a numerical procedure for finding such a solution. Here is a pseudocode:

```
$\mathbf{X} = \begin{bmatrix} 0.1, & 1.2, & 2.5 \end{bmatrix}^T$
for $k = 1$ to 10 do
    $\mathbf{F} = \begin{bmatrix} x_1 + x_2 + x_3 - 3 \\ x_1^2 + x_2^2 + x_3^2 - 5 \\ e^{x_1} + x_1x_2 - x_1x_3 - 1 \end{bmatrix}$
    $\mathbf{J} = \begin{bmatrix} 1 & 1 & 1 \\ 2x_1 & 2x_2 & 2x_3 \\ e^{x_1} + x_2 - x_3 & x_1 & -x_1 \end{bmatrix}$
    solve $\mathbf{JH} = \mathbf{F}$
    $\mathbf{X} = \mathbf{X} - \mathbf{H}$
end for
```

When programmed and executed on a computer, we found that it converges to $\boldsymbol{x} = (0, 1, 2)$, but when we change to a different starting vector, $(1, 0, 1)$, it converges to another root, $(1.2244, -0.0931, 1.8687)$. (Why?) ■

We can use mathematical software such as in Matlab, Maple, or Mathematica and their built-in procedures for solving the system of nonlinear equations (8). The important application area of solving systems of nonlinear equations is used in Chapter 16 on minimization of functions.

Fractal Basins of Attraction

The applicability of Newton's method for finding complex roots is one of its outstanding strengths. One need only program Newton's method using complex arithmetic.

The frontiers of numerical analysis and nonlinear dynamics overlap in some intriguing ways. Computer-generated displays with fractal patterns, such as in Figure 3.8, can easily be created with the help of the Newton iteration. The resulting pictures show intricately

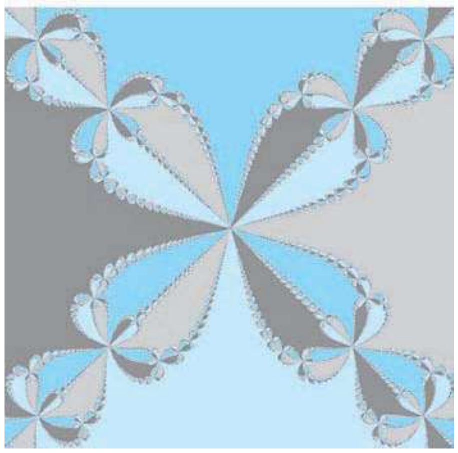
FIGURE 3.8 Basins of attraction

interwoven sets in the plane that are quite beautiful if displayed on a color computer monitor. One begins with a polynomial in the complex variable z. For example, $p(z) = z^4 - 1$ is suitable. This polynomial has four zeros, which are the fourth roots of unity. Each of these zeros has a **basin of attraction**, that is, the set of all points z_0 such that Newton's iteration, started at z_0, will converge to that zero. These four basins of attraction are disjoint from each other, because if the Newton iteration starting at z_0 converges to one zero, then it cannot also converge to another zero. One would naturally expect each basin to be a simple set surrounding the zero in the complex plane. But they turn out to be far from simple. To see what they are, we can systematically determine, for a large number of points, which zero of p the Newton iteration converges to if started at z_0. Points in each basin can be assigned different colors. The (rare) points for which the Newton iteration does not converge can be left uncolored. Computer Problem 3.2.27 suggests how to do this.

Summary

(1) For finding a zero of a continuous and differentiable function f, **Newton's method** is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n \geqq 0)$$

It requires a given initial value x_0 and two function evaluations (for f and f') per step.

(2) The errors are related by

$$e_{n+1} = -\frac{1}{2}\left(\frac{f''(\xi_n)}{f'(x_n)}\right) e_n^2$$

which leads to the inequality

$$|e_{n+1}| \leqq c|e_n|^2$$

This means that Newton's method has **quadratic convergence** behavior for x_0 sufficiently close to the root r.

