4 Modes

The goals of this chapter are:

- to illustrate the experimental way that an engineer studies systems, even abstract, mathematical systems;
- to illustrate what modes are by finding them for the Fibonacci system; and
- to decompose second-order systems into modes, explaining the decomposition using operators and block diagrams.

The first question is what a mode is. That question will be answered as we decompose the Fibonacci sequence into simpler sequences. Each simple sequence can be generated by a first-order system like the leaky tank and is called a **mode** of the system. By decomposing the Fibonacci sequence into modes, we decompose the system into simpler, first-order subsystems.

The plan of the chapter is to treat the Fibonacci system first as a black box producing an output signal F and to develop computational probes to examine signals. This experimental approach is how an engineer studies even abstract, mathematical systems. The results from the probes will show us how to decompose the signal into its modes. These modes are then reconciled with what the operator method predicts for decomposing the system.

Why describe the experimental, and perhaps harder, method for finding the modes before giving the shortcuts using operators? We know the operator expression for the Fibonacci system, and could just rewrite it using algebra. The answer is that the operator method has meaning only after you feel modes in your fingertips, a feeling developed only as you play with signals. Without first playing, we would be teaching you amazing feats of calculation on meaningless objects.

Furthermore, the experimental approach works even when no difference equation is available to generate the sequence. Engineers often characterize such unknown or partially known systems. The system might be:

- *computational:* Imagine debugging someone else's program. You send in test inputs to find out how it works and what makes it fail.
- *electronic:* Imagine debugging a CPU that just returned from the fabrication run, perhaps in quantities of millions, but that does not correctly divide floating-point numbers [12]. You might give it numbers to divide until you find the simplest examples that give wrong answers. From that data you can often deduce the flaw in the wiring.
- *mathematical:* Imagine computing primes to investigate the twin-prime conjecture [16], one of the outstanding unsolved problems of number theory. [The conjecture states that there are an infinite number of prime pairs p, $p+2$, such as $(3,5)$, $(5,7)$, etc.] The new field of experimental mathematics, which uses computational tools to investigate mathematics problems, is lively, growing, and a fertile field for skilled engineers [4, 14, 8].

So we hope that, through experimental probes of the Fibonacci sequence, you learn a general approach to solving problems.

4.1 Growth of the Fibonacci series

Section 1.1.2 estimated how fast the sequence $f[n]$ grew. It seemed to grow geometrically with an order of growth between $\sqrt{2}$ and 2. Our first project is to experimentally narrow this range and thereby to guess a closed form for the order of growth.

One probe to find the order of growth is to compute the successive ratios $f[n]/f[n-1]$. The ratios oscillated around 1.618, but this estimate is not accurate enough to guess a closed form. Since the oscillations in the ratio die out as n grows, let's estimate the ratio accurately by computing it for large n. Our tool for these experiments – our probe – is a computer that we program in Python, a clean, widely available language. Use any tool that fits you, perhaps another language, a graphing calculator, or a spreadsheet.

Section 2.3.2 offered this Python code to compute $f[n]$:

```
def f(n):
  if n < 2: return 1
  return f(n-1) + f(n-2)
```

But the code is slow when n is large. Here are the running times to evaluate $f[n]$ on a Pentium CoreDuo 1.8GHz processor:

n	10	15	20	25	30
time (ms)	0.17	1.5	21	162	1164

The times grow rapidly!

Exercise 21. What is the running time of this implementation?

The times might seem low enough to be usable, but imagine being on a desert island with only a graphing calculator; then the times might be a factor of 10 or of 100 longer. We would like to build an efficient computational probe so that it is widely usable.

An efficient function would store previously computed answers, returning the stored answer when possible rather than recomputing old values. In Python, one can store the values in a dictionary, which is analogous to a hash in Perl or an associative array in `awk`. The memoized version of the Fibonacci function is:

```
memo = {}
def f(n):
  if n < 2      : return 1
  if n in memo : return memo[n]
  memo[n] = f(n-1) + f(n-2)
  return memo[n]
```

Pause to try 20. What is the running time of the memoized function, in big-Oh notation?

