# A Kalman Filter for Bluetooth RSSI Tracking

Robert S. Huston

July 23, 2020

**Abstract**

This article investigates the application of a Kalman filter for estimating Bluetooth RSSI signal levels acquired via a mobile device. Three process models are explored: Gauss-Markov, Gauss-Markov with random bias, and integrated Gauss-Markov. Experimental data captured using an iPhone is used to demonstrate and compare the performance of each model.

# Contents

# 1 Introduction

Two fundamental architectures are representative of the reception of digital signals: direct-conversion and low-IF [5] [6]. The basic direct-conversion architecture is shown in Figure 1. The direct-conversion architecture supplies the baseband I and Q digital samples directly to the DSP.

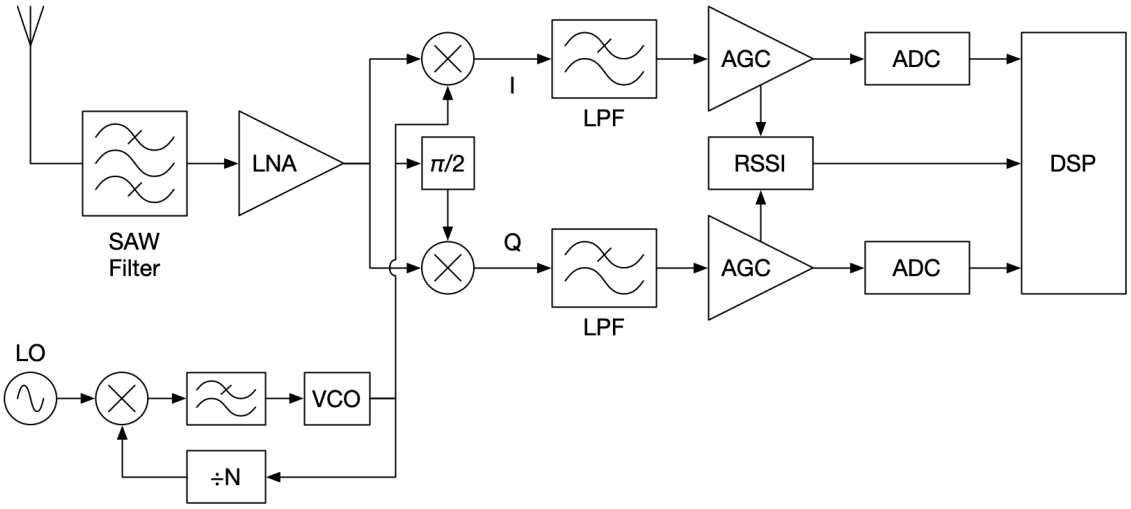



**Figure 1: Digital Receiver Architecture - Direct Conversion**

The basic low-IF architecture is shown in Figure 2. The low-IF architecture supplies the IF digital samples to the DSP.

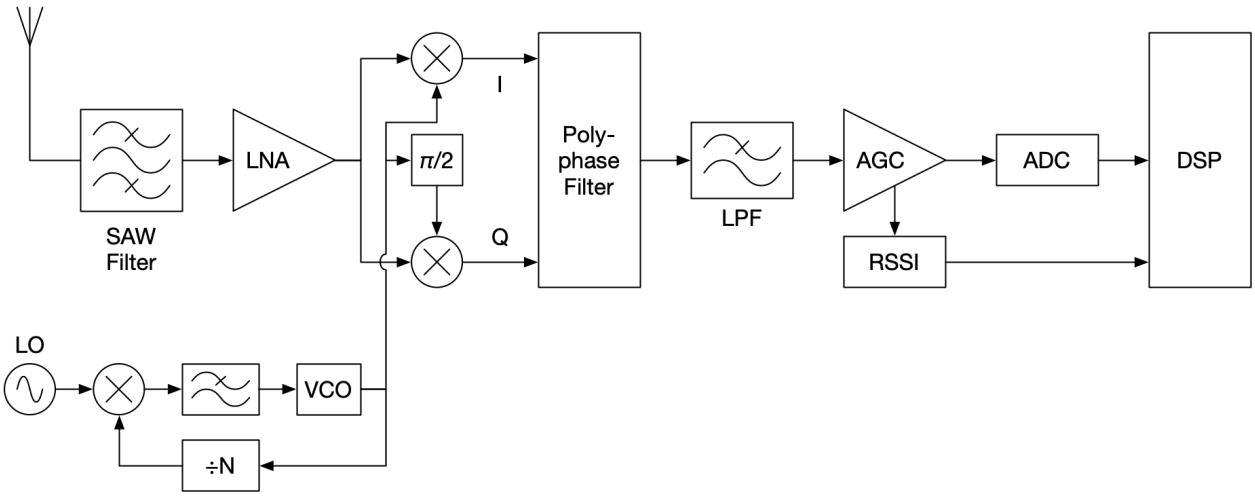



**Figure 2: Digital Receiver Architecture - Low IF**

While there are various engineering reasons for choosing one architecture over the other, the main point to realize is that the RSSI measurement is typically related to the AGC levels of the receiver [7]. Because the AGC is constantly adjusting its gain in order to track the incoming signal, the RSSI measurement as relayed by the signal processor will fluctuate.

In addition, the quality of the RSSI measurement is highly dependent on both the AGC operation as well as the processing strategies and algorithms implemented in the DSP. Of course, the position of the radio antenna itself also has an effect on measurement quality due to shadowing and polarization, and also due to multipath signals imposed by the physical layout of an indoor environment.

Many Bluetooth receiver architectures have been implemented in the mobile device industry. The reality is that Bluetooth RSSI measurements as acquired by a mobile device can be highly erratic and can be quite unreliable in their raw form. The application of a filter to refine the raw measurements is certainly warranted. In this paper, we explore the use of three Kalman filter models for refining the RSSI measurement of a mobile device into a more meaningful value.

We first explore the use a Gauss-Markov process to model the RSSI acquisition behavior of a Bluetooth device signal as seen by the mobile device. We choose this model because its autocorrelation function seems to match what we have observed from the signal behavior of experimental RSSI data captured over time, and because the resulting Kalman filter is scalar. We then present two experiments that demonstrate the performance of the Gauss-Markov Kalman filter.

Next, we explore augmenting the Gauss-Markov state with a random bias. It is hoped that this model better represents the behavior of a random RSSI level that is coupled with a Gauss-Markov process. Whereas the Gauss-Markov model is a scalar process, the Gauss-Markov with random bias model is a $2 \times 2$ matrix process.

Lastly, we explore the use of an integrated Gauss-Markov model. This model is a popular model used in many engineering applications because it characterizes a low-pass filtered Gauss-Markov process. The integrated Gauss-Markov model is also a $2 \times 2$ matrix process.

# 2 Gauss-Markov Process

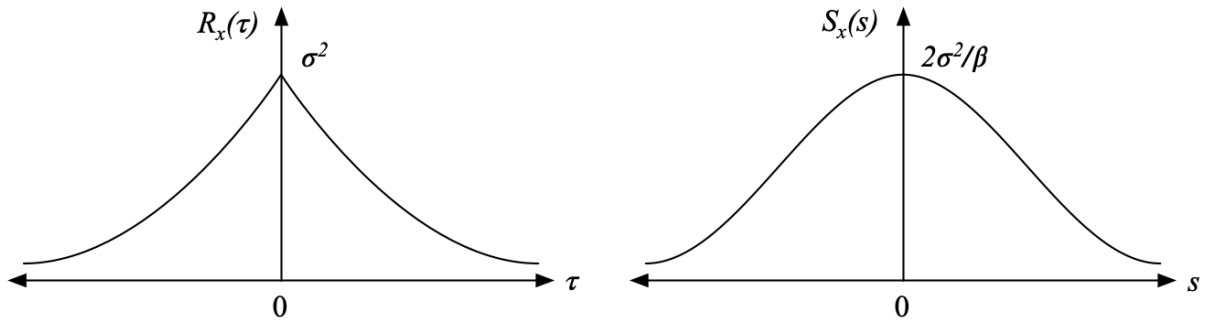
A stationary Gaussian process that has an exponential autocorrelation function is called a *Gauss-Markov* process [1]. If we designate $x(t)$ as a Gauss-Markov process, its autocorrelation function is of the form

$$R_x(\tau) = \sigma^2 e^{-\beta|\tau|} \tag{1}$$

where $\sigma^2$ is the process mean-square value, and $1/\beta$ is the process time constant. The Gauss-Markov process has a spectral density function of the form

$$S_x(s) = \frac{2\sigma^2\beta}{-s^2 + \beta^2} \tag{2}$$

Figure 3 illustrates the Gauss-Markov autocorrelation and spectral density functions.




**(a) Autocorrelation** **(b) Spectral Density**

**Figure 3: Gauss-Markov Autocorrelation and Spectral Density Functions**

The Gauss-Markov process is a popular model because it seems to fit a large number of physical processes, and because it is has a simple mathematical description. It basically describes a process where its values are more correlated when the time spacing between successive measurements is small, and less correlated when the time spacing between successive measurements is large. It is a suitable model when nothing is known about the underlying mechanics of the system state process and/or measurement acquisition process.

From linear system theory, we know that if we can factorize a spectral density function into LHP and RHP forms of the same factor function $G(\cdot)$:

$$S_x(s) = G(s)G(-s) \tag{3}$$

then $G(s)$ is the shaping filter that shapes unity white noise into the spectral density $S_x(s)$. Thus, for the Gauss-Markov spectral density, we have

$$S_x(s) = \left(\frac{\sqrt{2\sigma^2\beta}}{s+\beta}\right)\left(\frac{\sqrt{2\sigma^2\beta}}{-s+\beta}\right) \tag{4}$$

and so the shaping filter is

$$G(s) = \frac{\sqrt{2\sigma^2\beta}}{s+\beta} \tag{5}$$

and the corresponding differential equation is

$$\dot{x} = -\beta x + \sqrt{2\sigma^2\beta}\, w(t) \tag{6}$$

where $w(t)$ is a unity white noise process. Figure 4 shows the block diagram of the Gauss-Markov differential system.

