Chaos in the Swinging Atwood Machine (SAM)

Nathaniel Adams

November 23, 2015

1 Introduction to the SAM

A normal Atwood Machine is composed of two masses connected with a string, hanging from pulleys, (see figure 1). This system is often used to teach Newton's laws of motion due to its simplicity. However, it is not very interesting to study beyond that. When the two masses are not equal, the heavier one will accelerate down and the other will accelerate up at the same rate. When the masses are equal, there is no acceleration. In order to make the system more interesting, it is possible to introduce a second degree of freedom, giving rise to the Swinging Atwood Machine.

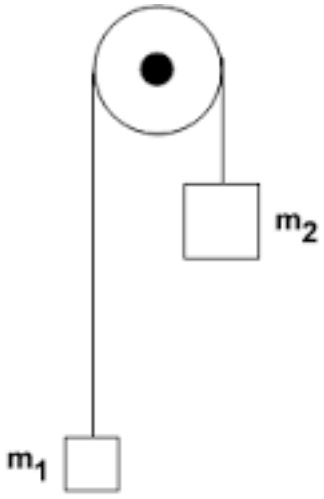


Figure 1: Atwood Machine, `http://physics.stackexchange.com/questions/118917/newtons-third-law-and-atwood-machines-confusion-about-tension`

In the Swinging Atwood Machine (see figure 2), a second degree of freedom is added by letting one of the masses swing from the pulley through an angle θ. Since the total energy of the system is no longer simply dependent on the height of the two masses, bounded orbiits can exist. Note that while one mass can move throughout the plane and is free in 2 dimensions, the mass constrained to moving vertically does not have another degree of freedom as its height is completely defined by r, the distance between the other mass and its pulley.

Figure 2: Swinging Atwood Machine, `https://en.wikipedia.org/wiki/Swinging_Atwood%27s_machine`

Many simplifications can be made to solve this system. The first is that there is no friction. Otherwise, the damping would force the system toowards $\theta = 0$, where it would behave like the regular Atwood Machine. All orbits would either asymptotically approach a fixed point or be unbounded and go to infinity. Another simplification is that the pulleys themselves are massless and have negligible radii.

2 Derivation of Hamiltonian Equations

Using classical physics, it it easy to derive the energy present in the system. $PE = mgh$ and $KE = \frac{1}{2}mv^2$

M has a potential energy of Mgr and a kinetic energy of $\frac{1}{2}M\dot{r}^2$

m has a potential energy of $-mgrcos(\theta)$ and a kinetic energy of $\frac{1}{2}m(\dot{r}^2 + r^2\dot{\theta}^2)$

The total potential energy of the system is given by $Mgr - mgrcos(\theta))$

The total kinetic energy of the system is given by $\frac{1}{2}M\dot{r}^2 + \frac{1}{2}m(\dot{r}^2 + r^2\dot{\theta}^2))$

The Lagrangian of the system is the difference between kinetic and potential energies:

$$L = \frac{1}{2}M\dot{r}^2 + \frac{1}{2}m(\dot{r}^2 + r^2\dot{\theta}^2)) - Mgr + mgrcos(\theta))$$

The Hamiltonian of the system is the total energy in the system:

$$H = \frac{1}{2}M\dot{r}^2 + \frac{1}{2}m(\dot{r}^2 + r^2\dot{\theta}^2)) + Mgr - mgrcos(\theta)) \tag{1}$$

Find the values of p_i for the system. For simplicity of notation, let $q_1 = r, p_1 = p_r, q_2 = \theta, p_2 = p_\theta$

$$p_r = \frac{\partial L}{\partial \dot{r}} = (M+m)\dot{r}$$
$$p_\theta = \frac{\partial L}{\partial \dot{\theta}} = mr^2\dot{\theta}$$

Plugging these back in gives $H = \frac{1}{2}(M+m)(\frac{p_r}{M+m})^2 + \frac{1}{2}mr^2(\frac{p_\theta}{mr^2})^2 + Mgr - mgrcos(\theta)$

$$H = \frac{p_r^2}{2(M+m)} + (\frac{p_\theta^2}{2mr^2})^2 + Mgr - mgrcos(\theta) \tag{2}$$

