

Hamiltonian Chaos in the Standard Map

Hamiltonian Mechanics

The Hamiltonian $H(q_i(t), p_i(t), t)$ in classical mechanics is a function of generalized positions q_i , generalized momenta p_i , $i = 1, \dots, N$, with dynamics

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \quad \dot{p}_i = -\frac{\partial H}{\partial q_i}, \quad (1)$$

where N is number of degrees of freedom.

Phase Space

Suppose that $H(q_i, p_i)$ does not depend explicitly on time t . The state of the system is represented by a point in a $2N$ -dimensional space with coordinates q_i, p_i . The system of $2N$ first order ODEs defines a unique trajectory in phase space

$$q_i = q_i(t; q_j(0), p_j(0)), \quad p_i = p_i(t; q_j(0), p_j(0)), \quad (2)$$

given an initial phase point $q_j(0), p_j(0)$. One and only one phase trajectory passes through any point in phase space, except for isolated limiting points.

Consider a *phase droplet* of initial points. Liouville's theorem[1] states that the volume of the phase droplet is conserved. If the density of initial points in the droplet is constant, the density remains constant in time. The shape of the droplet will, in general, change with time as it moves through the phase space. If droplet shape changes slowly or periodically, then the system is regular or periodic.

A trajectory is said to be chaotic if it diverges from neighboring points at an exponential rate. If droplet shape distorts drastically, then the motion is likely to be chaotic. Note that if a droplet expands exponentially in one direction, it must shrink exponentially in a perpendicular directions in order to preserve constant density!

Integrability

If H does not depend on time t

$$\frac{dH}{dt} = \frac{\partial H}{\partial p_i} \frac{dp_i}{dt} + \frac{\partial H}{\partial q_i} \frac{dq_i}{dt} = 0, \quad (3)$$

and H is conserved, and is said to be an *integral of the motion*.

The Hamiltonian system is said to be integrable if there are $N - 1$ additional independent conserved quantities (integrals of motion). All $N = 1$ time-independent Hamiltonian systems are integrable.

Non-integrable systems can have chaotic phase trajectories.

Extended Phase Space

If $H(q_i, p_i, t)$ depends explicitly on time, then phase trajectories can intersect. An *extended* phase space can be defined with t as $(2N + 1)$ -th coordinate. If the t dependence is periodic, then the extended phase space can be compactified, or made periodic, in the $(2N + 1)$ -th coordinate.

The extended phase space is convenient because only one phase trajectory can pass through any given point.

The Kicked Rotor and Standard Map

The kicked rotor is a simple example of a Hamiltonian system. Consider a pendulum with moment of inertia I in zero gravity that is given periodic impulsive kicks of fixed magnitude k at intervals of time τ . The time-dependent Hamiltonian for this system is

$$H(\theta, p_\theta, t) = \frac{p_\theta^2}{2I} + k \cos \theta \sum_n \delta(t - n\tau), \quad (4)$$

from which Hamilton's equations of motion can be derived:

$$\frac{dp_\theta}{dt} = k \sin \theta \sum_n \delta(t - n\tau), \quad (5)$$

$$\frac{d\theta}{dt} = \frac{p_\theta}{I}. \quad (6)$$

Because the momentum is constant between kicks, the angular position of the rotor changes linearly with time. Let θ_n, p_n be the position and momentum *just after* n -th kick

$$p_{n+1} = p_n + k \sin \theta_{n+1}, \quad (7)$$

$$\theta_{n+1} = \theta_n + \frac{\tau}{I} p_n. \quad (8)$$

Choosing units so $\tau/I = 1$ gives a two-dimensional map called the Standard Map[2]:

$$\theta_{n+1} = \theta_n + p_n \text{ modulo } 2\pi, \quad (9)$$

$$p_{n+1} = p_n + k \sin \theta_{n+1}. \quad (10)$$

By considering the change in a small rectangular area of phase space, it can be shown that the Standard Map is *area preserving*. A phase droplet can deform as it moves through phase space under the action of the map, but its area remains constant.

