Joe Sinotte

Final Project

Statement of the Problem:

Given a data set, in this case a binary and Markovian sequence, and a noisy observation of the data, is it possible to reconstruct the original data from the noisy observation?

Data:

As stated above, the data set used will be a binary Markovian sequence. Here, I will construct a variable $x = (x_1, \cdots, x_n)$ according to the following:

$$x_1 = \begin{cases} 0 & \text{with probability } 0.5 \\ 1 & \text{with probability } 0.5 \end{cases}$$

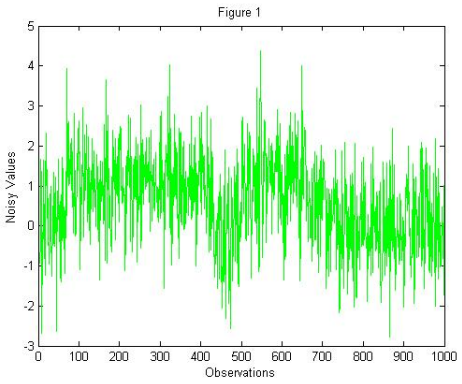
$$x_i = \begin{cases} x_{i-1} & \text{with probability } p \\ 1 - x_{i-1} & \text{with probability } 1 - p \end{cases}$$

Having generated the sequence of variables that will be used, the next step is to define a noisy observation of x, where the variable is observed with some error possible. This noisy variable will be defined as:

$$y_i = x_i + e_i$$

where,

$$e_i \sim N(0, \sigma^2)$$



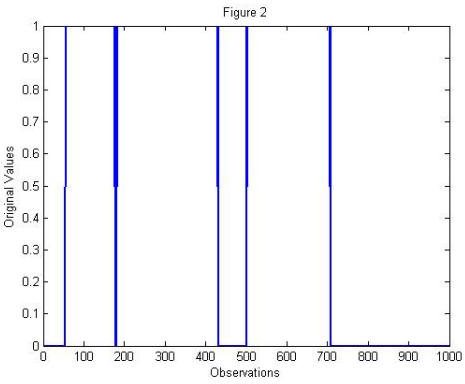


Figure 1 shows the values of the noisy observations and Figure 2 shows the original values of x. Both sets of variables were constructed according to the definitions above. Having constructed both sets of data, the goal of this exercise is now to determine whether or not the original data, x, can be reconstructed from the noisy data, y. In the sections to follow, I will describe the methodology used to approach this problem.

Dynamic Programming:

This project made use of an important computational tool called dynamic programming, which can be used to find optimal trajectories in situations where the decisions are made in stages, and decisions made in past stages have future consequences.

The particular method used to reconstruct the data is called the Maximum A Posteriori Method. This method involves taking a noisy observation and using a maximum likelihood technique to guess the most likely value for the reconstructed variable, given the observation. This process can be expressed mathematically as:

$$\{z_i\} = argmax\ Pr(\{x_i\}|\{y_i\}) = argmax\ log(Pr(\{x_i\}|\{y_i\}))$$

Which can be stated alternatively as:

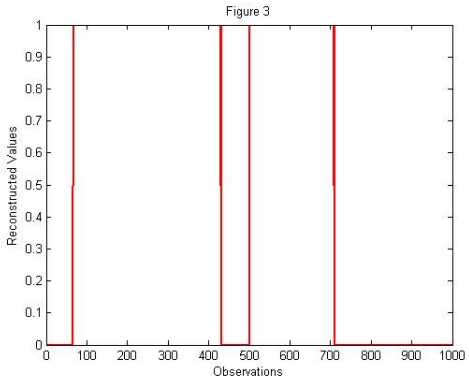
$$= argmax \sum_{i=1}^{N} \Big(- \frac{(y_i - x_i)^2}{2\sigma^2} \Big) + \sum_{i=2}^{N} log(1_{x_i = x_{i-1}} p + 1_{x_i \neq x_{i-1}} (1-p))$$

Hence, the values of z will denote the reconstructed variables which are compiled by maximizing the probability of the variable, x, conditioned on the observed noisy variable, y.

Results:

In the first portion of this exercise, the values for p and σ were set to 0.99 and 1, respectively. Then, using the Maximum A Posterior method described above, a dynamic program was implemented in Matlab to compute the reconstructed variable, z, from the noisy variable, y. This program was run 5 times and the rate of success for each iteration was computed as the proportion of reconstructed variables that matched the original variables, x. The average rate of success over these 5 iterations is given as:

$$average1 = 0.9798$$



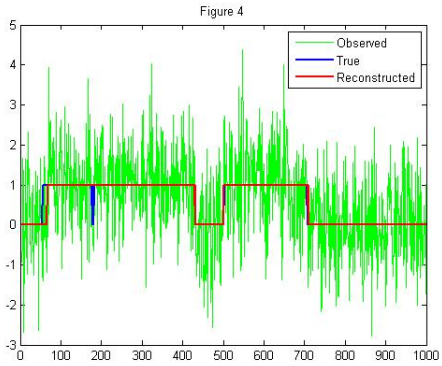
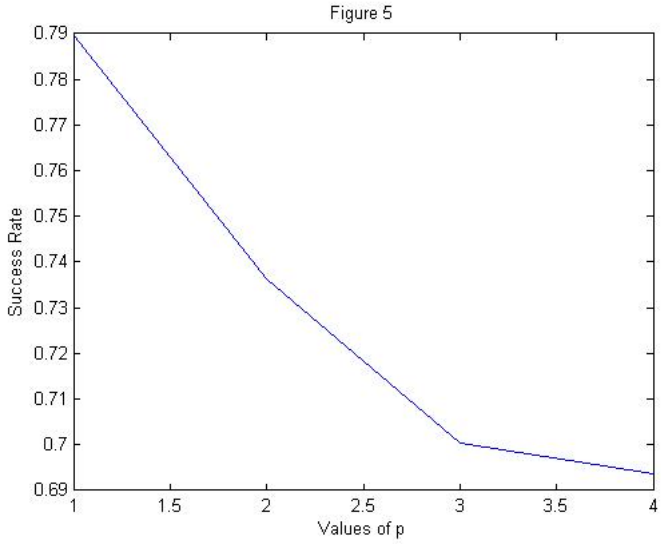


Figure 3 shows the reconstructed values, and Figure 4 shows the original values, x, the noisy values, y, and the reconstructed values, z, for the 5th iteration. Figure 4 as well as the average success rate reported above serve to demonstrate that this exercise was largely successful, but for few errors.

For the next portion of this project, I allow the values of p to vary while maintaining that $\sigma = 1$. Specifically, I will define p to be:

$$p = [0.9 \quad 0.8 \quad 0.7 \quad 0.6]$$

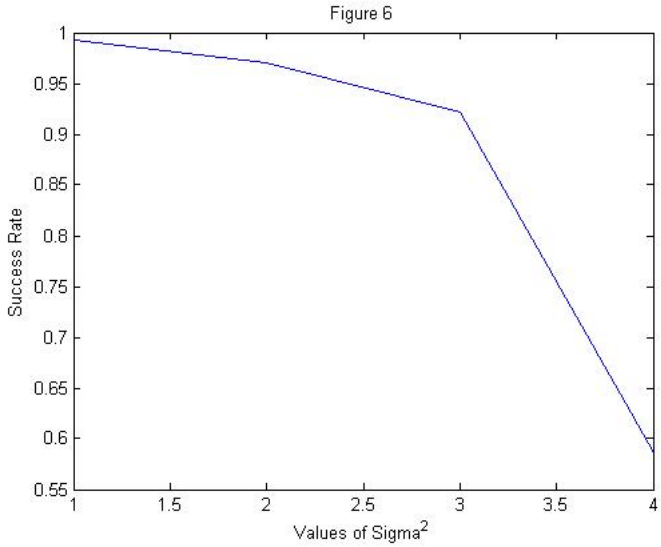
For each value of p, five repetitions were run and the average success rate was computed as before. Figure 5 shows these average success rates across the four possible values of p. As the figure shows, a smaller level of p corresponds to a lower probability of success. This is because as p gets smaller, it becomes more likely that x changes between observations, inducing more variation in the data and making it more difficult to successfully reconstruct values for z.



For the final portion of this project, I set $p = 0.99$ and allow the values of σ to vary. Specifically, I define σ as:

$$\sigma = [0.5 \quad 1 \quad 2 \quad 5]$$

For each value of σ, five repetitions were run and the average success rate was computed as before. Figure 6 shows these average success rates across the four possible values of σ. As the figure shows, a higher value for σ corresponds to a lower probability of success. This is because as σ gets smaller, the values of the noisy observations have greater variance making it more difficult to successfully reconstruct values for z.



Conclusion:

Overall, this exercise was successful. Using, the Maximum A Posterior Method, I have shown that reconstructed values can be found from noisy observations. In the process, I have also found that as the value of p decreases, it becomes more difficult to successfully reconstruct the data. Additionally, as the value of σ increases, it becomes more difficult to reconstruct the data.

