# Understanding and Applying Discrete Kalman Filtering

June 24, 2016

*Dr. Gamal ElBayoumi*

**Ahmed Ali ElSayed ElTahan**

# Contents

## What is a Kalman Filter and What Can It Do?

A Kalman filter is an optimal estimator - i.e. infers parameters of interest from indirect, inaccurate and uncertain observations. It is recursive so that new measurements can be processed as they arrive. (cf batch processing where all data must be present).

## Optimal in what sense?

If all noise is Gaussian, the Kalman filter minimizes the mean square error of the estimated parameters.

## What if the noise is NOT Gaussian?

Given only the mean and standard deviation of noise, the Kalman filter is the best linear estimator. Non-linear estimators may be better.

## Why is Kalman Filtering so popular?

- Good results in practice due to optimality and structure.
- Convenient form for online real time processing.
- Easy to formulate and implement given a basic understanding.
- Measurement equations need not be inverted.

## Word examples

- Determination of planet orbit parameters from limited earth observations.
- Tracking targets - e.g. aircraft, missiles using RADAR.
- Robot Localization and Map building from range sensors/ beacons.

## Why use the word "Filter"?

The process of finding the "best estimate" from noisy data amounts to "filtering out" the noise. However a Kalman filter also does not just clean up the data measurements, but also projects these measurements onto the state estimate.

## Formulating a Kalman Filter Problem

We require discrete time linear dynamic system description by vector difference equation with additive white noise that models unpredictable disturbances.

**STATE DEFINITION**: the state of a deterministic dynamic system is the smallest vector that summarises the past of the system in full. Knowledge of the state allows theoretically prediction of the future (and prior) dynamics and outputs of the deterministic system in the absence of noise.

## In Summary

Kalman filter is an algorithm for *optimal recursive* estimation of the states of the the system and it needs:

- Initial state estimate and error contrivance.
- Noisy measurements with known properties.
- System state space model.

## Process and Measurement Models

$$x_{k+1} = \phi_k x_k + B_k u_k + w_k \quad (1)$$
$$z_k = H_k x_k + v_k \quad (2)$$

## Definitions and Dimensions of Variables

| | | |
|---|---|---|
| $x_k$ | (n × 1) | State vector |
| $u_k$ | (l × 1) | Input-Control vector |
| $w_k$ | (n × 1) | Process noise vector |
| $z_k$ | (m × 1) | Observation vector |
| $v_k$ | (n × 1) | Measurement noise vector |
| $\phi_k$ | (n × 1) | State transition matrix |
| $B_k$ | (n × l) | Input-Control vector |
| $H_k$ | (m × n) | Observation matrix |
| $Q_k$ | (n × n) | Process noise covariance matrix |
| $R_k$ | (m × m) | Measurement noise covariance matrix |

## Noise

- $w_k$ is a zero mean white Gaussian noise vector at $t_k$

$$E[w_k {w_i}^T] = \begin{cases} Q_k & ; i = k \\ 0 & ; i \neq k \end{cases}$$

- $w_k$ is a zero mean white Gaussian noise vector at $t_k$

$$E[v_k {v_i}^T] = \begin{cases} R_k & ; i = k \\ 0 & ; i \neq k \end{cases}$$

## Notations

$\hat{}$ ≡ estimate.
$\sim$ ≡ perturbation.
$\hat{x}_k^-$ ≡ a prior estimate of $x_k$ (before the measurements at $t_k$).
$\hat{x}_k^+$ ≡ a posterior estimate of $x_k$ (after the measurements at $t_k$).

