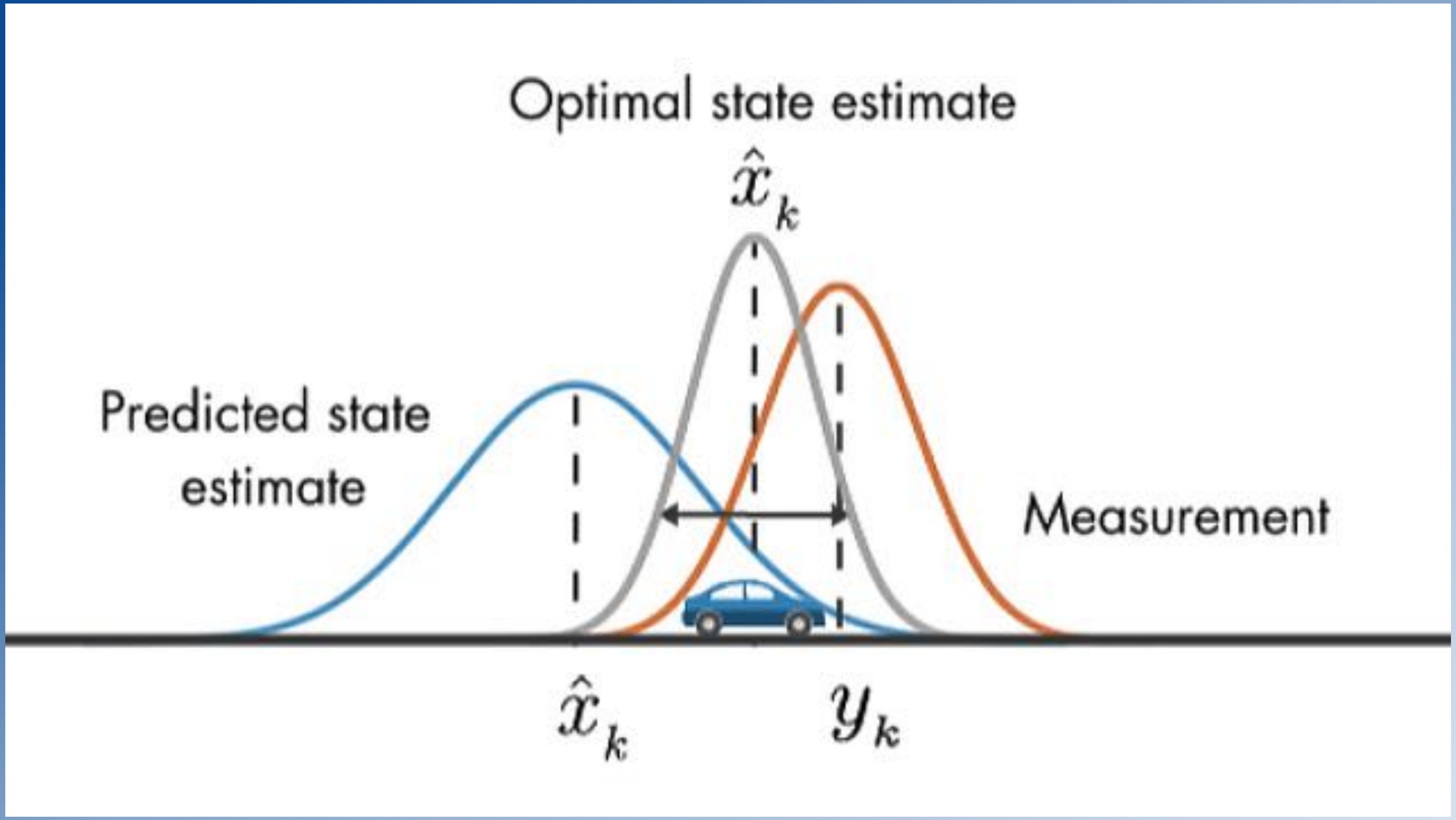


Lecture 16: Kalman Filter

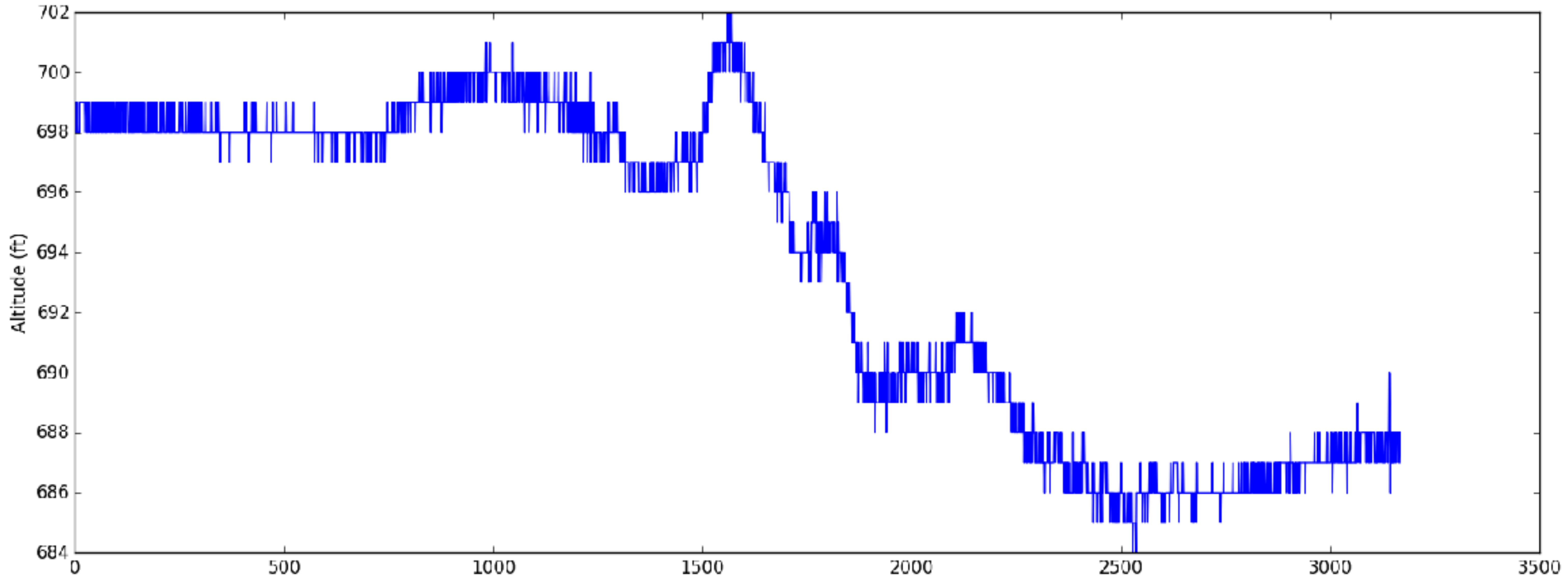
ENAE 380 Flight Software Systems

November 16, 2024

Kinda/sorta related but not really…

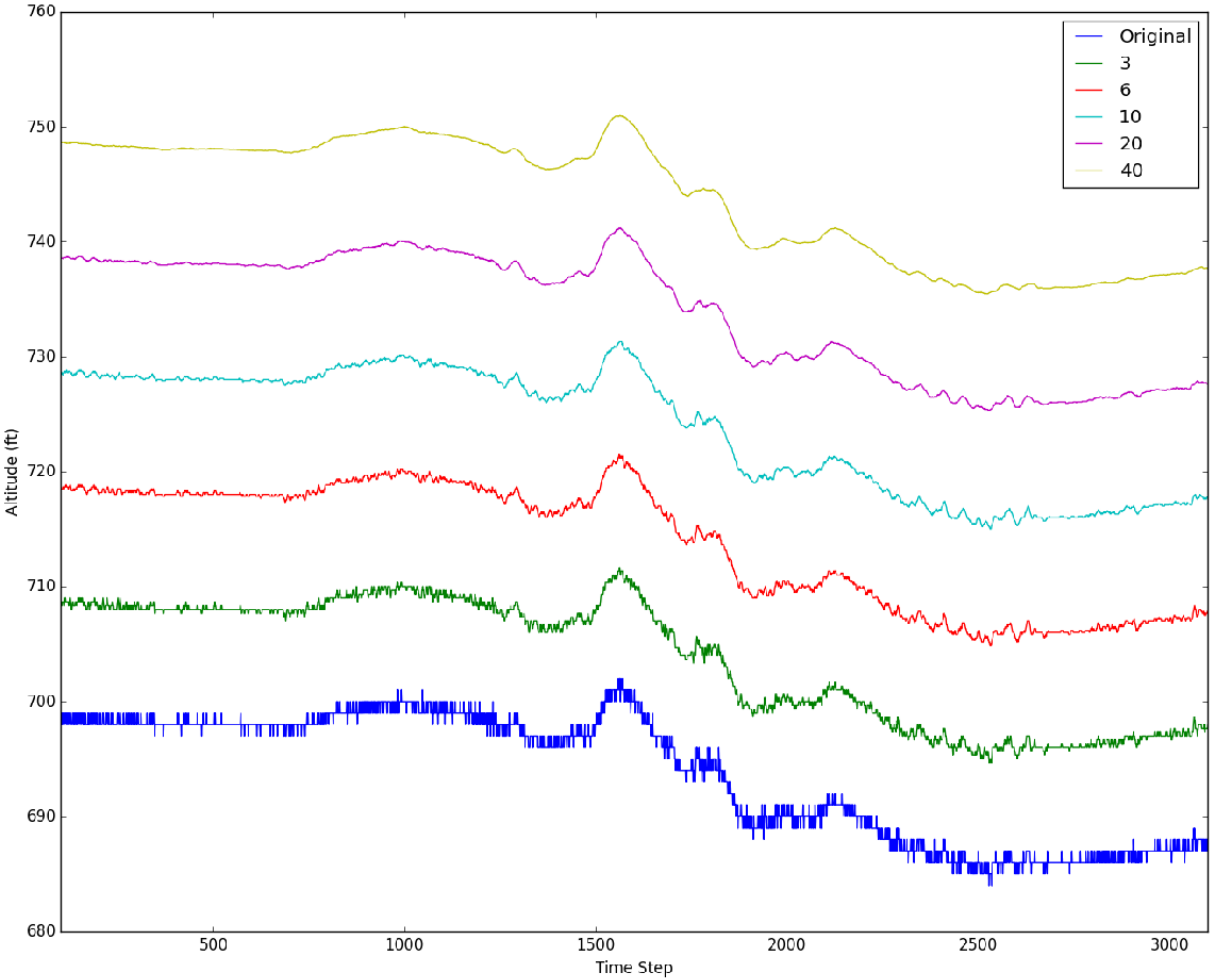
Moving Average

Altitude data from an aircraft



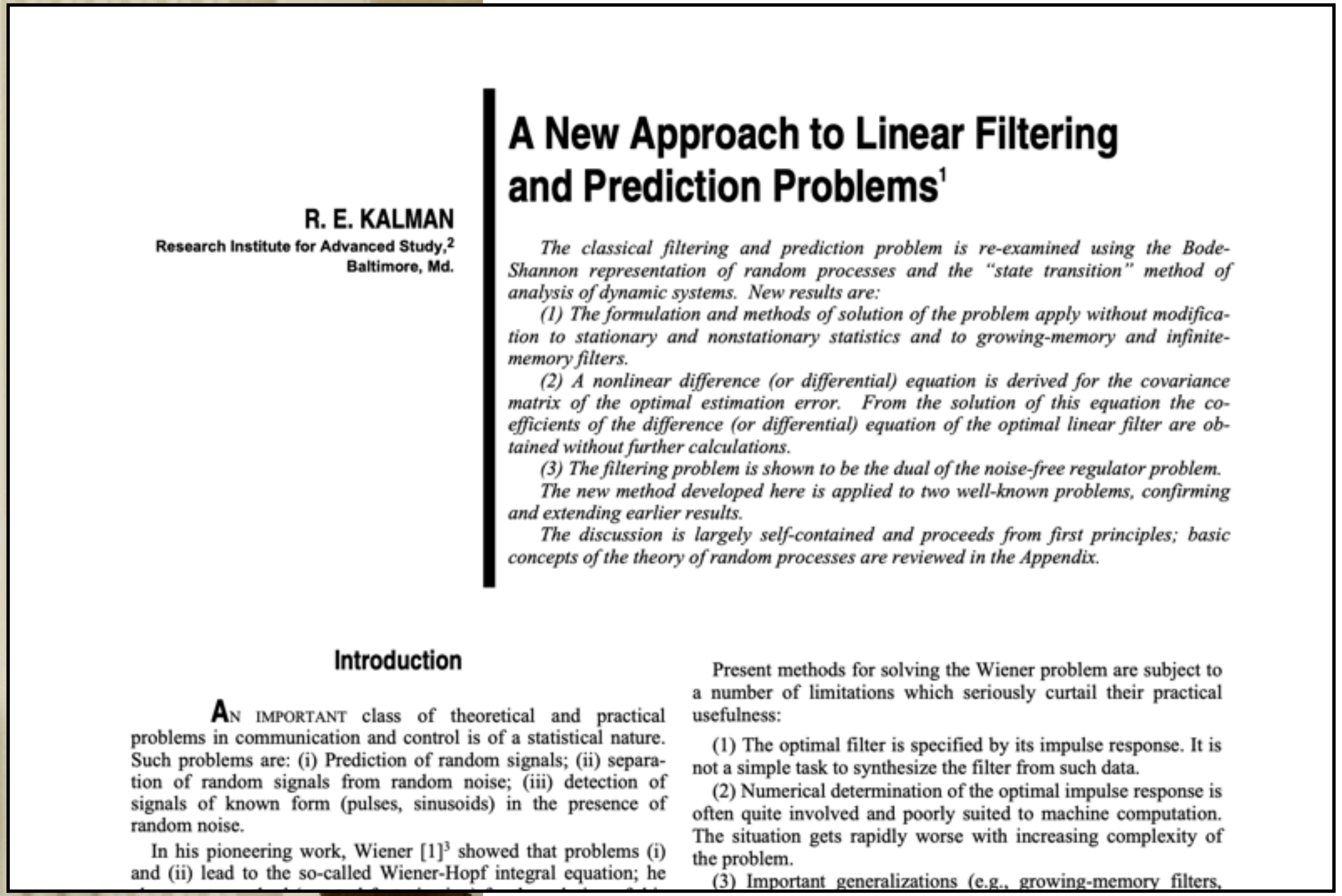
$$
\begin{aligned}
\bar{p}_{\mathrm{SM}} &= \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n} \\
&= \frac{1}{n} \sum_{i=0}^{n-1} p_{M-i}
\end{aligned}
$$

```python
import numpy as np
import matplotlib.pyplot as plt

arr = []

inp = open('altitude.txt', 'r')
for line in inp.readlines():
    for i in line.split():
        arr.append(float(i))

x = range(len(arr))

fig, ax = plt.subplots()
ax.plot(x,arr, label = "Original")

#Moving Average with variable window size
window_lst = [3,6,10,20,40]
arr_avg = np.zeros((len(window_lst),len(arr)))
for i,window in enumerate(window_lst):
    avg_mask = np.ones(window)/window
    arr_avg[i, :] = np.convolve(arr,avg_mask, 'same')
    ax.plot(x, arr_avg[i,:] + (i+1)*10, label=window)


ax.legend()
plt.ylabel('Altitude (ft)')
plt.xlabel('Time Step')
plt.xlim([100,3100])
plt.ylim([680,760])
plt.show()
```



760
750
740
730
720
710
700
690
680
Altitude (ft)
500
1000
1500
2000
2500
3000
Time Step
Original
3
6
10
20
40

Kalman Filter



A New Approach to Linear Filtering and Prediction Problems[1]

R. E. KALMAN
Research Institute for Advanced Study,[2]
Baltimore, Md.

The classical filtering and prediction problem is re-examined using the Bode-Shannon representation of random processes and the "state transition" method of analysis of dynamic systems. New results are:

(1) The formulation and methods of solution of the problem apply without modification to stationary and nonstationary statistics and to growing-memory and infinite-memory filters.

