Lecture6

February 21, 2019

1 Lecture 6: Simple Models for Chaos and he logistic map

Overview: * Revisit phase space plot and Poincare Map. * Properties of the Logistic Map.

```
In [1]: %matplotlib notebook
        import matplotlib.pyplot as plt # for plotting
        import numpy as np
        # import our Pendulum class from Particle1D
        from Particle1D import Pendulum
        import LogisticMap as lm
```

1.1 Exercise

- Open the file `LogisticMap.py` in a text editor and examine the contents, make sure that you understand all of the components.

1.2 The Logistic Map

```
In [2]: fig = plt.figure(figsize = [9,7])
        axs = fig.subplots(3,2)

        axs = list(axs.flatten())

        mus = [0.7,3.53,2.8,3.76,3.32,3.78]

        for ii in range(6):

            m = lm.LogisticMap(0.4, mu = mus[ii])
            m.iterate(80)

            axs[ii].plot(m.xList, ls = '--', marker = 'o', ms = 3, lw =0.5, label = "$\mu$ = {
            axs[ii].set_ylabel('$x_n$')
            axs[ii].set_xlabel('$n$')
            axs[ii].legend(loc = 4,fontsize = 12)
            axs[ii].set_ylim([0,1])
            axs[ii].set_xlim([0,81])
            axs[ii].set_yticks([0,0.5,1])


            fig.subplots_adjust(top = 0.95, bottom =0.09, left = 0.10, right =0.98, wspace = 0
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

- Try to find stable fixed points of period 1T, 2T, 4T, and higher by adjusting the parameters $\mu$ and x_0.

1.3 Period doubling and the logistic map

```
In [4]: # Generate a bifrucation diagram
        mu_start = 0
        mu_stop = 4
        mu_steps = 1000

        x0 = 0.8

        # Generate an array of mu values
        mus = np.linspace(mu_start, mu_stop, mu_steps)

        # containers for mu values and logistic map values
        mu_list = []
        m_list = []

        for mu in mus:
            m = lm.LogisticMap(x0, mu = mu)

            # initialize method removes transient
            m.initialize(100)
            m.iterate(1000)

            m_points = np.array(m.xList)
            m_points = np.unique(np.round(m_points,8))

            for m in m_points:
                mu_list.append(mu)
                m_list.append(m)

        fig = plt.figure(figsize = [8,5])
        ax = fig.add_subplot(111)
        ax.plot(mu_list, m_list,'.', markersize = 0.5)
        ax.set_xlabel('$\mu$')
        ax.set_ylabel('$x_n$')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
Out[4]: Text(0,0.5,'$x_n$')
```

1.4 Exercise

- Look up the helpfile on the function `np.unique`. What does it do? Why is it needed here and why must it be combined with a `round` function?

1.5 Period doubling route to chaos and the Pendulum

- Below we explore period doubling in the non-linear damped driven pendulum. You can explore the same phenomena for the logistic map by adjusting parameters for the plots above in the cell above this one.

```
In [4]:  # demonstate regime of chaotic behavior
         fig = plt.figure(figsize = [8,6])
         axs = fig.subplots(3,1)

         Fds = [1.4,1.45,1.465]

         for ii in range(3):
             p = Pendulum(nu = 1/2, Fd = Fds[ii], omega_d = 2./3. , x0 = 0.200, v0 = 0, tf = 42
             p.scipy_trajectory()

             axs[ii].plot(p.tarray, p.xv[:,0],lw = 2, label = "$F_d$ = {}".format(Fds[ii]))
             axs[ii].set_ylabel('$\\theta$ (rad)')
             axs[ii].legend(loc = 4, fontsize = 12)
             axs[ii].set_xlim([300,420])


             axs[-1].set_xlabel('t (s)')

         fig.subplots_adjust(top = 0.98, bottom =0.08, left = 0.14, right =0.98)
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

1.6 Exercise:

Construct Poincare Maps for each of the above traces. How many points should appear in each Poincare map?

1.7 Sensitivity to Initial Conditions

```
In [5]: fig = plt.figure(figsize = [9,6])
        axs = fig.subplots(2,2)
        axs = axs.flatten(())

        delta = 1e-10

        m1 = lm.LogisticMap(0.8, mu = 3.2)
        m1.iterate(80)
        m2 = lm.LogisticMap(0.8+delta, mu = 3.2)
        m2.iterate(80)
        diff1 = np.abs(np.array(m1.xList) - np.array(m2.xList))

        axs[0].plot(m1.xList, ls = '--', lw = 0.5, marker = 'o', ms = 2, label = "$x_0$ = 0.8"
        axs[0].plot(m2.xList, ls = '--', marker = 's', ms = 2, label = "$x_0$ = 0.8 + 1e-10")
        axs[2].semilogy(diff1, ls = '--', marker = 'o', ms = 2, label = "$\mu$ = 3.2")

        m3 = lm.LogisticMap(0.8, mu = 3.8)
        m3.iterate(80)
        m4 = lm.LogisticMap(0.8+delta, mu = 3.8)
        m4.iterate(80)
        diff2 = np.abs(np.array(m4.xList) - np.array(m3.xList))

        axs[1].plot(m4.xList, ls = '--', lw = 0.5,  marker = 'o', ms = 2, label = "$x_0$ = 0.8
        axs[1].plot(m3.xList, ls = '--', marker = 'o', ms = 2, label = "$x_0$ = 0.8 + 1e-10")
        axs[3].semilogy(diff2, ls = '--', marker = 'o', ms = 2, label = "$\mu$ = 3.8")

        axs[0].set_ylabel('$x_n$')
        axs[2].set_ylabel('$\Delta x_n$')
        for ax in axs:
            ax.set_xlabel('$n$')


        axs[0].set_title("$\mu = 3.2$")
        axs[1].set_title("$\mu = 3.8$")
        axs[0].legend(loc = 3,fontsize = 10, markerscale = 0.5)
        axs[1].legend(loc = 3,fontsize = 10, markerscale = 0.5)
        fig.subplots_adjust(top = 0.95, bottom =0.09, left = 0.12, right =0.98, wspace = 0.2, 
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
/Users/kemp/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: DeprecationWarning:
  This is separate from the ipykernel package so we can avoid doing imports until
```

1.8 Lyapunov Exponent for the logistic map

```
In [6]: x0 = 0.4
        mu_start = 1
        mu_stop = 4
        mu_steps = 1000
        mus = np.linspace(mu_start, mu_stop, mu_steps)

        lyapunov_list = []
        for mu in mus:
            m = lm.LogisticMap(x0, mu = mu)
            l = m.lyapunov(1000, 200)
            lyapunov_list.append(l)

        fig = plt.figure(figsize = [8,6])
        ax = fig.add_subplot(111)
        ax.plot(mus, lyapunov_list,'.', markersize = 1)
        ax.hlines(0,0,4, LineStyles = '--', lw = 1,colors = 'k')
        #ax.set_ylim([-2,1])
        ax.set_xlim([mu_start, mu_stop])
        ax.set_xlabel('$\mu$')
        ax.set_ylabel('$\lambda$')
```

```
/Users/kemp/Dropbox/Documents/Courses/2019/PHYS1600/Lectures/LogisticMap.py:38: RuntimeWarning
  lam += np.log(self.mu * abs(1.0 - 2.0 * x))
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
Out[6]: Text(0,0.5,'$\\lambda$')
```

1.9 Exercise

- Explore the Lyapunov exponent for different values of μ and x_0.
- Can you numerically compute the Lyapunov exponent for a given μ and compare with the exact result?

```
In [ ]:
```