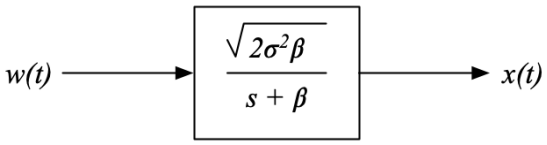



**Figure 4: Gauss-Markov Process Block Diagram**

The complementary solution is

$$x_c(t) = e^{-\beta t} \tag{7}$$

Hence, the discrete-time state transition function is

$$\phi_k = e^{-\beta\tau_k} \tag{8}$$

and the discrete-time process variance is

$$\begin{aligned} Q_k &= E[w_k^2] \\ &= \int_0^{\tau_k} \left(\sqrt{2\sigma^2\beta}\, e^{-\beta v}\right)^2 dv \\ &= \sigma^2\left(1 - e^{-2\beta\tau_k}\right) \end{aligned} \tag{9}$$

where $\tau_k = t_k - t_{k-1}$ represents the difference between two sample points in time denoted by $k$ and $k-1$.

The scalar discrete-time representation of the Gauss-Markov system is then

$$x_k = \phi_k x_{k-1} + w_k \tag{10}$$

$$z_k = x_k + v_k \tag{11}$$

where $z_k$ is the process measurement at the $k$th time point, $w_k$ is a white noise sequence representing process noise with known variance

$$E[w_k^2] = Q_k \tag{12}$$

and where $v_k$ is a white noise sequence representing measurement error with known variance

$$E[v_k^2] = R_k \tag{13}$$

# 3 Gauss-Markov Kalman Filter

The Kalman filter implementation of a Gauss-Markov process model can be implemented very efficiently since it involves only scalar arithmetic.

At the $k$th time point, we obtain a measurement $z_k$ at time $t_k$. We model our measurement error with a variance $R_k$. At each time point, the filter provides a best estimate, $\hat{x}_k$ and maintains a state estimate variance, $P_k$. We initialize the filter with $\hat{x}_0 = z_0$ and $P_0 = P0$, where $P0$ is a suitably chosen value based on empirical analysis.

For each acquisition event, $k$, we perform the following steps

1. Compute $\phi_k$ using (8)

2. Compute $Q_k$ using (9)

3. Compute state estimate prediction:

$$\hat{x}_k^- = \phi_k \, \hat{x}_{k-1} \tag{14}$$

4. Compute state estimate variance prediction:

$$P_k^- = \phi_k \, P_{k-1} \, \phi_k + Q_k \tag{15}$$

5. Compute Kalman gain:

$$K_k = \frac{P_k^-}{P_k^- + R_k} \tag{16}$$

6. Compute state estimate correction:

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \, (z_k - \hat{x}_k^-) \tag{17}$$

7. Compute state estimate variance correction:

$$P_k^+ = (1 - K_k) \, P_k^- \tag{18}$$

# 4 Experiments

A simple iOS application was developed that enables an iPhone to continuously scan for nearby Bluetooth devices, reading among other properties, their RSSI, UUID, and device name properties. Each reading gets timestamped and logged to a file that an be copied to a computer for further processing. A simple Python script was written to selectively filter and plot device data based on device UUID. The filter was a scalar Kalman filter using a Gauss-Markov process as the model.

Two experiments were performed. The first experiment involved walking the scanning iPhone (an iPhone Xs Max) away throughout the house (hallway and then dining room) from the development MacBook Pro computer (kitchen) and then moved back again. The second experiment involved taking both the scanning iPhone and a second iPhone outside and then walking the scanning iPhone from the corner of the back yard to the second iPhone and then away again. Figure 5 depicts a simple diagram of the walking routes taken (shown in red) for each experiment. The distance values are approximate.

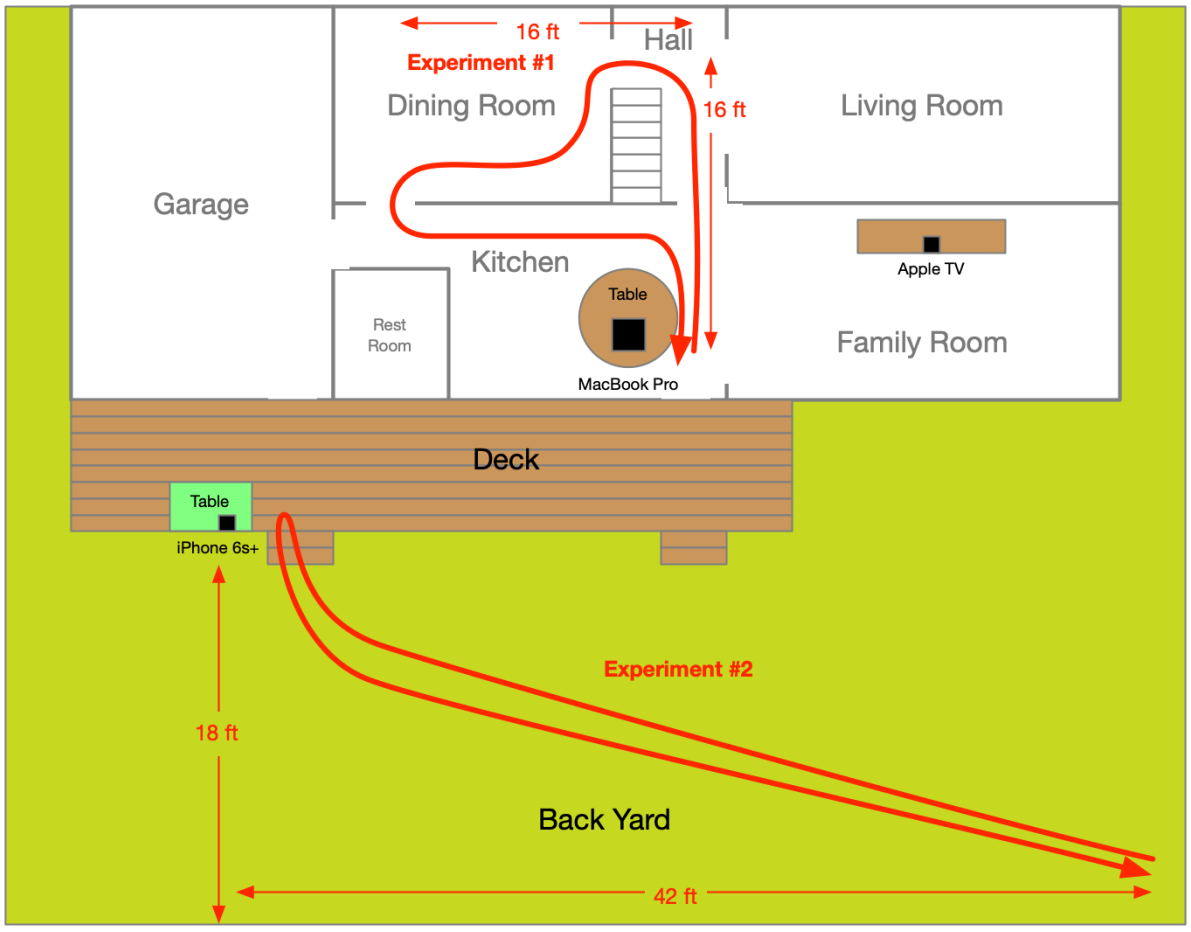



**Figure 5: Diagram of Experiment Walk Routes (Shown in Red)**

One particular note about the second experiment was that the second iPhone screen was active when the device was placed on a table on the deck before initiating the experiment, but, by the time the walk from the corner of the yard had reached the second iPhone, it was observed that the screen of the second iPhone had just gone to sleep. This can be seen in the rate of the acquired data, where the "awake" condition produced more Bluetooth scan responses than the "sleep" condition. This was a fortunate event, because it provided a perfect data set to illustrate why a Gauss-Markov process model is suitable.

The Gauss-Markov Kalman filter was initialized with the following parameters:

$$P_0 = 5 \ , \quad \sigma = 10 \ , \quad \beta = 0.01 \ , \quad R = 25$$

For the first experiment, the UUID of the development MacBook Pro (Red Green) was monitored. The scanning iPhone was walked away from and then back to the computer. The results can be seen in Figure 6. The RSSI measurements are blue, and the filtered RSSI values are in red.