(2) A nonlinear difference (or differential) equation is derived for the covariance matrix of the optimal estimation error. From the solution of this equation the coefficients of the difference (or differential) equation of the optimal linear filter are obtained without further calculations.

(3) The filtering problem is shown to be the dual of the noise-free regulator problem.

The new method developed here is applied to two well-known problems, confirming and extending earlier results.

The discussion is largely self-contained and proceeds from first principles; basic concepts of the theory of random processes are reviewed in the Appendix.

Introduction

An important class of theoretical and practical problems in communication and control is of a statistical nature. Such problems are: (i) Prediction of random signals; (ii) separation of random signals from random noise; (iii) detection of signals of known form (pulses, sinusoids) in the presence of random noise.

In his pioneering work, Wiener [1][3] showed that problems (i) and (ii) lead to the so-called Wiener-Hopf integral equation; he

Present methods for solving the Wiener problem are subject to a number of limitations which seriously curtail their practical usefulness:

(1) The optimal filter is specified by its impulse response. It is not a simple task to synthesize the filter from such data.

(2) Numerical determination of the optimal impulse response is often quite involved and poorly suited to machine computation. The situation gets rapidly worse with increasing complexity of the problem.

(3) Important generalizations (e.g., growing-memory filters,

https://qz.com/726338/the-code-that-took-america-to-the-moon-was-just-published-to-github-and-its-like-a-1960s-time-capsule/

https://github.com/chrislgarry/Apollo-11

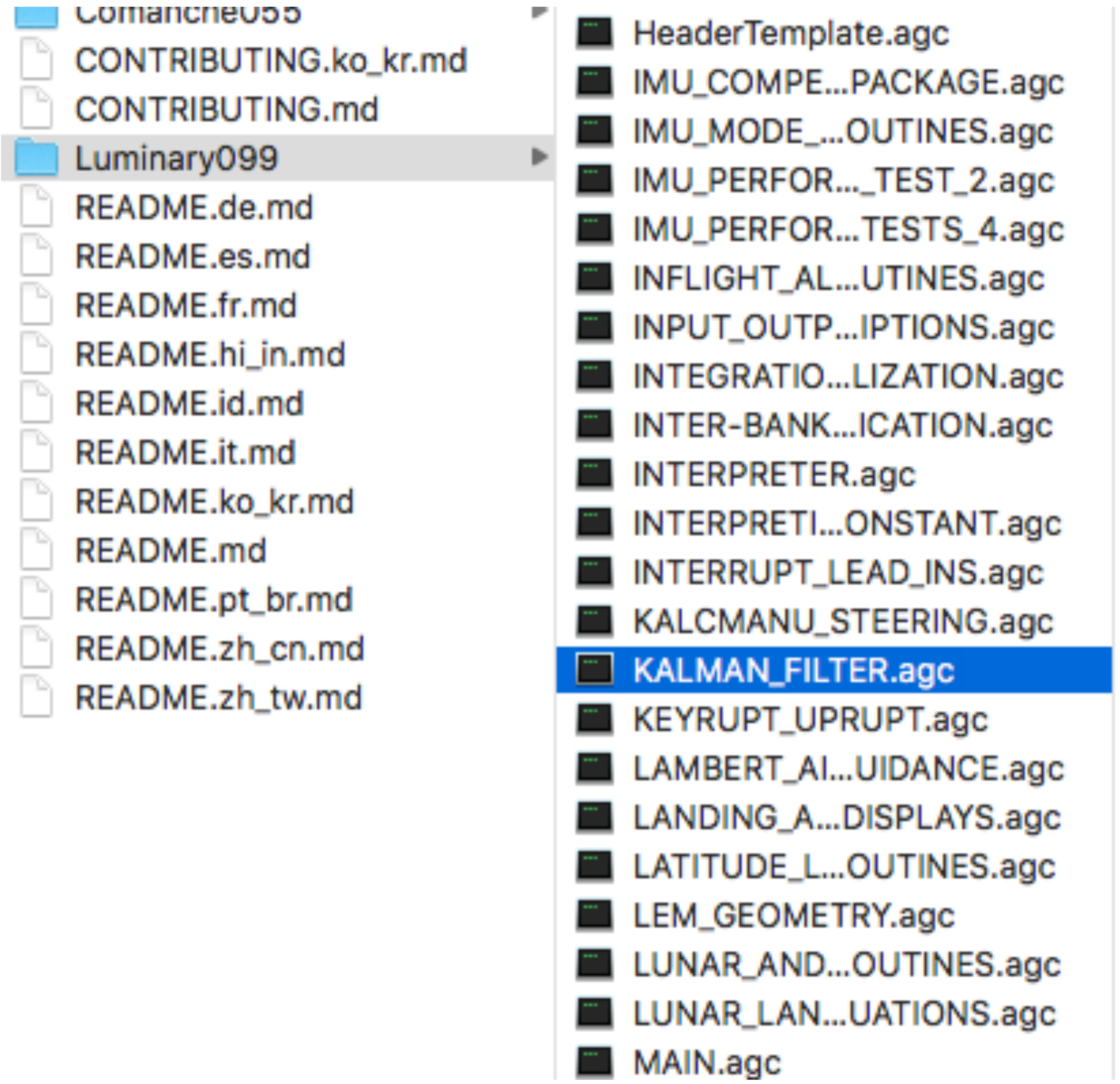
- AGC_BLOCK_TWO_SELF_CHECK.agc
- AGS_INITIALIZATION.agc
- ALARM_AND_ABORT.agc
- AOSTASK_AND_AOSJOB.agc
- AOTMARK.agc
- ASCENT_GUIDANCE.agc
- ASSEMBLY_AND_OPERATION_INFORMATION.agc
- ATTITUDE_MANEUVER_ROUTINE.agc
- BURN_BABY_BURN--MASTER_IGNITION_ROUTINE.agc
- CONIC_SUBROUTINES.agc
- CONTROLLED_CONSTANTS.agc
- DAPIDLER_PROGRAM.agc
- DAP_INTERFACE_SUBROUTINES.agc
- DISPLAY_INTERFACE_ROUTINES.agc
- DOWNLINK_LISTS.agc

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:    PLEASE CRANK THE
                TC      BANKCALL        #               SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOP00H        # TERMINATE
                TCF     P63SPOT3        # PROCEED       SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER         INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

