Homework 3

1. This problem is to look at noise models.

(a) Show that the error from the sum OR difference of two independent random measurements created from $y = 3a \pm 4b$, where a and b are both zero mean with variance σ_a^2 and σ_b^2 2 results in:

$$\sigma_y = \sqrt{9\sigma_a^2 + 16\sigma_b^2}$$

Perform 1000 run monte-carlo simulation to verify your results.

(b) Perform a 1000 run monte-carlo simulation (of 10 minutes) to look at the error growth of a random walk (integrated white noise). Use a white noise with 1-sigma value of 0.1 and 0.01 and compare the results. Plot the mean and standard deviation of the monte-carlo simulation along with one run of the simulation (show that the random walk is zero mean with a standard deviation is $\sigma_{\int w} = \sigma_w \Delta t \sqrt{k} = \sigma_w \sqrt{tx\Delta t}$ (where k is sample number)).

(c) Perform a 1000 monte-carlo simulation to look at the error growth of a 1st order markov process (integrated filtered noise) of the form $\dot{x} = -\frac{1}{\tau}x + w$. Use the same noise characteristics as above and compare the results with a 1 second and 100 second time constant (this results in 4 combinations). Comment on how changing the time constant and changing the standard deviation of the noise effects the error. Show that the 1st order markov process is zero mean with a standard deviation of $\sigma_x = \sigma_w \Delta t \sqrt{\frac{A^{2t} - 1}{A^2 - 1}}$ where $A = 1 - \frac{\Delta t}{\tau}$. Note that for a positive time constant (i.e. stable system) the standard deviation has a steady state value

Solution:

Solving for the variance of y:

$$\begin{aligned}
E\{(y - \bar{y})^2\} &= E\{((3a \pm 4b) - (3\bar{a} \pm 4\bar{b}))^2\} \\
&= E\{9a^2 - 18a\bar{a} + 24ab - 24a\bar{b} + 9\bar{a}^2 - 24\bar{a}b + 24 * \bar{a}\bar{b} + 16b^2 - 32b\bar{b} + 16\bar{b}^2\} \\
&= 9E\{a^2\} + 16E\{b^2\} \\
&= 9\sigma_a^2 + 16\sigma_b^2 \\
&= \sigma_y^2
\end{aligned}$$

Running a 1000 iteration monte carlo simulation where $a = b = 1$ and $\sigma_a = \sigma_b = 1$ results in the following statistics:

$$\begin{aligned}
\sigma_{y,theoretical} &= 5.0000 \\
\sigma_{y,sum} &= 5.1141 \\
\sigma_{y,diff} &= 4.9164
\end{aligned}$$

Performing a monte carlo on a random walk simulation of 10 minutes with sigma values of 0.01 and 0.1 results in the plots and MATLAB code below. As shown, the standard deviation values unboundedly increase with respect to time with magnitudes proportional to the sigma value of the noise. The mean of this simulation is also zero as expected due to white noise being zero mean.

Figure 1: Integrated White Noise.

```
time = 0:10*60;
dt = 1;
noise = [0.01, 0.1];
w = zeros(1000, length(time), 2);
sig = zeros(length(time), 2);
sig_mc = zeros(length(time), 2);
mu = zeros(length(time), 2);
mu_mc = zeros(length(time), 2);
% 2 standard deviations
for n = 1:2
    % 1000 MC runs
    for m = 1:1000
        % 10 minute run
        for t = 1:length(time)
            if t == 1
                w(m,t,n) = noise(n)*randn;
                sig(t,n) = noise(n) * dt * sqrt(t);
            else
                w(m,t,n) = w(m,t-1,n) + noise(n)*randn;
                sig(t,n) = noise(n) * dt * sqrt(t);
```

```
            end
        end
    end
end
```

Performing a similar monte carlo simulation on a first order markov process with varying time constants.

