+x}$$

Now let $x = [1; a1, a2, \cdots]$, and substitute it into the equation above to get

$$\begin{aligned}[1; a_1, a_2, \cdots] &= 1 + \frac{1}{[2; a_1, a_2, \cdots]} \\ &= [1; 2, a_1, a_2, \cdots] \quad \text{or} \quad [1; \overline{2}]\end{aligned}$$

Comparing the continued fractions term by term, we find that $a_i = 2$ for all $i = 1, 2, \cdots$.

Let's move along to the continued fraction for $\sqrt{3}$. By choosing a starting value carefully, we can convert $\sqrt{3}$ in a similar way to $\sqrt{2}$:

$$\begin{aligned}\sqrt{3} &= 1 + (\sqrt{3} - 1), \\ \frac{1}{\sqrt{3} - 1} &= \frac{\sqrt{3} + 1}{2} = 1 + \frac{\sqrt{3} - 1}{2}, \\ \frac{2}{\sqrt{3} - 1} &= \sqrt{3} + 1 = 2 + (\sqrt{3} - 1), \\ \frac{1}{\sqrt{3} - 1} &= 1 + \frac{\sqrt{3} - 1}{2}, \\ \cdots\end{aligned}$$

Thus

$$\sqrt{3} = [1; \overline{1, 2}]$$

We can check that $\sqrt{d}$ always has a periodic expansion, by using cfDecList() (Listing 11.5) and decimal.sqrt()

```
>>> decimal.getcontext().prec = 100
>>> cfDecList( D(7).sqrt(), 30)
[2, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1,
4, 1, 1, 1, 4, 1, 1]

>>> cfDecList( D(43).sqrt(), 30)
[6, 1, 1, 3, 1, 5, 1, 3, 1, 1, 12, 1, 1, 3, 1, 5, 1, 3, 1, 1, 12, 1, 1,
3, 1, 5, 1, 3, 1, 1, 12]

>>> cfDecList( D(94).sqrt(), 40)
[9, 1, 2, 3, 1, 1, 5, 1, 8, 1, 5, 1, 1, 3, 2, 1, 18, 1, 2, 3, 1, 1, 5, 1,
8, 1, 5, 1, 1, 3, 2, 1, 18, 1, 2, 3, 1, 1, 5, 1, 8]

>>> cfDecList( D(919).sqrt(), 100)
[30, 3, 5, 1, 2, 1, 2, 1, 1, 1, 2, 3, 1, 19, 2, 3, 1, 1, 4, 9, 1, 7, 1, 3,
6, 2, 11, 1, 1, 1, 29, 1, 1, 1, 11, 2, 6, 3, 1, 7, 1, 9, 4, 1, 1, 3, 2,
19, 1, 3, 2, 1, 1, 1, 2, 1, 2, 1, 5, 3, 60, 3, 5, 1, 2, 1, 2, 1, 1, 1, 2,
3, 1, 19, 2, 3, 1, 1, 4, 9, 1, 7, 1, 3, 6, 2, 11, 1, 1, 1, 29, 1, 1, 1, 11,
2, 6, 3, 1, 7, 1]
```

```
>>> cfDecList( D(991).sqrt(), 100)
[31, 2, 12, 10, 2, 2, 2, 1, 1, 2, 6, 1, 1, 1, 1, 3, 1, 8, 4, 1, 2, 1, 2, 3,
1, 4, 1, 20, 6, 4, 31, 4, 6, 20, 1, 4, 1, 3, 2, 1, 2, 1, 4, 8, 1, 3, 1, 1,
1, 1, 6, 2, 1, 1, 2, 2, 2, 10, 12, 2, 62, 2, 12, 10, 2, 2, 2, 1, 1, 2, 6,
1, 1, 1, 1, 3, 1, 8, 4, 1, 2, 1, 2, 3, 1, 4, 1, 20, 6, 4, 31, 4, 6, 20, 1,
4, 1, 3, 1, 1, 9]
```

