Bifurcation sets and critical curves

A preprint version of a “Mathematical graphics” column from

Mathematica in Education and Research.

Mark McClure

mcmcclure@unca.edu

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

Abstract

The bifurcation diagram is an iconic image of iterative real dynamics. It's structure can be illuminated using a family of polynomial curves, called critical curves. We can also apply these polynomials to pinpoint the locations of periodic components in the Mandelbrot set, the complex analog of the bifurcation diagram.

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. Real dynamics and the bifurcation diagram

■ 1.1 Basic iteration

This issue’s column returns to the topic of iterative dynamical systems, a subject rich in beautiful graphics. An excellent general introduction to this topic is [1]. In iterative dynamics, we study the iteration of a function f mapping a set to itself. In real dynamics, the iteration is performed on the set of real numbers $\mathbb{R}$, or perhaps an interval in $\mathbb{R}$. In complex dynamics, we iterate in $\mathbb{C}$, the complex numbers. Generally, we are interested in the long term behavior of an orbit $\{x_0, x_1, x_2, ...\}$, where x_0 is an arbitrary seed and $x_{n+1} = f(x_n)$. This exactly the type of sequence generated by **NestList** and related *Mathematica* commands. Since we are interested in the long term behavior of the orbit, we might generate a large number of terms, but only look at the last few.

Suppose, for example, that we are interested in the orbit of 0 under the function $f(x) = x^2 - 0.625$. We can examine this as follows.

```
f[x_] := x^2 - 0.625;
Take[NestList[f, 0., 100], -10]
```

```
{-0.435415, -0.435414, -0.435415, -0.435414,
 -0.435415, -0.435414, -0.435415, -0.435414, -0.435415, -0.435414}
```

It certainly appears that this orbit has converged to a fixed point. If $f(x) = x^2 - 0.9$, then the orbit of 0 converges to an orbit of period 2.

```
f[x_] := x² - 0.9;
Take[NestList[f, 0., 100], -10]

{-0.887298, -0.112702, -0.887298, -0.112702,
 -0.887298, -0.112702, -0.887298, -0.112702, -0.887298, -0.112702}
```

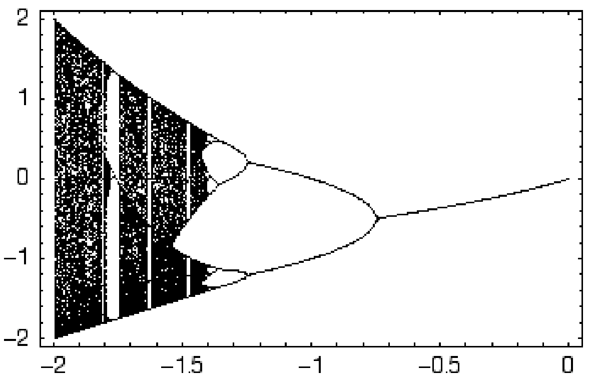
Experimentation with the orbit of 0 under functions of the form $f(x) = x^2 + c$ suggests a variety of possibilities. For example, $c = -1.625$ yields an orbit of period 5 and $c = -1.9$ yields no discernable pattern. The bifurcation diagram is an image which organizes and clarifies the possibilities.

▪ 1.2 Generating a simple bifurcation diagram

The idea behind the bifurcation diagram is fairly simple. Suppose that the family is $f_c(x) = x^2 + c$. Fix a c value. We iterate the function f_c from the starting point $x_0 = 0$ a large number of times and drop the first few iterates. This list represents the eventual orbit of 0 for the function f_c. We do this for a range of c values and then plot, for each c on the horizontal axis, the eventual orbit on the vertical axis. A intriguing pattern illustrating the dynamics then appears.

To implement this, we start by defining the family of functions in question and a function `eventualOrbit` which computes the eventual orbit for any of the given functions. We then make a table of eventual orbits and pair each term in an orbit with the corresponding c value. Finally, we plot the orbit data.

```
f_c_[x_] := x² + c;
eventualOrbit[c_] := Drop[NestList[f_c, 0.,
    500], 200];
orbitData = Table[{c, #} & /@ eventualOrbit[c],
   {c, -2, 0, .005}];
Show[Graphics[{PointSize[.003],
    orbitData /. p : {_?NumericQ, _?NumericQ} → Point[p]
   }]];
```



This is a very instructive image. It suggests that for c between -0.75 and 0 that f_c has a single attractive orbit. As c decreases past -0.75 that attractive orbit bifurcates into an attractive orbit of period 2. As c continues to decrease, further bifurcations occur. Past a certain point, chaos seems to reign in the very dark region. There appears to be strips of calm within the chaos, however.

■ 1.3 Critical polynomials

Despite the appeal of this basic image, it's fairly murky and suggests as many questions as it answers. How are we to interpret the dark, chaotic region? What of the relative strips of calm? There appear to be substructures enveloped by curves; can we quantify these? Critical polynomials provide answers to these questions, as eloquently described in [2].