LUNAR_LANDING_GUIDANCE_EQUATIONS.agc

```
        EXIT
        CAF  ZERO
        TS   FCOLD
        TS   FWEIGHT
        TS   FWEIGHT +1
VRTSTART     TS   WCHVERT
# Page 801
        CAF  TWO      # WCHPHASE = 2 ---> VERTICAL: P65,P66,P67
        TS   WCHPHOLD
        TS   WCHPHASE
        TC   BANKCALL    # TEMPORARY, I HOPE HOPE HOPE
        CADR     STOPRATE    # TEMPORARY, I HOPE HOPE HOPE
        TC   DOWNFLAG    # PERMIT X-AXIS OVERRIDE
        ADRES    XOVINFLG
        TC   DOWNFLAG
        ADRES    REDFLAG
        TCF  VERTGUID
```

```
#       "IT WILL BE PROVED TO THY FACE THAT THOU HAST MEN ABOUT THEE THAT
# Page 310
#       USUALLY TALK OF A NOUN AND A VERB, AND SUCH ABOMINABLE WORDS AS NO
#       CHRISTIAN EAR CAN ENDURE TO HEAR."
#                               HENRY 6, ACT 2, SCENE 4

# THE FOLLOWING ASSIGNMENTS FOR PINBALL ARE MADE ELSEWHERE

# RESERVED FOR PINBALL EXECUTIVE ACTION
#
#DSPCOUNT       ERASE                   # DISPLAY POSITION INDICATOR
#DECBRNCH       ERASE                   # +DEC, -DEC, OCT INDICATOR
#VERBREG        ERASE                   # VERB CODE
#NOUNREG        ERASE                   # NOUN CODE
```



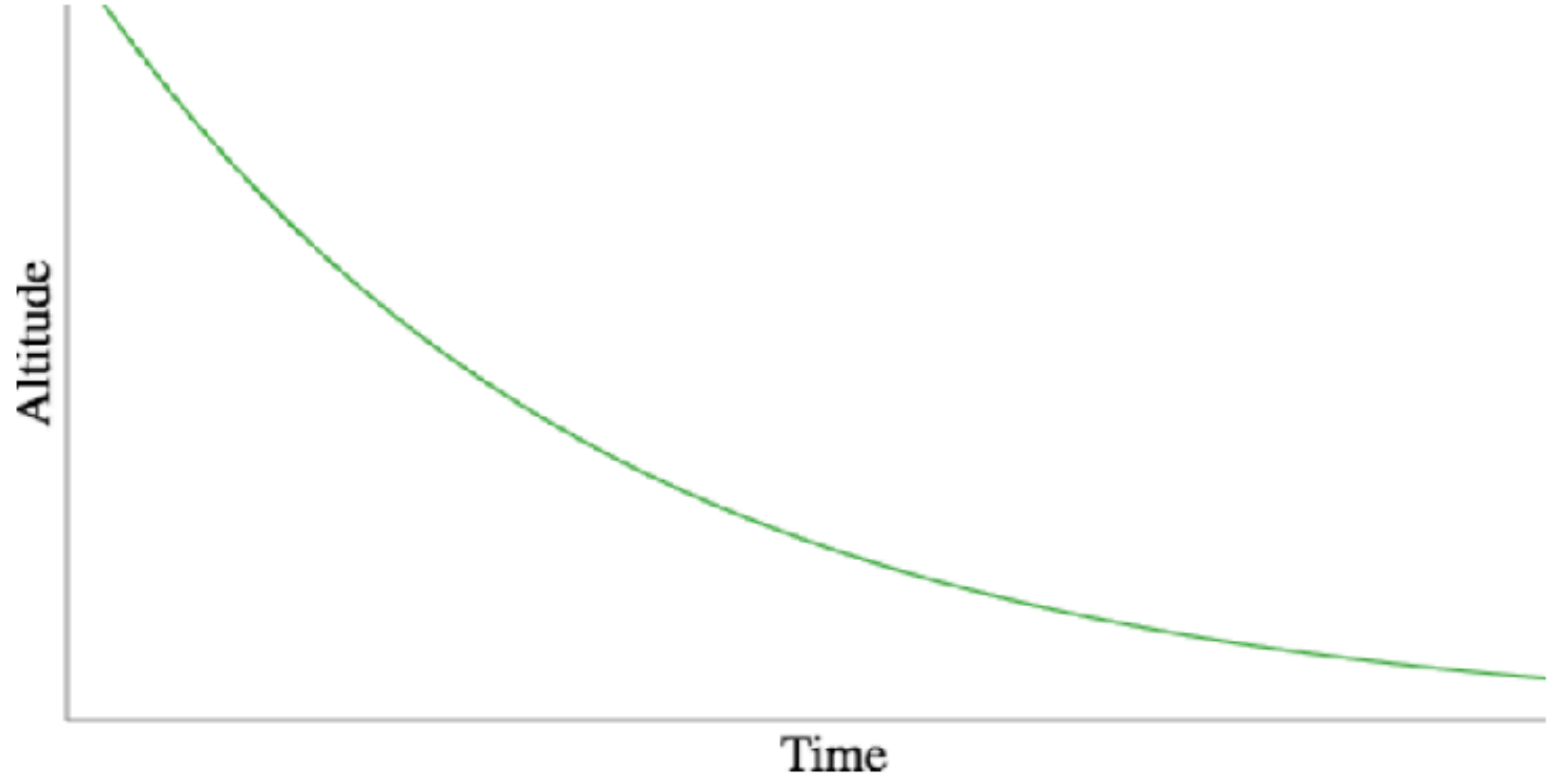

```
RATELOOP        CA      TWO
                TS      DAPTEMP6
                DOUBLE
                TS      Q
                INDEX   DAPTEMP6
                CCS     TJP
                TCF     +2
                TCF     LOOPRATE
                AD      -100MST6
                EXTEND
                BZMF    SMALLTJU
                INDEX   DAPTEMP6
                CCS     TJP
                CA      -100MST6
                TCF     +2
                CS      -100MST6
                INDEX   DAPTEMP6
                ADS     TJP
                INDEX   DAPTEMP6
                CCS     TJP
                CS      -100MS          # 0.1 AT 1
                TCF     +2
                CA      -100MS
```