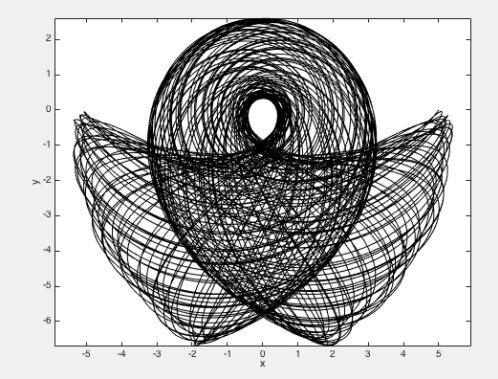
This equation can be used to find a system of 4 coupled differential equaitons describing the motion of the system. Given some initial condition, the equations below can be plugged into a numerical ODE solver like ode45 in Matlab to find its orbit.

$$\begin{aligned}
\dot{r} &= \frac{\partial H}{\partial p_r} = \frac{p_r}{M+m} \\
\dot{\theta} &= \frac{\partial H}{\partial p_\theta} = \frac{p_\theta}{mr^2} \\
\dot{p}_r &= -\frac{\partial H}{\partial r} = \frac{p_\theta^2}{mr^3} - Mg + mgcos(\theta) \\
\dot{p}_\theta &= -\frac{\partial H}{\partial \theta} = -mgrsin(\theta)
\end{aligned} \tag{3}$$

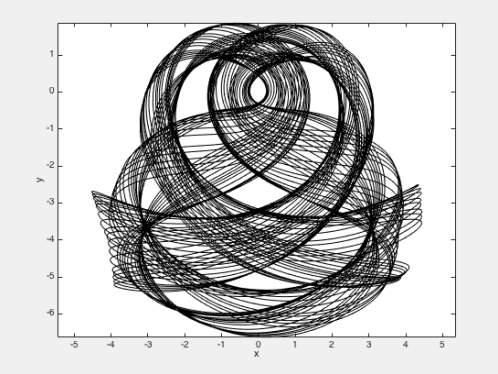
These equations give rise to many different orbits, depending on the values of parameters and initial conditions given (see figure 3).

3 Sensitive Dependence

Starting with some simple numerical models, it quickly became clear that the Swinging Atwood machine exhibited chaotic characteristics. To start, I plotted the orbits of two points starting very close to each other. The distance between these points grew exponentially over time until they reached a distance on the order of the initial radius apart (see figure 4). This shows the possibility of sensitive dependence in the swinging Atwood Machine. However, since this may not be true for initial conditions, a more rigorous method must be used to prove Chaos. To do this, I calculated the Lyapunov exponents of the orbits.



(a) $r = 5, \theta = \pi/2, p_r = 0, p_\theta = 0, m = 1, M = 3$



(b) $r = 5, \theta = \pi/3, p_r = 0, p_\theta = 0, m = 1, M = 2$

Figure 3: Some orbits of the SAM

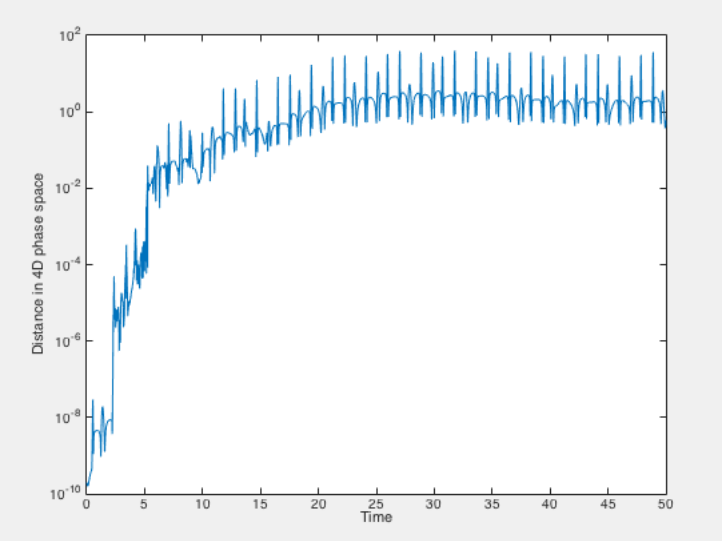


Figure 4: Sensitive dependence in the SAM

3.1 The Jacobian

The first step in calculating the Lyapunov exponents of a flow is finding the Jacobian of the differential equations. For this system it is:

$$J = \begin{pmatrix} \frac{\partial}{\partial r}\dot{r} & \frac{\partial}{\partial \theta}\dot{r} & \frac{\partial}{\partial p_r}\dot{r} & \frac{\partial}{\partial p_\theta}\dot{r} \\ \frac{\partial}{\partial r}\dot{\theta} & \frac{\partial}{\partial \theta}\dot{\theta} & \frac{\partial}{\partial p_r}\dot{\theta} & \frac{\partial}{\partial p_\theta}\dot{\theta} \\ \frac{\partial}{\partial r}\dot{p}_r & \frac{\partial}{\partial \theta}\dot{p}_r & \frac{\partial}{\partial p_r}\dot{p}_r & \frac{\partial}{\partial p_\theta}\dot{p}_r \\ \frac{\partial}{\partial r}\dot{p}_\theta & \frac{\partial}{\partial \theta}\dot{p}_\theta & \frac{\partial}{\partial p_r}\dot{p}_\theta & \frac{\partial}{\partial p_\theta}\dot{p}_\theta \end{pmatrix} = \begin{pmatrix} 0 & 0 & \frac{1}{M+m} & 0 \\ \frac{-2p_\theta}{mr^3} & 0 & 0 & \frac{1}{mr^2} \\ \frac{-3p_\theta^2}{mr^4} & -mgsin(\theta) & 0 & \frac{2p_\theta}{mr^3} \\ -mgsin(\theta) & mgrcos(\theta) & 0 & 0 \end{pmatrix}$$

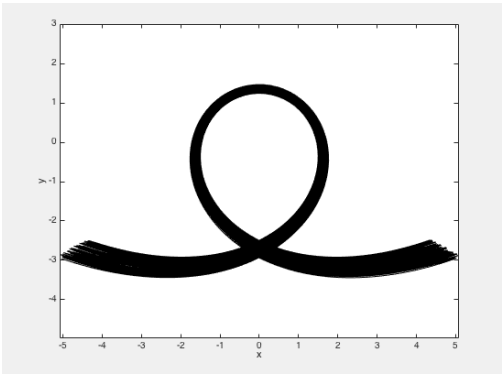
Then using Matlab, it is possible to iterate the ode45 solver for the values of the orbit as well as updating the Jacobian. Then, averaging the exponential growth of the Jacobian matrix over time gives values for the Lypunov exponents numerically. I used Matlab for all the coding. My programs are attached at the end of this document.

4 Parameter Variation

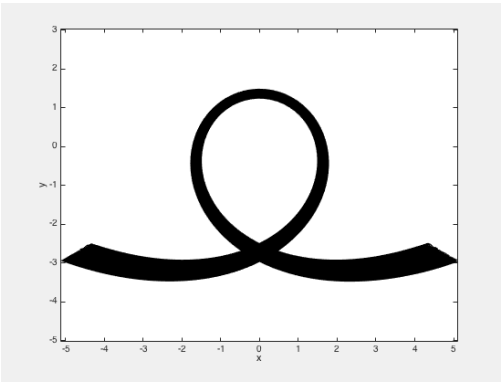
Most of the numerical exploration in this project came from observing what happened to Lyapunov exponents while changing all the different variables in the system.

4.1 Mass

The first observation about changing the values of mass in the system is that the ratio between masses M and m determines the characteristics of the system, but the the actual values do not. For example, the orbit of a system with m=1 and M=5 (see figure 5a) looks identical to that of a system with m=7 and M=35 (see figure 5b). The starting conditions for this orbit were randomly chosen to be $r = 5, \theta = \pi/3, p_r = 0, p_\theta = 0$. From here on, the ratio M/m will be denoted by μ, to be consistent with the literature.



(a) Mass Dependence, m=1 and M=5



(b) Mass Dependence, m=7 and M=35

Figure 5: $\mu = 5$

The next step in this exploration was to vary the value of μ in the system. I set m=1, M=1μ for simplicity. I chose an initial condition almost at random, while making sure it would not generally lie on a stable or unstable manifold. $r = 1, \theta = 0.5, p_r = 0.1, p_\theta = 0.1$ was a good set of initial conditions for this. I then varied μ between 1 and 100 to obtain the Lyapunov exponents. Since this is a system of 4 equations, there are 4 Lyapunov exponents returned by the code. The inital plot of these values is shown in figure 6. By increasing the number of averaging loops and increasing the number of μ values used to calculate the Lyapunov exponents, the figure became more detailed (see figure 7).