To understand critical polynomials, we need to introduce a bit more theory. Note that in all of our examples and in the construction of the bifurcation diagram, we started the iteration at 0. The reason for this is that 0 is the only critical point of f_c and critical points dominate the dynamical behavior of iteration. One example of the significance of critical points is provided by the following theorem, which is essentially theorem 9.3.1 in [3]

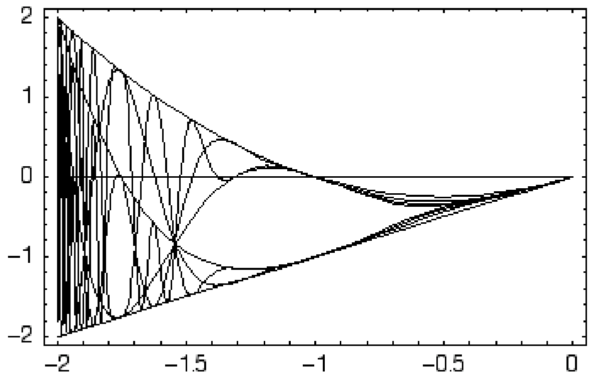
> Suppose the polynomial function f has an attractive periodic orbit. Then that orbit attracts a critical point of f.

Thus for $f_c(x) = x^2 + c$, we see that iteration starting from 0 is guaranteed to detect the presence of an attractive orbit. There is also an algebraic technique to find attractive orbits and this is where critical polynomials arise. Let f^n denote n-fold composition of the function f with itself. Then x_0 has period n for f if $f^n(x_0) = x_0$, but $f^m(x_0) \neq x_0$ for $m = 1, 2, ..., n-1$. Furthermore, it can be proved that this orbit is attractive precisely if $|f^{n\prime}(x_0)| < 1$. When $f^{n\prime}(x_0) = 0$, the orbit is called super-attractive. By the chain rule, $f^{n\prime}(x_0) = f'(x_0) f'(x_1) \cdots f'(x_{n-1})$, where $\{x_0, x_1, \ldots, x_{n-1}\}$ is the orbit of x_0. Thus an orbit is super-attractive if and only if there is a critical point in the orbit.

Now, the nth critical polynomial $Q_n(c)$ is defined to be the nth iterate of f_c at 0, i.e. $Q_n(c) = f_c^n(0)$. Note that f_c has a super-attractive orbit of period n if and only if $Q_n(c) = 0$ but $Q_m(c) \neq 0$ for any $m = 1, ..., n$. This is because $x = 0$ must be an element of any super-attractive orbit of f_c, thus $f_c^n(0) = Q_n(c) = 0$.

We now plot the critical polynomials. It turns out that there is a beautiful relationship between these curves and the bifurcation diagram; the critical curves form a “road map”, as it is called in [2].

```
f_c_[x_] := x^2 + c;
Q_n_[c_] := Nest[f_c, 0, n];
Plot[Table[Q_n[c], {n, 0, 8}] // Evaluate,
  {c, -2, 0}];
```



Recall that the roots of these polynomials locate windows of attractive periodic behavior in the bifurcation diagram. Furthermore, the critcal curve corresponding to the polynomial Q_0 is exactly the x-axis. Thus we see that intermingled with the chaotic regions are infinitely many regions of stable behavior. These regions of stability occur precisely where the curve Q_0, the horizontal line, intersect the other curves. The relationship between these roots and the windows can be seen most clearly by plotting the critical curves and the bifurcation diagram together.

There are a couple of points worth considering as we plot our final bifurcation diagram. First, for larger c values, larger than say c_4 where the period 4 bifurcation occurs, we don't need the critical curves to clarify anything. Thus we will use a very simple strategy for $c \geq c_4$ and a more detailed strategy for $c < c_4$. It's simple to find the approximate location of c_4 using the critical polynomial Q_4.

```
c4 = c /. FindRoot[Q₄[c] == 0, {c, -1.3}]
```