1. A Simple Example

$$\mathbf{altitude_{current\_time} = 0.98 * altitude_{previous\_time}}$$

0.95

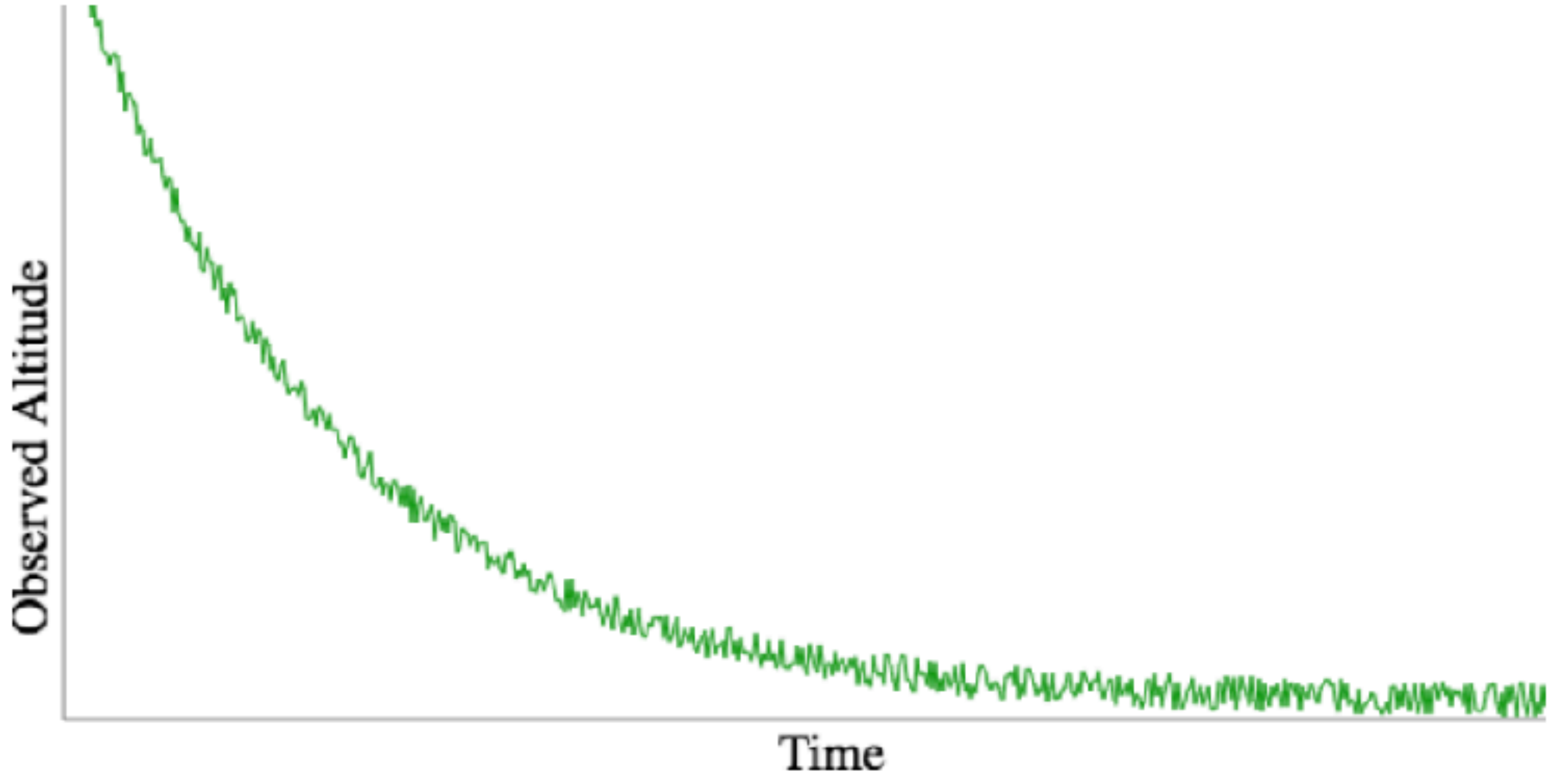
0.995



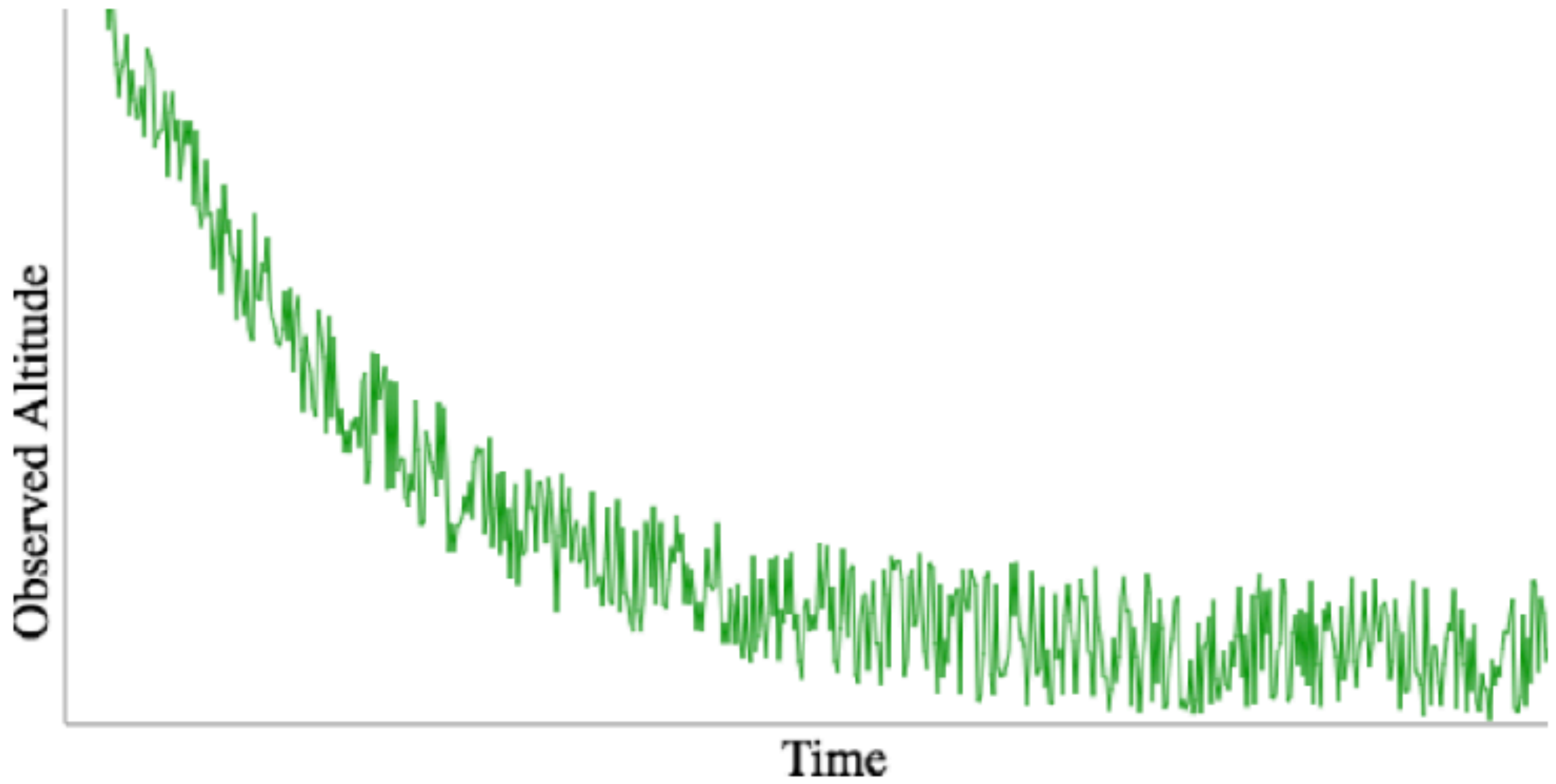
2. Dealing With Noise

$$\textbf{obs\_altitude}_{\textbf{current\_time}} = \textbf{altitude}_{\textbf{current\_time}} + \textbf{noise}_{\textbf{current\_time}}$$

5%



20%



3. Putting it Together

$$\mathbf{altitude_{current\_time} = 0.98*altitude_{previous\_time}}$$

$$\mathbf{obs\_altitude_{current\_time} = altitude_{current\_time} + noise_{current\_time}}$$

$$x_k = ax_{k-1}$$

$$z_k = x_k + v_k$$

x_k is current state of system
a is some constant
z_k is our current observation
v_k is the current noise measurement

3. Putting it Together

$$\textbf{altitude}_{\text{current\_time}} = \textbf{0.98*altitude}_{\text{previous\_time}} + \boxed{\textbf{turbulence}_{\text{current\_time}}}$$

$$\textbf{obs\_altitude}_{\text{current\_time}} = \textbf{altitude}_{\text{current\_time}} + \textbf{noise}_{\text{current\_time}}$$

$$x_k = ax_{k-1} + \boxed{w_k}$$

$$z_k = x_k + v_k$$

x_k is current state of system
a is some constant
z_k is our current observation
v_k is the current noise measurement
w_k is the process noise