C++ Programs to Simulate The Kicked Rotor

Like the Logistic Map, this system of equations is very easy to simulate, with numerical roundoff being the only source of error.

The following program generates a phase trajectory, given the kick magnitude k and initial values $\theta(0)$ and $p_\theta(0)$.

_____ Program 1: <http://www.physics.buffalo.edu/phy410-505/topic4/stdmap.cpp> _____

```
#include <cmath>
#include <cstdlib>
#include <fstream>
#include <iostream>
```

```

using namespace std;

const double pi = 4 * atan(1.0);

double k;           // strength of periodic kick
double theta;      // angular position
double p;           // angular momentum

void map() {
    theta += p;
    while (theta >= 2 * pi)
        theta -= 2 * pi;
    while (theta < 0)
        theta += 2 * pi;
    p += k * sin(theta);
}

int main() {

    cout << " Standard Map Phase Space Trajectory\n"
         << " -----\n"
         << " Enter kick amplitude k: ";
    cin >> k;
    cout << " Enter initial theta: ";
    cin >> theta;
    cout << " Enter initial p_theta: ";
    cin >> p;

    int n_steps = 1000;
    cout << " Generating trajectory with " << n_steps << " steps" << endl;
    ofstream file("stdmap.data");
    file << theta << '\t' << p << '\n';

    for (int i = 0; i < n_steps; i++) {
        map();
        file << theta << '\t' << p << '\n';
    }

    cout << " Trajectory written to file stdmap.data" << endl;
    file.close();
}

```

Fig. 1 shows a plot of a trajectory, which is certainly not periodic and appears to be chaotic.

OpenGL Program to Generate a Phase Portrait

To explore the various trajectories of Standard Map, it is helpful to use a program that allows several trajectories with different initial conditions to be generated and plotted on the same diagram. A Java Applet by Cross[3] allows the user to add trajectories by clicking on the phase space plot.

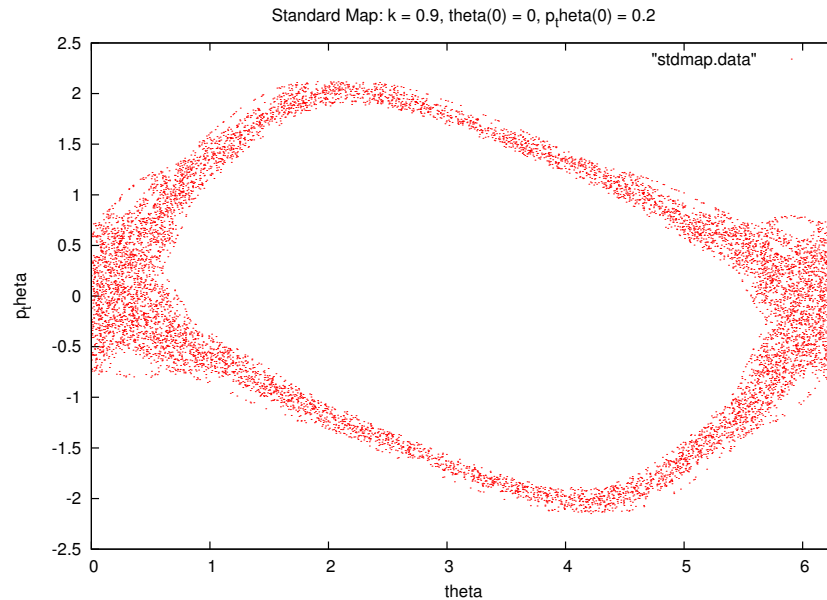


Figure 1: A single trajectory of the Standard Map for $k = 0.9$ with 10,000 points.

The following program use the OpenGL and GLUT libraries to implement interactive graphics.