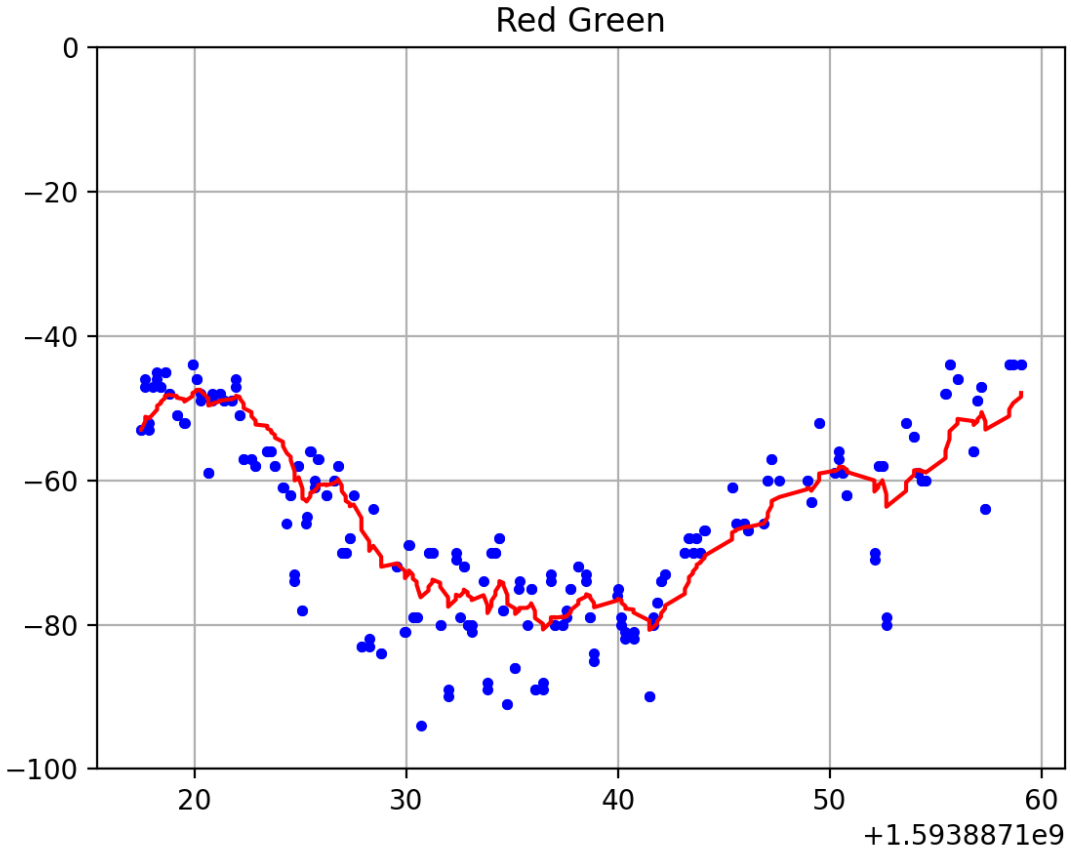



**Figure 6: Experiment #1 - Scanning MacBook Pro from Indoors**

For the second experiment, both the scanning iPhone and the second stationary iPhone (an iPhone 6s+) were outside in the back yard. The scanning iPhone was walked from the

corner of the yard to the stationary iPhone and back again. As was stated previously, the stationary iPhone screen went to sleep when the scanning iPhone reached the stationary iPhone location. The scanning iPhone was then walked away back to the corner of the yard. The results for the second iPhone can be seen in Figure 7.

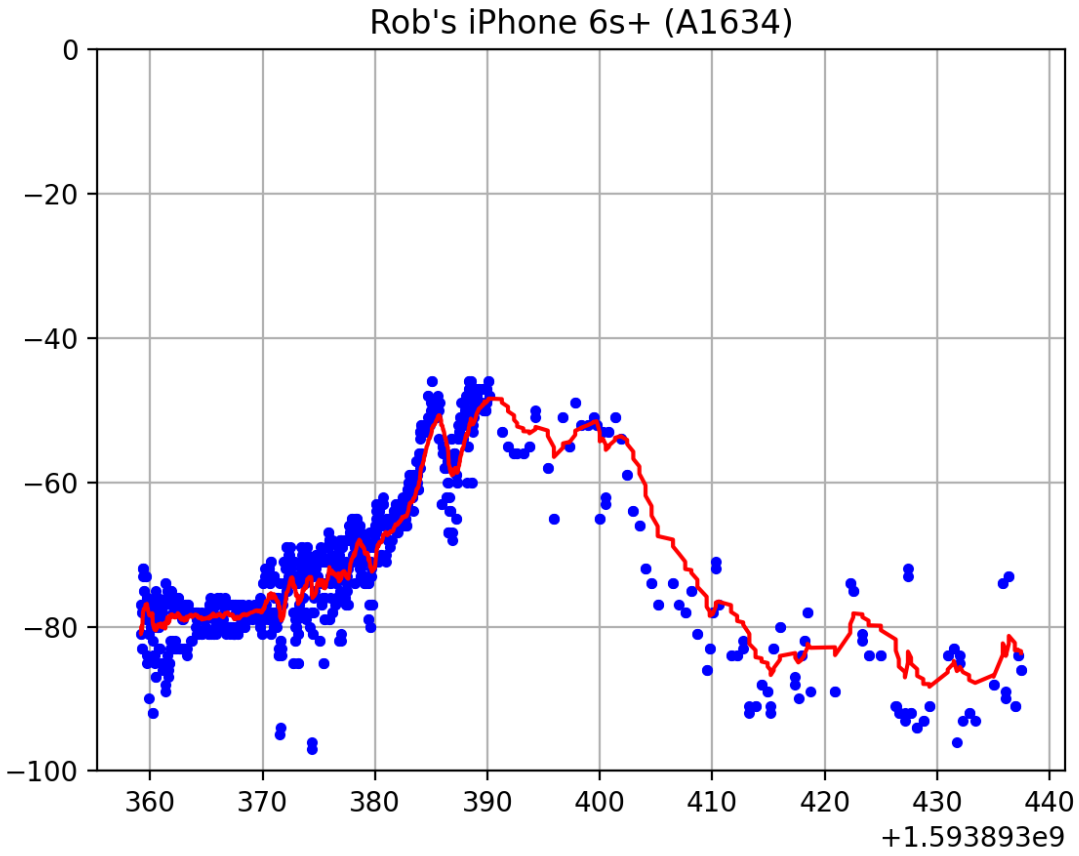



**Figure 7: Experiment #2 - Scanning Second iPhone from Back Yard**

Also available from the second experiment was the scanned Bluetooth signal data from both the development MacBook Pro computer in the kitchen, and an Apple TV in the family room. As shown in Figure 5, both devices are in rooms that face the back yard, and their distances from the scanning iPhone are comparable. The results for the MacBook Pro can be seen in Figure 8, and the results for the Apple TV can be seen in Figure 9.

