#12.2.4 Fixed points of Henon map:

$$X_{n+1} = Y_n + 1 - aX_n^2$$
$$Y_{n+1} = bX_n$$

$\Rightarrow$

$$x_n = (bx_n) + 1 - ax_n^2$$

$$\overset{b}{x_n(1-b)} \overset{c}{-1} + \overset{a}{ax_n^2} = 0$$

Quadratic

$$X_c = \frac{-(1-b) \pm \sqrt{(1-b)^2 + 4a}}{2a}$$

$$\Rightarrow Y_c = bX_c$$

$$\Rightarrow \sqrt{(1-b)^2 + 4a} > 0$$

$$a_0 > -\frac{(1-b)^2}{4}$$

#12.2.5

Jacobian:

$$\begin{pmatrix} \frac{\partial 1}{\partial x} & \frac{\partial 1}{\partial y} \\ \frac{\partial 2}{\partial x} & \frac{\partial 2}{\partial y} \end{pmatrix} = \begin{pmatrix} \overset{a}{-2ax_c} & \overset{b}{1} \\ \overset{c}{b} & \overset{d}{0} \end{pmatrix}$$

$X_{n+1} = eq\ 1$

$Y_{n+1} = eq\ 2$

@ critical point (X_c, Y_c)

Trace / determinant:

$$\tau = a + d = -2ax_c + 0$$
$$\Delta = ad - bc = 0 - b$$

Eigenvalues:
$$\lambda_{12} = \frac{\tau \pm \sqrt{\tau^2 - 4\Delta}}{2} = \frac{-2ax_c \pm \sqrt{(2ax_c)^2 + 4b}}{2}$$

#10.3.6 (Cubic Map)

$x_{n+1} = f(x_n) = r x_n - x_n^3$

$x_c = r x_c - x_c^3$ $\quad \boxed{x_c = 0, \pm\sqrt{r-1}}$

$1 = r - x^2$ $\quad$ exist for $r \geq 1$

$x_c = \sqrt{r-1}$

stability: $|f'(x_n)| < 1$

$f'(x_n, r) = r - 3x_n^2$ $\quad \boxed{\begin{array}{l} x_c = 0 \text{ is stable} \\ \text{if } |r| < 1 \end{array}}$