Matlab Program:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% Final Project %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear
clc

N = 1000;
sigma = 1;
p = .99;

for i = 1:5

    %%%% Generate Initial Value of x
    x(1) = (rand < .5);

    %%%% Generate Remaining Values of x
    for j = 2:N
        if rand < p
            x(j) = x(j-1);
        else
            x(j) = 1-x(j-1);
        end
    end

    %%%% Generate Noisy Observations
    for k =1:N
        y(k) = x(k)+normrnd(0,sigma);
    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%% Pseudo Code %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    %%% Step 1

    s(1,1) = -(y(1)^2)/(2*sigma^2);
    s(1,2) =-((y(1)-1)^2)/(2*sigma^2);

    %%% Step 2

    for k = 2:N
        for j = 1:2
            h(1) = s(k-1,1) - .5/sigma^2*(y(k)-(j-1))^2 + log((j==1)*p
                                                          + (j==2)*(1-p));
            h(2) = s(k-1,2) - .5/sigma^2*(y(k)-(j-1))^2 + log((j==2)*p
```

```
                                                    + (j==1)*(1-p));

            if h(1) > h(2)
                b(k,j) = 0;
                s(k,j) = h(1);
            else
                b(k,j) = 1;
                s(k,j) = h(2);
            end
        end
    end

    %%% Step 3

    z(N) = (s(N,1) < s(N,2));
    for k=N-1:-1:1
        z(k) = b(k+1,z(k+1)+1);
    end

    for k = 1:1000
        if z(k) == x(k)
            S(k) = 1;
        else
            S(k) = 0;
        end
        s = sum(S);
    end

    success(i) = s/N;
    average1 = mean(success)
end
```

```
figure(1); clf;
plot(1:N, y, 'g'); hold on;
plot(1:N, x, 'b', 'linewidth', 2);
plot(1:N, z, 'r', 'linewidth', 2);
legend('Observed', 'True', 'Reconstructed');
```

```
N = 1000;
sigma = 1;
```

```
p = [.9 .8 .7 .6];

for m = 1:4

    for i = 1:5

        %%%% Generate Initial Value of x
        x(1) = (rand < .5);

        %%%% Generate Remaining Values of x
        for j = 2:N
            if rand < p(m)
                x(j) = x(j-1);
            else
                x(j) = 1-x(j-1);
            end
        end

        %%%% Generate Noisy Observations
        for k =1:N
            y(k) = x(k)+normrnd(0,sigma);
        end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %%%%%%%%%%%%%%%%%%%%%%%%%% Pseudo Code %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %%% Step 1

        s(1,1) = -(y(1)^2)/(2*sigma^2);
        s(1,2) =-((y(1)-1)^2)/(2*sigma^2);

        %%% Step 2

        for k = 2:N
            for j = 1:2
                h(1) = s(k-1,1) - .5/sigma^2*(y(k)-(j-1))^2
                                + log((j==1)*p(m) + (j==2)*(1-p(m)));
                h(2) = s(k-1,2) - .5/sigma^2*(y(k)-(j-1))^2
                                + log((j==2)*p(m) + (j==1)*(1-p(m)));

                if h(1) > h(2)
                    b(k,j) = 0;
                    s(k,j) = h(1);
                else
                    b(k,j) = 1;
                    s(k,j) = h(2);
                end
            end
```

```
        end

        %%% Step 3

        z(N) = (s(N,1) < s(N,2));
        for k=N-1:-1:1
            z(k) = b(k+1,z(k+1)+1);
        end

        for k = 1:1000
            if z(k) == x(k)
                S(k) = 1;
            else
                S(k) = 0;
            end
            s = sum(S);
        end

        success(i) = s/N;
    end
    average2(m) = mean(success)
end

figure(2)
plot(average2)



N = 1000;
sigma = [.5 1 2 5];
p = .99;

for m = 1:4

    for i = 1:5

        %%%% Generate Initial Value of x
        x(1) = (rand < .5);

        %%%% Generate Remaining Values of x
        for j = 2:N
            if rand < p
                x(j) = x(j-1);
            else
                x(j) = 1-x(j-1);
            end
        end

        %%%% Generate Noisy Observations
```

```
for k =1:N
    y(k) = x(k)+normrnd(0,sigma(m));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% Pseudo Code %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% Step 1

s(1,1) = -(y(1)^2)/(2*sigma(m)^2);
s(1,2) =-((y(1)-1)^2)/(2*sigma(m)^2);

%%% Step 2

for k = 2:N
    for j = 1:2
        h(1) = s(k-1,1) - .5/sigma(m)^2*(y(k)-(j-1))^2
                                    + log((j==1)*p + (j==2)*(1-p));
        h(2) = s(k-1,2) - .5/sigma(m)^2*(y(k)-(j-1))^2
                                    + log((j==2)*p + (j==1)*(1-p));

        if h(1) > h(2)
            b(k,j) = 0;
            s(k,j) = h(1);
        else
            b(k,j) = 1;
            s(k,j) = h(2);
        end
    end
end

%%% Step 3

z(N) = (s(N,1) < s(N,2));
for k=N-1:-1:1
    z(k) = b(k+1,z(k+1)+1);
end

for k = 1:1000
    if z(k) == x(k)
        S(k) = 1;
    else
        S(k) = 0;
    end
    s = sum(S);
end

success(i) = s/N;
```

```
    end
    average3(m) = mean(success);
end

figure(3)
plot(average3)
```