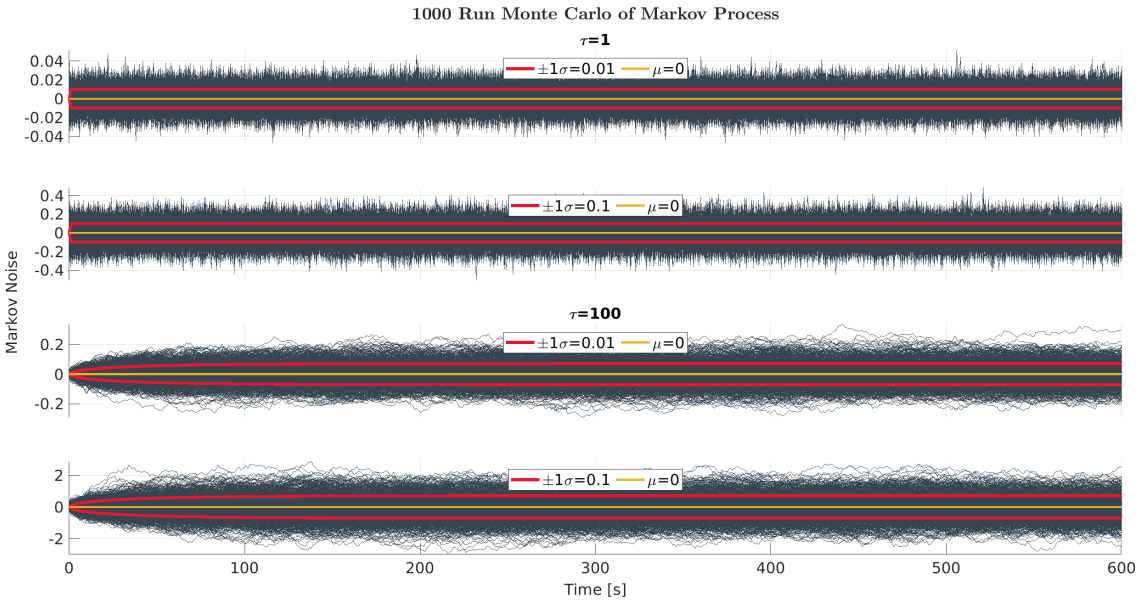

Figure 2: Markov Process Noise.

```
time = 0:10*60;
dt = 1;
noise = [0.01, 0.1];
time_const = [1, 100];
x = zeros(1000, length(time), 4);
sig = zeros(length(time), 4);
sig_mc = zeros(length(time), 4);
mu = zeros(length(time), 4);
mu_mc = zeros(length(time), 4);
% 2 standard deviations
for i = 1:2
    n = noise(i);
    % 2 time constants
    for s = 1:2
        tau = time_const(s);
        if i == 1
            ii = i + s - 1;
        else
```

```
            ii = i + s;
        end
        % 1000 MC runs
        for m = 1:1000
            % 10 minute run
            for t = 1:length(time)
                w = n*randn;
                if t == 1
                    x(m,t,ii) = w*dt;
                else
                    xDot = -x(m,t-1,ii)/tau + w;
                    x(m,t,ii) = x(m,t-1,ii) + xDot*dt;
                end
                A = 1 - (dt / tau);
                sig(t,ii) = n * dt * sqrt((A^(2*time(t)) - 1) / (A^2 - 1));
            end
        end
    end
end
```

As shown, while the larger time constant makes the standard deviation grow more gradually, the magnitude of the errors are significantly larger. The noise with the time constant of 1 almost instantly grows to its maximum threshold. Errors due to larger standard deviation values also prove to result in larger values. The markov process is also zero mean.

2. Determine the expected uncertainty for an L1-L5 ionosphere free pseudorange measurement and L2-L5 ionosphere free pseudorange. Assuming all measurements have the same accuracy (L1, L2, L5) which will provide the best ionosphere estimate?

Solution:

In order to create an ionosphere free psuedorange, the following formula can be used with respect the the different GPS carrier frequencies and psuedorange measurements.

$$\rho_{IF} = \frac{f_a^2}{f_a^2 - f_b^2}\rho_a - \frac{f_b^2}{f_a^2 - f_b^2}\rho_b$$

Solving for the L1-L5 measurement:

$$\begin{aligned}\rho_{IF} &= \frac{(1575.42(10^6))^2}{(1575.42(10^6))^2-(1176.45(10^6))^2}\rho_{L1} - \frac{(1176.45(10^6))^2}{(1575.42(10^6))^2-(1176.45(10^6))^2}\rho_{L5}\\ &= 2.2606\rho_{L1} - 1.2606\rho_{L5}\\ E\{(\rho_{IF}-\bar{\rho}_{IF})^2\} &= 5.11\rho_{L1}^2 - 5.69\rho_{L1}\rho_{L5} - 10.22\rho_{L1}\bar{\rho}_{L1} + 5.69\rho_{L1}\bar{\rho}_{L5} + 1.58\rho_{L5}^2 + 5.69\rho_{L5}\bar{\rho}_{L1} - 3.17\rho_{L5}\bar{\rho}_{L5} + 5.11\bar{\rho}_{L1}^2 - 5.69\bar{\rho}_{L1}\bar{\rho}_{L5} + 1.58\bar{\rho}_{L5}^2\\ &= E\{5.1103(\rho_{L1}^2-\bar{\rho}_{L1}^2) + 1.5891(\rho_{L5}^2-\bar{\rho}_{L5}^2)\}\\ &= 5.1103\sigma_{L1}^2 + 1.5891\sigma_{L5}^2\\ &= \sigma_{IF,L1L5}^2\end{aligned}$$

Similarly:

$$\begin{aligned}\sigma_{IF,L2L5}^2 &= 150.1928\sigma_{L2}^2 + 126.6822\sigma_{L5}^2\\ \sigma_{IF,L1L2}^2 &= 6.4807\sigma_{L1}^2 + 2.3893\sigma_{L2}^2\end{aligned}$$

Assuming a 1-sigma value of 1, the standard deviations are as follows:

$$\begin{aligned}\sigma_{IF,L1L5} &= 2.5883 \text{ m}\\ \sigma_{IF,L2L5} &= 17.3316 \text{ m}\\ \sigma_{IF,L1L2} &= 2.9782 \text{ m}\end{aligned}$$

This concludes that the L1-L5 combination would provide the best measurement.