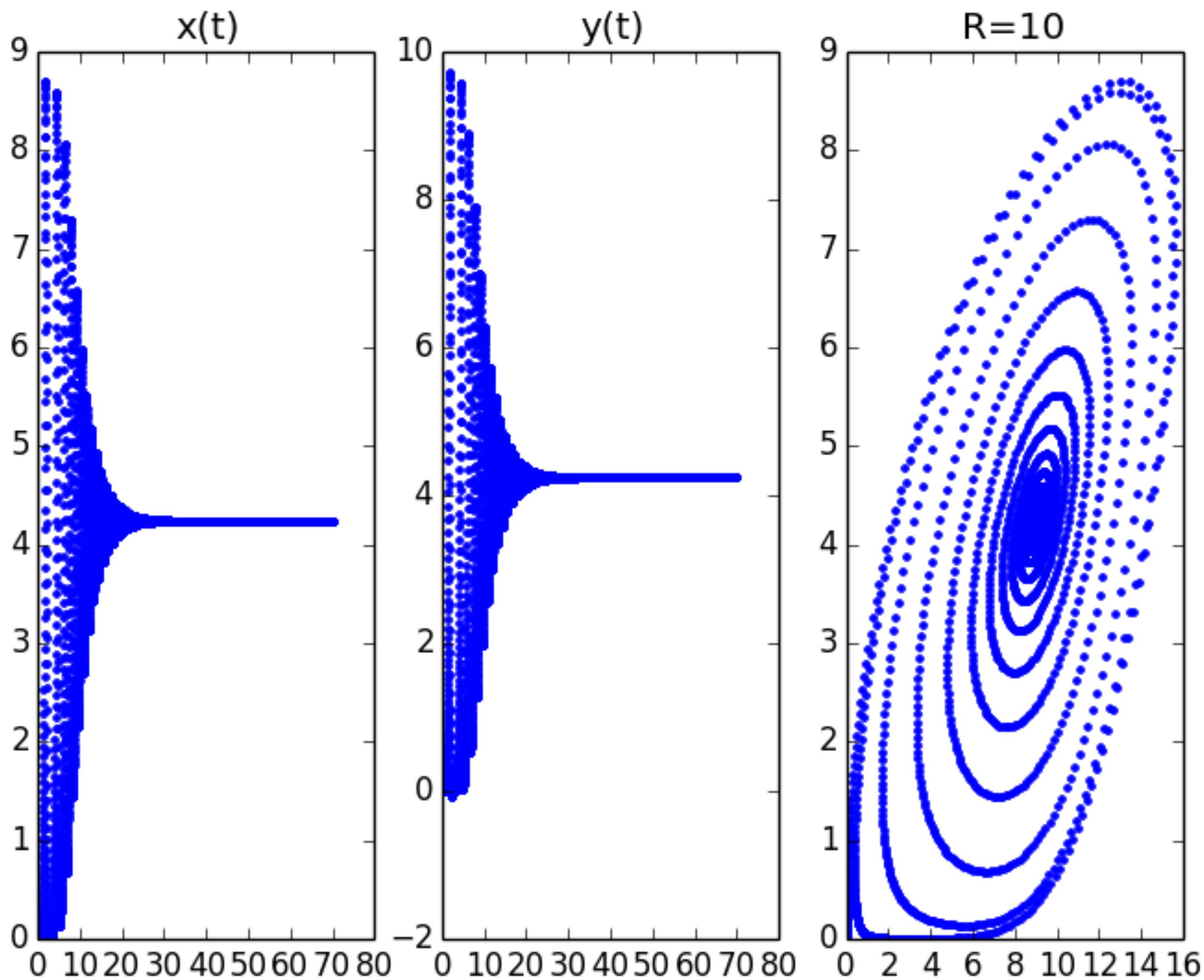
$f'(0,1) = 1$

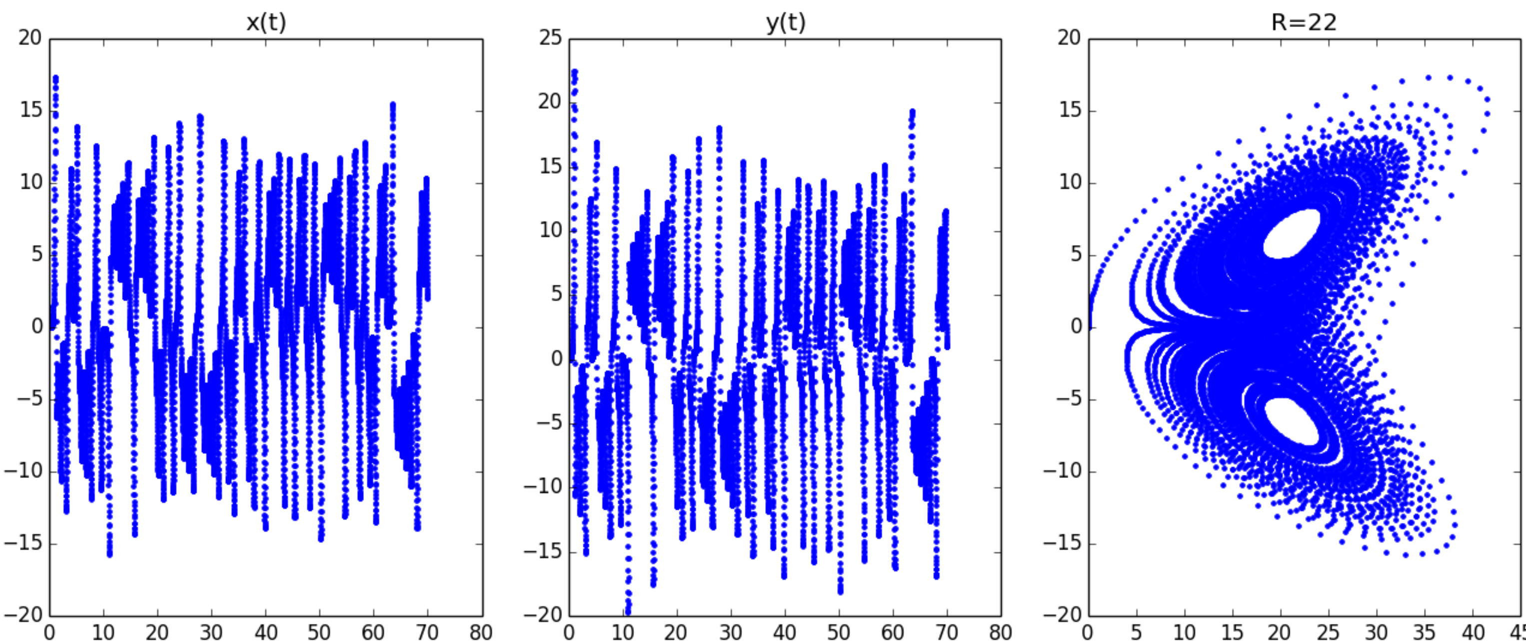
$f'(\pm\sqrt{r-1}, r) = 1.5 - 3(.5) = 0$ $\quad \boxed{\begin{array}{l} x_c = \pm\sqrt{r-1} \text{ is} \\ \text{stable for } 1 < r < 2 \end{array}}$