The new function runs in linear time – a faster probe! – so we can inexpensively compute $f[n]$ for large n. Here are the ratios $f[n]/f[n-1]$:

n	$f[n]/f[n-1]$
5	1.6000000000000009
10	1.6181818181818181817
15	1.6180327868852459
20	1.6180339985218033
25	1.6180339886704431
30	1.6180339887505408
35	1.6180339887498896
40	1.6180339887498949
45	1.6180339887498949

These values are very stable by $n = 45$, perhaps limited in stability only by the precision of the floating-point numbers.

Let's see what closed form would produce the ratio 1.6180339887498949 at $n = 45$. One source for closed forms is your intuition and experience. Another wonderful source is the Inverse Symbolic Calculator.
By using the Inverse Symbolic Calculator, you increase your repertoire of closed form and thereby enhance your intuition.

Pause to try 21. Ask the Inverse Symbolic Calculator about 1.6180339887498949.

The Inverse Symbolic Calculator thinks that 1.6180339887498949 is most likely the positive root of $x^2 - x - 1$ or, equivalently, is the golden ratio ϕ:

$$\phi \equiv \frac{1+\sqrt{5}}{2}$$

Let's use that hypothesis. Then

$$f[n] \sim \phi^n.$$

But we do not know the constant hidden by the $\sim$ symbol. Find that constant by using the Inverse Symbolic Calculator one more time. Here is a table of the ratio $f[n]/\phi^n$. With luck it converges to a constant. And it does:

n	$f[n]/\phi^n$
0	1.0000000000000000
10	0.7236250692647178 1
20	0.72360679895785285
30	0.72360679775005809
40	0.72360679774997805
50	0.72360679774997783
60	0.72360679774997749
70	0.72360679774997727
80	0.72360679774997705
90	0.72360679774997672
100	0.72360679774997649

Around $n = 10$, the ratios look like $\sqrt{3}-1 \approx 0.732$ but later ratios stabilize around a value inconsistent with that guess.

Pause to try 22. Ask the Inverse Symbolic Calculator about 0.72360679774997649. Which of the alternatives seem most reasonable?

The Inverse Symbolic Calculator provides many closed forms for 0.72360679774997649. A choice that contains $\sqrt{5}$ is reasonable since ϕ contains $\sqrt{5}$. The closed form nearest to 0.72360679774997649 and containing $\sqrt{5}$ is $(1+1/\sqrt{5})/2$, which is also $\phi/\sqrt{5}$. So the Fibonacci sequence is roughly

$$f[n] \approx \frac{\phi}{\sqrt{5}}\phi^n.$$

4.2 Taking out the big part from Fibonacci

Now let's **take out the big part** by peeling away the $\frac{\phi}{\sqrt{5}}\phi^n$ contribution to see what remains. Define the signal F_1 by

$$f_1[n] = \frac{\phi}{\sqrt{5}}\phi^n.$$

This signal is one mode of the Fibonacci sequence. The shape of a mode is its order of growth, which here is ϕ. The amplitude of a mode is the prefactor, which here is $\phi/\sqrt{5}$. The mode shape is a characteristic of the system, whereas the amplitude depends on the input signal (for this example, the input signal was the impulse). So we often have more interest in the shape than in the amplitude. However, here we need shape and amplitude in order to determine the signal and peel it away.

So tabulate the residual signal $F_2 = F - F_1$:

n	$f_2[n] = f[n] - f_1[n]$
0	+0.27639320225002106
1	−0.17082039324993681
2	+0.10557280900008426
3	−0.06524758424985277
4	+0.04032522475023104
5	−0.02492235949962307
6	+0.01540286525060708
7	−0.00951949424901599
8	+0.00588337100158753
9	−0.00363612324743201
10	+0.00224724775415552

The residual signal starts small and gets smaller, so the main mode F_1 is an excellent approximation to the Fibonacci sequence F. To find a closed form for the residual signal F_2, retry the successive-ratios probe:

n	$f_2[n]/f_2[n-1]$
1	−0.61803398874989446
2	−0.61803398874989601
3	−0.61803398874989390
4	−0.61803398874989046
5	−0.61803398874993953
6	−0.61803398874974236
7	−0.61803398875029414
8	−0.61803398874847626
9	−0.61803398875421256
10	−0.61803398873859083

The successive ratios are almost constant and look suspiciously like $1 - \phi$, which is also $-1/\phi$.

Exercise 22. Show that $1 - \phi = -1/\phi$.