3. Show that the differential GPS problem is linear. In other words derive the following expression:

$$\delta\rho = \begin{bmatrix} uv_x & uv_y & uv_z & 1 \end{bmatrix} \begin{bmatrix} r_x \\ r_y \\ r_z \\ c\delta t_{ab} \end{bmatrix}$$

Solution:

To solve for the delta psuedorange, each individual psuedorange is considered.

$$\begin{aligned}\rho_a^j &= R_a^j + c(t_a - t^j) + I^j + T^j + \nu_a^j\\ \rho_b^j &= R_b^j + c(t_b - t^j) + I^j + T^j + \nu_b^j\end{aligned}$$

Assuming each psuedorange has the same noise characteristics as well as equivalent atmospheric

errors, when differencing the psuedoranges:

$$\rho_a^j - \rho_b^j = R_a^j - R_b^j + c(t_a - t_b) + \sqrt{2}\nu^j$$

Taking the partial derivatives of this equation where u is the unit vector between the receiver and satellite, R is the range between the receiver and satellite, and r is the position of the receiver:

$$\Delta\rho_{ab}^j = \frac{u_a^j}{R_a^j} r_a^j - \frac{u_b^j}{R_b^j} r_b^j + c\Delta t_{ab} + \sqrt{2}\nu^j$$

From here, the small angle approximation can be used because the satellites are extremely far away compared to the distance between the receivers. This simplifies the equation by letting the direction and ranges from each receiver to the satellite be the same. Finishing the solution:

$$\begin{aligned}
\Delta\rho_{ab}^j &= \frac{u_a^j}{R^j} r_a^j - \frac{u_b^j}{R^j} r_b^j + c\Delta t_{ab} + \sqrt{2}\nu^j \\
&= \frac{u^j}{R^j}\Delta r_{ab} + c\Delta t_{ab} + \sqrt{2}\nu^j \\
&= uv_x^j \Delta r_x + uv_y^j \Delta r_y + uv_z^j \Delta r_z + c\Delta t_{ab} + \sqrt{2}\nu^j
\end{aligned}$$

Where uv is the unit vector of each axis between receiver a to a satellite. When generalized to all satellites, it is equivalent the the equation defined in the problem statement. Assuming position a is known, position b can be found by adding Δr_{ab} to the known position.

4. Set up your own 2D planar trilateration problem. Place the SVs at (0,300) (100,400), (700,400), and (800,300). Generate a range measurement for a base station at (400,0) and a user at (401,0).

(a) Solve for the position of the user using 2 SVs and then 4 SVs assuming no clock errors. How does the PDOP change for the two cases?

(b) Solve for the position of the user assuming you need to solve for the user clock bias. What is the PDOP with all 4 satellites.

(c) Calculate a differential solution between the base and user using a single difference model and assuming you must solve for a clock bias between the base station and user. What is the PDOP with all 4 satellites?

(d) Calculate a differential solution between the base and user using a double difference model to remove the clock bias between the base station and user. What is the PDOP with all 4 satellites?

(e) Assuming the range error is zero mean with unit variance, what is the order of accuracy in the above 4 solution methods?

Solution:

The following code was used to generate ranges and calculate the user position assuming there were no clock errors.

```
x_sv = [  0, 300; ...
        100, 400; ...
        700, 400; ...
        800, 300];
x_r = [400, 0];
x_u = [401, 0];

R_r = sqrt(sum((x_sv-x_r).^2, 2));
R_u = sqrt(sum((x_sv-x_u).^2, 2));

% PART A and B
x1 = [0;0]; x2 = [0;0];
e1 = 1; e2 = 1;
while (e1 > 1e-6)
  % 2 sv solution
  u1 = x_sv(1:2,:) - x1';
  R1 = sqrt(sum(u1.^2,2));
  uv1 = u1./R1;

  H1 = -uv1;
  y1 = R_u(1:2,:) - R1;

  dx1 = pinv(H1)*y1;
  x1 = x1 + dx1;
  e1 = norm(dx1);
end
while (e2 > 1e-6)
  % 4 sv solution
  u2 = x_sv - x2';
  R2 = sqrt(sum(u2.^2,2));
  uv2 = u2./R2;

  H2 = -uv2;
  y2 = R_u - R2;

  dx2 = pinv(H2)*y2;
  x2 = x2 + dx2;
  e2 = norm(dx2);
end
```

```
DOP1 = inv(H1'*H1);
DOP2 = inv(H2'*H2);
```