In summary:

$$\begin{aligned}
\sqrt{7} &= [2; \overline{1, 1, 1, 4}] \\
\sqrt{43} &= [6; \overline{1, 1, 3, 1, 5, 1, 3, 1, 1, 12}] \\
\sqrt{94} &= [9; \overline{1, 2, 3, 1, 1, 5, 1, 3, 1, 5, 1, 1, 3, 2, 1, 18}] \\
\sqrt{919} &= [30; \overline{3, 5, 1, 2, 1, 2, 1, 1, 1, 2, \cdots, 2, 1, 5, 3, 60}] \\
\sqrt{991} &= [31; \overline{2, 12, 10, 2, 2, 2, 1, 1, 2, 6, \cdots, 2, 10, 12, 2, 62}]
\end{aligned}$$

The periods for $\sqrt{919}$ and $\sqrt{991}$ have 62 and 60 terms respectively. Each period always ends with $2\lfloor\sqrt{d}\rfloor$ and, excluding the last term, is a palindrome (it reads the same forwards and backwards).

11.4 The Law of Best Approximation

If the rational r/s is closer to an irrational x than the n-th convergent $r_n = p_n/q_n$ of x, then $s > q_n$. In other words, the n-th convergent is the 'best' approximation to the irrational for that denominator size. If you want a better approximation, then you'll need a larger denominator, which can be found in subsequent convergents in the continued fraction. We could prove this, but instead refer to Sec. 2.12, where the topic is dealt with in a more general way.

The following examples generate 100-term continued fractions and convergents for π, e, and ϕ.

```
>>> cfPi = cfDecList("3.14159265358979323846264338327950288419716939
93751058209749445923078164062862089986280348253421170679", 100)
>>> convergents(cfPi, 5)
[(1, 1), (22, 7), (333, 106), (355, 113), (103993, 33102)]

>>> cfE = cfDecList("2.718281828459045235360287471352662497757247093
69995957496696762772407663035354759457138217852516642742", 100)
>>> convergents(cfE)
[(1, 1), (3, 1), (8, 3), (11, 4), (19, 7), (87, 32), (106, 39),
(193, 71), (1264, 465), (1457, 536)]

>>> cfPhi = cfDecList("1.61803398874989484820458683436563811772030917
98057628621354486227052604628189024497072072041893911374", 100)
>>> convergents(cfPhi, 20)
[(1, 1), (2, 1), (3, 2), (5, 3), (8, 5), (13, 8), (21, 13), (34, 21),
```

```
(55, 34), (89, 55), (144, 89), (233, 144), (377, 233), (610, 377),
(987, 610), (1597, 987), (2584, 1597), (4181, 2584), (6765, 4181),
(10946, 6765)]
```

We can say just how close the convergents get to the irrational. Let $\alpha = [a_0; a_1, a_2, ...]$, and let the n-th convergent be p_n/q_n, with $p_0 = a_0$ and $q_0 = 1$. Then

$$\frac{1}{(a_{n+1}+2)q_n^2} < \left|\alpha - \frac{p_n}{q_n}\right| < \frac{1}{a_{n+1}\, q_n^2}$$

This implies that larger denominators in a convergent is a sign that the convergent is getting closer to the irrational's actual value. For instance, consider the fourth convergent for π:

$$[3; 7, 15, 1, 292] = \frac{355}{113} = 3.14159\underline{292035}\cdots$$

It differs from π (3.141592 65359...) by no more than $1/(292 \cdot 113^2) \approx 2.2e - 7$.

This also explains why $\phi = [1; 1, 1, 1, \cdots]$ is considered the most difficult number to approximate by rationals, since $a_{n+1}\, q_n^2$, grows as slowly as possible.

11.4.1 Convergents for ϕ. All of the numerators and denominators of the convergents for ϕ above are Fibonacci numbers. Indeed, each numerator is the successor of its denominator in the Fibonacci series.