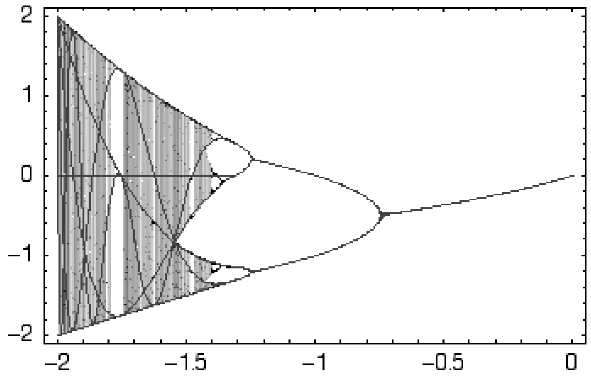
```
-1.3107
```

The other point to consider is that we need to shade the bifurcation diagram gray so that we can see the critical curves. In fact, we can use this as an opportunity to highlight the density of the orbits in the bifurcation diagram. We will do this by plotting a fairly large number of points, breaking each orbit into bins of points close to one another (as measured by the variable **`res`**), and using the **`Frequencies`** command in the **`Statistics`DataManipulation``** package to determine the local density of the orbit. Finally, the points are shaded according to this local density. The following code implements this plan.

```
Needs["Statistics`DataManipulation`"];
eventualOrbit[c_] := {c, #} & /@
    Drop[NestList[f_c, 0., 500], 100] /; c ≥ c4;
eventualOrbit[c_] := {c, #} & /@
    Drop[NestList[f_c, 0, 2000], 500] /; c < c4;
shadedColumn[c_] := {GrayLevel[.2],
    Point /@ eventualOrbit[c]} /; c ≥ c4;
res = 2^12;
shadedColumn[c_] := Module[
    {points, countedPoints,
     maxFrequency, shadedPoints},
    points = N@Floor[res eventualOrbit[c]] / res;
    countedPoints = Frequencies[points];
    maxFrequency = Max[First /@ countedPoints];
    shadedPoints = {GrayLevel[N[(1 - (First[#] / maxFrequency))]^3],
    Point[Last[#]]} & /@ countedPoints] /; c < c4;
shadedOrbits = Table[shadedColumn[c], {c, -2, 0, .005}];
bifPic = Graphics[{PointSize[0.002], shadedOrbits}];
```

To put everything together, we replot the critical curves over an appropriate interval and shade them a bit darker than the bifurcation diagram.

```
criticalCurves = Plot[Table[Qn[c], {n, 0, 6}] // Evaluate,
    {c, -2, c4}, DisplayFunction → Identity] /.
   Graphics[g_, opts__] → Graphics[{
      GrayLevel[0.2], g}, opts];
Show[{bifPic, criticalCurves}];
```



Note how the critical curve Q_0 intersects other critical curves where the stable periodic windows occur.

2. Critical polynmials and complex dynamics

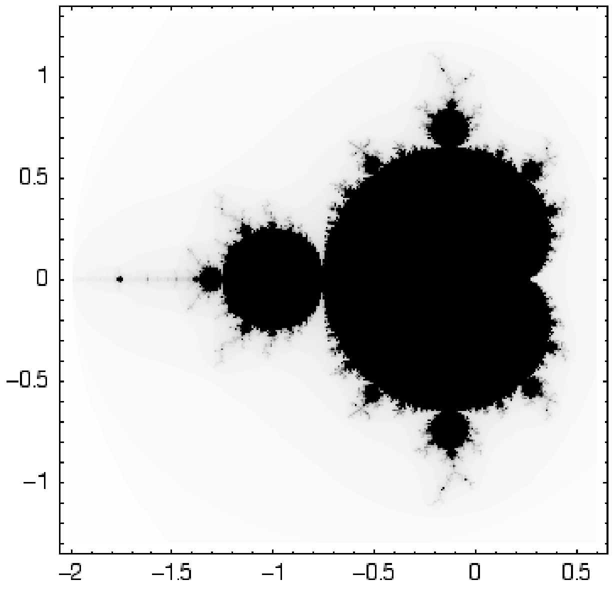
■ 2.1 Complex dynamics and the Mandelbrot set

Iteration may be performed in the complex plane just as it may be performed on the real line, although different visualization techniques need to be used. The Mandelbrot set is analagous to the bifurcation diagram in this setting. Suppose we iterate the complex function $f_c(z) = z^2 + c$ starting from the critical point $z = 0$. On a very coarse level, it can be proved (as in section 17.1 of [1]) that there are only two possible end behaviors:

1) The orbit of 0 stays bounded by 2 or
2) The orbit of 0 diverges to infinity.

The Mandelbrot set is defined to be the set of all c values which falls into the first of these categories. This dichotomy can be used to plot an image of the Mandelbrot set fairly easily. Note that some version of the following code appears in many other places, for example [4]. We choose code which is as simple as possible for our purposes. The basic idea is to use **`NestWhileList`** to iterate f_c while the orbit stays bounded by 2, or until some maximum number (100 in this case) of iterations is reached. We then use **`DensityPlot`** to show the results.

```
MandelbrotCount[c_] := Length[NestWhileList[
    #² + c &, N[c], Abs[#] ≤ 2 &, 1, 100]];
MandelbrotPic = DensityPlot[-MandelbrotCount[x + ⅈ y],
   {x, -2, .6}, {y, -1.3, 1.3}, PlotPoints → 300,
   Mesh → False, AspectRatio → Automatic
];
```



As is well known, this image consists of a main cardioid together with a number of circles, or bulbs, attached to it. Off of each decorative bulb hangs another sequence of such bulbs, etc. If we zoom in, we can see smaller copies of the whole set repeated. In fact, the interiors of these components all correspond to different types of periodic behavior exhibited by f_c for various choices of c. For example, if c is chosen from the main cardoid, then f_c will have an attractive fixed point. If c is chosen from the large circle just to the left of the main cardioid, then f_c will have an attractive orbit of period 2.