Looking at the dop values for both 2 and 4 satellites, the PDOP changes significantly when the number of satellites is doubled. This is due to the fact that there is large ambiguity in the y-axis when using only the first two satellites, which is fixed utilizing the rest. Still, both solutions provide answers that are extremely close to $x = \begin{bmatrix} 401 & 0 \end{bmatrix}$, being less than 10^{14} away from the actual answer.

$$PDOP_1 = 5.0577$$
$$PDOP_2 = 1.0000$$

Repeating the least squares solution assuming you also have to determine the clock bias first means that the 2 satellite solution will no longer work as there is three unknowns. The position solution remains almost exact for the 4 satellite solution where $PDOP = 5.0498$.

Using a single difference solution between the base station and user (as described in problem 3), the estimated user position is $x = \begin{bmatrix} 401 & -0.0014 \end{bmatrix}$ and $PDOP = 5.04975$. In this case, a differential solution seems to be worse than a standard solution because there is no source of error in the range measurements. Using the small angle approximation here makes the estimation slightly worse. The code below is for a single difference solution.

```
H3 = [(x_sv-x_r)./R_r, ones(4,1)];
y3 = R_r - R_u;
dx3 = pinv(H3)*y3;
x3 = [x_r,0]' + dx3;
DOP3 = inv(H3'*H3);
```

Solving using double difference techniques removes the need to calculate a clock bias, but increases the the number of measurements from satellites needed by one due to the differencing of all measurements to a reference. The estimated user position is $x = \begin{bmatrix} 401 & -0.0014 \end{bmatrix}$ and $PDOP = 4.2051$. The code below is for a double difference solution.

```
u = (x_sv-x_r)./R_r;
dR = R_r - R_u;
H4 = u(2:4,:) - u(1,:);
y4 = dR(2:4) - dR(1);
dx4 = pinv(H4)*y4;
x4 = x_r' + dx4;
DOP4 = inv(H4'*H4);
```

Based on the estimations and PDOP estimations, the following is a list of best to worst solution accuracies for this simulated scenario:

1. Perfect Clock (A)
2. Double Differenced (D)
3. Imperfect Clock (B)
4. Single Differenced (C)

Looking at the GDOP (which would be equivalent to a covariance with a unit variance) for part B and C does not change this answer as they still have the highest values.

5. (Bonus for Undergrads/Required for Grads). Repeat problem #4 using 4 and 8 SV positions from Lab #2 (4,7,8,9,16,21,27,30). Comment on any difference or similarities with the planar problem in #3.

Solution:

Finding a spot where both receivers in the dynamic data sets had 8 satellites was difficult, therefore the first timestep this occurred was used. This resulted in satellites [3, 6, 11, 14, 17, 19, 24, 30] being used. Running through the same steps as problem 3, the position outputs on a geoplot are shown below as well as the PDOP values of each part for both 4 and 8 satellites.