$= 1.2 - 3(.2) < 1$

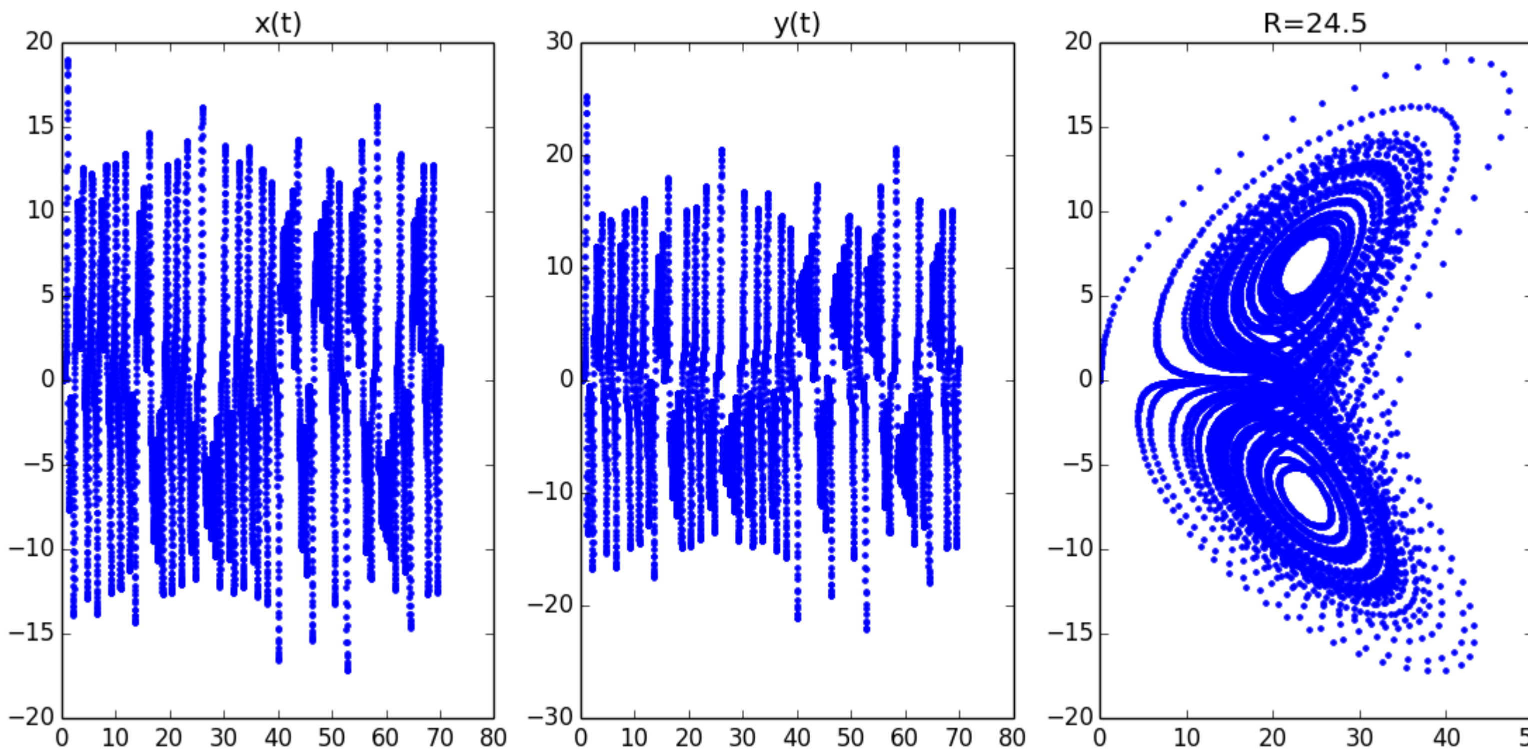