■ 2.2 Critical polynomials and the Mandelbrot set

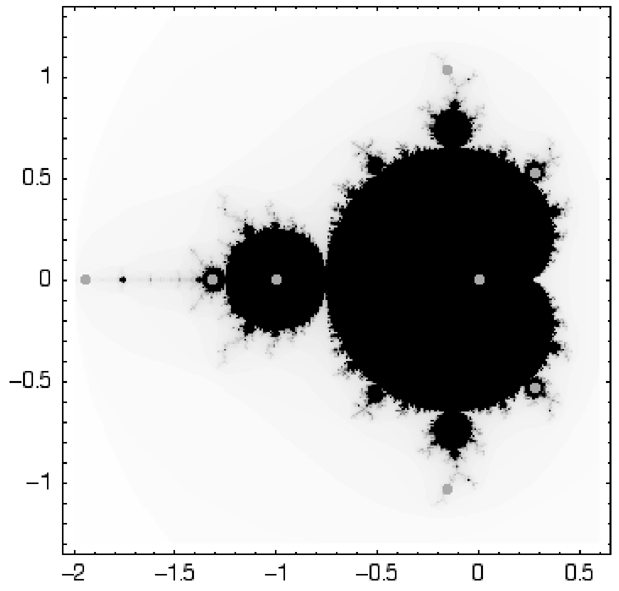
The critical polynomials may be used to pinpoint the locations of components in the Mandelbrot set correpsonding to various types of periodic behaviour. To illustrate this statement, consider the roots of the polynomial $Q_4(c)$.

```
f_c_[z_] := z² + c;
Q_n_[c_] := Nest[f_c, 0, n];
complexPoints = c /. NSolve[Q₄[c] == 0, c]
```

```
{-1.9408, -1.3107, -1., -0.15652 - 1.03225 ⅈ,
 -0.15652 + 1.03225 ⅈ, 0., 0.282271 - 0.530061 ⅈ, 0.282271 + 0.530061 ⅈ}
```

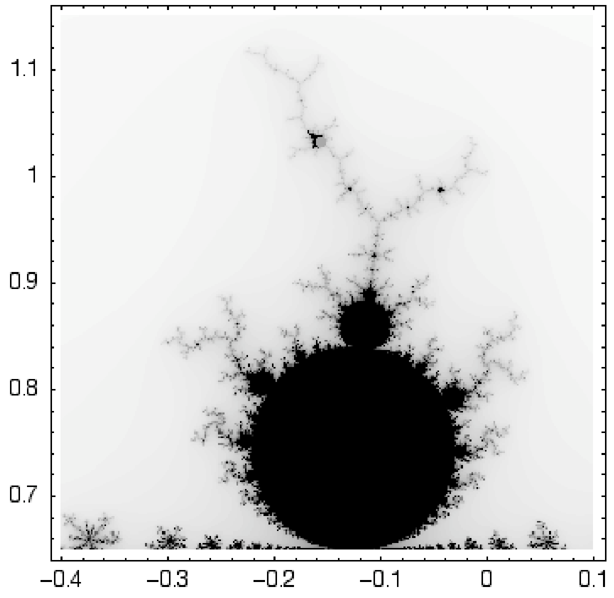
Note that there are eight roots, as expected. We can check their locations in the Mandelbrot set.

```
planePoints = {Re[#], Im[#]} & /@ complexPoints;
Show[MandelbrotPic,
  Epilog → {PointSize[.02], GrayLevel[.6],
    Point /@ planePoints}];
```



Note that eight components have been naturally identified, although we need to zoom in to clearly see what's going on near three of them. For example, here is a zoom in near the highest point.

```
MandelbrotZoomPic = DensityPlot[-MandelbrotCount[x + ⅈ y],
   {x, -0.4, 0.1}, {y, 0.65, 1.15}, PlotPoints → 300,
   Mesh → False, Epilog →
    {PointSize[.02], GrayLevel[.6], Point /@ planePoints}
];
```



Now if we choose a c value from the interior of any of these components, we will find that $f_c^{\,4}(z)$ has an attractive fixed point. This is not quite the same as saying that f_c has an attractive orbit of period 4, since it could be that f_c has an attractive fixed point or an attractive orbit of period 2. This clearly does help us identify the periodic components, however.