So $f_2[n] \sim (-\phi)^{-n}$. To evaluate the amplitude, divide $f_2[n]$ by the mode shape $(-\phi)^{-n}$. Here is a table of those results:

n	$f_2[n]/(-\phi)^{-n}$
1	0.27639320225002090
2	0.27639320225002140
3	0.27639320225002101
4	0.27639320225001901
5	0.27639320225003899
6	0.27639320224997083
7	0.27639320225014941
8	0.27639320224951497
9	0.27639320225144598
10	0.27639320224639063

Those values stabilize quickly and look like one minus the amplitude of the ϕ^n mode. So the amplitude of the $(-\phi)^n$ mode is $1 - \phi/\sqrt{5}$, which is also $1/(\phi\sqrt{5})$. Thus the residual signal, combining its shape and amplitude, is

$$f_2[n] = \frac{1}{\phi\sqrt{5}}(-\phi)^{-n}.$$

Now combine the F_1 and F_2 signals to get the Fibonacci signal:

$$\begin{aligned} f[n] &= f_1[n] + f_2[n] \\ &= \frac{\phi}{\sqrt{5}}\phi^n + \frac{1}{\phi\sqrt{5}}(-\phi)^{-n}. \end{aligned}$$

This closed form, deduced using experiment, is the famous Binet formula for the n^{th} Fibonacci number.

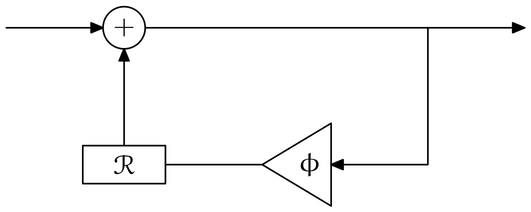
Exercise 23. Use peeling away and educated guessing to find a closed form for the output signal when the impulse is fed into the following difference equation:

$$y[n] = 7y[n-1] - 12y[n-2] + x[n].$$

4.3 Operator interpretation

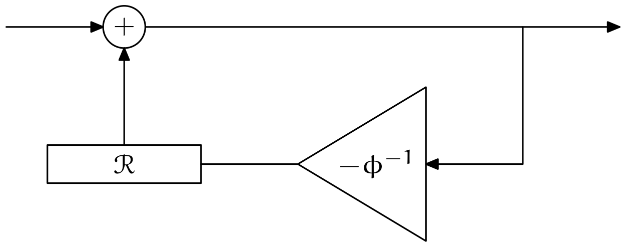
Next we interpret this experimental result using operators and block diagrams. Modes are the simplest persistent responses that a system can make, and are the building blocks of all systems, so we would like to find the operator or block-diagram representations for a mode.

The Fibonacci signal decomposed into two simpler signals F_1 and F_2 – which are also the modes – and each mode grows geometrically. Geometric growth results from one feedback loop. So the ϕ^n mode is produced by this system



with the system functional $(1-\phi\mathcal{R})^{-1}$.

The $(-\phi)^{-n}$ mode is produced by this system



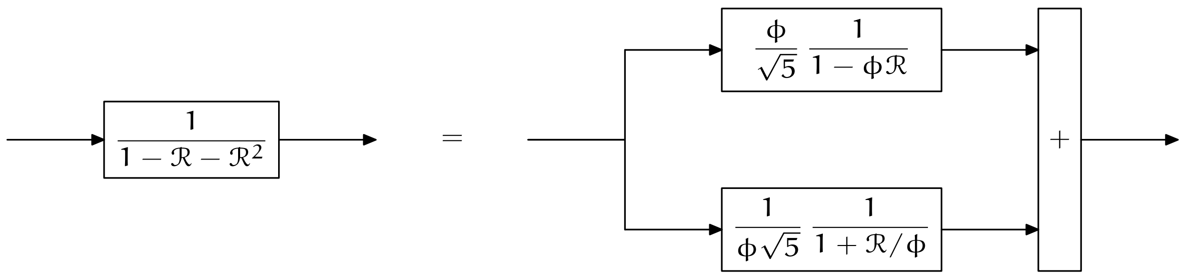
with the system functional $(1+\mathcal{R}/\phi)^{-1}$.

The Fibonacci system is the sum of these signals scaled by the respective amplitudes, so its block diagram is a weighted sum of the preceding block diagrams. The system functional for the Fibonacci system is a weighted sum of the pure-mode system functionals.