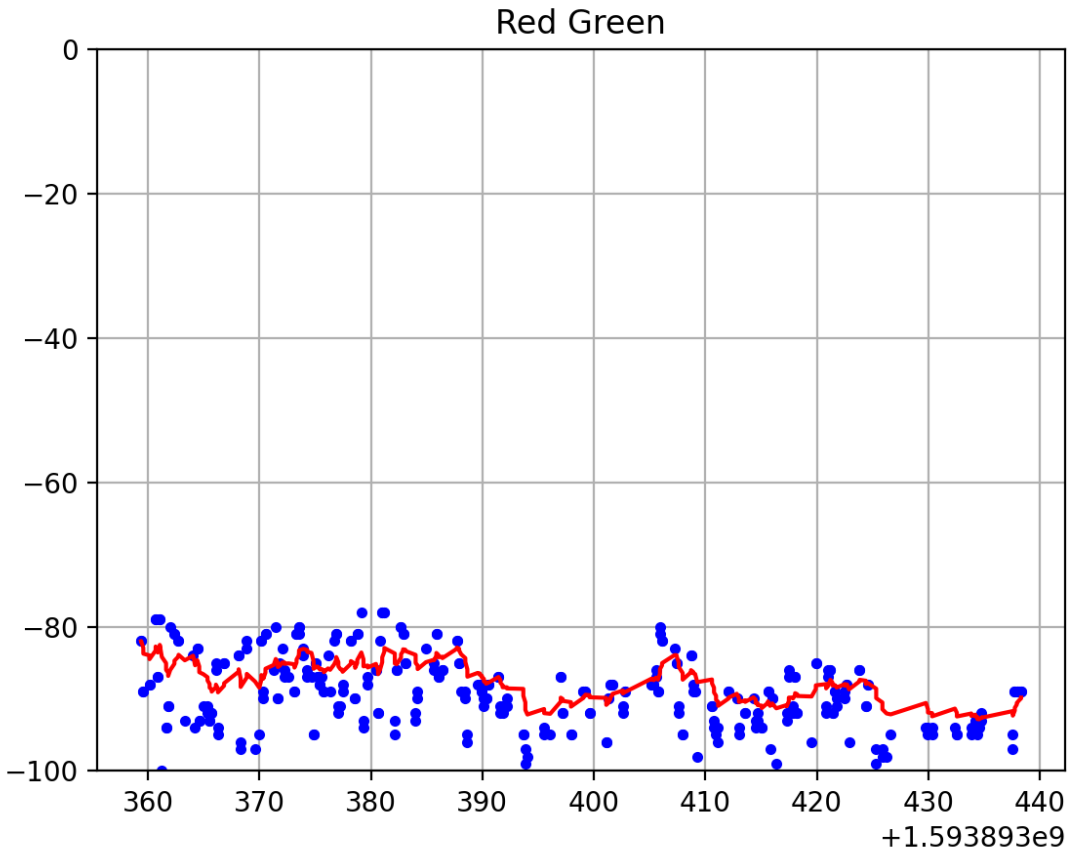



**Figure 8: Experiment #2 - Scanning MacBook Pro from Back Yard**

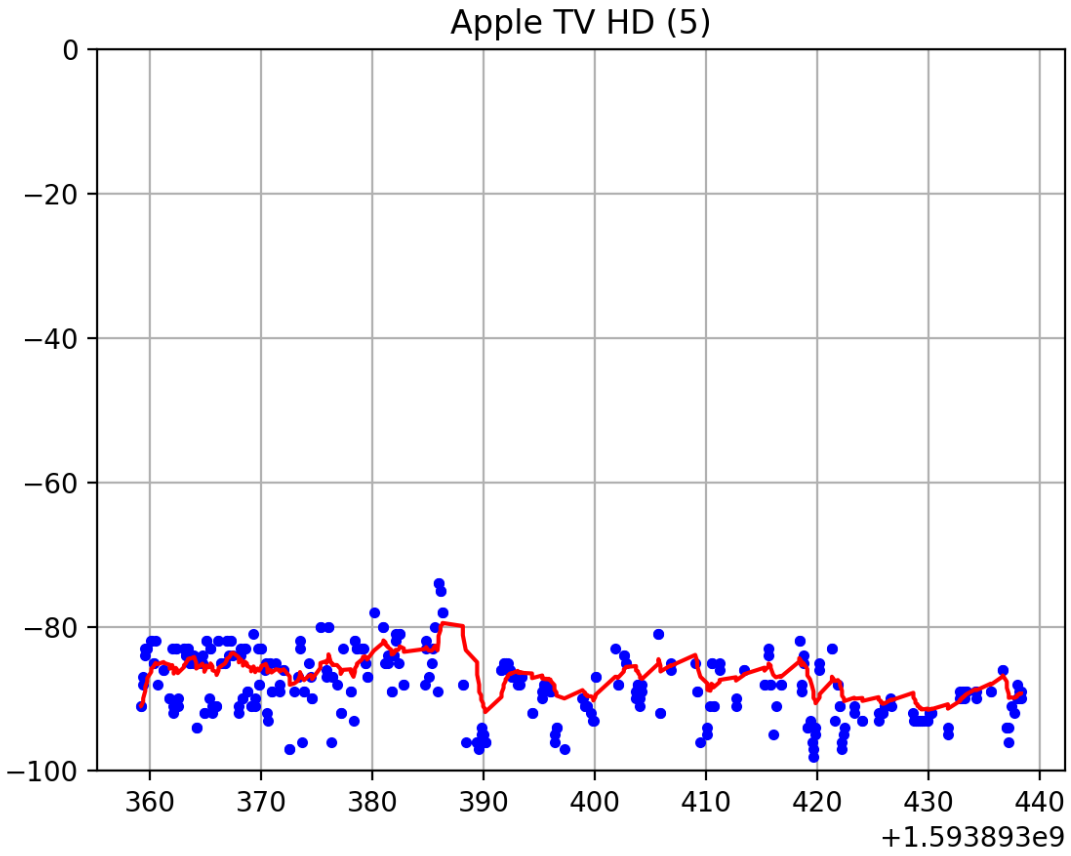



**Figure 9: Experiment #2 - Scanning Apple TV from Back Yard**

# 5 Gauss-Markov Process with Random Bias

The Gauss-Markov process can be augmented with a random bias process to model certain systems that seem to have a Gauss-Markov behavior offset by a random bias [1]. The model differential equations are

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & -\beta_g \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{2\sigma_g^2\beta_g} \end{bmatrix} \begin{bmatrix} w_b \\ w_g \end{bmatrix} \tag{19}$$

$$y = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + v \tag{20}$$

where $v$ is the measurement-error zero-mean Gaussian random process.

Figure 10 illustrates the block diagram for the Gauss-Markov process with random bias.

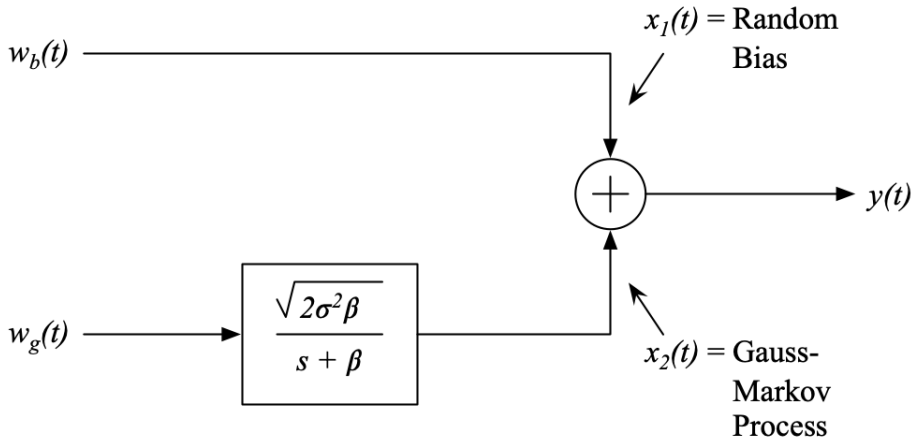



**Figure 10: Gauss-Markov with Random Bias Block Diagram**

The vector discrete-time representation of the Gauss-Markov with random bias is

$$\mathbf{x}_k = \mathbf{\Phi}_k \mathbf{x}_{k-1} + \mathbf{w}_k \tag{21}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{22}$$

where $\mathbf{w}_k$ is a white noise sequence representing process noise with known covariance

$$E[\mathbf{w}_k \mathbf{w}_k^T] = \mathbf{Q}_k \tag{23}$$

and where $\mathbf{v}_k$ is a white noise sequence representing measurement error with known measurement error covariance

$$E[\mathbf{v}_k \mathbf{v}_k^T] = \mathbf{R}_k \tag{24}$$

In order to specify the discrete-time matrix values, as before we define the discrete-time difference to be $\tau_k = t_k - t_{k-1}$.

The discrete-time state transition matrix is

$$\mathbf{\Phi}_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{-\beta_g \tau_k} \end{bmatrix} \tag{25}$$

The process covariance matrix is

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_b^2 & 0 \\ 0 & \sigma_g^2 \left(1 - e^{-2\beta_g \tau_k}\right) \end{bmatrix} \tag{26}$$

where $\sigma_b^2$ is the mean-square value of the random bias, $\sigma_g^2$ is the Gauss-Markov process mean-square value, and $1/\beta_g$ is the Gauss-Markov process time constant.

Lastly, the measurement transformation matrix is

$$\mathbf{H}_k = \begin{bmatrix} 1 & 1 \end{bmatrix} \tag{27}$$

# 6 Gauss-Markov with Random Bias Kalman Filter

The Kalman filter implementation of a Gauss-Markov with random bias process model can be implemented efficiently since it involves only $2 \times 2$ matrix arithmetic.

At the $k$th time point, we obtain a measurement $\mathbf{z}_k$ at time $t_k$. We model our measurement error with a covariance $\mathbf{R}_k$. At each time point, the filter provides a best estimate, $\hat{\mathbf{x}}_k$ and maintains a state estimate covariance, $\mathbf{P}_k$. We initialize the filter with $\hat{\mathbf{x}}_0 = \mathbf{z}_0$ and $\mathbf{P}_0 = \mathbf{P0}$, where $\mathbf{P0}$ is a suitably chosen value based on empirical analysis.

For each acquisition event, $k$, we perform the following steps

1. Compute $\mathbf{\Phi}_k$ using (25)

2. Compute $\mathbf{Q}_k$ using (26)

3. Compute state estimate prediction:

$$\hat{\mathbf{x}}_k^- = \mathbf{\Phi}_k \hat{\mathbf{x}}_{k-1} \tag{28}$$

4. Compute state estimate variance prediction:

$$\mathbf{P}_k^- = \mathbf{\Phi}_k \mathbf{P}_k \mathbf{\Phi}_k^T + \mathbf{Q}_k \tag{29}$$

5. Compute Kalman gain:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left( \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1} \tag{30}$$

6. Compute state estimate correction:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left( \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^- \right) \tag{31}$$

7. Compute state estimate variance correction:

$$\mathbf{P}_k^+ = \left( \mathbf{I} - \mathbf{K}_k \mathbf{H}_k \right) \mathbf{P}_k^- \tag{32}$$

# 7 Experiments - Gauss-Markov with Random Bias

Using the data from our previous experiments, we can evaluate the Kalman filter using the Gauss-Markov with random bias model. Our Kalman filter was initialized with the following parameters:

$$\mathbf{P}_0 = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}, \quad \sigma_b = 0.5, \quad \begin{matrix} \sigma_g = 1 \\ \beta_g = 0.1 \end{matrix}, \quad R = 25$$

Using the Gauss-Markov with Random Bias model on the data of the first experiment for the UUID of the MacBook Pro computer produces the results shown in Figure 11. As before, the RSSI measurements are blue, and the filtered RSSI values are in red.

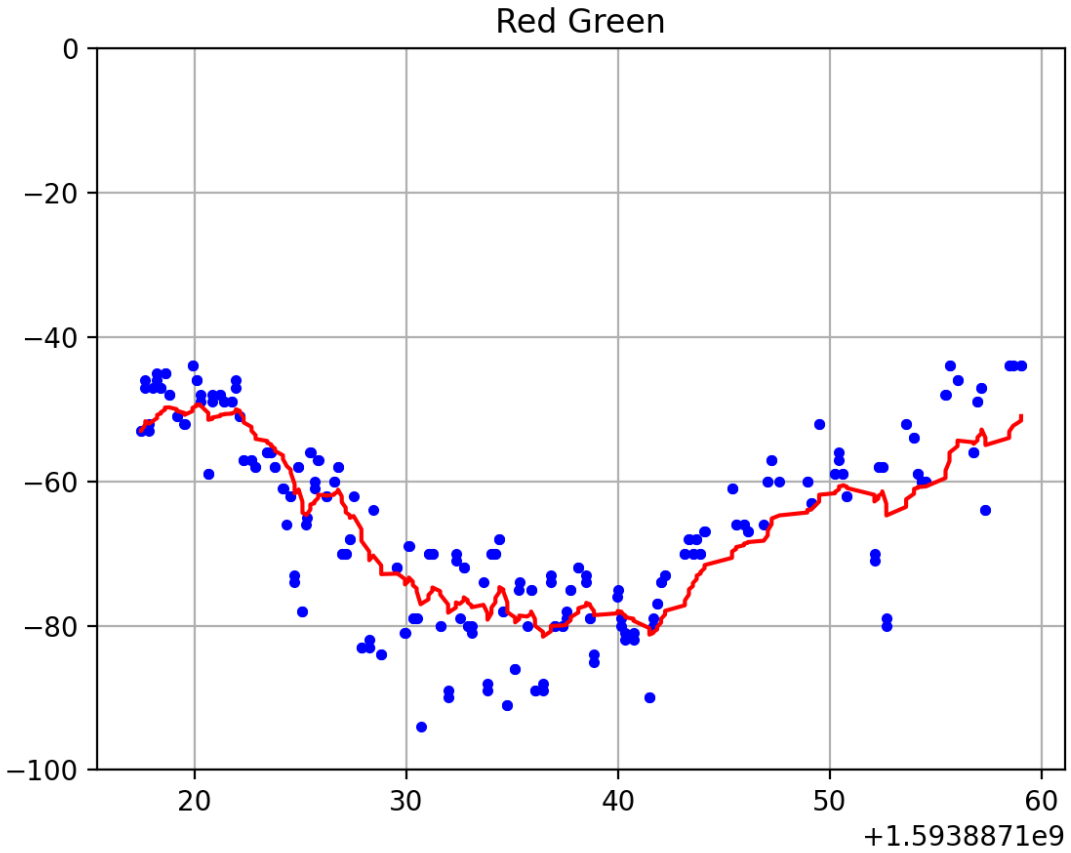



**Figure 11: Experiment #1 - MacBook Pro, Gauss-Markov with Random Bias**

Using the Gauss-Markov with Random Bias model on the data of the second experiment for the UUID of the stationary iPhone produces the results shown in Figure 12.