■ 2.3 Parameterizing the components

Once we've located a periodic component, it is possible to parameterize its boundary. Consider the main cardioid, for example. The interior of this cardioid consists of all c values such that f_c has an attractive fixed point, i.e. there is a point z_0 such that $f_c(z_0) = z_0$ and $|f_c{}'(z_0)| < 1$. The boundary of this component consists of c values so that f_c has a neutral fixed point, i.e. $|f_c{}'(z_0)| = 1$. Put another way, $f_c{}'(z_0) = e^{2\pi i t}$ for some t in [0, 1]. Thus, consider the two following two equations in the two unknowns c and z:

$$f_c(z) = z \text{ and } f_c{}'(z) = e^{2\pi i t}.$$

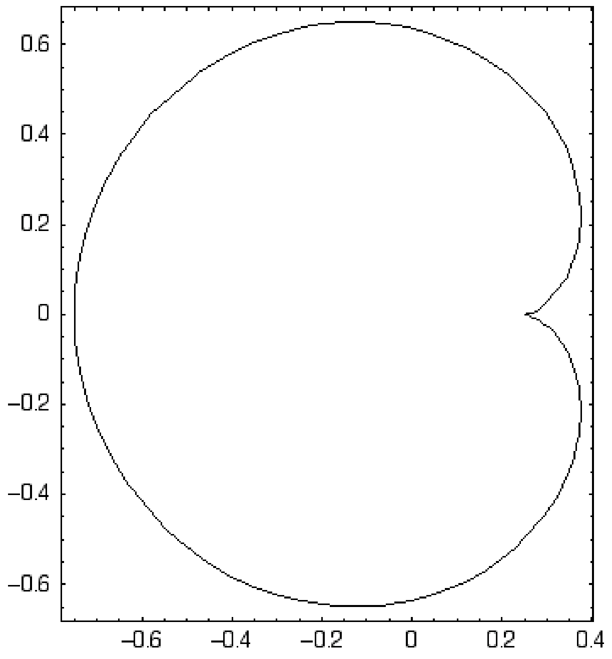
If we solve these for c and z to obtain c as a function of t, we generate a parameterization of the cardioid. Here is the parameterization.

```
f_c_[z_] := z^2 + c;
{p1[t_]} = c /. Solve[{f_c[z] == z, D[f_c[z], z] == e^(2 π i t)}, {c, z}]
```

$$\left\{-\frac{1}{4}\, e^{2\,i\,\pi\,t}\,\left(-2+e^{2\,i\,\pi\,t}\right)\right\}$$

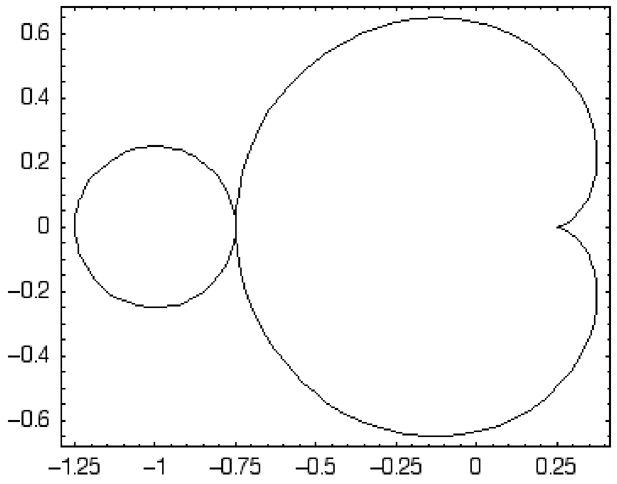
Here is a plot of the main cardioid.

```
complexToPoint[z_] := {Re[z], Im[z]};
mainCardioidPic = ParametricPlot[complexToPoint[p1[t]], {t, 0, 1},
   AspectRatio → Automatic];
```



We can parameterize other periodic components in a similar way. For example, here is the period two bulb, together with the main cardioid. Note that some care is required to select the correct solution returned by **Solve**.

```
f2[z_] = Nest[f_c, z, 2];
sols = c /. Solve[{f2[z] == z, D[f2[z], z] == ⅇ^(2 π ⅈ t)}, {c, z}];
p2[t_] = sols[[4]];
bulb2Pic = ParametricPlot[complexToPoint[p2[t]], {t, 0, 1},
   DisplayFunction → Identity];
Show[{mainCardioidPic, bulb2Pic}];
```



Higher order period components can be described in a similar manner, but require numerical solvers. The function **ComponentParameterization** attempts to find a numerical parameterization about a complex parameter **c0** with an attractive orbit of period **per**. Note that, like most numerical procedures, there is the possibility of numerical error. All of *Mathematica*'s numerical functions have methods to bound such error which generate warning messages when there is a potential problem. There are also options to control these methods. In particular, the **FindRoot** command in the **ComponentParameterization** function will occasionally lead to warning messages. There are several approaches to deal with this. One possibility that eliminates most warning messages is to increase the **WorkingPrecision** and **MaxIterations** options. However, this slows things down considerably and has no visible impact on the images. Furthermore, the pictures generated agree very well theory. Thus we choose to turn the warning messages off. Readers who prefer not to turn off the warning messages might want to comment out the following **Off** statement and perhaps uncomment the **SetOptions** command.