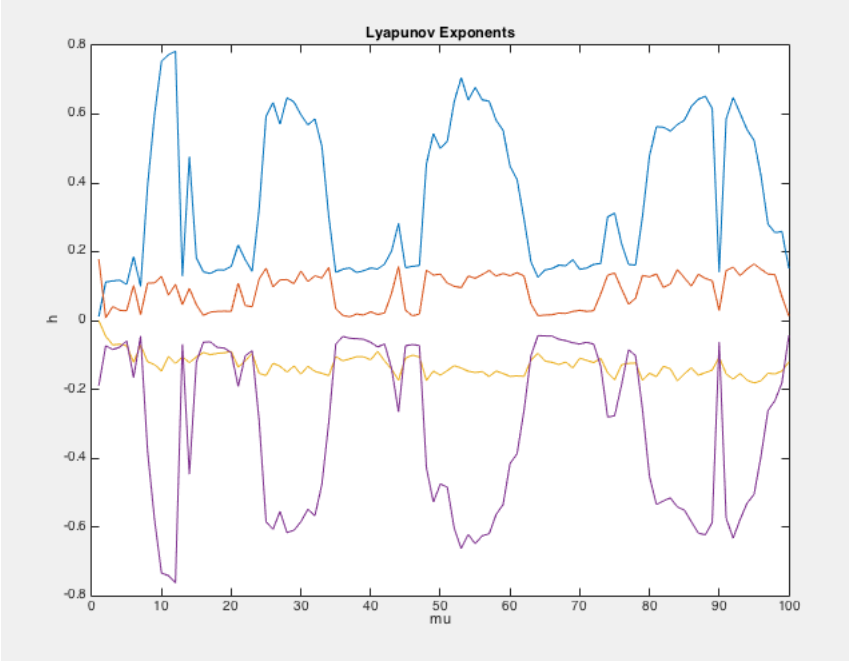


Figure 6: μ Dependence

4.2 Initial Conditions

Since sensitive dependence is a property of chaotic system, there should be a lyapunov exponent greater than zero independent of the starting point. To make sure I didn't just randomly guess some initial condition with a positive lyapunov exponent, I varied the 4 initial conditions one at a time to see how they would impact the values of h. Figure 8 shows the impact that varying initial conditions has on the lyapunov exponents. These plots assume that three variables stay fixed at $r = 1, \theta = 0.5, p_r = 0.1, p_\theta = 0.1$ while the fourth is varied through a range of values. μ here is fixed at 10, since that parameter gives rise one of the largest h values in the previous graphs. That indicates it should be easiest to see any changes that may occur at that value.

The results show that changing the initial values of θ, p_r, and p_θ have very little effect on the Lyapunov exponents calculated for the system. The initial value of r does apepar to effect the values of h. By increasing

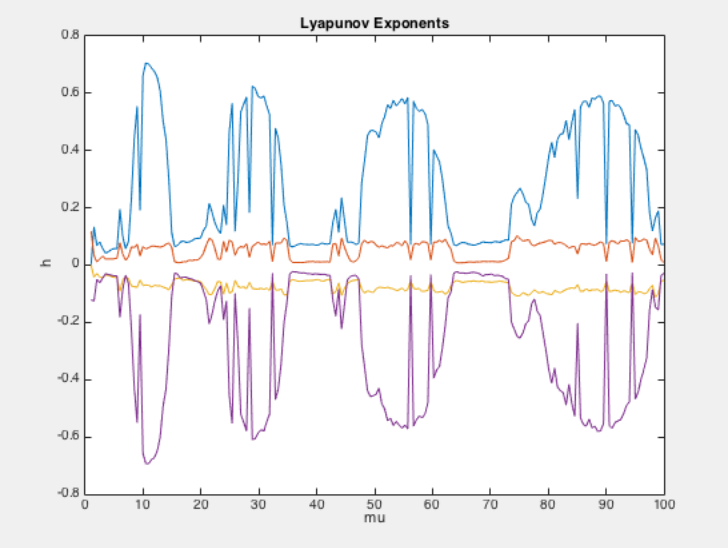
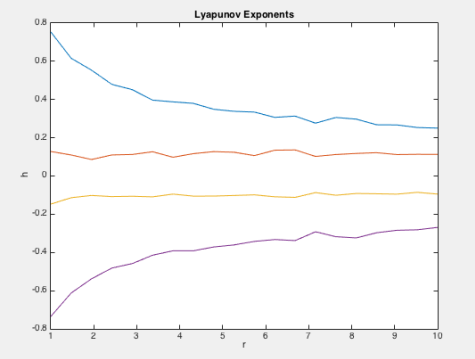
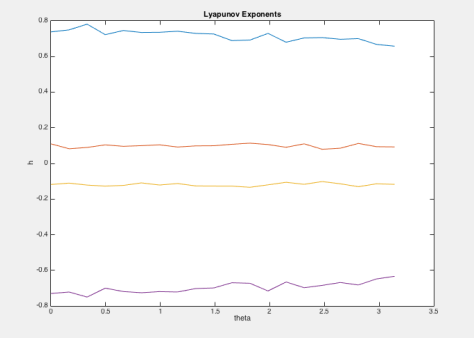


