Parametric L-Systems and borderline fractals

A preprint version of a “Mathematical graphics” column from Mathematica in Education and Research.

Mark McClure

Department of Mathematics
University of North Carolina at Asheville
Asheville, NC 28804

mcmcclure@unca.edu

Abstract

L-systems have been widely used to describe fractal curves and generate realistic images of plants. In this column, we discuss L-systems that depend upon a parameter and their application to borderline fractals.

Note: To reduce the size of the file, the graphics in this file have all been converted to bitmap form. Of course, they may be regenerated within *Mathematica*.

■ Initialization

1. Introduction

An L-system is a type of tool that can be used to generate artificial plants, fractal curves, and other complicated objects on a computer. The basic components of an L-system are an initial string or axiom, a list of replacement rules that act recursively on the axiom to construct more complicated strings, and a technique to generate an image from these strings. While the axiom and replacement rules might be very simple, recursive application of the replacement rules can generate very complicated strings which, in turn, can lead to interesting graphics.

Translation of the string into a graphic is sometimes called *turtle graphics* due to the following interpretation. Imagine you hand a string to a turtle who interprets the individual characters as instructions. Typically, "F" is interpreted to mean step forward one unit and draw a line, "+" is interpreted to mean rotate counter-clockwise through some pre-specified angle (say 60°), and "-" is interpreted to mean rotate clockwise through the same angle. Given this interpretation, the string "$F -- F -- F$" would generate an equilateral triangle.

Now suppose that each "F" in the axiom is replaced by a more complicated string, say "$F + F -- F + F$", and this process is repeated several times. As we can see using **`Nest`** and **`StringReplace`**, this generates a much more complicated string.

```
axiom = "F--F--F";
replacementRule = "F" → "F+F--F+F";
Nest[StringReplace[#, replacementRule] &,
 axiom, 2]
```

```
F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+
  F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F
```

The recursive nature of this process has led to a very complicated string and turtle interpretation of such strings can lead to stunning images.

The notion of an L-system originated in the work of Aristid Lindenmayer and was further developed by Lindenmayer, Prusinkiewicz, and others. Their main objective was to obtain an efficient means to model plant growth on a computer, but L-systems are also a natural way to describe many fractal curves. The definitive description of the L-system concept is contained in [1] and an L-system generator based on this work is available online [2]. There are a number of L-system implementations in *Mathematica* as well (for example, [3,4,5]). The code described in this column is based in large part on the work of Wagon [5], although a number of extensions have been made. In particular, we focus on L-systems where more complicated substitutions involving a parameter are involved and investigate how these can lead to borderline fractals.

2. Implemention of parametric L-systems

The basic idea behind a parametric L-system is to include a parameter and allow the substitution rule to affect the parameter as well. We'll illustrate the technique by generating a famous fractal set called the Koch snowflake. The typical L-system technique to generate this set is to iteratively apply the simple rule "F" → "$F + F -- F + F$" starting from the axiom "$F -- F -- F$". Using a parametric approach, we could represent the rule as "$F(x)$" → "$F(x/3)\, r(\pi/3)\, F(x/3)\, r(-2\,\pi/3)\, F(x/3)\, r(\pi/3)\, F(x/3)$" and the axiom as "$F(1)\, r(-2\,\pi/3)\, F(1)\, r(-2\,\pi/3)\, F(1)$". The turtle could then interpret the parameters as step size, rotation angle, or some other appropriate geometric quantity. While this is a bit more complicated, it is also more flexible, allowing more general step sizes and rotations angles to be used.