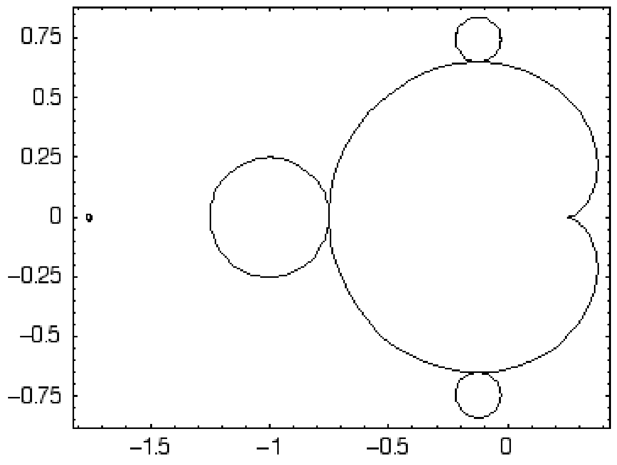
```
Off[FindRoot::lstol, FindRoot::cvmit];
(* SetOptions[FindRoot,
   WorkingPrecision → 17, MaxIterations → 1000];*)

ComponentParameterization[c0_, per_] := Module[
   {iteratedf, data, cFunc},
   iteratedf[z_] = Nest[#^2 + c &, z, per];
   data = Table[{t, c /.
       FindRoot[{iteratedf[z] == z, D[iteratedf[z], z] == ⅇ^(2 π ⅈ t)},
        {{c, c0}, {z, c0}}]},
     {t, 0, 1, 1 / 40}];
   Interpolation[data]
  ];
```

Now we'd like to apply **ComponentParameterization** to plot the components corresponding to some higher periods. We can do this by using the critical polynomials to locate *c* values with super-attracting orbits and passing these *c* values to

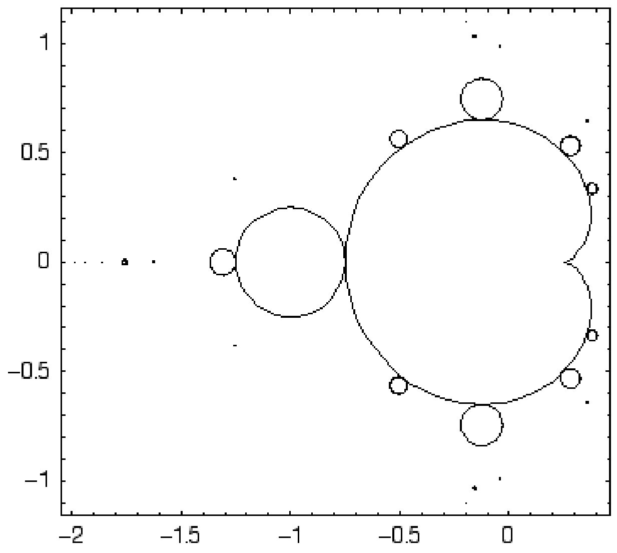
ComponentParameterization. Since **ComponentParameterization** expects the correct period as the second argument and the roots of Q_n includes points with smaller period, we need to keep track of the c values we've found so far, and remove these from the c values returned by **NSolve**. For example, when we search for the period 3 components, the critical polynomial will return these along with $c = 0$, since f_0 has a fixed point. Taking this all into acount, here are plots of the period 3 components.

```
csSoFar = {0., -1.};
Qn_[c_] := Nest[fc, 0, n];
complexPoints = Select[c /. NSolve[Q3[c] == 0, c],
   ! MemberQ[Chop[csSoFar - #], 0] &];
csSoFar = Union[complexPoints, csSoFar];
bulbs3 = ParametricPlot[Evaluate[
      complexToPoint@ComponentParameterization[#, 3][t]],
     {t, 0, 1}, AspectRatio → Automatic,
     DisplayFunction → Identity] & /@ complexPoints;
Show[{mainCardioidPic, bulb2Pic, bulbs3}];
```