It does not appear that the Gauss-Markov with random bias model offers a noticeable improvement over the Gauss-Markov model. Given the added processing complexity in going

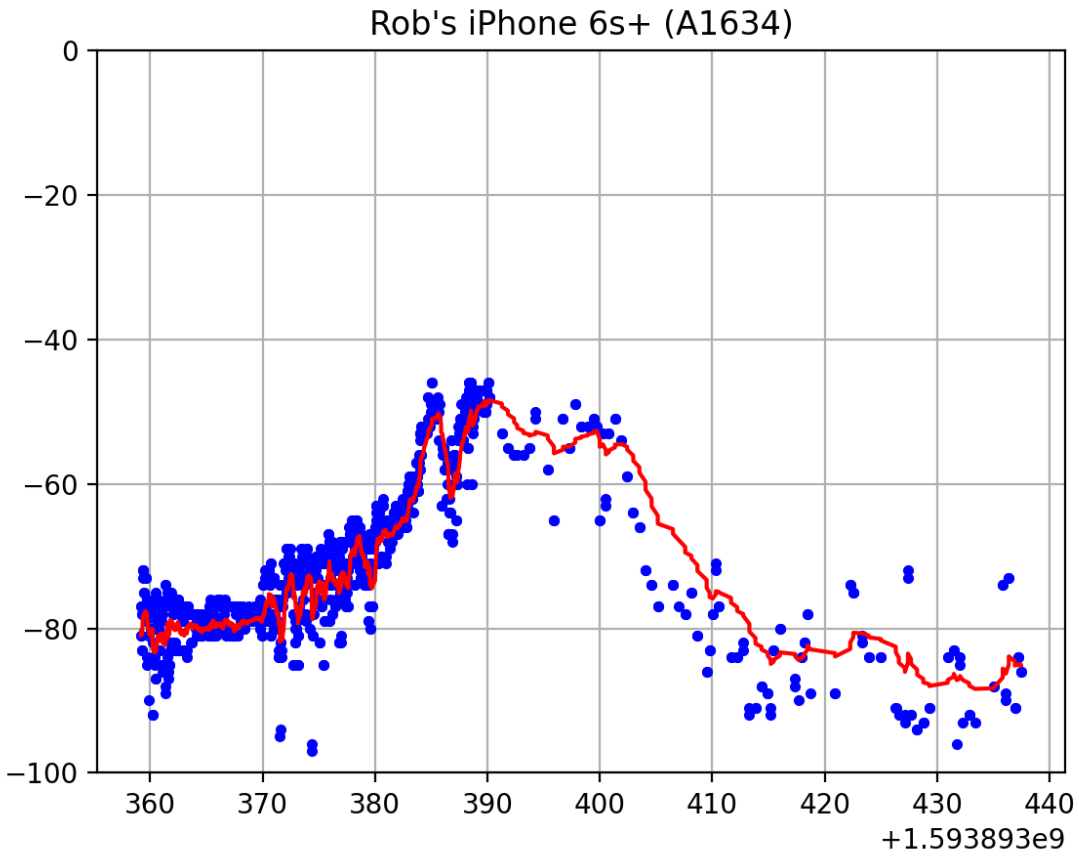



**Figure 12: Experiment #2 - iPhone, Gauss-Markov with Random Bias**

from a scalar filter to a $2 \times 2$ matrix filter, this filter model is not a viable choice for this particular application.

For completeness, the back yard results for the MacBook Pro can be seen in Figure 13, and the results for the Apple TV can be seen in Figure 14.

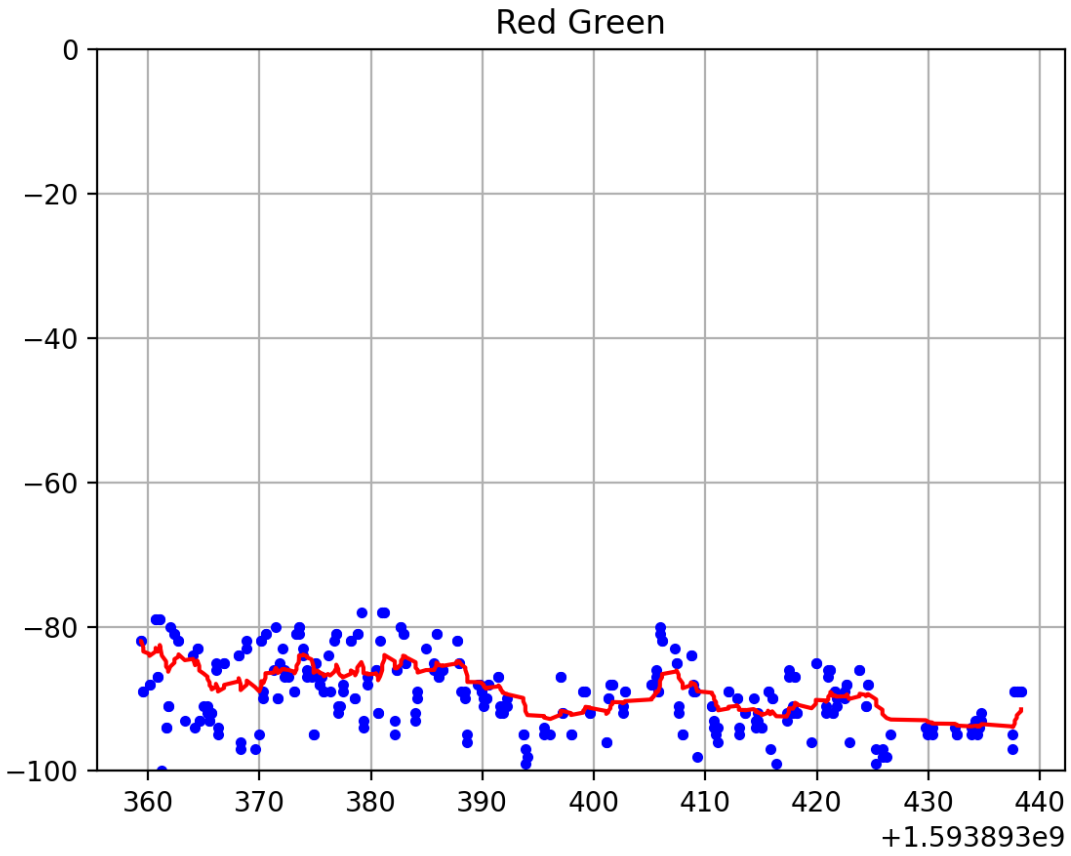



**Figure 13: Experiment #2 - MacBook Pro, Gauss-Markov with Random Bias**

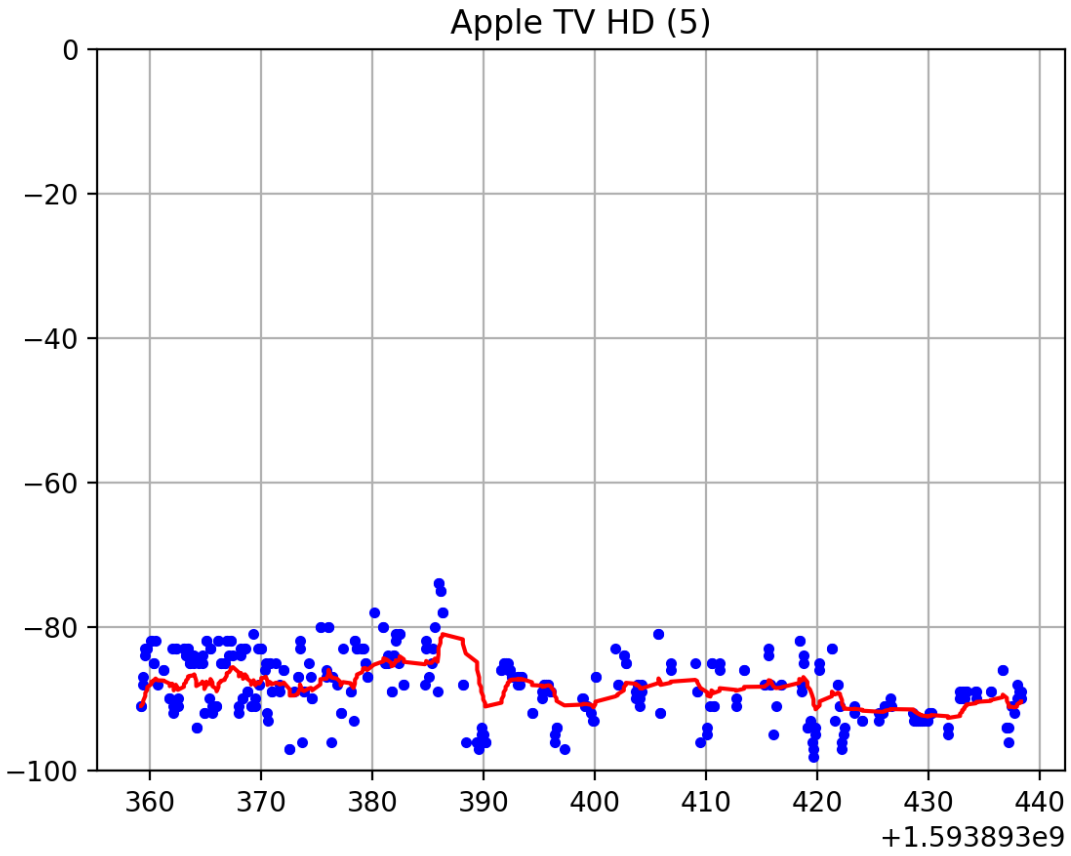



**Figure 14: Experiment #2 - Apple TV, Gauss-Markov with Random Bias**

# 8 Integrated Gauss-Markov Process

A useful process model based on the Gauss-Markov process is the integrated Gauss-Markov process [1]. This model is a popular model used in many engineering applications because it characterizes a low-pass filtered Gauss-Markov process. The model differential equations are

$$\begin{bmatrix} \dot{x_1} \\ \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \\ 0 & -\beta \end{bmatrix} \begin{bmatrix} x_1 \\ \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \\ \sqrt{2\sigma^2\beta} \end{bmatrix} w \tag{33}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \\ x_2 \end{bmatrix} + v \tag{34}$$

where $v$ is the measurement-error zero-mean Gaussian random process.

