Homework Assignment № 6

Kevin Mack, Clarkson University

11/15/2017

Problem Statement:

In this assignment the task is to use Data Assimilation techniques to model the dynamics of the Rossler equations when the initial conditions have added noise. The Rossler equations, also known as the Rossler attractor, are a system of three non-linear ordinary differential equations that exhibit chaotic dynamics; this is because it is particularly sensitive to the initial conditions of the system. This means that when noise is added to the initial conditions, the error of modeling the system diverges as time progresses. The Kalman filter is a perfect use-case for this scenario because it allows the system to retroactively update its estimate of the initial conditions based on the error of the model in time. This assignment will investigate the effectiveness of the ensemble Kalman Filter (EnKF) in minimizing the error involved in modeling the dynamics of the Rossler equations when noise is present.

Part 1: Model the Rossler Equations

First, the Rossler equations given by (1) are modeled in Matlab using the **ode45** command. The equations are dependent on three constants $a, b,$ and c, and three initial conditions $x_0, y_0,$ and z_0. The Matlab solution is given by Listing 1, and the graph is given by Figure 1.

$$\begin{aligned}\frac{dx}{dy} &= -y - z\\ \frac{dy}{dt} &= x + ay\\ \frac{dz}{dt} &= b + z(x - c)\end{aligned} \tag{1}$$

Listing 1: Matlab code – Simulate Rossler Attractor

```
%% EE520 HW6
% Set constants and time interval
a = 0.2; b = 0.2; c = 40.0;
t = 0:.0001:100;

% set initial conditions
x_init = [1 1.001];
y_init = [1 1];
z_init = [1 1];
```

```

% Rossler equations
f = @(t,x) [ -x(2)-x(3); x(1)+a*x(2); b+x(1)*x(3)-c*x(3)];

% ODE solver for Rossler equations with two sets of initial conditions
[t1,x1] = ode45(f, t, [x_init(1) y_init(1) z_init(1)]);
[t2,x2] = ode45(f, t, [x_init(2) y_init(2) z_init(2)]);

% plot solution of Rossler equations
figure
hold on
grid on
grid minor
title('Rossler Equations')
plot3(x1(:,1), x1(:,2), x1(:,3), 'g-')
plot3(x2(:,1), x2(:,2), x2(:,3), 'b-')

x_1 = x1(:,1); y_1 = x1(:,2); z_1 = x1(:,3);
x_2 = x2(:,1); y_2 = x2(:,2); z_2 = x2(:,3);
```

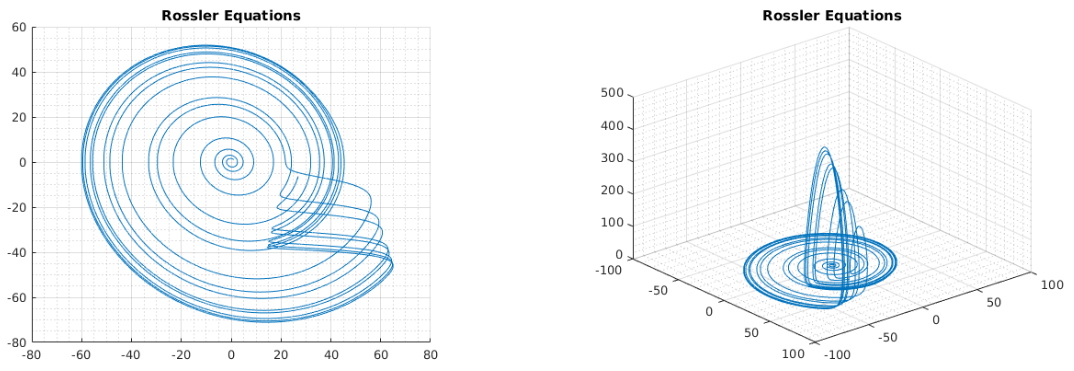


Figure 1: This figure depicts two angles of the same solution to the Rossler equations from time $t = 0$ to $t = 100$ for $a = 0.2$, $b = 0.2$, and $c = 8.0$, with initial conditions $x_0 = 1$, $y_0 = 1$, and $z_0 = 1$. From the graphs it can be seen that the attractor follows an outward spiral in the x, y plane, until reaching an unstable point in the system. At the unstable point, the surface folds, and rises in the z plane. The system oscillates chaotically between these two states as time progresses.