4. State Estimation

x_k is current state of system
a is some constant
z_k is our current observation
v_k is the current noise measurement
w_k is the process noise

$$x_k = a x_{k-1} + w_k$$

$$z_k = x_k + v_k \longrightarrow x_k = z_k - v_k$$

$$\boxed{\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})}$$

Kalman's insight

g is a "gain"

g=0

$$\hat{x}_k = \hat{x}_{k-1}$$

g=1

$$\hat{x}_k = z_k$$

5. Computing the Gain

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

5. Computing the Gain

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

$$z_k = x_k + v_k$$

5. Computing the Gain

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

$$z_k = x_k + v_k$$

$$g_k = \frac{p_{k-1}}{p_{k-1} + r}$$

r is how noisy the output is
p_k is a prediction error

$$p_k = (1 - g_k)p_{k-1}$$

5. Computing the Gain

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

$$z_k = x_k + v_k$$

$$g_k = \frac{p_{k-1}}{p_{k-1} + r}$$

r is how noisy the output is
p_k is a prediction error

$$p_k = (1 - g_k)p_{k-1}$$

$p_{k-1} = 0$, then $g_k = 0$

$p_{k-1} = 1$, then $g_k = 1/(1+r)$

5. Computing the Gain

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

$$z_k = x_k + v_k$$

$$g_k = \frac{p_{k-1}}{p_{k-1} + r}$$

r is how noisy the output is
p_k is a prediction error

$$p_k = (1 - g_k)p_{k-1}$$

$p_{k-1} = 0$, then $g_k = 0$

$p_{k-1} = 1$, then $g_k = 1/(1+r)$

$g_k = 0$, then $p_k = p_{k-1}$

$g_k = 1$, then $p_k = 0$

6. Prediction and Update

$$x_k = a x_{k\text{-}1}$$

what happened to a?

$$\hat{x}_k = \hat{x}_{k-1} + g_k(z_k - \hat{x}_{k-1})$$

prediction phase of Kalman Filter

$$\hat{x}_k = a\hat{x}_{k-1}$$
$$p_k = a p_{k-1} a$$

7. Running the Filter

PREDICT

$$\hat{x}_k = a\hat{x}_{k-1}$$

$$p_k = ap_{k-1}a$$

UPDATE

$$g_k = \frac{p_{k-1}}{p_{k-1} + r}$$

$$\hat{x}_k \leftarrow \hat{x}_k + g_k(z_k - \hat{x}_k)$$

$$p_k \leftarrow (1 - g_k)p_k$$

7. Running the Filter

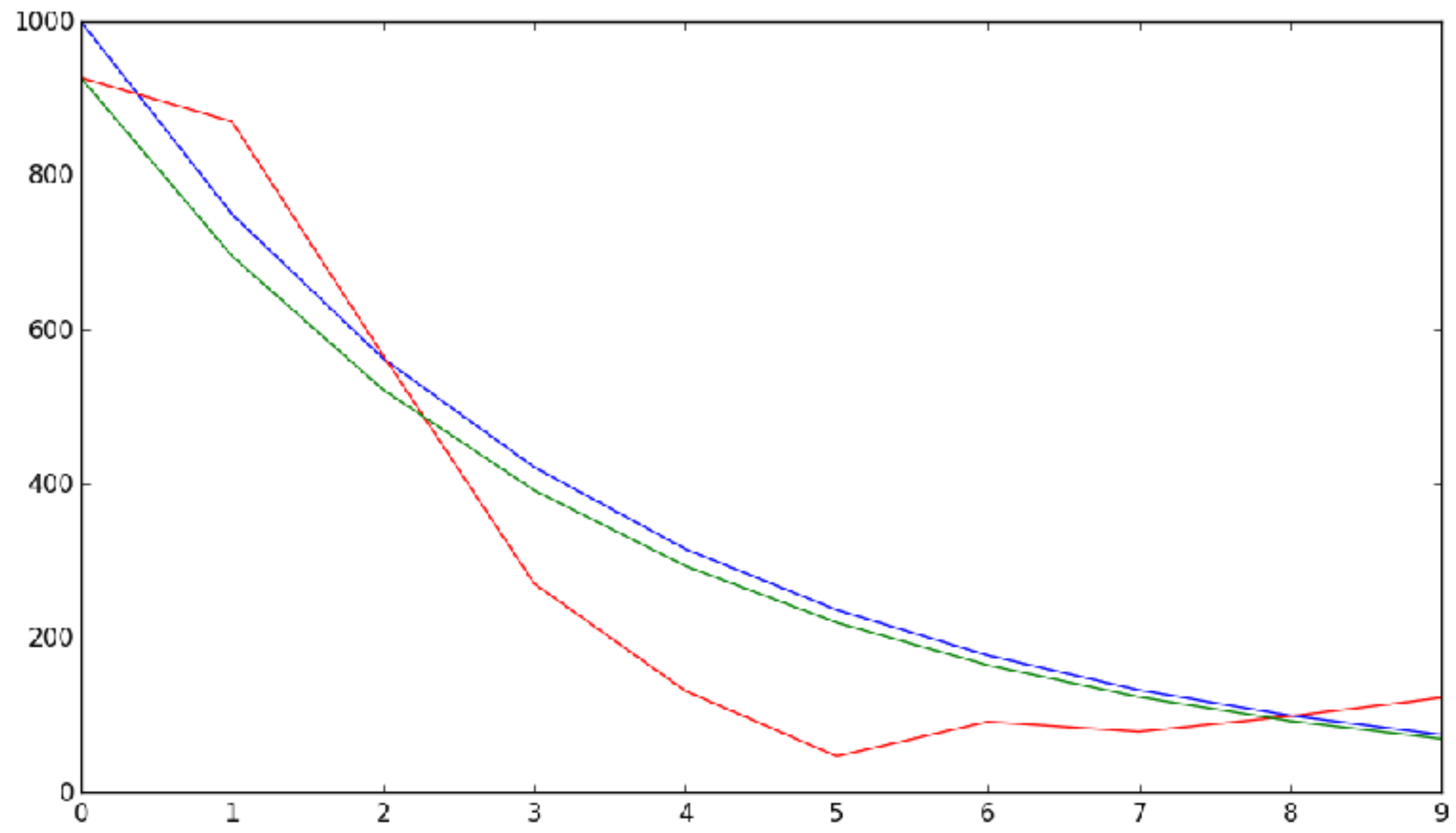
```
import numpy as np
import matplotlib.pyplot as plt

k = np.linspace(0,9,10)
x = [1000, 750, 563, 422, 316, 237, 178, 133, 100, 75]
z = [927,870,567,271,132,47,92,79,99,123]

r = 200
a = 0.75
x_est = [z[0]]
p_est = [1]
g_est = [0]

x_hat = z[0]
p = 1

for i in range(len(k)-1):
    x_hat = a*x_hat
    p = a*p*a

    g = p/(p+r)
    x_hat = x_hat + g*(z[i]-x_hat)
    p = (1-g)*p

    p_est.append(p)
    x_est.append(x_hat)
    g_est.append(g)

fig, ax = plt.subplots()
ax.plot(k,x,'b',k,z,'r', k, x_est, 'g')
plt.show()
```