So let's add the individual system functionals and see what turns up:

$$\begin{aligned} F(\mathcal{R}) &= F_1(\mathcal{R}) + F_2(\mathcal{R}) \\ &= \frac{\phi}{\sqrt{5}}\frac{1}{1-\phi\mathcal{R}} + \frac{1}{\phi\sqrt{5}}\frac{1}{1+\mathcal{R}/\phi} \\ &= \frac{1}{1-\mathcal{R}-\mathcal{R}^2}. \end{aligned}$$

That functional is the system functional for the Fibonacci system derived directly from the block diagram (Section 3.5.2)! So the experimental and operator approaches agree that these operator block diagrams are equivalent:


$\frac{1}{1-\mathcal{R}-\mathcal{R}^2}$
=
$\frac{\phi}{\sqrt{5}}\frac{1}{1-\phi\mathcal{R}}$
$\frac{1}{\phi\sqrt{5}}\frac{1}{1+\mathcal{R}/\phi}$
+

where, to make the diagram easier to parse, system functionals stand for the first- and second-order systems that they represent.

Exercise 24. Write the system of difference equations that corresponds to the parallel-decomposition block diagram. Show that the system is equivalent to the usual difference equation

$$f[n] = f[n-1] + f[n-2] + x[n].$$

The equivalence is obvious neither from the block diagrams nor from the difference equations directly. Making the equivalence obvious needs either experiment or the operator representation. Having experimented, you are ready to use the operator representation generally to find modes.

4.4 General method: Partial fractions

So we would like a way to decompose a system without peeling away and guessing. And we have one: the method of partial fractions, which shows the value of the operator representation and system functional. Because the system functional behaves like an algebraic expression – or one might say, because it is an algebraic expression – it is often easier to manipulate than is the block diagram or the difference equation.

Having gone from the decomposed first-order systems to the original second-order system functional, let's now go the other way: from the original system functional to the decomposed systems. To do so, first factor the $\mathcal{R}$ expression:

$$\frac{1}{1-\mathcal{R}-\mathcal{R}^2} = \frac{1}{1-\phi\mathcal{R}}\,\frac{1}{1+\mathcal{R}/\phi}.$$

This factoring, a series decomposition, will help us study poles and zeros in a later chapter. Here we use it to find the parallel decomposition by using the technique of partial fractions.

The partial fractions should use the two factors in denominator, so guess this form:

$$\frac{1}{1-\mathcal{R}-\mathcal{R}^2} = \frac{a}{1-\phi\mathcal{R}} + \frac{b}{1+\mathcal{R}/\phi},$$

where a and b are unknown constants. After adding the fractions, the denominator will be the product $(1-\phi\mathcal{R})(1+\mathcal{R}/\phi)$ and the numerator will be the result of cross multiplying:

$$a(1+\mathcal{R}/\phi) + b(1-\phi\mathcal{R}) = a + (a/\phi)\mathcal{R} + b - b\phi\mathcal{R}.$$

We want the numerator to be 1. If we set $a = \phi$ and $b = 1/\phi$, then at least the $\mathcal{R}$ terms cancel, leaving only the constant $a + b$. So we chose a and b too large by the sum $a + b$, which is $\phi + 1/\phi$ or $\sqrt{5}$. So instead choose

$$\begin{aligned} a &= \phi/\sqrt{5}, \\ b &= 1/(\phi\sqrt{5}). \end{aligned}$$

If you prefer solving linear equations to the guess-and-check method, here are the linear equations:

$$\begin{aligned} a + b &= 1, \\ a/\phi - b\phi &= 0, \end{aligned}$$

whose solutions are the ones deduced using the guess-and-check method.

The moral: To find how a system behaves, factor its system functional and use partial fractions to decompose that factored form into a sum of first-order systems. With that decomposition, you can predict the output signal because you know how first-order systems behave.

You can practice the new skill of decomposition with the following question:

Exercise 25. Look again at the system

$$y[n] = 7y[n-1] - 12y[n-2] + x[n].$$

Decompose the operator representation into a sum of two modes and draw the corresponding block diagram (using block diagram elements). When the input signal X is the impulse, do the operator and block-diagram decompositions produce the same closed form that you find by peeling away and guessing?