Part 2: Investigate the Effects of Initial Conditions

It is now important to investigate the effects that the initial conditions have on the system, due to the fact that the Rossler attractor is known to behave chaotically. It is expected that as time progresses, the difference between systems with different initial conditions will diverge. How quickly these systems diverge from each other will depend on how chaotic the system is and how different the initial conditions are from each other. In this test case, the parameters will be the same as in Part 1, and the initial conditions will be the only change in the model. The code to generate the solution to the Rossler equations is given in Listing 1, while the code to generate the difference between test cases and plot the results is given in Listing 2.

Listing 2: Matlab code – Find Error in System Model

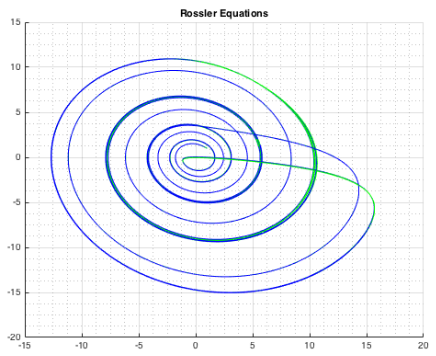
```
% Find error between two different sets of initial conditions
e = zeros(1, size(t,2));
for i = 1:size(t,2)
    e(i) = norm([x_1(i)-x_2(i),y_1(i)-y_2(i),z_1(i)-z_2(i)]);
end

% Create error plots
figure
title('Error Plot')
subplot(1,2,1)
hold on
grid on
grid minor
plot(t, e)
xlabel('Time')
ylabel('Euclidean Norm of Error')
subplot(1,2,2)
hold on
grid on
grid minor
plot(t, log10(e))
xlabel('Time')
ylabel('Euclidean Norm of Error (Log Scale)')
```

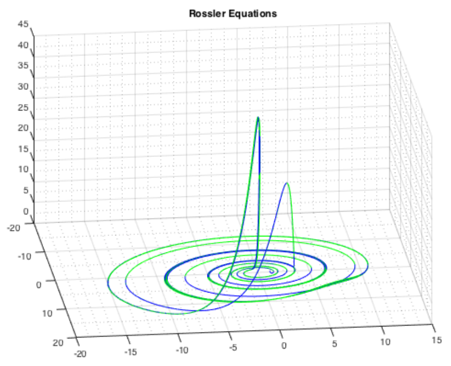
In order to demonstrate the chaotic nature of the attractor, one more case for initial conditions will be considered. Note that it is also possible to change the parameters of the model (a, b, and c) in order to obtain similar results. It is also likely that given more time, the solutions will continue to diverge until the solutions appear to be visibly different on the graphs. The third solution is given in Figure 4, and has comparably more divergent solutions than Figure 2.

Part 3: Investigate the Effects of Noise

It is important to inspect how noise in the initial conditions affects the solution of the system. In Part 2, it appears that initial conditions are extremely important in keeping the solutions from diverging, however more inspection is needed. Here we will look at the model dynamics for several trials with random noise added to the initial conditions. The code to introduce this noise in the system is given by Listing 3, with the model dynamics compared to the original model in



(a) Solution with initial conditions $x_0 = 1.0$, $y_0 = 1.0$, and $z_0 = 1.0$.



(b) Solution with initial conditions $x_0 = 1.0$, $y_0 = 1.0$, and $z_0 = 1.001$.

Figure 2: In this figure both solutions (different initial conditions) to the Rossler are plotted on the same graph, and it is difficult to tell at this scale how different they are. This suggests that the Rossler equations either need more time to develop, or need different model parameters in order to demonstrate highly chaotic behavior at this scale. The differences between models are shown in Figure 3.