This is easily proved: ϕ can be expressed as the equation $\phi^2 - \phi - 1 = 0$, or equivalently

$$\phi = 1 + \frac{1}{\phi}$$

This can be expanded into the continued fraction $[1; 1, 1, 1, \cdots]$, or written as the sequence:

$$\phi_{n+1} = 1 + \frac{1}{\phi_n}, \quad \phi_0 = 1.$$

Suppose

$$\phi_n = \frac{r_n}{t_n}$$

Then

$$\phi_{n+1} = \frac{r_{n+1}}{t_{n+1}} = 1 + \frac{t_n}{r_n} = \frac{t_n + r_n}{r_n}$$

Equate the corresponding numerators and denominators:

$$\begin{aligned} r_{n+1} &= t_n + r_n \\ t_{n+1} &= r_n \end{aligned}$$

which can be rewritten as a Fibonacci series for generating r's or a series to produce t's.

Another way to obtain the series is by utilizing the general recurrence relation for the numerators and denominators of convergents in Equ. (11.1). Recall:

$$\begin{aligned} p_k &= a_k p_{k-1} + p_{k-2}, \quad k = 2, ..., n \\ q_k &= a_k q_{k-1} + q_{k-2} \end{aligned}$$

In the case of ϕ, all of the a_k quotients = 1 and so the p_k and q_k relations simplify to Fibonacci series.

11.4.2 Convergent Relative Errors.

convergentsErr() is a slight variant of Listing 11.3 in cfLib.py which prints the relative error of each convergent in terms of the previous one. It shows a rapid decrease in the error, and also an oscillation around the 'true' value. For example, for π:

```
>>> convergentsErr(cfPi)
22/7 = 3.142857142857143 ; rel error: 0.047619047619047596
333/106 = 3.141509433962264 ; rel error: -0.00042881646655234076
355/113 = 3.1415929203539825 ; rel error: 2.6575247814301264e-05
103993/33102 = 3.1415926530119025 ; rel error: -8.50976198402835e-08
[(1, 1), (22, 7), (333, 106), (355, 113), (103993, 33102)]
```

Exercises

(1) There is a deep result which says that for rationals (and irrationals) the proportion of 1's, 2's, 3's, and 4's that occur in the quotients of the continued fraction will be in the approximate proportions $log_2(4/3) \approx 0.41504$, $log_2(9/8) \approx 0.16992$, $log_2(16/15) \approx 0.09311$, and $log_2(25/24) \approx 0.05890$.

These constants can be estimated by calculating the average number of 1's, 2's, 3's, and 4's in the quotient list, excluding the first integer term. For example, Listing 11.6 (quotFreqs.py) averages the number of 1's:

```
r = random.random()
print(r)
cs = cfList(r)
print(cs)
num1s = cs.count(1)
quotLen = len(cs)-1   # don't count leading 0
print(f"num 1s = {num1s}, est p1 = {num1s/quotLen:.5f}")
```

Listing 11.6. Counting quotients

Here are two runs of the program:

```
> python quotFreqs.py
0.2795543177484361
[0, 3, 1, 1, 2, 1, 2, 1, 6, 1, 2]
num 1s = 5, est p1 = 0.50000
```

```
> python quotFreqs.py
0.041800404500606736
[0, 23, 1, 12, 43, 3, 1, 7, 1, 3, 3]
num 1s = 3, est p1 = 0.30000
```

Modify Listing 11.6 to gather more empirical information over $n = 10000$ trials to see if the 1's, 2's, 3's, and 4's averages really do approach these constants.

(2) We can use the continued fraction for $\sqrt{d}$ to find the integer solutions of Pell's equations $x^2 - dy^2 = \pm 1$. It has been shown that the (x, y) solutions (if they exist at all) will be one of the convergent pairs (p_i, q_i).

Some outputs from our pell.py:

```
> python pell.py
d? 61
Success: 10 (29718, 3805) -1
Success: 21 (1766319049, 226153980) 1

> python pell.py
d? 109
Success: 14 (8890182, 851525) -1
Success: 29 (158070671986249, 15140424455100) 1
```

The two 'Success' lines indicate that solutions were found for both Pell forms (± 1). The bracketed values are the (x, y) solutions, and the initial integer is the index position of the matching convergent. This index is very suggestive of a general pattern for Pell equation solutions. What is that pattern?