To implement this in *Mathematica*, it makes more sense to work with a list of expressions rather than with a string, even though L-systems were initially defined in terms of strings. That is, it's easier to work in *Mathematica* with **F[x_]** than it is to work with "F(x)". Thus our fundamental objects will be lists of *Mathematica* expressions corresponding to turtle directives. The two simplest turtle directives are **F[x]**, meaning move forward x units and draw a line, and **r[θ]**, meaning rotate through the angle θ. Thus the axiom and rule for the Koch snowflake can be represented as follows.

```
axiom = {F[1], r[-2 π / 3], F[1], r[-2 π / 3], F[1]};
KochRule = F[x_] :→ {F[x / 3], r[π / 3], F[x / 3],
    r[-2 π / 3], F[x / 3], r[π / 3], F[x / 3]};
```

We now iteratively apply the rule to the axiom and store the result as **instructions**.

```
instructions = Flatten[Nest[# /. KochRule &, axiom, 5]];
```

The variable **instructions** now contains a fairly complicated list. In [5], Wagon devised an elegant way to interpret this as a graphic. We first define a few variables to contain the image, specify the location and direction of the turtle, and define a rotation matrix.

```
lines = {};
lastpt = {0, 0};
dir = {1, 0};
```

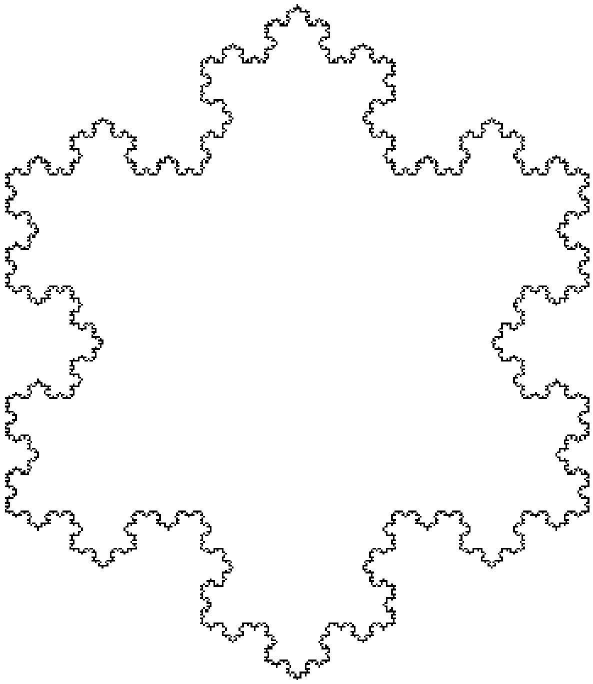
```
rotate[θ_] := N[{
    {Cos[θ], -Sin[θ]},
    {Sin[θ], Cos[θ]}}];
```

We now define a list of substitutions that will be performed on the **instructions**. Note that these substitutions will be performed in order on each element in the list **instructions** and each substitution results in a modification of **lines**, **lastpt**, or **dir**.

```
turtleInterpretation = {
   F[x_] :→ (lines = {Line[{lastpt, lastpt += x dir}], lines}),
   r[t_] :→ (dir = rotate[t].dir;)};
```

Finally, we perform the substitutions and display the resulting image contained in **lines**.

```
instructions /. turtleInterpretation;
Show[Graphics[lines],
  AspectRatio → Automatic];
```



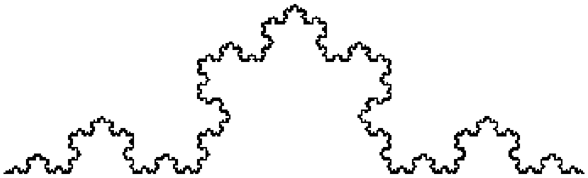
3. Examples

The algorithm described in the previous section has been encapsulated in the **ShowPLSystem** command defined in the **PLSystem** package. We now load this package, after removing a couple of symbols we've used to avoid shadowing.

```
Remove[r, F];
Needs["PLSystem`"];
```