(a) $PDOP_{4sv} = 2.7265$
$PDOP_{8sv} = 1.0837$

(b) $PDOP_{4sv} = 3.5137$
$PDOP_{8sv} = 1.7370$

(c) $PDOP_{4sv} = 3.5137$
$PDOP_{8sv} = 1.7370$

(d) $PDOP_{4sv} = 3.3109$
$PDOP_{8sv} = 1.4743$

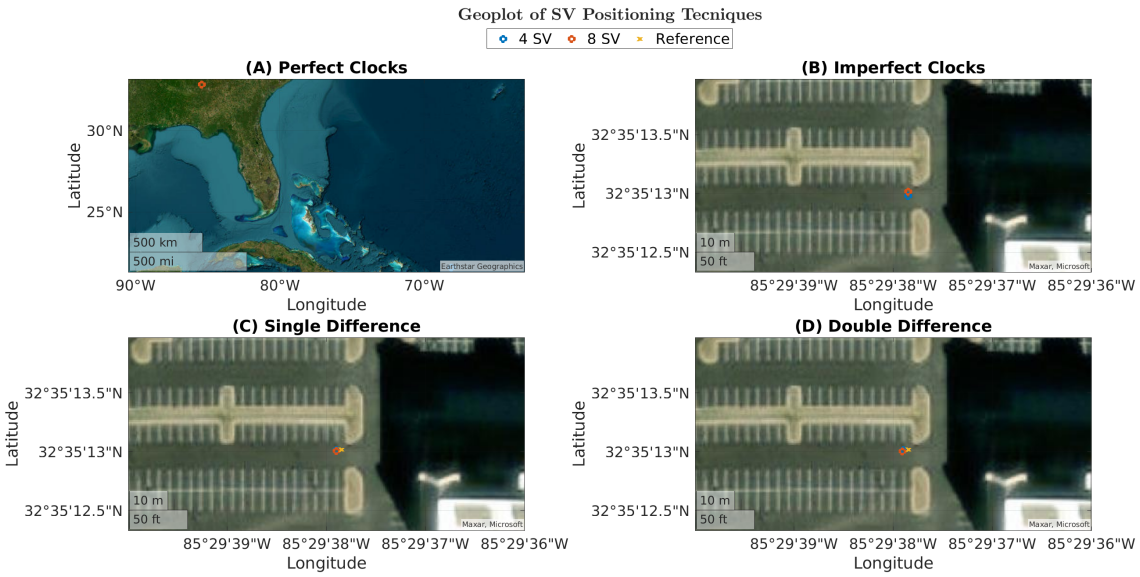

Figure 3: Position Estimations with Different Techniques.

In *part a*, is is obvious that the 4 satellite estimate provides an extremely poor estimate, likely due to a combination of poor satellite geometry and not accounting for clock bias. The 8 satellite geometry provides a more reasonable solution that is tenths or degrees off in latitude and longitude, but provides a height estimate deep inside the earth (-102037m, for reference the 4 SV solution places the receiver at -500793m). Moving to *part b*, accounting for clock bias provides really good positioning. While the PDOP is higher, the position solution is more correct. *Part c* and *part d* also provide good position solutions having the same level of accuracy as *part b* (all are in the same area of the MRI building parking lot). The only difference being that the PDOP value for the double differenced solution as smaller. Ranking the solutions by order of accuracy:

1. Double Differenced (D)
2. Single Differenced (C)
3. Imperfect Clock (B)
4. Perfect Clock (A)

While the perfect clock provided the best PDOP values, it is obvious that this values is independent of the actual position solution giving it no bearing on the accuracy.

6. Chapter 2, Problem 1a and 1b for PRN#4. Repeat 1a for PRN #7.

Solution:

The following is a function made to generate the C/A code for a GPS satellite:

```
function G2i = caGen(s1, s2)
```

```
% initialize to all ones
G1 = ones(1,10);
G2 = ones(1,10);
G2i = zeros(1,1023);
for i = 1:1023
    % create ca code
    G2i(i) = bitSum(G2(s1), G2(s2));
    G2i(i) = bitSum(G2i(i), G1(10));
    % update G1
    newBit = bitSum(G1(3), G1(10));
    G1 = [newBit, G1(1:9)];
    % update G2
    newBit = 0;
    for j = [2,3,6,8,9,10]
        newBit = bitSum(newBit, G2(j));
    end
    G2 = [newBit, G2(1:9)];
end
% XOR function
function s = bitSum(a,b)
    s = a+b;
    if s > 1
        s = 0;
    end
end
end
```

For PRN#4 the values of $s1$ and $s2$ are 5 and 9, and PRN#7 they are 1 and 8 (provided in the GPS specification document `https://www.gps.gov/technical/icwg/IS-GPS-200N.pdf`). Plotting the first and last 16 chips of PRN#4 and PRN#7:

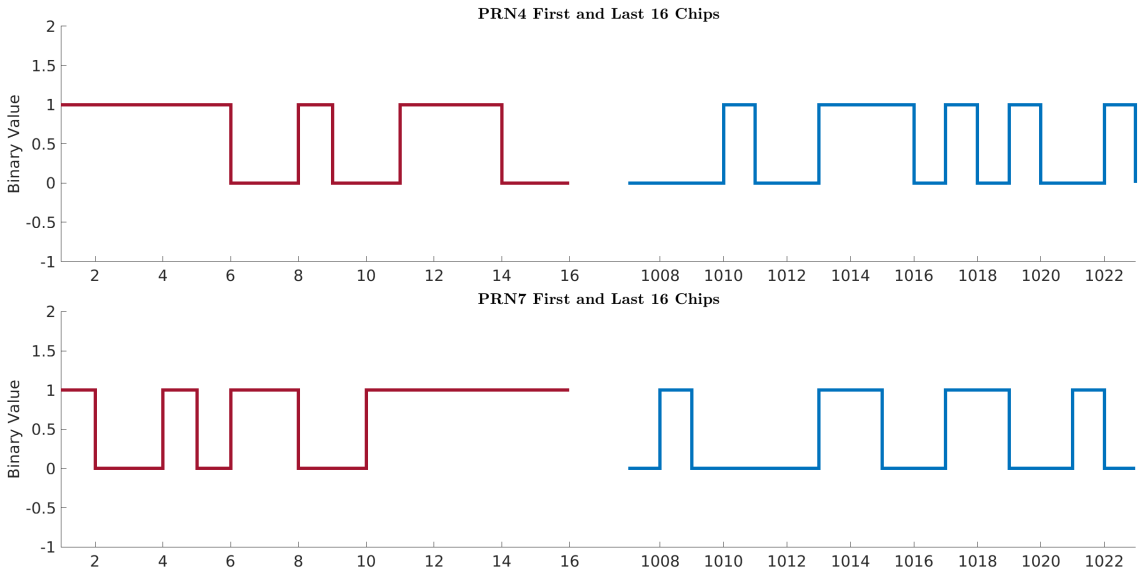

Figure 4: Beginning and Ending Chips of PRNs.