$= 3 - 6 > 1$

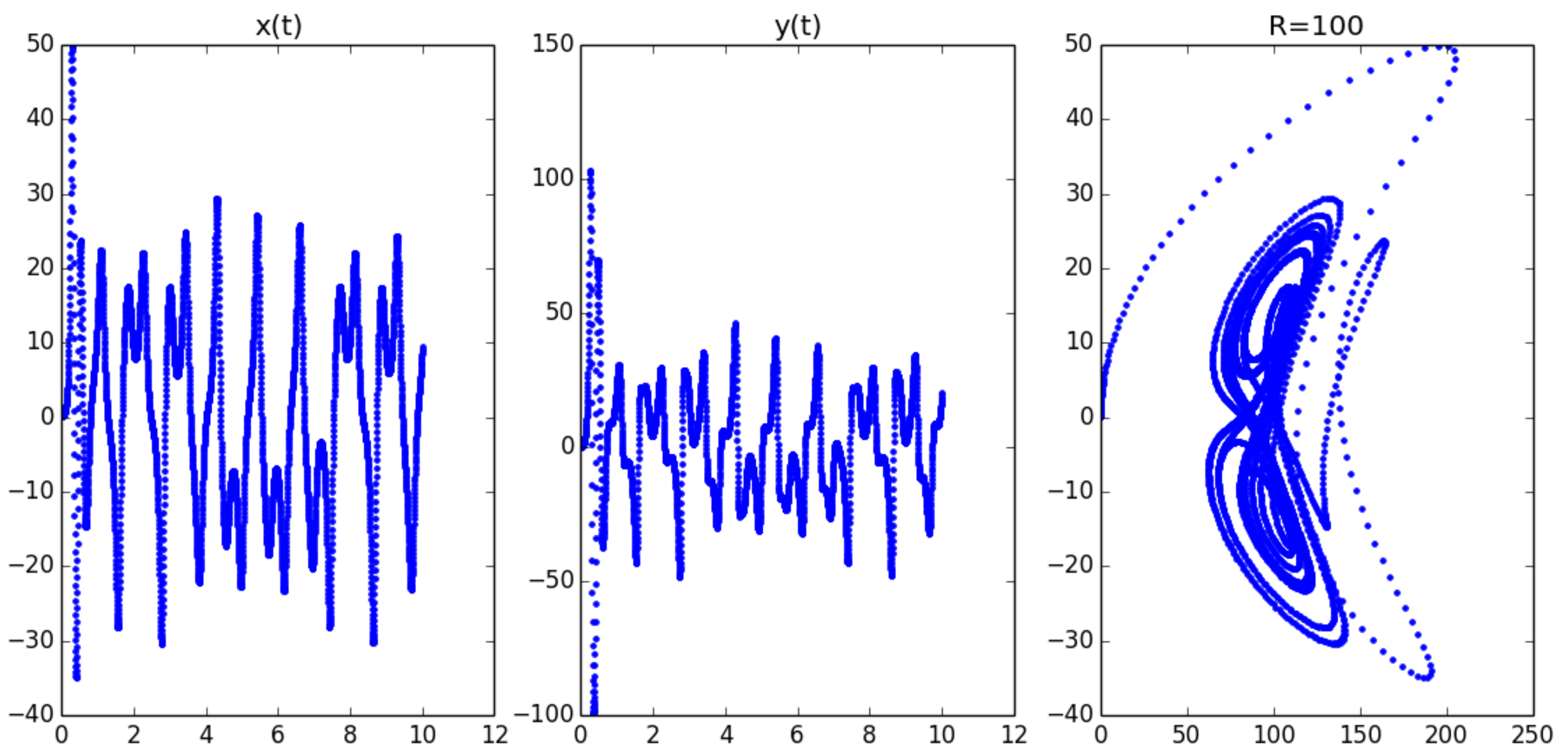
b)