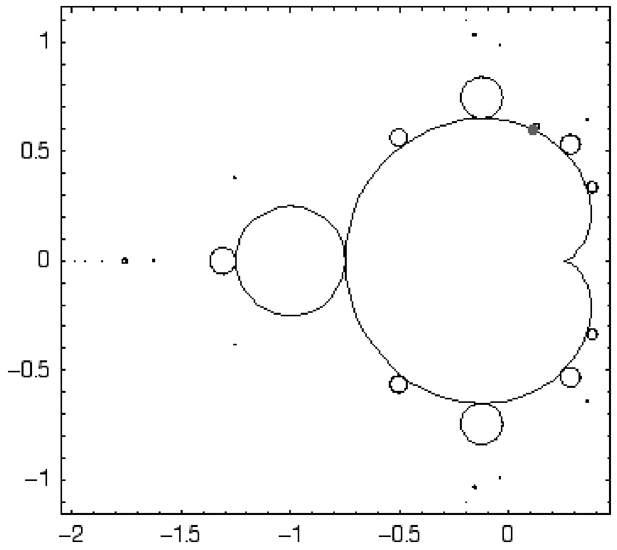
Here are the next couple of periods.

```
bulbs4To5 = Table[
   complexPoints = Select[c /. NSolve[Qn[c] == 0, c, 22],
     ! MemberQ[Chop[csSoFar - #], 0] &];
   csSoFar = Union[complexPoints, csSoFar];
   ParametricPlot[Evaluate[
       complexToPoint@ComponentParameterization[#, n][t]],
      {t, 0, 1}, AspectRatio → Automatic,
      DisplayFunction → Identity] & /@ complexPoints,
   {n, 4, 5}];
regionsTo5 = Show[{mainCardioidPic,
    bulb2Pic, bulbs3, bulbs4To5}];
```



We could clearly go on, but we generate *lots* of very small regions. On the other hand, there are still plenty of interesting visible regions hanging right off of the larger regions we've already plotted. The parameterizations we have chosen can be used to identify the locations of these. In particular, a bulb corresponding to period q is attached to the main cardioid at each point **`p1[p/q]`** where p/q is in lowest terms. For example, we can parameterize a bulb with period 7 and indicate it's location in the Mandelbrot set as follows.

```
p = ComponentParameterization[p1[2 / 7], 7];
ParametricPlot[complexToPoint[p[t]], {t, 0, 1},
  DisplayFunction → Identity];
Show[%, regionsTo5,
  DisplayFunction → $DisplayFunction,
  AspectRatio → Automatic,
  Epilog → {PointSize[.02], GrayLevel[.3],
    Point[complexToPoint[p1[2 / 7]]]},
  Axes → False];
```



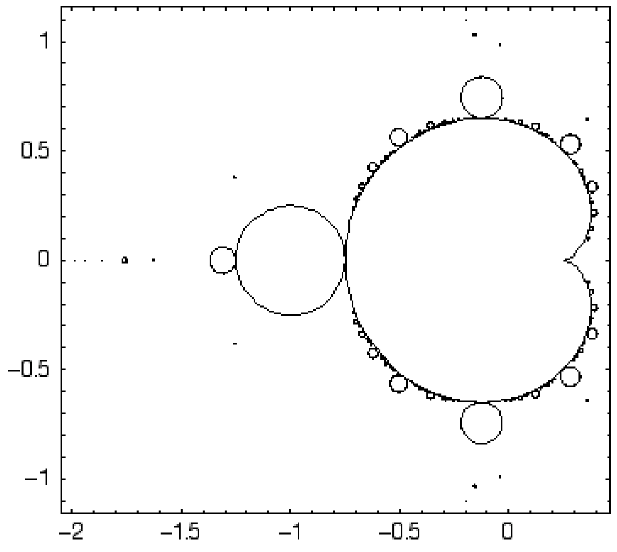
We now want to parameterize and display a few more bulbs attached to the main cardioid. To do this, we first need to list out some rational numbers between 0 and 1. Of course, there are a number of approaches to do this. As it happens, there is one technique ideally suited for our purposes. Suppose we would like to locate the fraction t between p/q and r/s so that $p_1(t)$ has the largest possible bulb attached. In [5], it is shown that $t = (p+r)/(q+s)$ is that fraction. Thus the following recursive construction, called the Farey tree, generates an appropriate list of rational numbers. Note that we delete the fractions which have already been listed.

```
FareyFractions[1] := {0, 1};
FareyFractions[2] := {0, 1 / 2, 1};
FareyFractions[n_] := FareyFractions[n] = Union[
    Partition[FareyFractions[n - 1], 2, 1] /. {x_, y_} :→
      (Numerator[x] + Numerator[y]) / (Denominator[x] + Denominator[y]),
    FareyFractions[n - 1]];
fractions = Complement[FareyFractions[8],
  FareyFractions[4], {1 / 5, 4 / 5}]
```

```
{1/8, 1/7, 2/13, 1/6, 3/17, 2/11, 3/16, 4/19, 3/14, 5/23, 2/9, 5/22, 3/13, 4/17, 5/19, 4/15, 7/26, 3/11,
 8/29, 5/18, 7/25, 2/7, 7/24, 5/17, 8/27, 3/10, 7/23, 4/13, 5/16, 6/17, 5/14, 9/25, 4/11, 11/30, 7/19,
 10/27, 3/8, 11/29, 8/21, 13/34, 5/13, 12/31, 7/18, 9/23, 9/22, 7/17, 12/29, 5/12, 13/31, 8/19, 11/26, 3/7,
 10/23, 7/16, 11/25, 4/9, 9/20, 5/11, 6/13, 7/13, 6/11, 11/20, 5/9, 14/25, 9/16, 13/23, 4/7, 15/26, 11/19,
 18/31, 7/12, 17/29, 10/17, 13/22, 14/23, 11/18, 19/31, 8/13, 21/34, 13/21, 18/29, 5/8, 17/27, 12/19, 19/30,
 7/11, 16/25, 9/14, 11/17, 11/16, 9/13, 16/23, 7/10, 19/27, 12/17, 17/24, 5/7, 18/25, 13/18, 21/29, 8/11,
 19/26, 11/15, 14/19, 13/17, 10/13, 17/22, 7/9, 18/23, 11/14, 15/19, 13/16, 9/11, 14/17, 5/6, 11/13, 6/7, 7/8}
```