$$\hat{x}_k = a\hat{x}_{k-1}$$
$$p_k = a p_{k-1} a$$

$$g_k = \frac{p_{k-1}}{p_{k-1} + r}$$

$$\hat{x}_k \leftarrow \hat{x}_k + g_k(z_k - \hat{x}_k)$$

$$p_k \leftarrow (1 - g_k) p_k$$



8. A More Realistic Model

$$x_k = ax_{k-1}$$
$$z_k = x_k + v_k$$

$$\longrightarrow$$

$$x_k = ax_{k-1} + \boldsymbol{bu_k}$$
$$z_k = \boldsymbol{c}x_k + v_k$$

9. Modifying the Estimates

$$x_k = ax_{k-1} + bu_k$$

$$z_k = cx_k + v_k$$

PREDICT

$$\hat{x}_k = a\hat{x}_{k-1} + bu_k$$
$$p_k = ap_{k-1}a$$

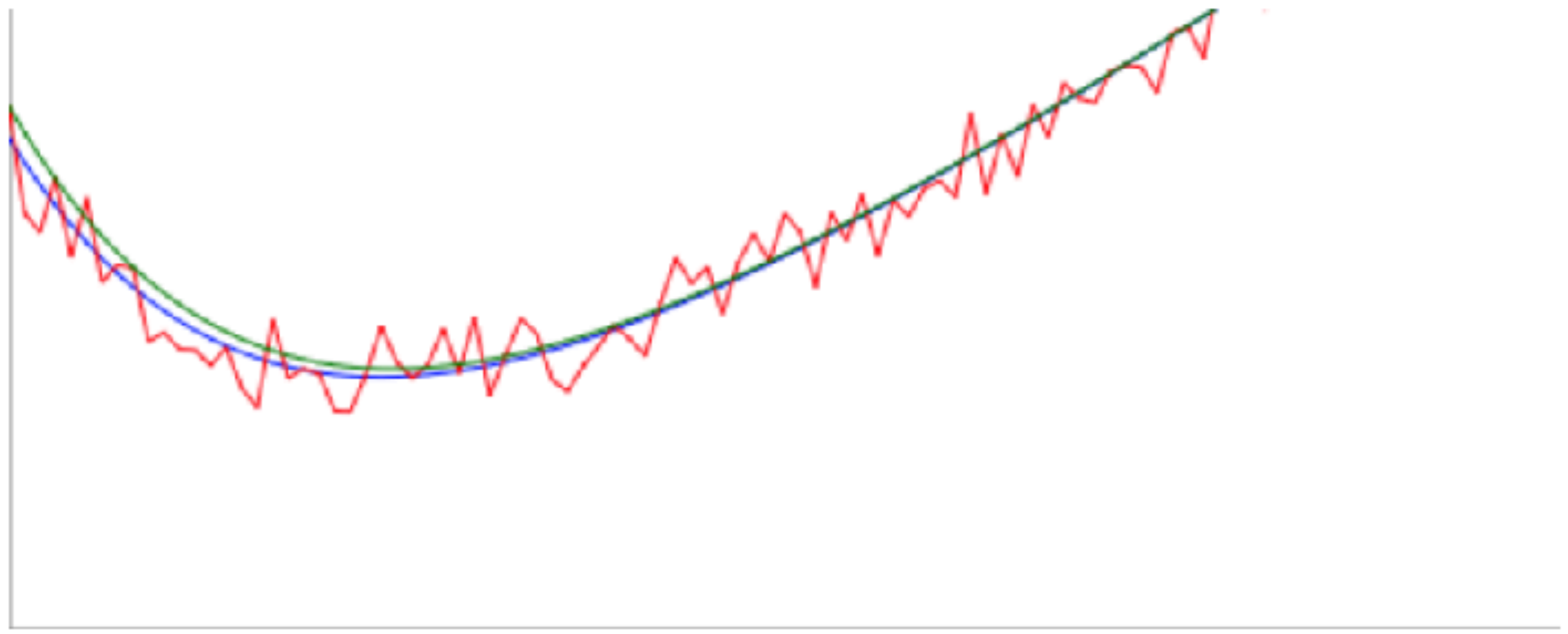
UPDATE

$$g_k = \frac{p_k c}{cp_k c + r}$$

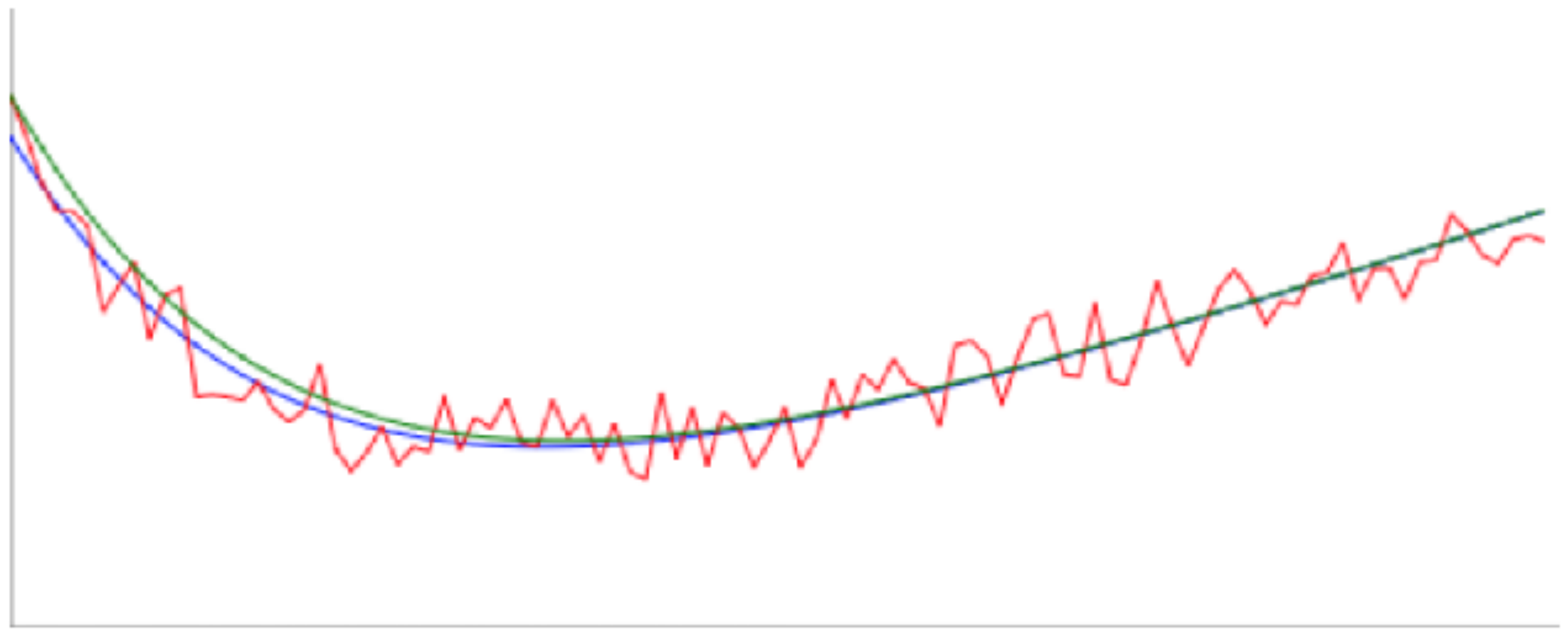
$$\hat{x}_k \leftarrow \hat{x}_k + g_k(z_k - c\hat{x}_k)$$