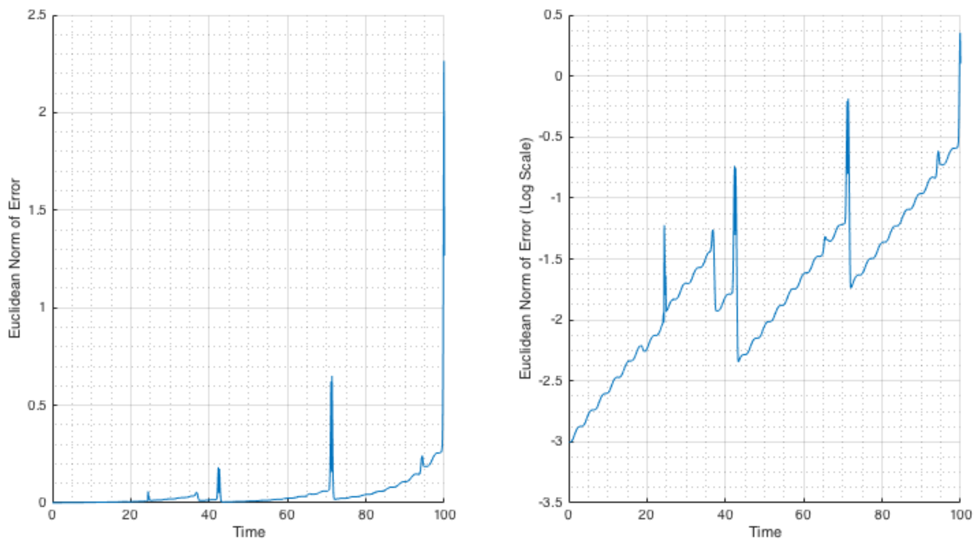
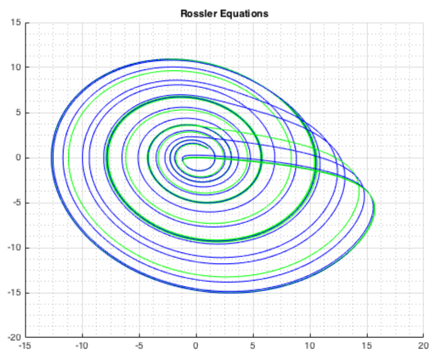


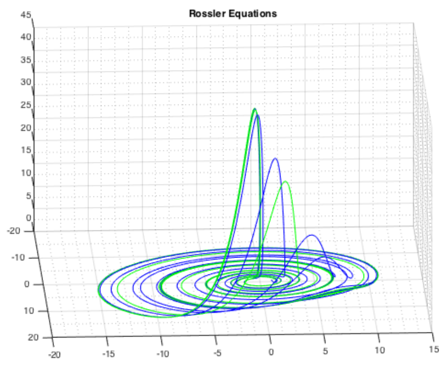
Figure 3: The Figure shows the difference between the two systems whose initial conditions are slightly different. It can be seen that as time progresses, the general trend of the error is to diverge. There are also spikes of high error which correspond to the points where the solution of the attractor folds over, and rises or falls quickly in the z-plane.

Listing 1. The results are given by Figure 6. It is clear from the results here that it is important to consider the noise in the initial conditions in order to allow for accurate modeling of a given Rossler attractor. In Part 4, the EnKF will be used to keep the error introduced by this noise from causing the solution to diverge from the solution given by the original initial conditions.

Listing 3: Matlab code – Produce Error Dynamics

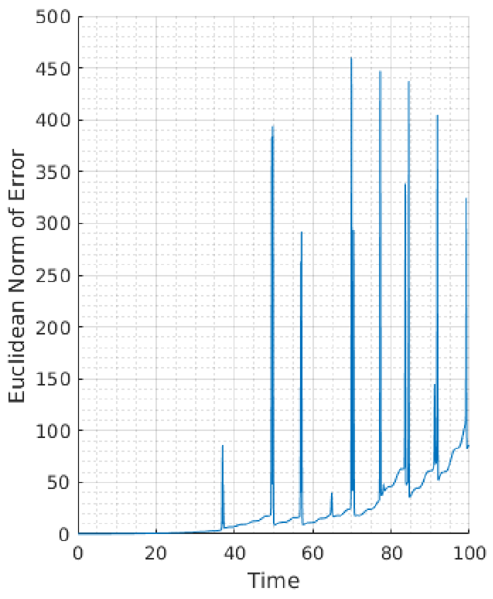


(a) Solution with initial conditions $x_0 = 1.0$, $y_0 = 1.0$, and $z_0 = 1.0$.



(b) Solution with initial conditions $x_0 = 1.0$, $y_0 = 1.0$, and $z_0 = 1.1$.