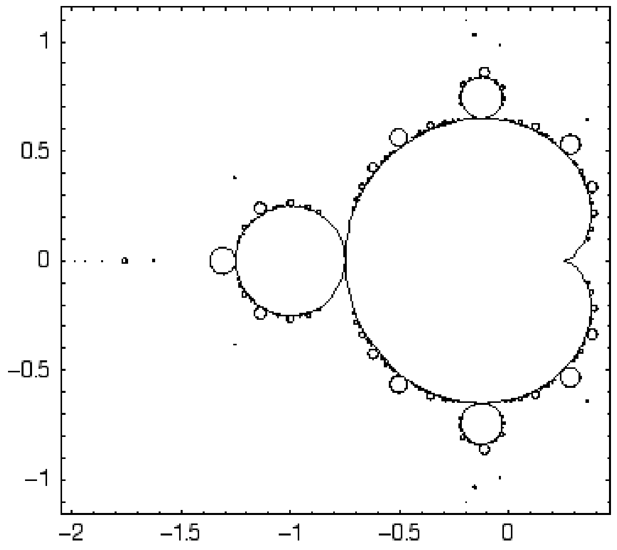
Here are the corresponding bulbs.

```
moreBulbs = fractions /. x_?NumericQ :→
    ParametricPlot[Evaluate[
      complexToPoint@ComponentParameterization[p1[x],
         Denominator[x]][t]], {t, 0, 1},
     DisplayFunction → Identity];
Show[{regionsTo5, moreBulbs}];
```



Finally, we plot some of the corresponding bulbs off of the period 2 and 3 regions.

```
fractions2 = Complement[FareyFractions[6], {1 / 2}];
moreBulbs2 = fractions2 /. x_?NumericQ :→
    ParametricPlot[Evaluate[
      complexToPoint@ComponentParameterization[p2[x],
         2 Denominator[x]][t]], {t, 0, 1},
     DisplayFunction → Identity];
fractions3 = FareyFractions[5];
p13 = ComponentParameterization[p1[1 / 3], 3];
moreBulbs13 = fractions3 /. x_?NumericQ :→
    ParametricPlot[Evaluate[
      complexToPoint@ComponentParameterization[p13[x],
         3 Denominator[x]][t]], {t, 0, 1},
     DisplayFunction → Identity];
p23 = ComponentParameterization[p1[2 / 3], 3];
moreBulbs23 = fractions3 /. x_?NumericQ :→
    ParametricPlot[Evaluate[
      complexToPoint@ComponentParameterization[p23[x],
         3 Denominator[x]][t]], {t, 0, 1},
     DisplayFunction → Identity];
Show[{regionsTo5, moreBulbs, moreBulbs2,
   moreBulbs13, moreBulbs23}];
```



While not stunning in the same sense as the many colored images of the Mandelbrot set, this image is very instructive and illuminates the many substructures we see in other images.

References

[1] Devaney, R.L. *A First Course in Chaotic Dynamical Systems*. Addison-Wesley, Reading, Mass. 1992.

[2] Neidinger, R. D. and Annen, R. J., "The road to chaos is filled with polynomial curves." *Am. Math. Monthly* **103** #2 (1996) 640-653.

[3] Beardon, A.F. *Iteration of Rational Functions*. Springer-Verlag, NY. 1991.

[4] Dickau, R. M. Compilation of iterative and list operations in "Tricks of the trade." *The Mathematica Journal,* **7** #1 (1997)14-15.

[5] Devaney, R.L. “The Mandelbrot set and the Farey tree.” *Am. Math. Monthly* **106** (1999) 289-302.

[6] Devaney, R.L. “The Mandelbrot set and the Farey tree.” *Am. Math. Monthly* **106** (1999) 289-302.