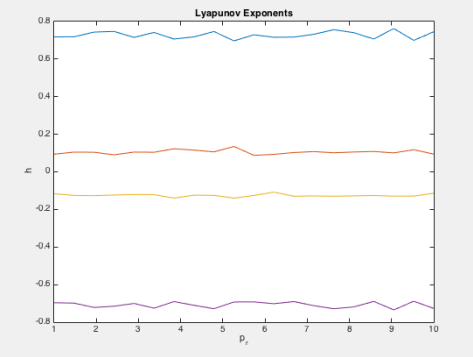
Figure 7: Detailed μ Dependence



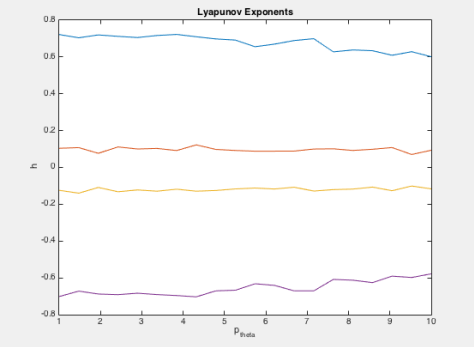
(a) $r \in [1, 10]$



(b) $\theta \in [0, \pi]$



(c) $p_r \in [1, 10]$



(d) $p_\theta \in [1, 10]$

Figure 8: Variations of initial conditions

the initial radius, the lyapunov exponents grow closer to zero. However, the curves look like they level off before approaching zero (or h2). Physically, this means that two nearby points will grow apart faster when the initial conditions involve a smaller radius and should diverge slower with a larger initial radius. Having said that, varying r should not make a chaotic orbit become periodic or vice versa.

5 Analysis

In theory, Lyapunov exponents of a Hamiltonian system should come in pairs. That means for every exponent h, there must also be an exponent of -h. This implies that the sum of all the exponents must be zero. Therefore any Hamiltonian flow must be volume preserving in $\mathbb{R}^{2n}$, which is one statement of Liouville's Theorem (Kathleen T. Alligood, Section 9.6). Since this is a continous Hamiltonian flow, one of the Lyapunov exponents must also equal zero (Gerald Jay Sussman, Section 4.2). This makes sense since starting a small distance farther along the orbit should cause the final value to move a small distance along the orbit as well, without exponential growth or decay. In this direction, h must equal zero.

In the case of the Swinging Atwood Machine, there are 4 coupled differential equations, leading to 4 Lyapunov exponents. However, by the theorems above, they must be $(h, 0, 0, -h)$. The numerical solutions give two numbers close to zero, but not exactly zero. This fact can be used to put an error bound around the plot of h. Since there should only be one h value in question, but there are actually 4 exponents, I averaged h_1 and $-h_4$ to get the value of h. I averaged h_2 and $-h_3$ to get the error at each value of μ, since in theory this should equal zero. I multiplied this average error value by 2 to be conservative about the accuracy in the plot. Using the more detailed μ data plotted earlier, I created a new analyzed image (figure 9). While the graph may seem busy, the important parts to notice are the values of μ where the lower red line dips below zero. Any point where h could be at or below zero indicates a point that could show non-chaotic, integrable behavior. Points that remain above zero for the entire error range are almost certainly chaotic.