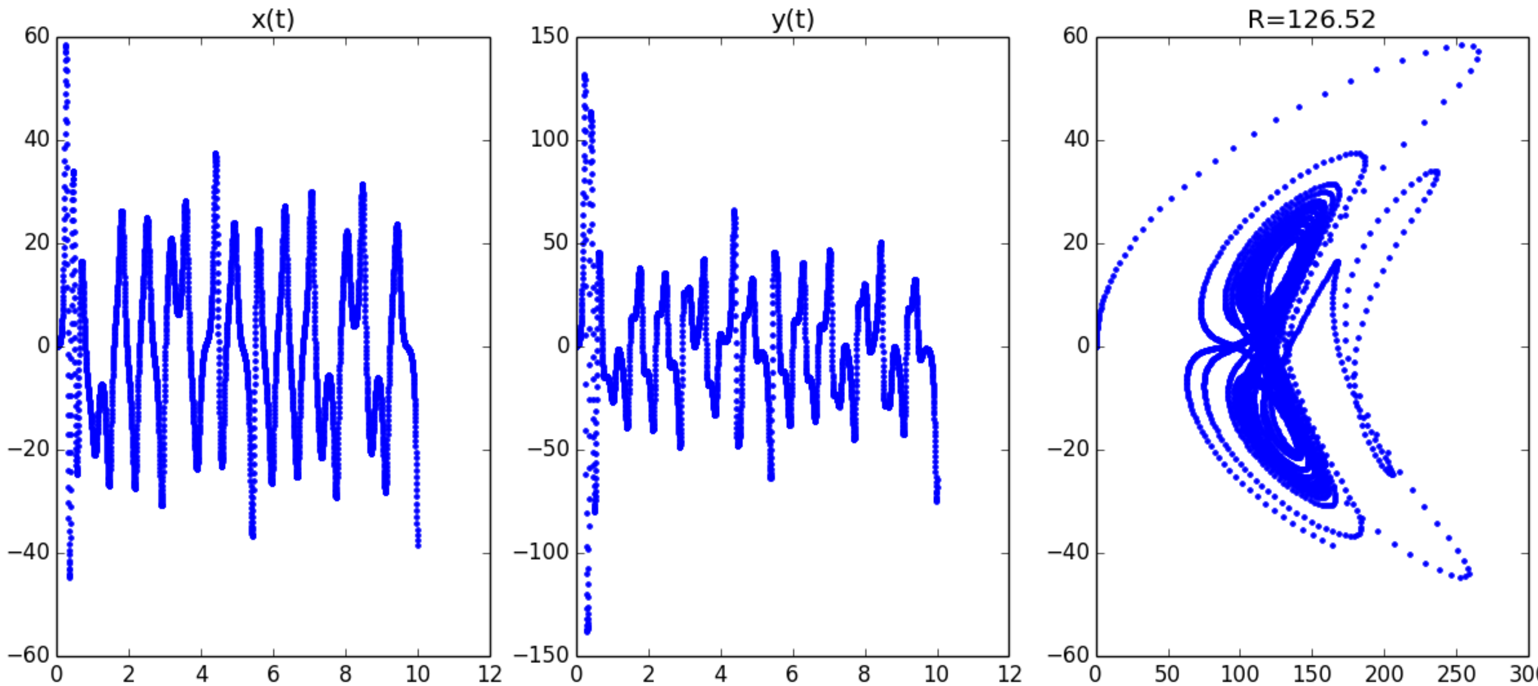

x(t)
y(t)
R=10
0
10
20
30
40
50
60
70
80
−2
2
4
6
8
9
12
14
16

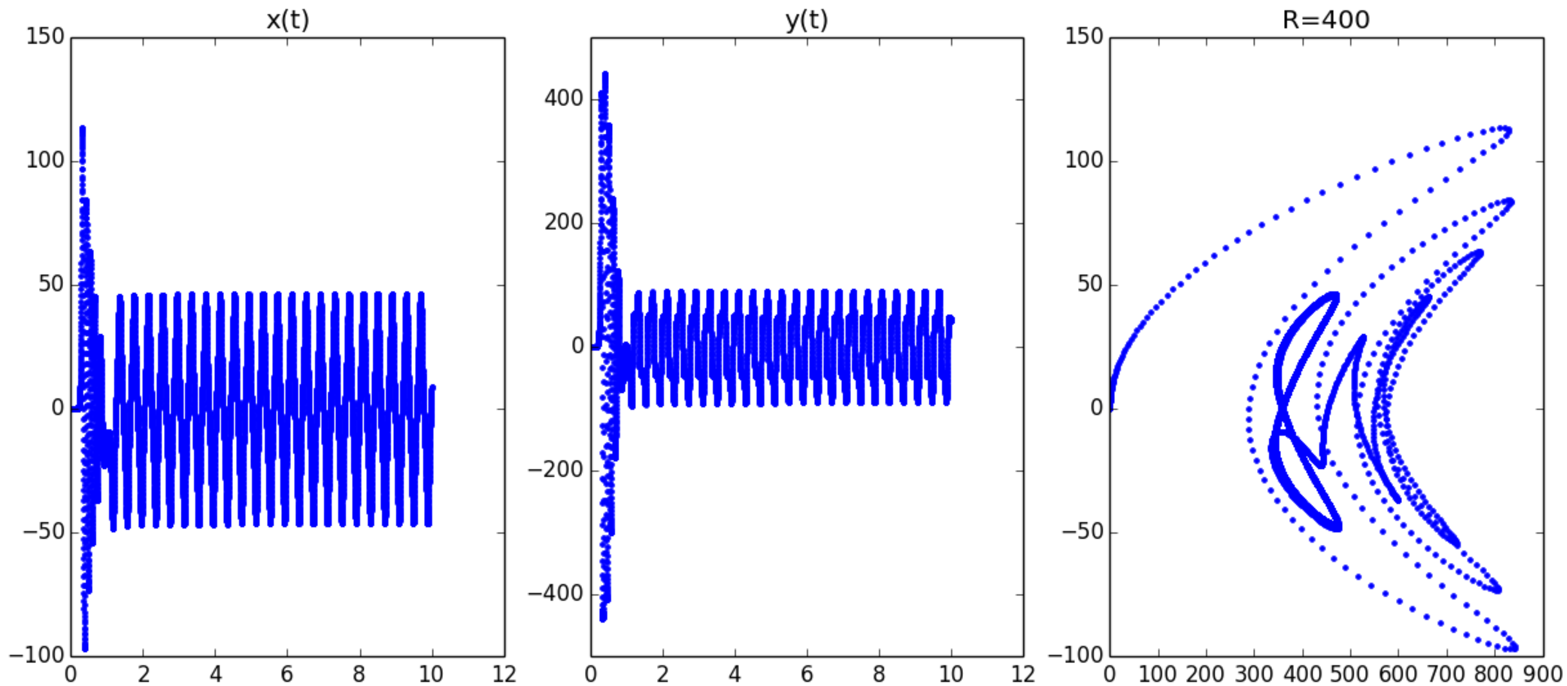


x(t)
y(t)
R=22


x(t)
y(t)
R=24.5

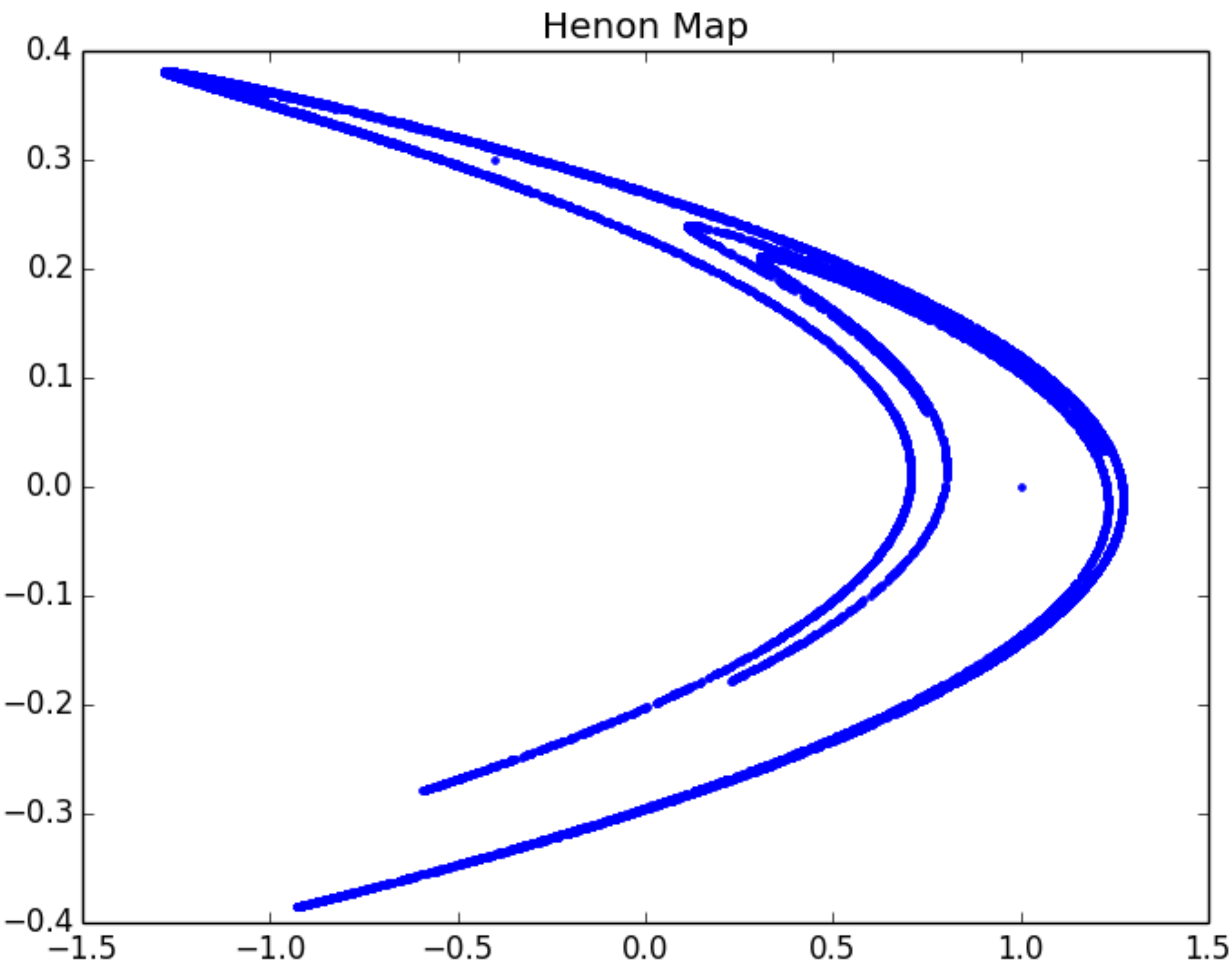

x(t)
y(t)
R=100


x(t)
y(t)
R=126.52



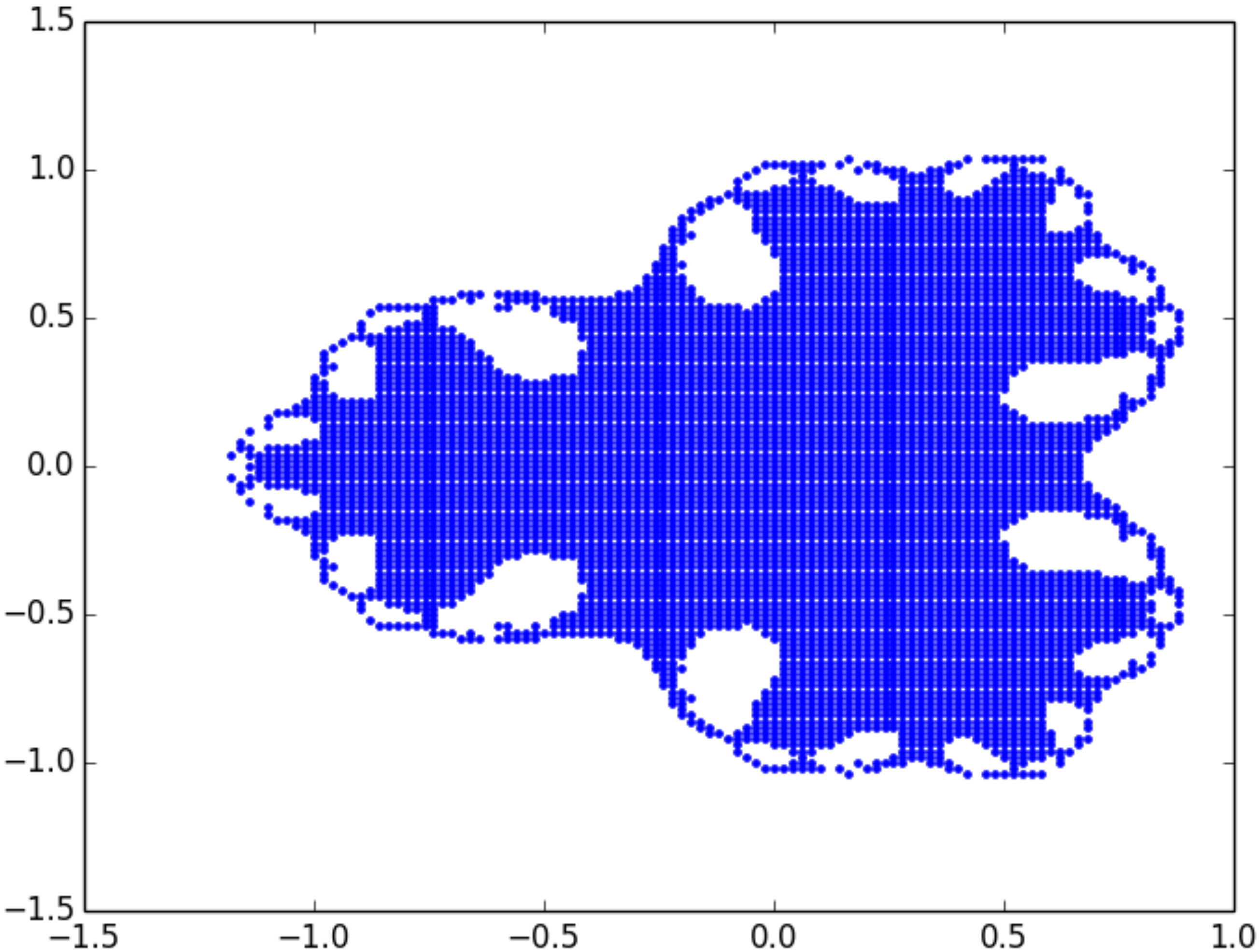
x(t)
y(t)
R=400
150
100
50
0
−50
−100
400
200
−200
−400
0
2
4
6
8
10
12
100
200
300
400
500
600
700
800
900

```python
import numpy as np
import matplotlib.pyplot as plt

##lorentz equations
def function():
    sigma = 10   ##Prandtl number
    r = 400    ##Rayleigh number
    b = 8/3
    z=y=x=.00001

    t = 0
    dt=.001
    while t < 10.0000:
        xdot = sigma*(y-x)
        ydot = r*x-y-x*z
        zdot = x*y-b*z
        x += xdot*dt
        y += ydot*dt
        z += zdot*dt
        t+= dt
        plt.subplot(1,3,1)
        plt.plot(t,x,'.b-')
        plt.title('x(t)')
        plt.subplot(1,3,2)
        plt.plot(t,y,'b.-')
        plt.title('y(t)')
        plt.subplot(1,3,3)
        plt.plot(z,x,'b.-')
        plt.title('Z(x)')



#plt.xlim(5,100)
function()
plt.title('R=400')
#plt.legend()
plt.show()
```