Figure 4: In this figure both solutions (different initial conditions) to the Rossler are plotted on the same graph, and the difference in the solutions is visible. As the solution spirals out from the center, the two solutions begin to diverge. When the attractor folds over the solutions become even more dissimilar and result in large amounts of error. These graphs show how even a factor 0.1 can impact the solution of the system, and the importance of introducing Data Assimilation when initial conditions have noise.



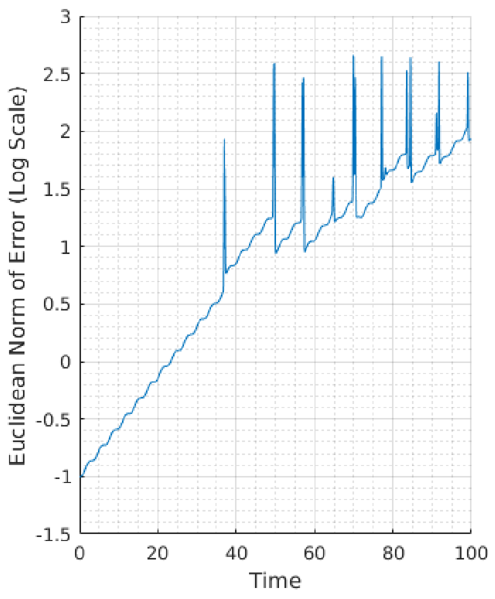


Figure 5: These plots depict the difference between the two systems whose initial conditions are different. It can be seen that as time progresses, the general trend of the error is once again to diverge. Again there are spikes of high error which correspond to the points where the solution of the attractor folds over, and rises or falls quickly in the z-plane.

```
t = 0:0.001:100;
a = 0.2; b = 0.2; c = 8.0;

x0 = [1 1 1];

```

```
f = @(t,x) [ -x(2)-x(3); x(1)+a*x(2); b+x(1)*x(3)-c*x(3)];
[t, xsol] = ode45(f, t, x0);
x_true=xsol(:,1); y_true=xsol(:,2); z_true=xsol(:,3);

sigma2=1;% Error variance for initial conditions

figure
hold all
title('Numerical Approximation')
for j=1:8
    xic = x0+sigma2*randn(1,3);
    [t, xsol] = ode45(f, t, xic);
    x=xsol(:,1);
    subplot(4,2,j)
    hold on
    plot(t,x_true,'k')
    plot(t,x,'b','Linewidth',[2])
    xlabel('t')
    ylabel('x')
end
```

Part 4: Apply the Kalman Filter and Compare Results

In this section the ensemble Kalman Filter method is used to keep the system with noise from diverging from the true solution. The EKF is used to enhance our prediction of the future states of the system, given noisy initial conditions. The goal is to find a way to map the experimental observations to the true values of the system. This is represented in (2) below,

$$\mathbf{y}(t) = \mathbf{H}\mathbf{x}(t) + \mathbf{q_3}, \tag{2}$$

where $\mathbf{y(t)}$ are the observations at time t of the state vector $\mathbf{x}$, $\mathbf{H}$ is the matrix which is used to map the state vector to the observations, and $\mathbf{q_3}$ is the observation error. In our simulation the observation error is normally distributed, zero mean, and of unit variance. For the EKF algorithm we assume the $\mathbf{H{=}I}$ where $\mathbf{I}$ is the identity matrix. This gives the update equation

$$\mathbf{x}_{k+1} = \mathbf{x}_{0_{k+1}} + \mathbf{K}_{k+1}\big(\mathbf{y}_{k+1} - \mathbf{x}_{0_{k+1}}\big), \tag{3}$$

where $\mathbf{K}_{k+1}$ is the Kalman gain matrix given by

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1}\big(\mathbf{P}_{k+1} + \mathbf{R}\big)^{-1} \tag{4}$$

and $\mathbf{R}$ is the noise covariance matrix. Here, $\mathbf{P}_{k+1}$ is a 3×3 matrix given by

$$\mathbf{P}_{k+1} = \mathbf{J}(\mathbf{f})\mathbf{P}_k\mathbf{J}(\mathbf{f})^T \tag{5}$$