_____ Program 2: <http://www.physics.buffalo.edu/phy410-505/topic4/stdmap-gl.cpp> _____

```
#include <cmath>
#include <cstdlib>
#include <iostream>
#include <string>
#include <sstream>
using namespace std;

// Include OpenGL and GLUT Libraries
// See http://www.physics.buffalo.edu/phy410-505/opengl.html

#ifdef __APPLE__
# include <GLUT/glut.h>
#else
# include <GL/glut.h>
#endif

const double pi = 4 * atan(1.0);

double k;
double theta;
double p;

void map() {
    theta += p;
    while (theta >= 2 * pi)
```

```

    theta -= 2 * pi;
while (theta < 0)
    theta += 2 * pi;
p += k * sin(theta);
}

double pMax;

void getInput() {
    cout << " Standard Map Phase Space Portrait\n"
         << " -----\n"
         << " Enter kick amplitude k: ";
    cin >> k;
    cout << " Enter maximum p to plot: ";
    cin >> pMax;
}

double theta0;
double p0;

void initialize() {
    theta = theta0;
    p = p0;
}

int N = 1000;
bool clear = true;

// define red, green, blue, yellow, cyan, magenta
GLubyte color_table[6][3] = {
    {255, 0, 0}, {0, 255, 0}, {0, 0, 255},
    {0, 255, 255}, {255, 0, 255}, {255, 255, 0}
};
int color = 0;

void display() { // display callback function to draw graphics
    if (clear) {
        glClear(GL_COLOR_BUFFER_BIT);
        clear = false;
        glColor3ub(127, 127, 127);
        glBegin(GL_LINES);
        glVertex2d(pi, -pMax); glVertex2d(pi, pMax);
        double p = 0;
        while (p < pMax) {
            glVertex2d(0, p); glVertex2d(2*pi, p);
            glVertex2d(0, -p); glVertex2d(2*pi, -p);
            p += pi;
        }
        glEnd();
    } else {
        glColor3ubv(color_table[color++]);
    }
}

```

```

        color %= 6;
        glBegin(GL_POINTS);
            for (int i = 0; i < N; i++) {
                glVertex2d(theta, p);
                map();
            }
        glEnd();
    }
    glutSwapBuffers();
}

void reshape(int w, int h) {    // reshape callback function
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0, 2*pi, -pMax, pMax, -1, 1);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    clear = true;
}

void mouse(int button, int state, int x, int y) {
    if (button == GLUT_LEFT_BUTTON) {
        if (state == GLUT_DOWN) {
            int w = glutGet(GLUT_WINDOW_WIDTH);
            theta0 = 2 * pi * x / double(w);
            int h = glutGet(GLUT_WINDOW_HEIGHT);
            p0 = -pMax + 2 * pMax * (h - y) / double(h);
            initialize();
            glutPostRedisplay();
        }
    } else if (button == GLUT_RIGHT_BUTTON) {
        if (state == GLUT_DOWN) {
            clear = true;
            glutPostRedisplay();
        }
    }
}

int main(int argc, char *argv[]) {
    glutInit(&argc, argv);
    getInput();
    initialize();
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowSize(400, 400);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Standard Map");
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glShadeModel(GL_FLAT);
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
}

```

```
glutMouseFunc(mouse);  
glutMainLoop();  
}
```

Homework Problem

Explore the phase space of the standard map for different values of the kicking parameter k (for example 0, 0.1, 0.5, 0.7, 0.9, 1.3, 4.0, 4.5) and describe the nature of the trajectories in terms of motion of the rotor. Simulate a small rectangle of initial points. How does the shape of this rectangle evolve for different values of k ?

References

- [1] Wikipedia: Liouville's Theorem, [http://en.wikipedia.org/wiki/Liouville's_theorem_\(Hamiltonian\)](http://en.wikipedia.org/wiki/Liouville's_theorem_(Hamiltonian)).
- [2] Wikipedia "Standard Map", http://en.wikipedia.org/wiki/Standard_map, Scholarpedia, "Chirikov Standard Map", http://www.scholarpedia.org/article/Chirikov_standard_map.
- [3] M. Cross, "Chaos on the Web", http://www.cmp.caltech.edu/~mcc/Chaos_Course/.