## Kalman Filter Loop

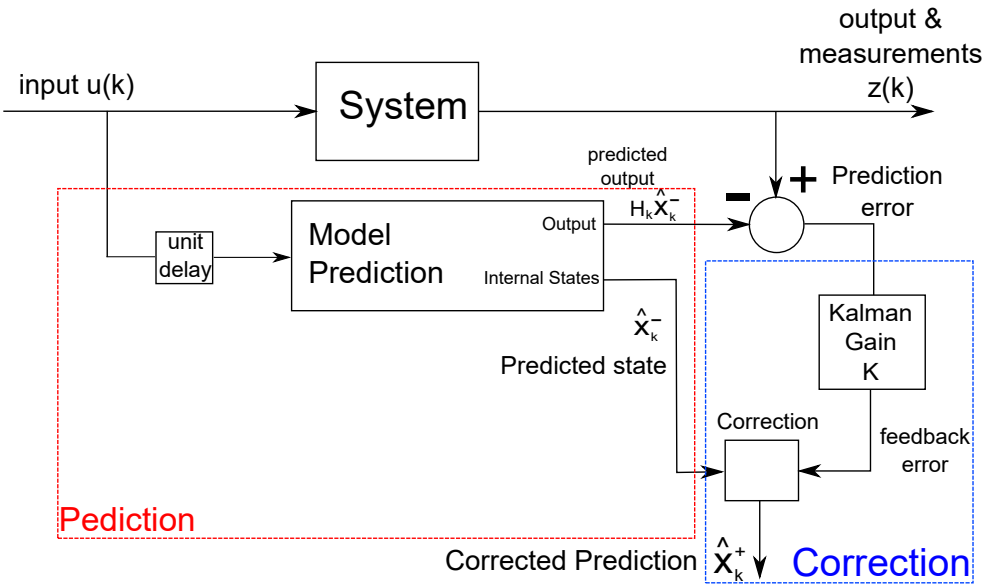


Figure 1: Kalman Filter Loop Schematic

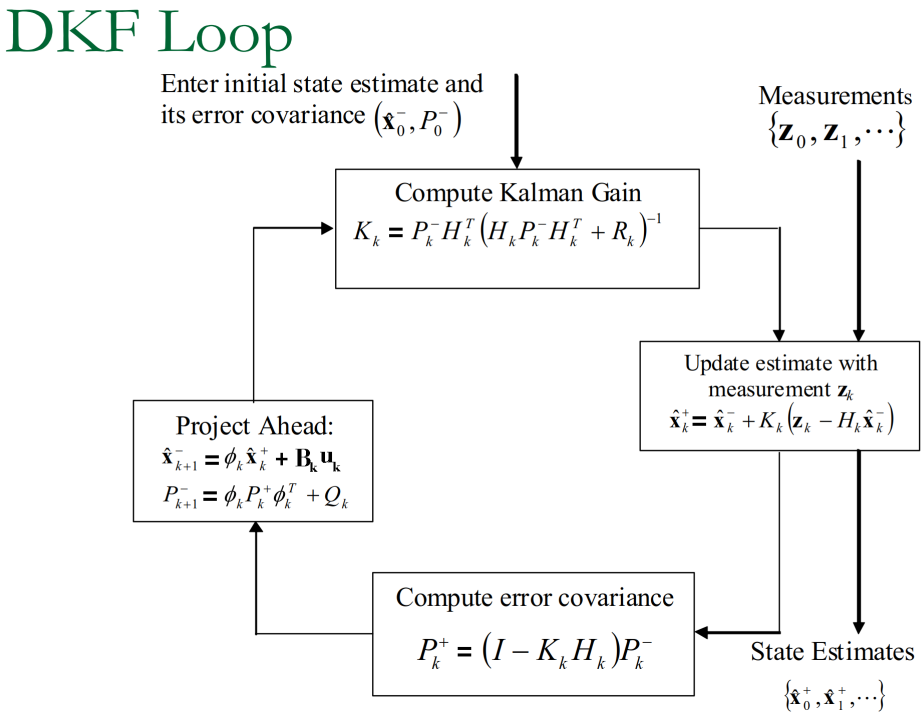


Figure 2: Kalman Filter Loop in Details

## Falling Body Example

Consider a object falling and a constant gravitational field. Let $y(t)$ denote the height of the object, then

$$\ddot{y}(t) = -g \tag{3}$$
$$\Rightarrow \dot{y}(t) = \dot{y}(t_0) - g(t - t_0) \tag{4}$$
$$\Rightarrow y(t) = y(t_0) + \dot{y}(t_0)(t - t_0) - \frac{g}{2}(t - t_0)^2 \tag{5}$$

As a discrete time system with time increment of $t - t_0 = 1$

$$y(k+1) = y(k) + \dot{y}(k) - \frac{g}{2} \tag{6}$$

the height $y(k+1)$ depends on the previous velocity and height at time k.