Figure 15 shows the block diagram of the integrated Gauss-Markov differential system.

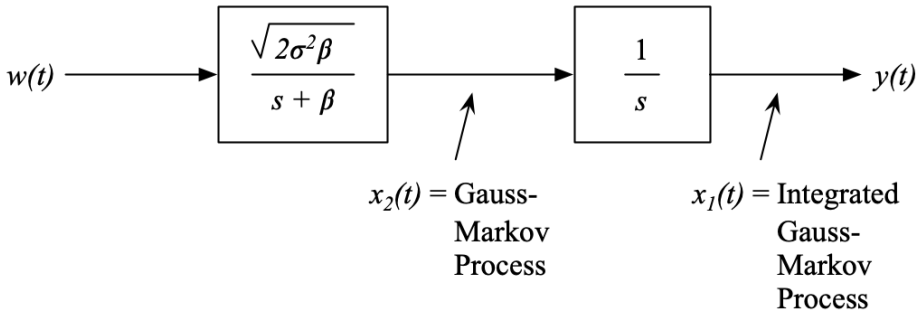



**Figure 15: Integrated Gauss-Markov Process Block Diagram**

The vector discrete-time representation of integrated Gauss-Markov system is

$$\mathbf{x}_k = \mathbf{\Phi}_k \mathbf{x}_{k-1} + \mathbf{w}_k \tag{35}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{36}$$

where $\mathbf{w}_k$ is a white noise sequence representing process noise with known covariance

$$E[\mathbf{w}_k \mathbf{w}_k^T] = \mathbf{Q}_k \tag{37}$$

and where $\mathbf{v}_k$ is a white noise sequence representing measurement error with known measurement error covariance

$$E[\mathbf{v}_k \mathbf{v}_k^T] = \mathbf{R}_k \tag{38}$$

In order to specify the discrete-time matrix values, as before we define the discrete-time difference to be $\tau_k = t_k - t_{k-1}$.

The discrete-time state transition matrix is

$$\mathbf{\Phi}_k = \begin{bmatrix} 1 & \frac{1}{\beta}\left(1 - e^{-\beta\tau_k}\right) \\ 0 & e^{-\beta\tau_k} \end{bmatrix} \tag{39}$$

The process mean-square values for the process covariance matrix are

$$E[w_1 w_1] = \frac{2\sigma^2}{\beta}\left[\tau_k - \frac{2}{\beta}\left(1 - e^{-\beta\tau_k}\right) + \frac{1}{2\beta}\left(1 - e^{-2\beta\tau_k}\right)\right] \tag{40}$$

$$E[w_1 w_2] = 2\sigma^2\left[\frac{1}{\beta}\left(1 - e^{-\beta\tau_k}\right) - \frac{1}{2\beta}\left(1 - e^{-2\beta\tau_k}\right)\right] \tag{41}$$

$$E[w_2 w_2] = \sigma^2\left(1 - e^{-2\beta\tau_k}\right) \tag{42}$$

and so the process covariance matrix is

$$\mathbf{Q}_k = \begin{bmatrix} E[w_1 w_1] & E[w_1 w_2] \\ E[w_1 w_2] & E[w_2 w_2] \end{bmatrix} \tag{43}$$

Lastly, the measurement transformation matrix is

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{44}$$

# 9 Integrated Gauss-Markov Kalman Filter

The Kalman filter implementation of an integrated Gauss-Markov process model can be implemented efficiently since it involves only $2 \times 2$ matrix arithmetic.

At the $k$th time point, we obtain a measurement $\mathbf{z}_k$ at time $t_k$. We model our measurement error with a covariance $\mathbf{R}_k$. At each time point, the filter provides a best estimate, $\hat{\mathbf{x}}_k$ and maintains a state estimate covariance, $\mathbf{P}_k$. We initialize the filter with $\hat{\mathbf{x}}_0 = \mathbf{z}_0$ and $\mathbf{P}_0 = \mathbf{P0}$, where $\mathbf{P0}$ is a suitably chosen value based on empirical analysis.

For each acquisition event, $k$, we perform the following steps

1. Compute $\mathbf{\Phi}_k$ using (39)

2. Compute $\mathbf{Q}_k$ using (43)

3. Compute state estimate prediction:

$$\hat{\mathbf{x}}_k^- = \mathbf{\Phi}_k \hat{\mathbf{x}}_{k-1} \tag{45}$$

4. Compute state estimate variance prediction:

$$\mathbf{P}_k^- = \mathbf{\Phi}_k \mathbf{P}_k \mathbf{\Phi}_k^T + \mathbf{Q}_k \tag{46}$$

5. Compute Kalman gain:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left( \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1} \tag{47}$$

6. Compute state estimate correction:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left( \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^- \right) \tag{48}$$

7. Compute state estimate variance correction:

$$\mathbf{P}_k^+ = \left( \mathbf{I} - \mathbf{K}_k \mathbf{H}_k \right) \mathbf{P}_k^- \tag{49}$$

# 10 Experiments - Integrated Gauss-Markov Model

Using the data from our previous experiments, we can evaluate the Kalman filter using the integrated Gauss-Markov model. Our Kalman filter was initialized with the following parameters:

$$\mathbf{P}_0 = \mathbf{I} \ , \quad \sigma = 0.2 \ , \quad \beta = 0.1 \ , \quad R = 5$$

Using the integrated Gauss-Markov model on the data of the first experiment for the UUID of the MacBook Pro computer produces the results shown in Figure 16. As before, the RSSI measurements are blue, and the filtered RSSI values are in red.

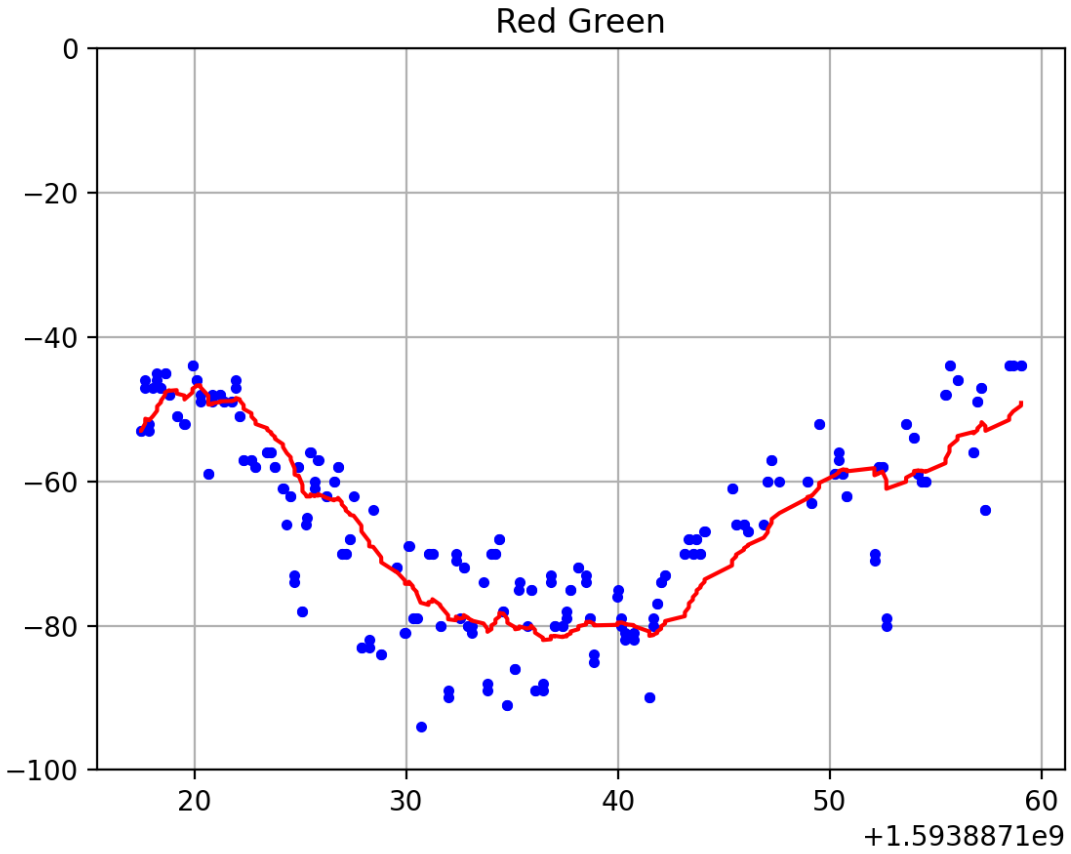



**Figure 16: Experiment #1 - MacBook Pro, Integrated Gauss-Markov**

Using the integrated Gauss-Markov model on the data of the second experiment for the UUID of the stationary iPhone produces the results shown in Figure 17.

Unlike the results of the Gauss-Markov with random bias model, it definitely appears that the integrated Gauss-Markov model offers a substantial improvement over the Gauss-Markov model. The filtered estimate is much more centered within the RSSI measurement data than that of the Gauss-Markov model, and the filtered estimate is smoother yet still responsive.