Since the PRNs repeat every 1023 chips, the C/A code should repeat in chips 1023-2046. Verifying this by duplicating the code above and running the generator twice reveals that the first 16 bits are exactly the same in both sequences. Plotting the autocorrelation between chips 1-1023 and chips 1024-2046 of PRN#4 which confirms that the sequence repeats every 1023 chips:

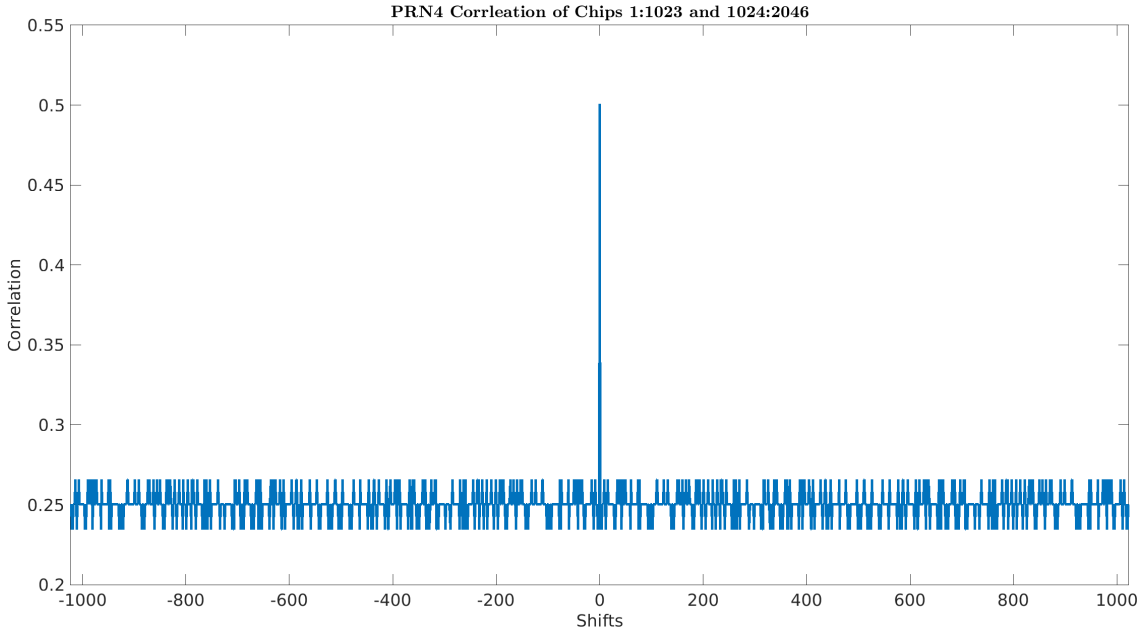

Figure 5: Autocorrelation of PRN#4 Chips 1-1023 and 1024-2046.

7. Using your PRN sequence for PRN 4 and 7, repeat problem #2 from HW#1. Compare the results to the results for your made up sequence.

(a) Plot the histogram on each sequence

(b) Plot the spectral analysis on each sequence

(c) Plot the autocorrelation each sequence with itself (i.e. a sequence delay crosscorrelation)

(d) Plot the cross autocorrelation between the two sequences

Solution:

Below are the plots for each part and each PRN.