According to another study (J. Casasayas), $\mu = 3$ is the only value that can possibly give an integrable solution analytically. The paper goes on to show that $\mu = 4n^2 - 1$ may also appear to be integrable from numerical simulations, but they are not because the Hamiltonian is the only conserved quantity along an orbit except when $\mu = 3$. Examining these values (3, 15, 35, 63, 99) with respect to figure 9 gives a pretty neat comparison. These values seem to occur exactly the ends of the "hills" that occur in my graph. At $\mu = 3$, h is small and it is hard to make out exactly what is happening, but at 15, it comes back to zero after a hill. At 35, it comes back to zero after a hill, etc... Casasayas' paper goes on to explain why these are not actually integrable, but it goes into more complicated analysis that goes beyond the scope of this course.

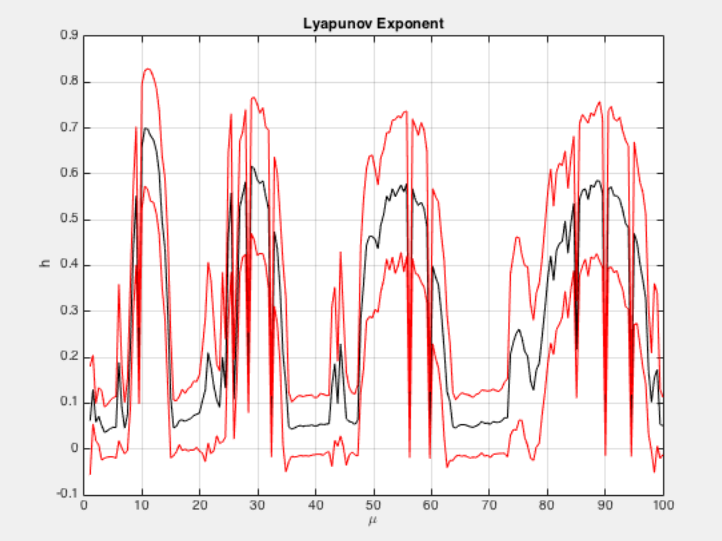


Figure 9: μ Dependence with Error

6 Further Exploration

After completing this project, there are many directions that further research in this topic could head. One is simply getting more accurate values of h. By averaging the iterative time-1 map over more loops, the solution should slowly converge to the exact values of h for the system. However, this would require more computational power over more time. If I could just let code sit for weeks, it might give more accurate solutions, but that is not practical in a 10 week course.

Another possibility is to extend the range of μ (like $[1, 1000]$) and see if it the numerical solution follows the same pattern between values of $4n^2 - 1$ that held for $\mu \in [1, 100]$. It would also be interesting to see what the apparently random dips in the "hills" correspond to: do they have physical meaning or are they just anomalies of the numerical method? I could also extend the range of the initial r to see if h ever converged to zero or if it asymptotically approached some positive number. Physically, I would expect that the starting radius would not affect whether a system is chaotic or not, but it might be a worthwhile study. I could also try running the initial condition variations for some initial point other than $\mu = 10, r = 1, \theta = 0.5, p_r = 0.1, p_\theta = 0.1$ and see if the trends still looked the same.

Another extension to the project could involve looking at the analytical solutions to the Swinging Atwood Machine. While most orbits are chaotic, any μ could have periodic orbits if the initial conditions are chosen carefully. This would tie into an exploration of the integrable case $\mu = 3$ that should have a completely defined solution family.

To anyone interested in learning more about the Liouville Theorem and the behavior of Lyapunov exponents in general in $\mathbb{R}^{2n}$ phase space, Chapters 3 and 4 of the Sussman-Wisdom textbook give a very in-depth explanation of these properties.

References

[1] Gerald Jay Sussman, Jack Wisdom. *Structure and Interpretation of Classical Mechanics*. Cambridge. MIT Press. (2001). Web.
http://www.fisica.net/mecanicaclassica/struture_and_interpretation_of_classical_mechanics_by_gerald_jay_sussman.pdf

[2] J. Casasayas, A. Nunes, N. Tufillaro. *Swinging Atwoods Machine : Integrability and Dynamics*. Journal de Physique. 51 (16): 1495-1500, 1990. Web.
https://hal.archives-ouvertes.fr/jpa-00212480/document

[3] Kathleen T. Alligood, Tim D, Sauer, James A. Yorke. *Chaos: An Introduction to Dynamical Systems* New York. Springer. (1996). Print.