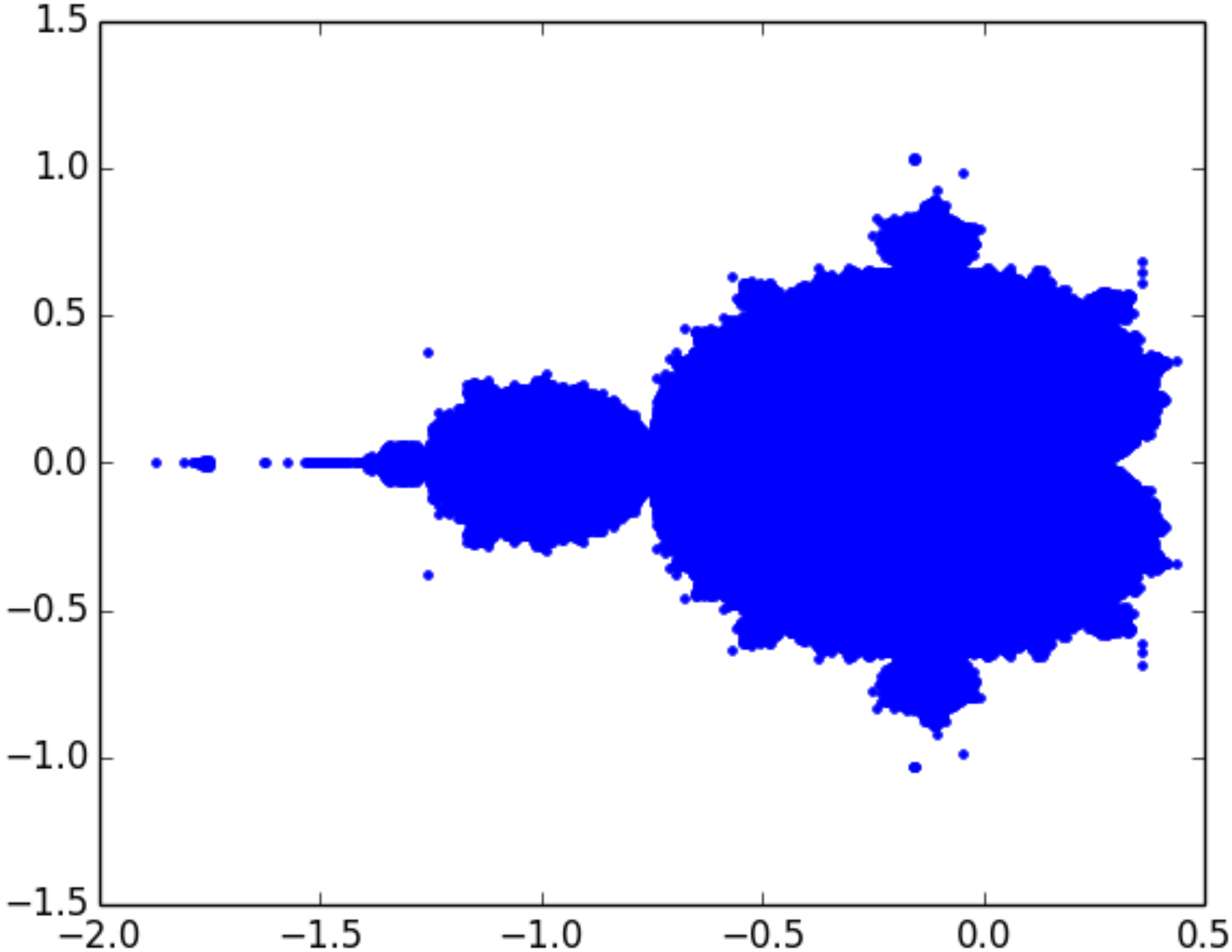
Henon Map
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

```python
import matplotlib.pyplot as plt
import numpy as np
x=[]
y=[]
yi=0
xi=0
a=1.4
b=.3
for i in range (1,10000):
    x1=yi+1-a*xi**2  #x_n +1
    y1=b*xi          #y_n +1
    x.append(x1)
    y.append(y1)
    xi=x1
    yi=y1
    i +=1
    plt.plot(xi,yi,'b.-')
plt.title('Henon Map')
plt.show()
```



1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0

```python
import matplotlib.pyplot as plt
import numpy as np



y=x=np.arange(-1.50,1.50,.02)
def function():

        for i in x:
                for c in y:
                        t=0
                        k=complex(0.5,0)
                        z=complex(i,c)
                        while t<10:
                                z=z*z*z+k
                                t+=1
                        if abs(z)>100:
                                plt.plot(i,c,'b.-')
function()
plt.show()


```


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5

```
import matplotlib.pyplot as plt
import numpy as np

y=x=np.arange(-5,5,.005)

def function():
        for i in x:
                for c in y:
                        t=0
                        k=complex(i,c)
                        z=0
                        while t<100:
                                z=z*z+k
                                t+=1
                        if abs(z)<1000000000:
                                plt.plot(i,c,'b.-')
function()
plt.show()
```