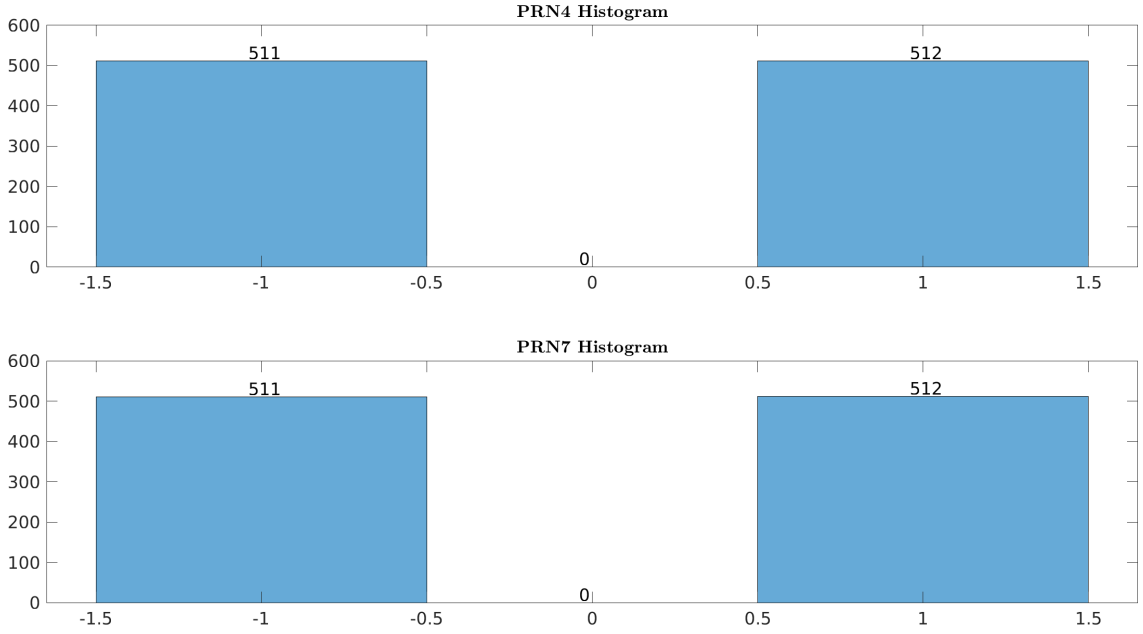

Figure 6: Histograms of PRN#4 and PRN#7.

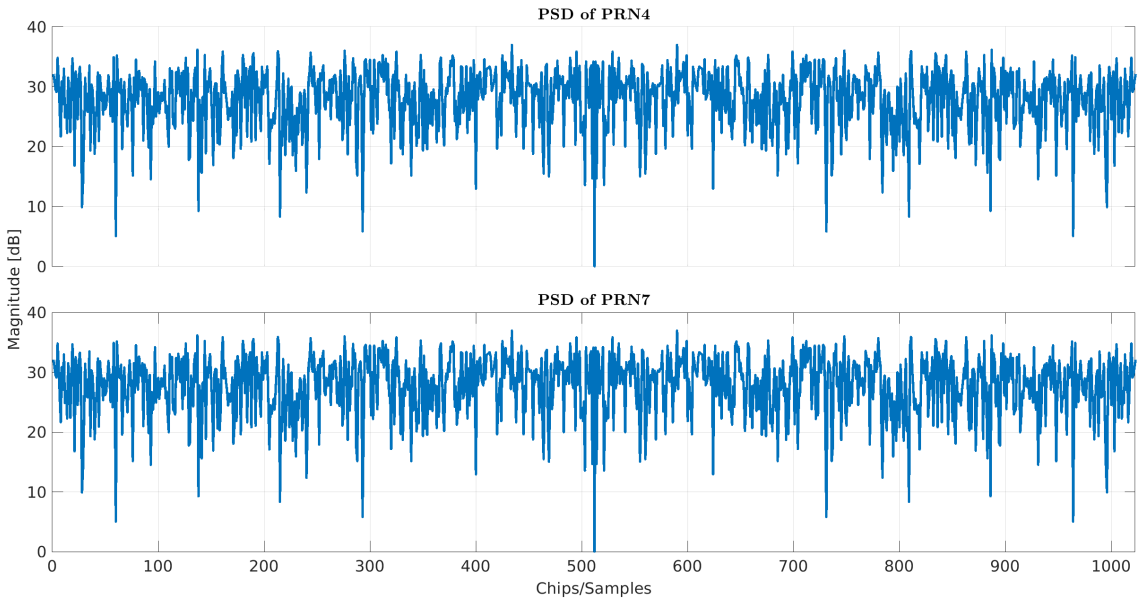

Figure 7: PSDs of PRN#4 and PRN#7.