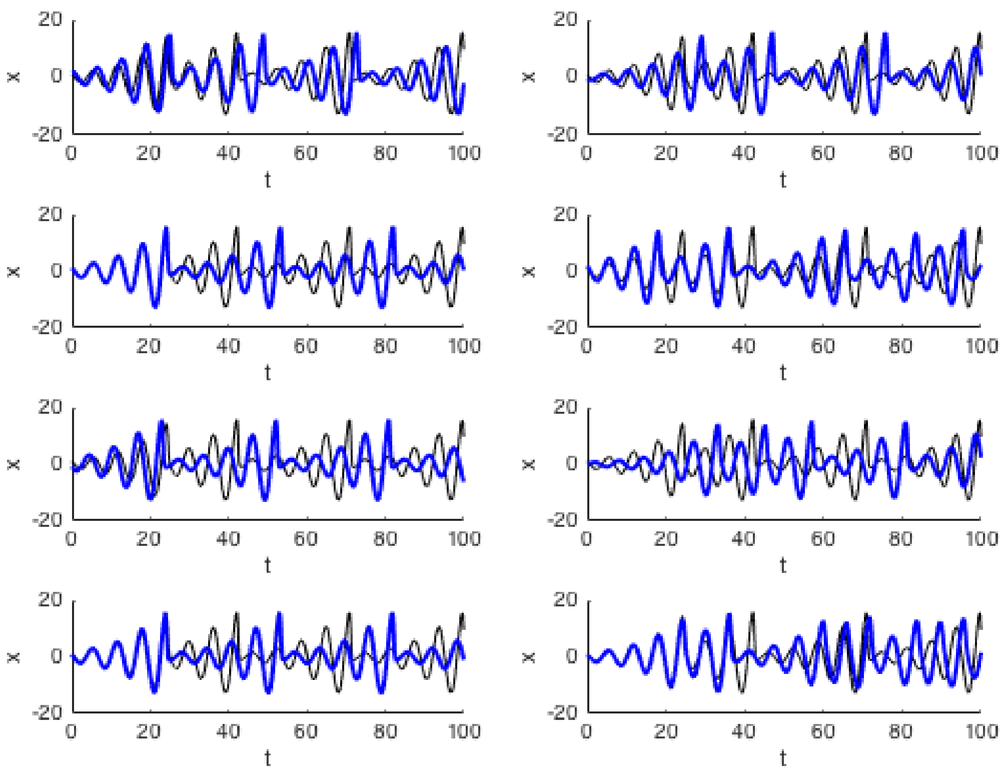


Figure 6: The figure displays the model dynamics for x(t) over eight trials (eight noise realizations). The black line represents the true model dynamics, while the blue line represents the model dynamics of the system with added noise. The noise involved is normally distributed, zero mean, and of unit variance. This is a small perturbation of noise, but is similar to the noise introduced manually in Part 2, which was shown to produce divergence of the solutions.

where $\mathbf{J}(\mathbf{f})$ is the Jacobian matrix of the Rossler equations. Using the update equation in (3), with the equations (4) and (5), it is possible to continuously update the estimate of the initial conditions in order to perform noise reduction. The results of applying this filter are compared to a test case where no filter is present. In Figure 7 the results of the experiment clearly display that the EKF has drastically reduced the model error, stopping the model from diverging as time progresses (for the chosen error variance). The code to produce these results is given in Listing 4.

Listing 4: Matlab code – Kalman Filter with Noisy Initial Conditions

```
tdata=t(1:50:end);
n=length(tdata);
xn=randn(n,1); yn=rand(n,1); zn=randn(n,1);
sigma3=1;
xdata=x_true(1:50:end)+sigma3*xn;
ydata=y_true(1:50:end)+sigma3*yn;
zdata=z_true(1:50:end)+sigma3*zn;