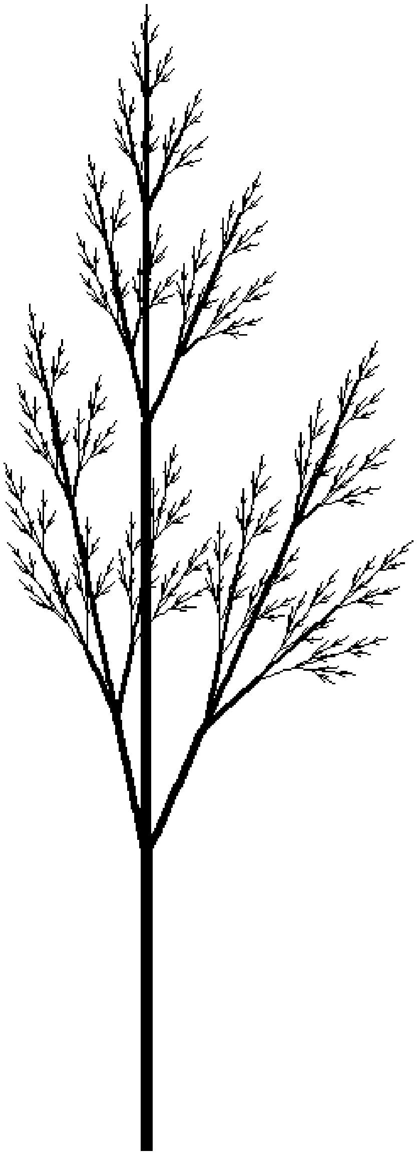
The package recognizes the standard string notation used by many L-system implementations, provided no parameters are used. Thus, here is a look at the Koch curve, which is part of the Koch snowflake.

```
ShowPLSystem["F" → "F+F--F+F", "F", 6,
  Angle → π / 3];
```



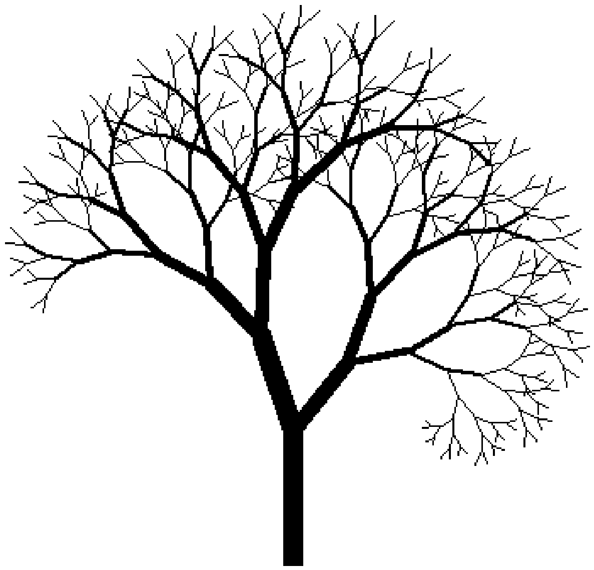
While the main objective in this column is to explore borderline fractals, care has been taken to make the package capabilities and notation consistent with [1] where possible. We refer the interested reader to [1] and give a few more examples here. For example, open and closed brackets ("[" and "]") may be used to construct branching structures, "!" may be used to decrement the width segments by a pre-specified factor, and miscellaneous symbols may be used in the expansion rules, which are then subsequently ignored by the turtle.

```
ShowPLSystem[{"F" → "FF",
   "X" → "F![+!X][--!X]FX"}, "X", 7,
  Angle → 12 Degree,
  InitialDirection → {0, 1},
  InitialWidth → .03,
  WidthRatio → .8];
```



When parameters do appear, we must represent the rules and axiom usings lists of *Mathematica* expressions. Again, this allows much greater flexibility and, therefore, variety. In the following example, **A[l,w]** might represent one type of branch with a specified length and width, while **B[l,w]** represents another type. The types are very similar, but differ slightly in how they decompose into smaller branches. Note that "+"/"-", "!", "[", and "]" have been replaced by **r**, **setWidth**, **push**, and **pop** respectively.

```
ShowPLSystem[{
   A[l_, w_] ⧴ {setWidth[w], F[l],
     push, r[24 Degree], B[0.75 l, 0.7 w], pop,
     push, r[-22 Degree], A[0.85 l, 0.8 w], pop},
   B[l_, w_] ⧴ {setWidth[w], F[l],
     push, r[19 Degree], A[0.8 l, 0.8 w], pop,
     push, r[-39 Degree], B[0.7 l, 0.6 w], pop}},
  B[1, .035], 9,
  InitialDirection → {0, 1},
  PlotRange → All];
```