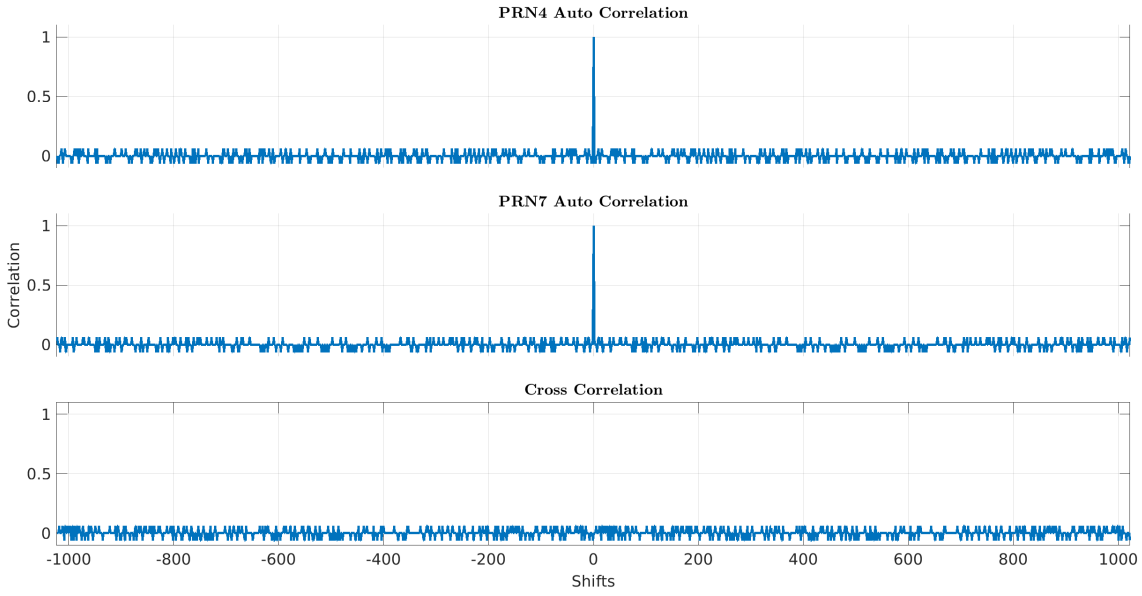

Figure 8: Correlations of PRN#4 and PRN#7.

Looking at the histograms, it is obvious that the assortment of 0 (-1) and 1 is super even. The PSD displays that the noise is random on the C/A code with a mean value just under 30. And the correlations prove that each PRN is only correlated with itself at one timepoint and completely uncorrelated with other PRNs.

Comparing these results to those from HW#1, each of the plots look almost identical (except for the randomness in each one from HW#1). This proves that GPS PRNs contain the properties of

psuedorandom sequences.

8. Take your C/A code from problem above (i.e. PRN #4 and #7) and multiply it times the L1 Carrier (your C/A code must be in the form -1 and +1). Perform a spectral analysis (magnitude) on the resultant signal. You will need to make sure to "hold" your C/A code bits for the correct length of time (I suggest using a sample rate 10x the L1 carrier frequency - meaning each chip of the C/A code will be used for 10 samples of the sine wave).

Solution:

The following is a PSD of the modulation of the C/A code with the L1 carrier signal. As expected peaks at the L1 carrier frequency for both PRNs.

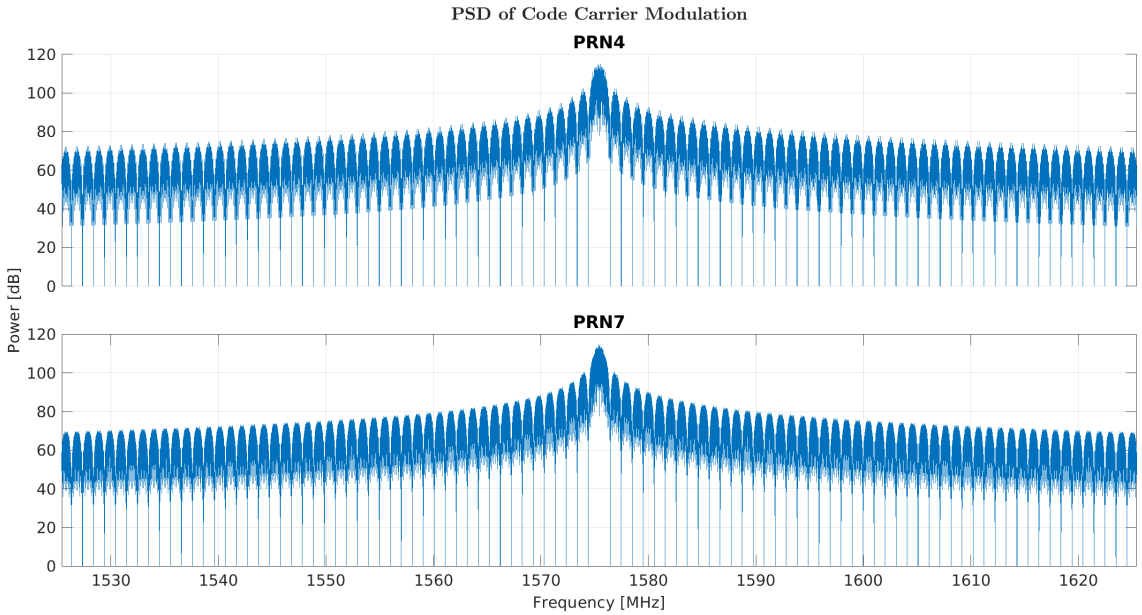

Figure 9: Code - Carrier Modulation