```

```matlab
x_da = zeros(size(t,2), 3);

for j=1:length(tdata)-1
    tspan=0:0.001:.05;
    [tspan,xsol]=ode45(f,tspan,xic);
    xic0=[xsol(end,1); xsol(end,2); xsol(end,3)];
    xdat=[xdata(j+1); ydata(j+1); zdata(j+1)];
    K=sigma2/(sigma2+sigma3); % compute K
    xic=xic0+(K*[xdat-xic0]); % new initial conditions
    x_da = [x_da; xsol(1:end-1,:)];% estimated time dynamics of the ↩
        solution
end

% Compute and plot error dynamics
error_noisy_ode = abs(x_true-x);
error_kalman_ode = abs(x_true-x_da(1:end-1,1));

figure
hold all
subplot(2,2,1)
hold on
grid on
grid minor
plot(t,x_true,'k')
plot(t,x,'b','Linewidth',[2])
xlabel('t')
title('Model Dynamics')

subplot(2,2,2)
hold on
grid on
grid minor
plot(error_noisy_ode)
axis([0 1e5 0 25])
xlabel('t')
ylabel('Abs(Error)')
title('Error in Model Dynamics')

subplot(2,2,3)
hold on
grid on
grid minor
plot(t,x_true,'k')
plot(t,x_da(1:end-1,1),'b','Linewidth',[2])
xlabel('t')
title('Model Dynamics')

```

```
subplot(2,2,4)
hold on
grid on
grid minor
plot(error_kalman_ode)
axis([0 1e5 0 25])
xlabel('t')
ylabel('Abs(Error)')
title('Error in Model Dynamics with Kalman Filter')
```

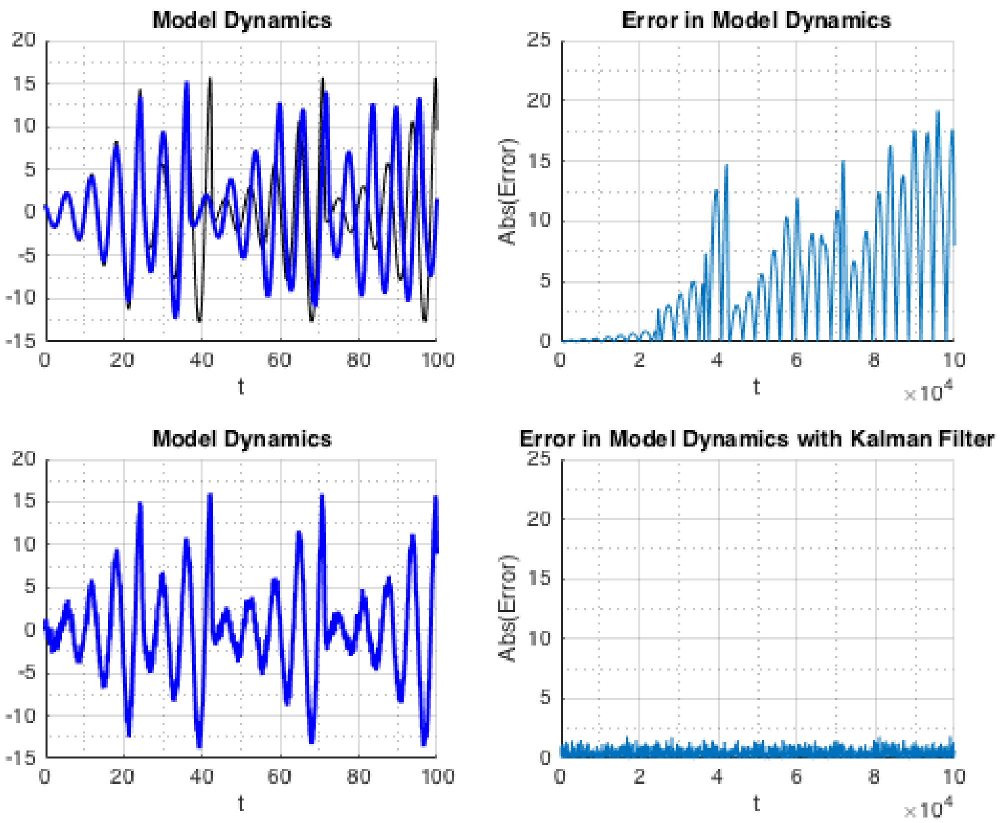


Figure 7: The figure displays the model dynamics for x(t) over one trial. The black line represents the true model dynamics, while the blue line represents the model dynamics of the system with added noise. The noise involved is again normally distributed, zero mean, and of unit variance. In the figure which shows the error in model dynamics with no EKF (top right), the error clearly diverges as time increases. In comparison, the error in the x(t) dynamics when the EKF is applied (bottom right) stays relatively the same throughout time (and is small in comparison). This is a display of the effectiveness of the EKF in this particular scenario, and its importance in systems which are sensitive to initial conditions.