$$p_k \leftarrow (1 - g_k c)p_k$$

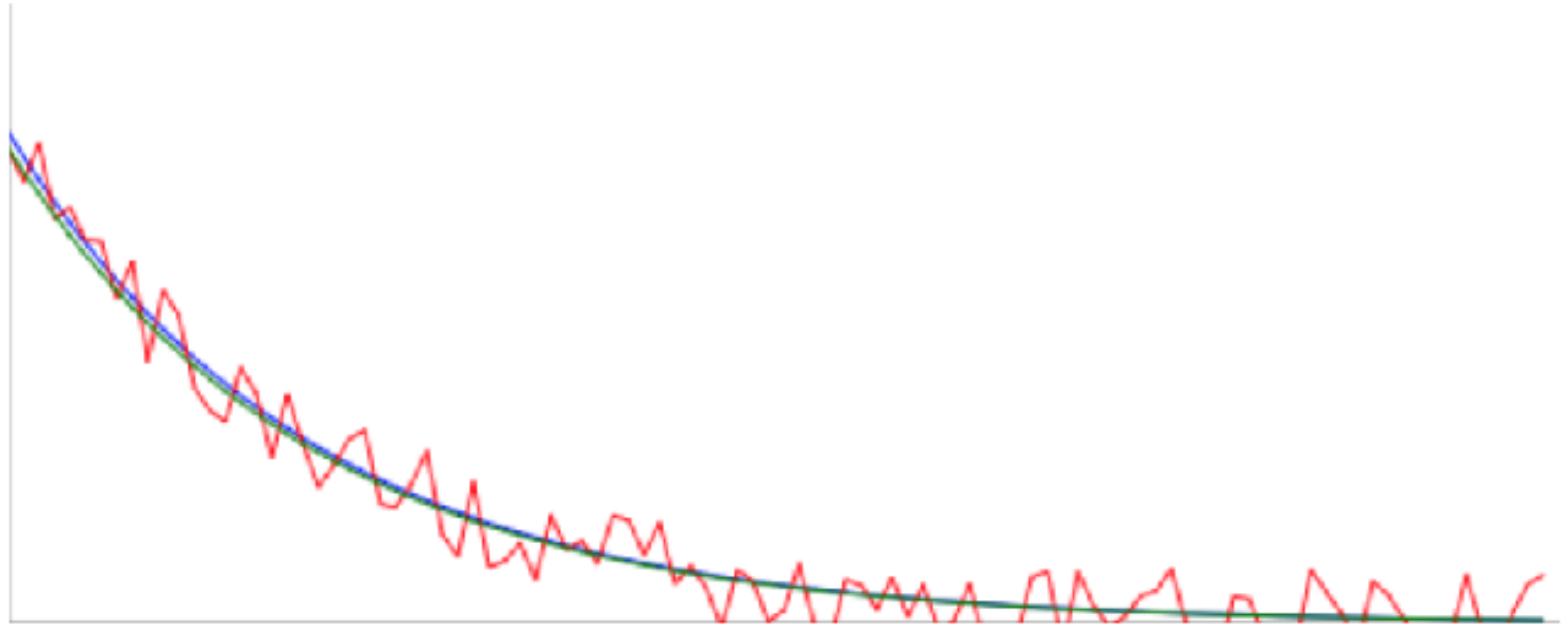
9. Modifying the Estimates



b = 1



b = .5



b = 0

10. Adding Velocity to the System

$$\mathbf{altitude}_{\mathbf{current\_time}} = \mathbf{0.98*altitude}_{\mathbf{previous\_time}}$$

$$x_k = ax_{k-1}$$

$$\mathbf{distance = velocity * time}$$

$$\text{distance}_{\text{current}} = \text{distance}_{\text{previous}} + \text{velocity}_{\text{previous}}*(\text{time}_{\text{current}}-\text{time}_{\text{previous}})$$

$$\text{distance}_{\text{current}} = \text{distance}_{\text{previous}} + \text{velocity}_{\text{previous}}*\text{timestep}$$

11. Linear Algebra

$$distance_{current} = distance_{previous} + velocity_{previous} * timestep$$

$$x_k = a x_{k-1}$$

$$x_k \equiv \begin{bmatrix} distance_k \\ velocity_k \end{bmatrix}$$

11. Linear Algebra

$$\text{distance}_{\text{current}} = \text{distance}_{\text{previous}} + \text{velocity}_{\text{previous}} * \text{timestep}$$

$$x_k = a x_{k-1}$$

$$x_k \equiv \begin{bmatrix} distance_k \\ velocity_k \end{bmatrix} \quad A = \begin{bmatrix} 1 & timestep \\ 0 & 1 \end{bmatrix} \qquad x_k = A x_{k-1}$$

11. Linear Algebra

$$distance_{current} = distance_{previous} + velocity_{previous} * timestep$$

$$x_k = ax_{k-1}$$

$$x_k \equiv \begin{bmatrix} distance_k \\ velocity_k \end{bmatrix} \quad A = \begin{bmatrix} 1 & timestep \\ 0 & 1 \end{bmatrix} \qquad x_k = Ax_{k-1}$$

$$\begin{bmatrix} distance_k \\ velocity_k \end{bmatrix} = \begin{bmatrix} 1 & timestep \\ 0 & 1 \end{bmatrix} \begin{bmatrix} distance_{k-1} \\ velocity_{k-1} \end{bmatrix}$$

11. Linear Algebra

$$distance_{current} = distance_{previous} + velocity_{previous} * timestep$$

$$x_k = a x_{k-1}$$

$$x_k \equiv \begin{bmatrix} distance_k \\ velocity_k \end{bmatrix} \quad A = \begin{bmatrix} 1 & timestep \\ 0 & 1 \end{bmatrix} \qquad x_k = A x_{k-1}$$

$$\begin{bmatrix} distance_k \\ velocity_k \end{bmatrix} = \begin{bmatrix} 1 & timestep \\ 0 & 1 \end{bmatrix} \begin{bmatrix} distance_{k-1} \\ velocity_{k-1} \end{bmatrix}$$

$$\begin{bmatrix} distance_k \\ velocity_k \\ acceleration_k \end{bmatrix} = \begin{bmatrix} 1 & timestep & 0 \\ 0 & 1 & timestep \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} distance_{k-1} \\ velocity_{k-1} \\ acceleration_{k-1} \end{bmatrix}$$

12. Prediction and Update Revisited

MODEL

$$x_k = Ax_{k-1} + Bu_k$$
$$z_k = Cx_k + v_k$$

PREDICT

UPDATE

12. Prediction and Update Revisited

MODEL

$$x_k = Ax_{k-1} + Bu_k$$
$$z_k = Cx_k + v_k$$

PREDICT

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_k$$
$$P_k = AP_{k-1}A^T$$

UPDATE

$$G_k = P_kC^T(CP_kC^T + R)^{-1}$$
$$\hat{x}_k \leftarrow \hat{x}_k + G_k(z_k - C\hat{x}_k)$$
$$P_k \leftarrow (I - C_kC)P_k$$

13. Sensor Fusion Intro

$$z_k = Cx_k + v_k$$

$$\begin{bmatrix} barometer_k \\ compass_k \\ pitot_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} altitude_{k-1} \\ heading_{k-1} \\ airspeed_{k-1} \end{bmatrix}$$

$$\begin{bmatrix} barometer_k \\ compass_k \\ pitot_k \\ gps_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} altitude_{k-1} \\ heading_{k-1} \\ airspeed_{k-1} \end{bmatrix}$$

14. Sensor Fusion Example

$$\hat{x}_k = A\hat{x}_{k-1} = 1 * \hat{x}_{k-1} = \hat{x}_{k-1}$$

$$z_k = Cx_k + v_k = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x_k + v_k$$

14. Sensor Fusion Example

$$\hat{x}_k = A\hat{x}_{k-1} = 1 * \hat{x}_{k-1} = \hat{x}_{k-1}$$

$$z_k = Cx_k + v_k = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x_k + v_k$$

Now have A, C. What is R?
Covariance matrix

$$R = \begin{bmatrix} 0.64 & 0 \\ 0 & 0.64 \end{bmatrix}$$

12. Prediction and Update Revisited

MODEL

$$x_k = Ax_{k-1} + Bu_k$$
$$z_k = Cx_k + v_k$$

PREDICT

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_k$$
$$P_k = AP_{k-1}A^T$$

UPDATE

$$G_k = P_kC^T(CP_kC^T + R)^{-1}$$
$$\hat{x}_k \leftarrow \hat{x}_k + G_k(z_k - C\hat{x}_k)$$
$$P_k \leftarrow (I - C_kC)P_k$$

14. Sensor Fusion Example

MODEL

$$x_k = Ax_{k-1} + Bu_k$$
$$z_k = Cx_k + v_k$$

PREDICT

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_k$$
$$P_k = AP_{k-1}A^T$$

$$x_k = Ax_{k-1} + Bu_k + w_k$$
$$P_k = AP_{k-1}A^T + Q$$

14. Sensor Fusion Example