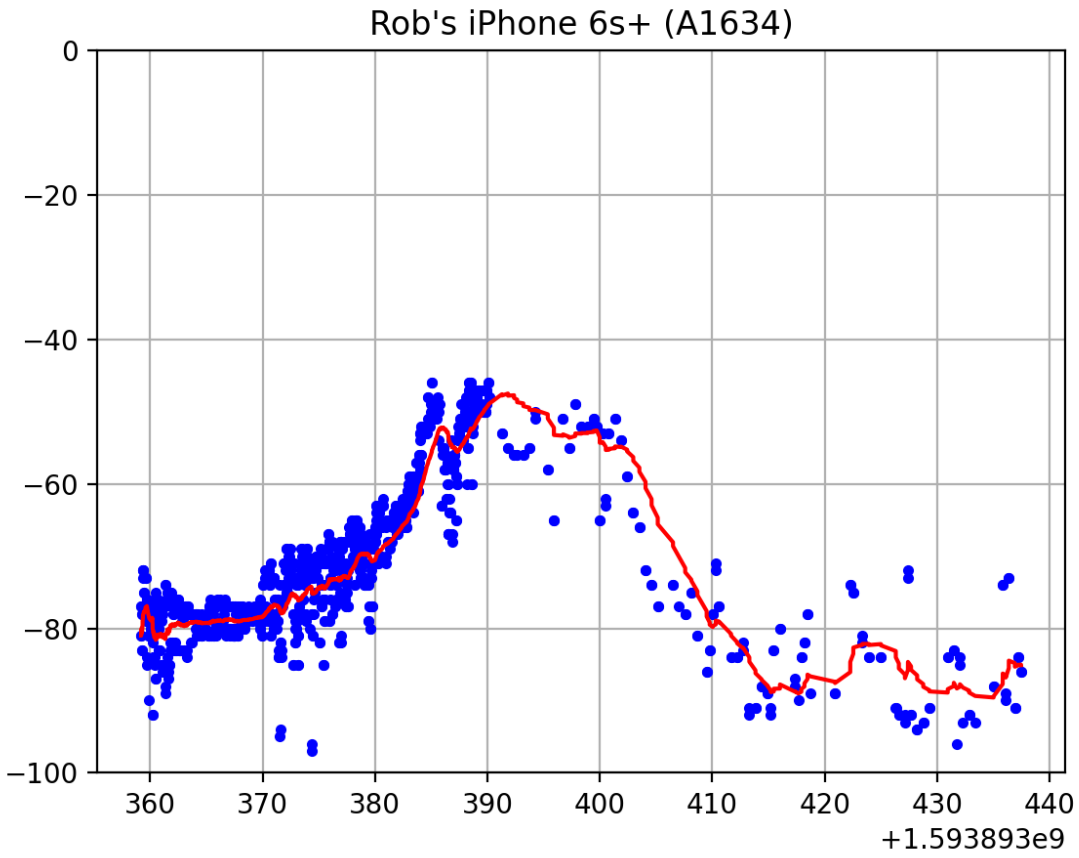



**Figure 17: Experiment #2 - Stationary iPhone, Integrated Gauss-Markov**

The added processing complexity in going from a scalar filter to a $2 \times 2$ matrix filter is worth the the effort for the use of this filter model for this particular application. In fact, because it is a two-state filter, the matrix operations can be explicitly coded for each element, thereby improving efficiency.

For completeness, the back yard results for the MacBook Pro can be seen in Figure 18, and the results for the Apple TV can be seen in Figure 19.

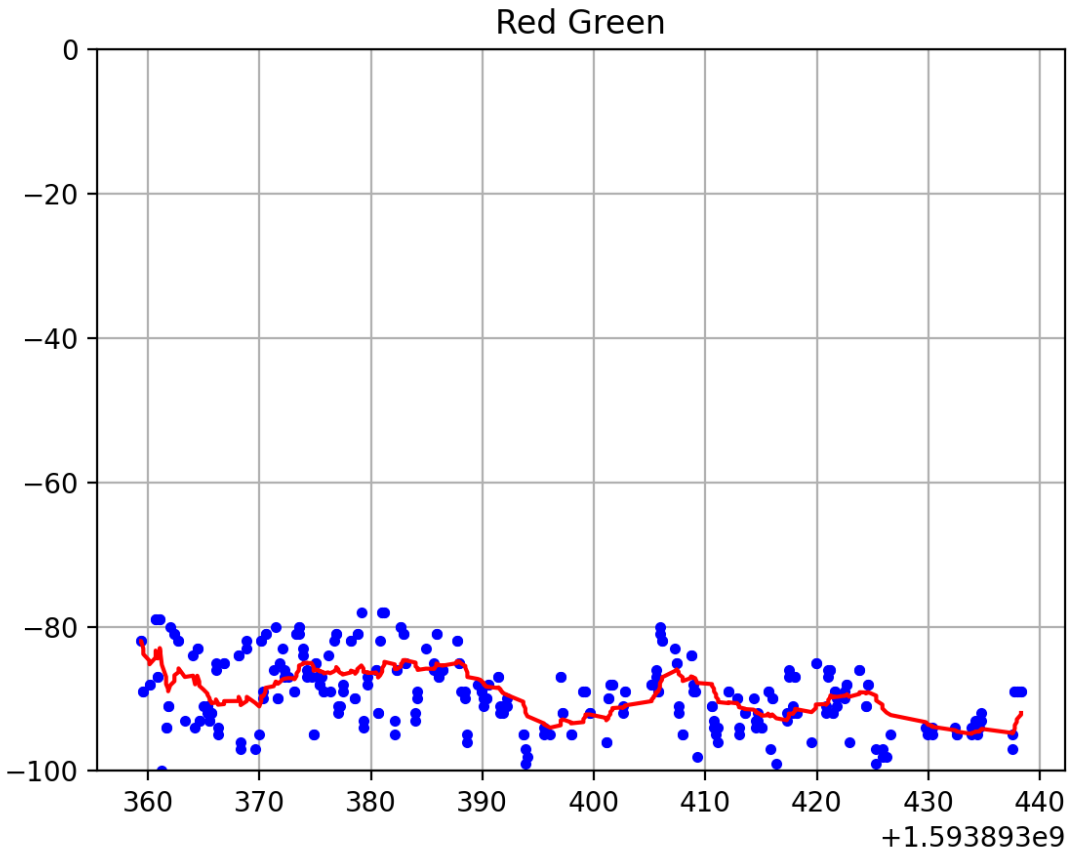



**Figure 18: Experiment #2 - MacBook Pro, Integrated Gauss-Markov**

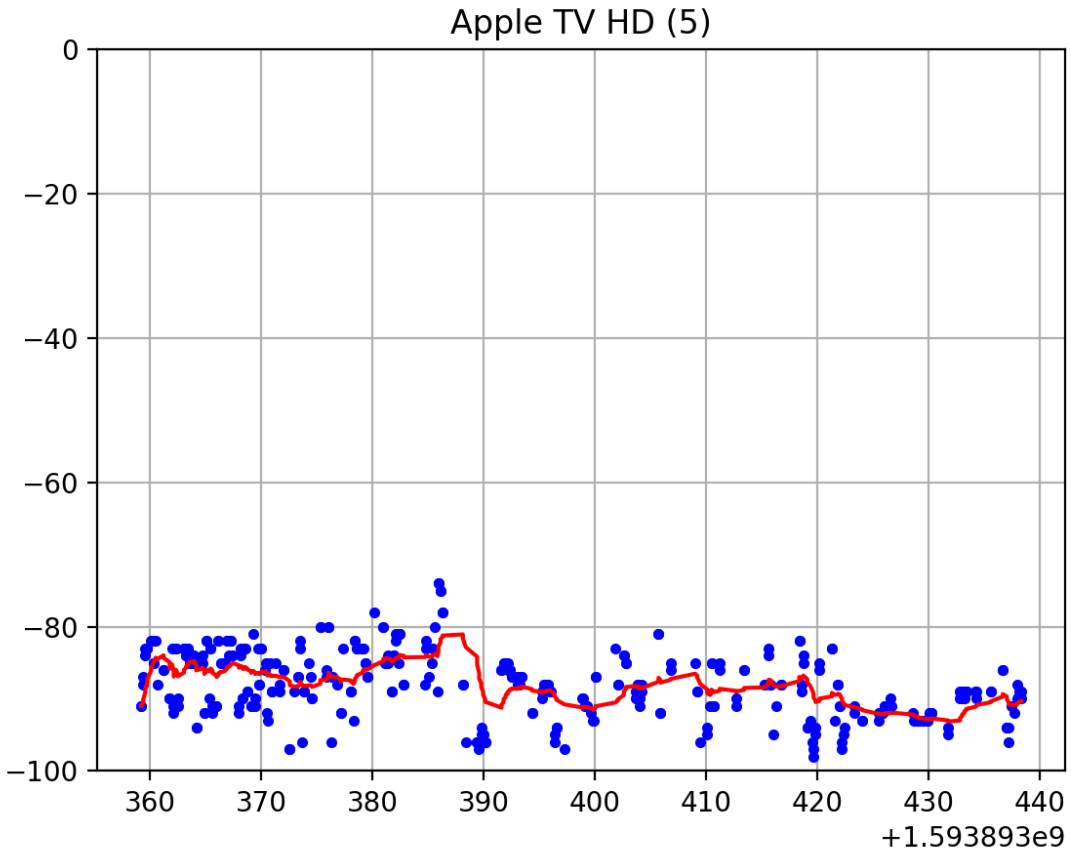
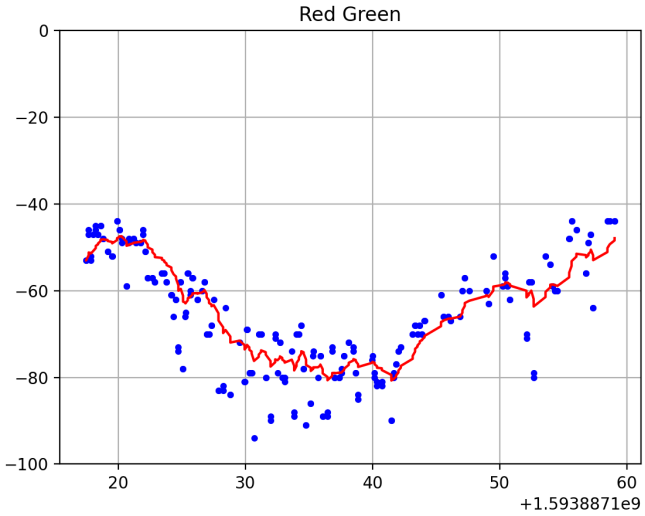