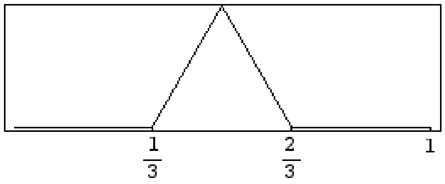


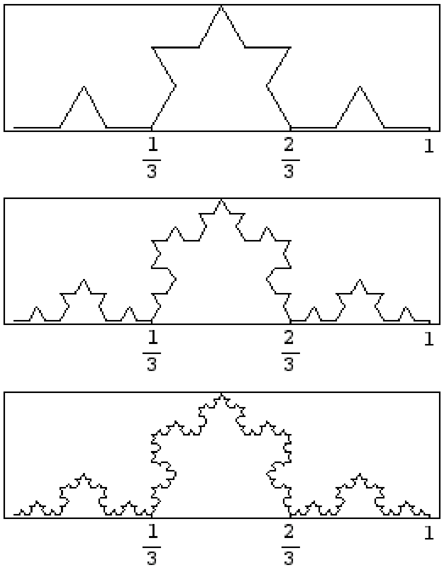
All the turtle directives for two-dimensional graphics described on page 209 of [1] are supported. The **TurtleDirectiveButtons[]** command returns a list of the turtle directives with pointers to their usage statements.

4. Borderline fractals

To introduce the concept of a borderline fractal, we begin by looking at several approximations to Koch's curve using the parametric formulation.

```
KochRule = F[t_] ⧴
   {F[t / 3], r[π / 3], F[t / 3], r[-2 π / 3],
    F[t / 3], r[π / 3], F[t / 3]};
KochAxiom = F[1];
ShowPLSystem[KochRule, KochAxiom, #,
    FrameTicks → {{1 / 3, 2 / 3, 1}, {}, {}, {}},
    Frame → True] & /@ Range[4];
```





Clearly, these curves are converging to a limit curve, which we call the Koch curve. By all accounts, this is an example of a fractal curve, but what exactly is a fractal curve? The definition suggested by Mandelbrot essentially boils down to this: a fractal curve is one whose fractal dimension is larger than one. As we will see, however, there are curves naturally described using parametric L-systems whose fractal dimension is one, but that are qualitatively similar to the Koch curve. A related way to classify a curve is in terms of local rectifiability.

■ Rectifiability

A curve is said to be rectifiable, if it is of finite length. Furthermore, there is a natural way to describe the length of the curves generated by L-systems. Suppose that the n^{th} level approximation of a curve described by a parametric L-system consists of N_k segments of length ℓ_k. Then the length of the n^{th} level approximation is simply $N_k \; \ell_k$. The limit of this expression as $k \to \infty$ (possibly infinite) is defined to be the length of the curve. For the Koch curve, we have $N_k = 4^k$ and $\ell_k = 3^{-k}$. Thus the length of the Koch curve is $\lim_{k\to\infty} (4/3)^k = \infty$. Furthermore, the length of any connected piece of the Koch curve containing more than a single point is also infinite. A curve having this property is said to be non-locally rectifiable and we might consider such a curve to be fractal.