[4] N. Tufillaro. *Motions of a Swinging Atwoods Machine*. Journal de Physique. 46 (9): 1693-1702, 1985. Web.
https://hal.inria.fr/file/index/docid/210094/filename/ajp-jphys_1985_46_9_1495_0.pdf

[5] Nicholas B. Tufillaro, Tyler A. Abbott, David J. Griffiths. *Swinging Atwood's Machine*. American Journal of Physics. 52 (10): 895-903, 1984. Web.
http://step.ipgp.fr/images/f/f3/SAM.pdf

[6] Parker W. Moody. *Modeling a Swinging Atwood Machine*. Journal of the Advanced Undergraduate Physics Laboratory Investigation. 1 (1): 1-7, (2013). Web.
http://opus.ipfw.edu/cgi/viewcontent.cgi?article=1120&context=jaupli

Except when stated otherwise, figures are my own work, created using Matlab.

```
% Nathaniel Adams
% Code adapted from lyapflow

% This code allows the user to vary any of the system parameters and
% calculate the lyapunov exponents. To change which parameter is being
% varied, comment and uncomment the appropriate lines below. To change the
% values that the parameter takes on, change the arguments of linspace()
% appropriately.

clc
clear

mus=10;
% mu=10;
% mus=linspace(1,100,200);
% th0s=linspace(0,pi,20);
% r0s=linspace(1,10,20);
% rp0s=linspace(1,10,20);
% pth0s=linspace(1,10,20);

h = zeros(4,numel(mus));
g=9.81;
m=1;
for i=1:numel(mus)
mu=mus(i);
M=mu*m;
% th0=th0s(i);
% pth0=pth0s(i);
% r0=r0s(i)
% rp0=rp0s(i);

f=@(t,z) [z(3)/(M+m)
    z(4)/(m*z(1).^2)
    z(4).^2/(m*z(1)^3)-M*g+m*g*cos(z(2))
    -m*g*z(1)*sin(z(2))];

% Measure Lyapunov Exponents

% Varying K and L changes how accurate the solution is. In general,
% increasing these values increases precision.
K = 50;                                 % how many averaging loops
L = 100;                                % how many its per meas step

z = [1; 0.5; 0.1; 0.1]; % Initial condition

for k=1:K
  J = eye(4);                   % Id is where Jacobean starts
  for l=1:L
    [z, Jz] = SAM_map(z,m,M);
    J = Jz*J;                         % update Jacobean
    [Q,R] = qr(J);                    % re-orthogonalize
    J = Q*diag(diag(R));              % but keep them correct lengths
  end
  rL = abs(diag(R));
  h(:,i) = h(:,i) + log(rL)/L;                % sum up lyap exps from each run
end
h(:,i) = h(:,i)/K;

% Show progress
fprintf('%f, %f, %f, %f, mu=%f\n',h(1,i),h(2,i),h(3,i),h(4,i),mu)

end
```

```
function [z, DFz] = SAM_map(x0,m,M)
% Converts the ODE flow for the SAM into a map, taking one time step.
% This function includes updating the Jacobian, halping find the lyapunov
% exponents. Based on the code lorentz_time1map.

g=9.81;

% ODE
f=@(t,z) [z(3)/(M+m)
    z(4)/(m*z(1).^2)
    z(4).^2/(m*z(1)^3)-M*g+m*g*cos(z(2))
    -m*g*z(1)*sin(z(2))];

% Jacobian
Df=@(z) [0, 0, 1/(M+m), 0
    -2*z(4)/(m*z(1).^3), 0, 0, 1/(m*z(1).^2)
    -3*z(4).^2/(m*z(1)^4), -m*g*sin(z(2)), 0, 2*z(4)/(m*z(1)^3)
    -m*g*sin(z(2)), -m*g*z(1)*cos(z(2)), 0, 0];

% Initial matrix is I4
J0=eye(4);

% Reshape to pass through ode45
G=@(t,z) [f(t,z(1:4,:))
    Df(z(1:4,:))*z(5:8,:);
    Df(z(1:4,:))*z(9:12,:);