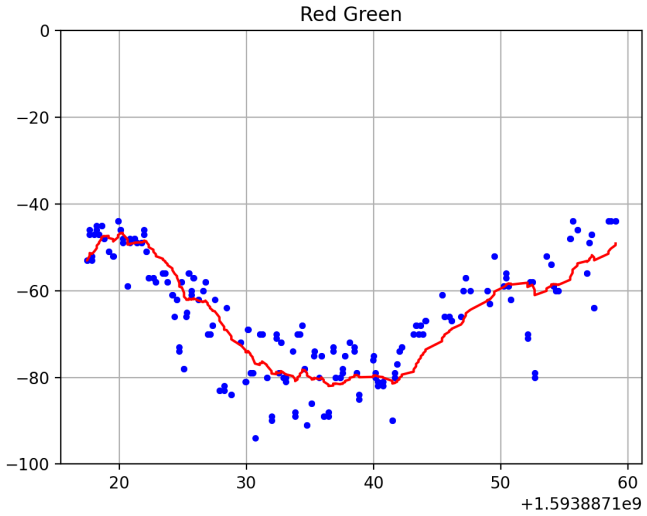

**Figure 19: Experiment #2 - Apple TV, Integrated Gauss-Markov**

# 11 Summary

It is evident that, for estimating RSSI signal levels acquired via a mobile device, the integrated Gauss-Markov Kalman filter produces superior results over the Gauss-Markov Kalman filter. Figures 20 and 21 illustrate this comparison.

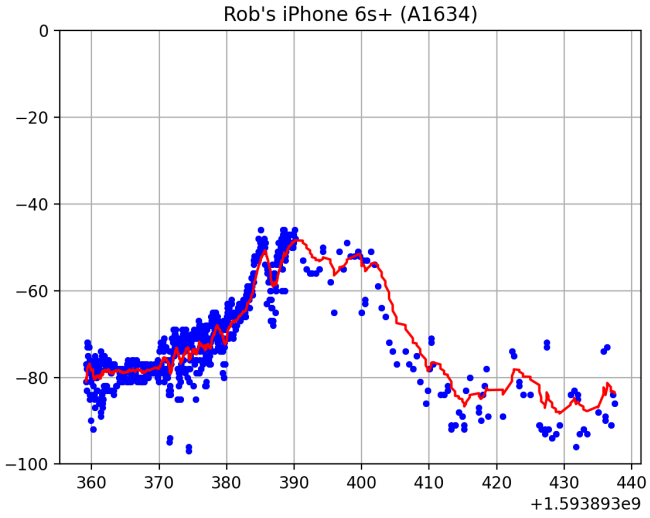



**(a) Gauss-Markov**

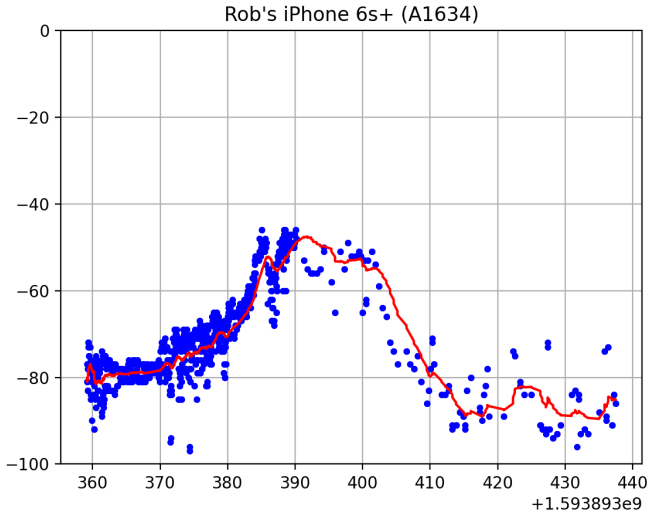



**(b) Integrated Gauss-Markov**

**Figure 20: Experiment #1 Model Comparisons**




**(a) Gauss-Markov**




**(b) Integrated Gauss-Markov**

**Figure 21: Experiment #2 Model Comparisons**

While additional refining of tuning the filter parameters is needed before producing a production-worthy filter, it can be concluded that the feasibility of using a Kalman filter for estimating RSSI signal level is justified, and it is recommended that the integrated Gauss-Markov process model be the preferred choice for implementation. By taking advantage of the symmetry of the **P** matrix and the structure of the **H** matrix, the implementation of the integrated Gauss-Markov Kalman Filter can be made quite efficient.

You can access the source for this project from my GitHub repo:
https://github.com/rshuston/RSSISniffer

# 12 Appendix A - IGM, KF Non-Matrix Realization

Define the discrete-time difference to be $\tau_k = t_k - t_{k-1}$. The discrete-time formulation for the integrated Gauss-Markov system is

$$\begin{bmatrix} x_0 \\ \\ x_1 \end{bmatrix}_k = \begin{bmatrix} \phi_{00} & \phi_{01} \\ \\ \phi_{10} & \phi_{11} \end{bmatrix}_k \begin{bmatrix} x_0 \\ \\ x_1 \end{bmatrix}_{k-1} + \begin{bmatrix} w_0 \\ \\ w_1 \end{bmatrix}_k$$

$$z_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ \\ x_1 \end{bmatrix}_k + v_k$$

where the state transition matrix is

$$\begin{bmatrix} \phi_{00} & \phi_{01} \\ \\ \phi_{10} & \phi_{11} \end{bmatrix}_k = \begin{bmatrix} 1 & \frac{1}{\beta}\left(1 - e^{-\beta\tau_k}\right) \\ \\ 0 & e^{-\beta\tau_k} \end{bmatrix}$$

and where $\mathbf{w}_k$ is a white noise sequence representing process noise with known covariance

$$E[\mathbf{w}_k \mathbf{w}_k^T] = \mathbf{Q}_k$$

and where $v_k$ is a scalar white noise sequence representing measurement error with known measurement error covariance

$$E[v_k^2] = r_k$$

The process covariance matrix, $\mathbf{Q}_k$, is

$$\mathbf{Q}_k = \begin{bmatrix} q_{00} & q_{01} \\ \\ q_{01} & q_{11} \end{bmatrix}$$

where

$$q_{00} = \frac{2\sigma^2}{\beta}\left[\tau_k - \frac{2}{\beta}\left(1 - e^{-\beta\tau_k}\right) + \frac{1}{2\beta}\left(1 - e^{-2\beta\tau_k}\right)\right]$$

$$q_{01} = 2\sigma^2 \left[ \frac{1}{\beta} \left( 1 - e^{-\beta \tau_k} \right) - \frac{1}{2\beta} \left( 1 - e^{-2\beta \tau_k} \right) \right]$$

$$q_{11} = \sigma^2 \left( 1 - e^{-2\beta \tau_k} \right)$$

Taking advantage of the symmetry of the covariance matrices and that $\phi_{10} = 0$, the non-matrix implementation of the integrated Gauss-Markov Kalman filter is as follows (the $k$ subscripts have been omitted for readability):

Prediction:

$$\begin{aligned} x_0^- &= \phi_{00} x_0 + \phi_{01} x_1 \\ x_1^- &= \phi_{11} x_1 \end{aligned}$$

$$\begin{aligned} p_{00}^- &= (\phi_{00} p_{00} + \phi_{01} p_{01}) \phi_{00} + (\phi_{00} p_{01} + \phi_{01} p_{11}) \phi_{01} + q_{00} \\ p_{01}^- &= (\phi_{00} p_{01} + \phi_{01} p_{11}) \phi_{11} + q_{01} \\ p_{11}^- &= \phi_{11} p_{11} \phi_{11} + q_{11} \end{aligned}$$

Kalman gain:

$$\begin{aligned} k_0 &= \frac{p_{00}^-}{p_{00}^- + r} \\ k_1 &= \frac{p_{01}^-}{p_{00}^- + r} \end{aligned}$$

Correction:

$$\begin{aligned} x_0^+ &= x_0^- + k_0 (z - x_0^-) \\ x_1^+ &= x_1^- + k_1 (z - x_0^-) \end{aligned}$$

$$\begin{aligned} p_{00}^+ &= (1 - k_0) p_{00}^- \\ p_{01}^+ &= (1 - k_0) p_{01}^- \\ p_{11}^+ &= (-k_1) p_{01}^- + p_{11}^- \end{aligned}$$

# 13 Appendix B - General References


[1] Brown, R. G., 1983,
*Introduction to Random Signal Analysis and Kalman Filtering*,
John Wiley & Sons, Inc., New York, NY.

[2] Sorenson, H. W. (Ed.), 1985,
*Kalman Filtering: Theory and Application*,
IEEE Press, New York, NY.

[3] Chui, C. K. and Chen, G., 1987,
*Kalman Filtering with Real-Time Applications*,
Springer-Verlag, New York, NY.

[4] Grewal, M. S., and Andrews, A. P., 1993,
*Kalman Filtering: Theory and Practice*,
Prentice-Hall, Englewood Cliffs, NJ.

[5] Sheng, Xia, Emira, Xin, Moon, Valero-Lopez, and Sanchez-Sinencio, 2002,
*A Monolithic CMOS Low-IF Bluetooth Receiver*,
IEEE Custom Circuits Conference, 2002, pp. 247-250

[6] Emira, Valdes-Garcia, Xia, Moheildin, Valero-Lopez, Moon, Xin, and Sanchez-Sinencio, 2004,
*A BiCMOS Bluetooth/Wi-Fi Receiver*,
IEEE Radio Frequency Integrated Circuits Symposium, 2004, pp. 519-522

[7] Johnsrud and Aaberge, 2010,
*RSSI Interpretation and Timing*,
Texas Instruments Design Note DN505