■ Dimension

For a curve of infinite length, the quantities N_k and ℓ_k can be used to define a notion of fractal dimension which distinguishes the relative sizes of infinitely long curves. In fact the divider dimension d (see [6]) is simply

$$d = -\lim_{k\to\infty} \frac{\log N_k}{\log \ell_k}.$$

Although we won't go into an explanation of why this is a reasonable definition here, we can show that if the curve is rectifiable, $d = 1$ as we would expect. If the curve is rectifiable, then there is some positive finite number L (the length) so that $N_k \; \ell_k \to L$ as $k \to \infty$. If we simply take the logarithm of both sides and solve for $-\log(N_k)/\log(\ell_k)$, we get

$$-\frac{\log N_k}{\log \ell_k} \sim 1 - \frac{L}{\log \ell_k} \to 1,$$

as expected.

Now applying this formula to the Koch curve we find that the dimension is

$$d = -\lim_{k\to\infty} \frac{\log 4^k}{\log 3^{-k}} = -\frac{k}{-k} \lim_{k\to\infty} \frac{\log 4}{\log 3} = \frac{\log 4}{\log 3} \approx 1.262\,.$$

Thus, we'd certainly call this a fractal curve.

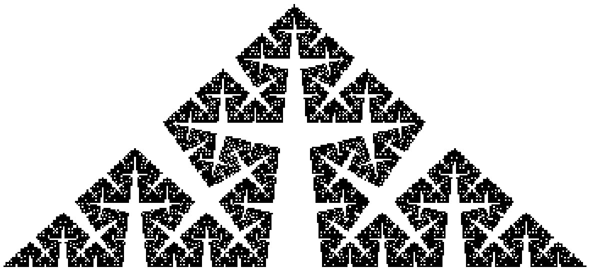
■ More examples

We now generalize the Koch curve construction to obtain curves of other dimensions and fractal-like curves which have dimension one. (Thus the term borderline fractal.) Suppose that $\{\ell_k\}_{k=0}^{\infty}$ is a sequence so that $\ell_0 = 1$ and $\ell_k / 4 \leq \ell_{k+1} < \ell_k / 2$. Note that such a sequence must decrease monotonically to zero. Given such a sequence, there is a natural generalization of the Koch curve so that the n^{th} level approximation consists of 4^k pieces of length ℓ_k. The zeroth level approximation is simply the unit interval and the $(n+1)^{\text{st}}$ level approximation is obtained from the n^{th} level approximation by replacing each segment of length ℓ_k with four segments of length ℓ_{k+1} in the same manner as for the Koch curve. A precise description may be given using a parametric L-system.

```
curveRule = F[x_, k_] :→
   {F[a[k] x, k + 1], r[ArcCos[1 / (2 a[k]) - 1]],
    F[a[k] x, k + 1], r[-2 ArcCos[1 / (2 a[k]) - 1]],
    F[a[k] x, k + 1], r[ArcCos[1 / (2 a[k]) - 1]],
    F[a[k] x, k + 1]};
curveAxiom = F[1, 0];
```

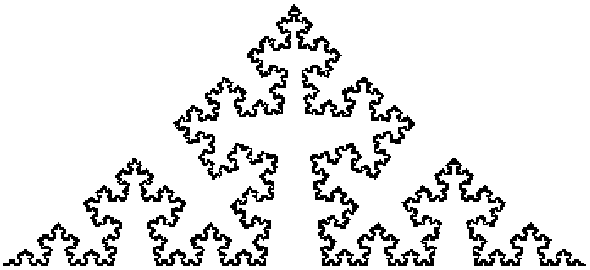
Note that the term **a[k]** indicates how the length scales from the level k to level $k+1$. Thus to generate the Koch curve we could take **a[k]=1/3**, so that $\ell_k = 1/3^k$. We can generate a curve of any dimension we like by choosing **a[k]** to be the appropriate constant between $1/4$ and $1/2$. For example, here is a curve of dimension $-\log(4)/(\log(.45)) \approx 1.736$. Note how thick this curve seems compared to the Koch curve.

```
a[k_] := .45;
ShowPLSystem[curveRule, curveAxiom, 7];
```