(3) For an $N \times N$ system of nonlinear equations $\mathbf{F}(\mathbf{X}) = \mathbf{0}$, **Newton's method** is written as

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(k)}\right) \qquad (k \geqq 0)$$

which involves the Jacobian matrix $\mathbf{F}'\left(\mathbf{X}^{(k)}\right) = \mathbf{J} = \left[\left(\partial f_i\left(\mathbf{X}^{(k)}\right)/\partial x_j\right)\right]_{N\times N}$. In practice, one solves the **Jacobian linear system**

$$\left[\mathbf{F}'(\mathbf{X}^{(k)})\right]\mathbf{H}^{(k)} = -\mathbf{F}\left(\mathbf{X}^{(k)}\right)$$

using Gaussian elimination and then finds the next iterate from the equation

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mathbf{H}^{(k)}$$

Additional References

For additional details and sample plots, see Kincaid and Cheney [2002] or Epureanu and Greenside [1998]. For other references on fractals, see Crilly, Earnshall, and Jones [1991], Feder [1998], Hastings and Sugihara [1993], and Novak [1998].

f. Select starting values, and solve

$$\begin{cases} \sin(x+y) = e^{x-y} \\ \cos(x+6) = x^2y^2 \end{cases}$$

24. Investigate the behavior of Newton's method for finding complex roots of polynomials with real coefficients. For example, the polynomial $p(x) = x^2 + 1$ has the complex conjugate pair of roots $\pm i$ and Newton's method is $x_{n+1} = \frac{1}{2}(x_n - 1/x_n)$. First, program this method using real arithmetic and real numbers as starting values. Second, modify the program using complex arithmetic but still using only real starting values. Finally, use complex numbers as starting values. Observe the behavior of the iterates in each case.

25. Using Problem 3.2.40, find a complex root of each of the following:

a. $z^3 - z - 1 = 0$

b. $z^4 - 2z^3 - 2iz^2 + 4iz = 0$

c. $2z^3 - 6(1+i)z^2 - 6(1-i) = 0$

d. $z = e^z$

Hint: For the last part, use Euler's relation $e^{iy} = \cos y + i \sin y$.

26. In the Newton method for finding a root r of $f(x) = 0$, we start with x_0 and compute the sequence $x_1, x_2, \ldots$ using the formula $x_{n+1} = x_n - f(x_n)/f'(x_n)$. To avoid computing the derivative at each step, it has been proposed to replace $f'(x_n)$ with $f'(x_0)$ in all steps. It has also been suggested that the derivative in Newton's formula be computed only every other step. This method is given by

$$\begin{cases} x_{2n+1} = x_{2n} - \dfrac{f(x_{2n})}{f'(x_{2n})} \\ x_{2n+2} = x_{2n+1} - \dfrac{f(x_{2n+1})}{f'(x_{2n})} \end{cases}$$

Numerically compare both proposed methods to Newton's method for several simple functions that have known roots. Print the error of each method on every iteration to monitor the convergence. How well do the proposed methods work?

27. **(Basin of attraction)** Consider the complex polynomial $z^3 - 1$, whose zeros are the three cube roots of unity. Generate a picture showing three basins of attraction in the complex plane in the square region defined by $-1 \leqq \text{Real}(z) \leqq 1$ and $-1 \leqq \text{Imaginary}(z) \leqq 1$. To do this, use a mesh of 1000×1000 pixels inside the square. The center point of each pixel is used to start the iteration of Newton's method. Assign a particular basin color to each pixel if convergence to a root is obtained with $nmax = 10$ iterations. The large number of iterations suggested can be avoided by doing some analysis with the aid of Theorem 1, since the iterates get within a certain neighborhood of the root and the iteration can be stopped. The criterion for convergence is to check both $|z_{n+1} - z_n| < \varepsilon$ and $|z_{n+1}^3 - 1| < \varepsilon$ with a small value such as $\varepsilon = 10^{-4}$ as well as a maximum number of iterations. *Hint:* It is best to debug your program and get a crude picture with only a small number of pixels such as 10×10.

28. (Continuation) Repeat for the polynomial $z^4 - 1 = 0$.

29. Write **real function** $Sqrt(x)$ to compute the square root of a real argument x by the following algorithm: First, reduce the range of x by finding a real number r and an