    Df(z(1:4,:))*z(13:16,:);
    Df(z(1:4,:))*z(17:20,:)];

% Solve Numerically
[ts,zs] = ode45(G,[0 1], [x0; J0(:)],odeset('abstol',1e-3));

% Reshape solution back into its matrix form
z = reshape(zs(end,1:4),[4,1]);
J = zs(end,5:end);
DFz = reshape(J,[4 4]);
end
```

```
% Nathaniel Adams
% SAM Simulation

% After changing the initial conditions and parameter mu, this  code will
% give an image of the resulting orbit from t=0 to t=300 using the built
% in ode45 command.

% Initial conditions
z0=[5,pi/2,0,0];
% System parameter
mu=3;


% Don't change code past here

% Finish defining system parameters
m=1;
M=mu*m;
g=9.81;

% Hamilton's equations given by
% r_dot=p_r/(M+m)                       z(1)
% th_dot=p_th/(mr^2)                    z(2)
% pr_dot=p_th^2/(mr^3)-Mg+mg*cos(th)    z(3)
% pth_dot=-mgr*sin(th)                  z(4)

f=@(t,z) [z(3)/(M+m)
    z(4)/(m*z(1).^2)
    z(4).^2/(m*z(1)^3)-M*g+m*g*cos(z(2))
    -m*g*z(1)*sin(z(2))];

tmax=300; % Range of t

sol=ode45(f,[0 tmax],z0); % odeset('abstol', 1e-12)

t=0:0.01:tmax;
z=deval(sol,t);

% Convert Polar Coordinates to Cartesian
% (r, theta) -> (x,y)
x=z(1,:).*sin(z(2,:));
y=z(1,:).*-cos(z(2,:));

% Plot resulting orbit
figure(1)
plot(x,y,'k')
xlabel('x')
ylabel('y')
axis equal
```

```
% Nathaniel Adams
% This code shows, in general, a point near z0 will usually iterate away
% from zn. mu, z0 and z0e can be changed as necessary.

mu=7;
z0=[1,0.5,0.1,0.1];
z0e=z0+1E-10*[1,1,1,1];


% System parameters
m=1;
M=mu*m;
g=9.81;

% System of ODEs
f=@(t,z) [z(3)/(M+m)
    z(4)/(m*z(1).^2)
    z(4).^2/(m*z(1)^3)-M*g+m*g*cos(z(2))
    -m*g*z(1)*sin(z(2))];

tmax=50;

% Solve point 1
sol=ode45(f,[0 tmax],z0);
t=0:0.01:tmax;
z=deval(sol,t);
% Solve point 2
sole=ode45(f,[0 tmax],z0e);
te=0:0.01:tmax;
ze=deval(sole,te);

e=abs(z-ze);

% Calculate distance
err=sqrt(e(1,:).^2+e(2,:).^2+e(3,:).^2+e(4,:).^2);

% Plot distance between points vs time
figure(1)
semilogy(t,err)
xlabel('Time')
ylabel('Distance in 4D phase space')


x=z(1,:).*sin(z(2,:));
y=z(1,:).*-cos(z(2,:));

xe=ze(1,:).*sin(ze(2,:));
ye=ze(1,:).*-cos(ze(2,:));

% Plot updating orbit over time
for i=1:numel(x)-1
    figure(2)
    plot(x(i),y(i),'ob',xe(i),ye(i),'or')
    axis([min(x) max(x) min(y) max(y)])
    xlabel('x')
    ylabel('y')
    grid on
    pause(0.05)
end
```

```
% Nathaniel Adams
% Watch an orbit evolve over time.

% Vary these two lines only to produce any given orbit
mu=3;
z0=[1,0.5,0.1,0.1];


% System parameters
m=1;
M=mu*m;
g=9.81;

% System of ODEs
f=@(t,z) [z(3)/(M+m)
    z(4)/(m*z(1).^2)
    z(4).^2/(m*z(1)^3)-M*g+m*g*cos(z(2))
    -m*g*z(1)*sin(z(2))];

tmax=300;

sol=ode45(f,[0 tmax],z0); % odeset('abstol', 1e-12)

t=0:0.02:tmax;
z=deval(sol,t);

% Convert to cartesian coordinates
x=z(1,:).*sin(z(2,:));
y=z(1,:).*-cos(z(2,:));

% Graph solution
figure(1)
for i=1:numel(x)-1
    plot([x(i),x(i+1)],[y(i),y(i+1)],'k')
    axis([min(x) max(x) min(y) max(y)])
    xlabel('x')
    ylabel('y')
    hold on
    pause(0.01)
end
```