If **a[k]=1/4**, then $\ell_k = 1/4^k$ and the dimension is one. In fact, the limit curve is simply the unit interval. Now suppose that **a[k]** is the k^{th} term in a sequence decreasing to $1/4$. Then we might expect the dimension to be one, but the curve could still be non-rectifiable. Here's an example of such a curve using a parametric L-system.

```
a[k_] := 1 / 4 + 1 / (k + 5);
ShowPLSystem[curveRule, curveAxiom, 7];
```



Note that the curve still appears very fractal in nature. It is non-rectifiable; in fact, the length of the n^{th} level approximation is the following expression.

```
          n
ℓ[n_] :=  ∏ a[k];
         k=1
totalLength = 4^n ℓ[n]
```

$$\frac{(6+n)\ (7+n)\ (8+n)\ (9+n)}{3024}$$

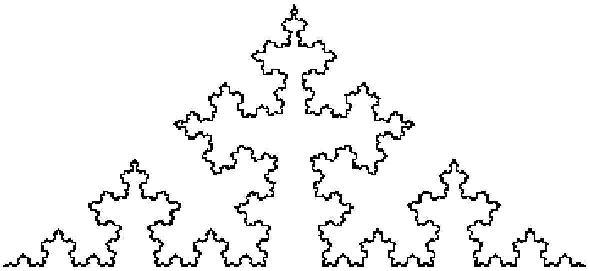
Note that as $n \to \infty$, the total length diverges to infinity. However, the rate of divergence is only comparable to n^4. This rate would have to be exponential to force the fractal dimension to be larger than 1. In fact, *Mathematica* can compute the dimension.

```
Limit[-Log[4^n] / Log[ℓ[n]], n → ∞]

1
```

If $\ell_n \to 0$ a little faster, then the curve may even be rectifiable. Consider the following example.

```
a[k_] := 1 / 4 + 1 / (k^2 + 5);
ShowPLSystem[curveRule, curveAxiom, 7];
```



We can compute the length of the n^{th} level approximation in the same manner.

```
          n
ℓ[n_] :=  ∏ a[k];
         k=1
totalLength = 4^n ℓ[n]
```

$$\left(\sqrt{5}\ \mathrm{Csch}[\sqrt{5}\ \pi]\ \mathrm{Gamma}[(1-3\,\mathrm{i})+n]\ \mathrm{Gamma}[(1+3\,\mathrm{i})+n]\ \mathrm{Sinh}[3\,\pi]\right) \Big/ \left(3\ \mathrm{Gamma}[1-\mathrm{i}\,\sqrt{5}+n]\ \mathrm{Gamma}[1+\mathrm{i}\,\sqrt{5}+n]\right)$$

Although this is a more complicated expression, *Mathematica* can still compute the length of the limit curve.

```
Limit[4^n ℓ[n], n → ∞]
```

$$\frac{1}{3}\sqrt{5}\,\mathrm{Csch}[\sqrt{5}\,\pi]\,\mathrm{Sinh}[3\,\pi]$$

Of course, the dimension should still be one.

```
Limit[-Log[4^n] / Log[ℓ[n]], n → ∞]
```

```
1
```

We invite the reader to construct a curve with dimension two and area zero.

References

1. Lindenmayer, A. and Prusinkiewicz, P., *The Algorithmic Beauty of Plants*. Spring-Verlag, New York, 1990.

2. Prusinkiewicz, P., *The Virtual Laboratory*. http://algorithmicbotany.org/virtual_laboratory/.

3. Jacob, C. Modeling growth with L-systems & *Mathematica*. *Mathematica* in Education and Research, **4** #3 (1995) 12-19.

4. Maeder, R., *The Mathematica programmer* (Chapter 8). AP Professional, Boston, 1994.

5. Wagon, S., *Mathematica in action,* 2nd ed. (Chapter 7). Spring-Verlag, New York, 1999.

6. Falconer, K., *Fractal Geometry: Mathematical Foundations and Applications* (Section 3.5). John Wiley & Sons, New York, 1990.