We can define the state as

$$x(k) = \begin{bmatrix} y(k) & \dot{y}(k) \end{bmatrix}$$

and then the state equation becomes

$$x(k+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} -0.5 \\ -1 \end{bmatrix} g \tag{7}$$
$$= Ax(k) + Bu(k) \tag{8}$$

Assuming we observe or measure the height of the ball directly. The measurement equation is:

$$z(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) + v(k) \tag{9}$$
$$= Hx(k) + v(k) \tag{10}$$

The variance of $v(k)$ needs to be known for implementing a Kalman filter.

Given the initial state and covariance, we have sufficient information to find the optimal state estimate using the Kalman filter equations.

The following Data are used to simulate the system.

| **Parameter** | **Value** |
|---|---|
| n | 2 |
| m | 1 |
| l | 2 |
| $K_{final}$ | 42 |
| g | 9.81 |
| Q | 100 |
| R | 2e5 |

A MATLAB code has been used for the simulation and the results are given in Fig.3 and Fig.4

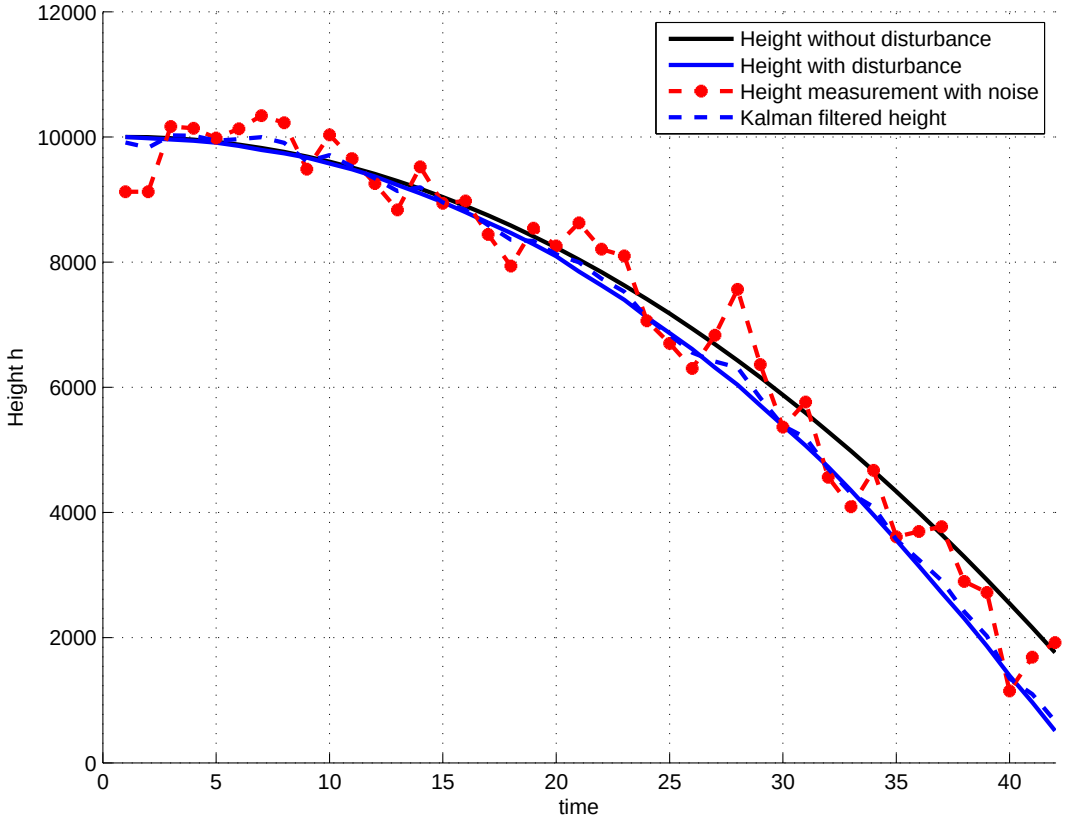


Figure 3: Position Estimate

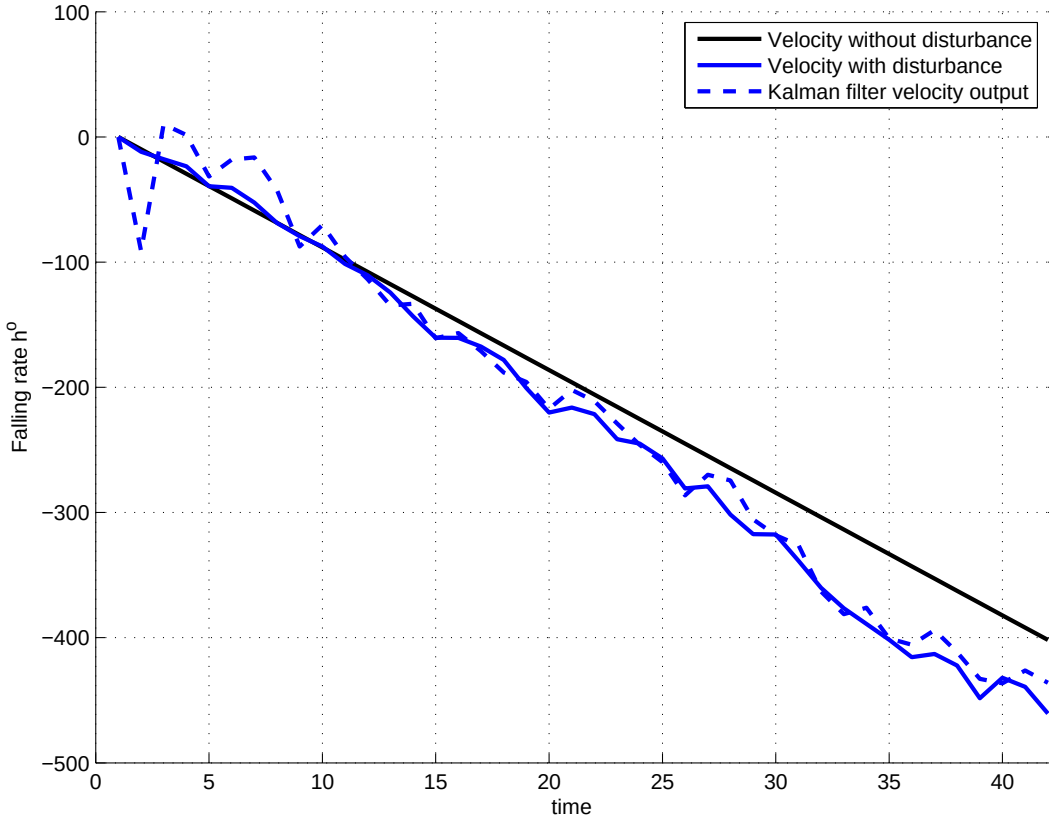


Figure 4: Velocity Estimate

MATLAB code

```
clc; clear all; close all;

%% Define Parameters
n = 2;                  % number of states
m = 1;                 % number of measurements
K_final =  42;    % number of iterations
t = 1:K_final;     % iteration vector

phi = [1 1; 0 1]; % State transition matrix
B = [-0.5; -1];    % Input-Control vector
g = 9.8;             % gravity
H = [1 0];          % measurements - output vector
In = eye(n);
Im = eye(m);
q = 100;             % Variance of the the system disturbance output
r = 200000;       % Variance of the the system measurements output
v = random('Normal', 0, sqrt(r), m,K_final);    % disturbance noise
w = random('Normal', 0, sqrt(q), n,K_final);   % measurements noise

R = v*v'*Im;        % Measurement noise covariance matrix
Q = w*w'*In;       % Process noise covariance matrix

w = mat2cell(w, [n], [ones(1,K_final)]);
v = mat2cell(v, [m], [ones(1,K_final)]);

```

```
xhat_p = cell(1, K_final);          % posterior estimate of x_k
xhat_m = cell(1, K_final);         % prior estimate of x_k
z = cell(1, K_final);                    % measurments
x_pure = cell(1,K_final);           % process output without noise
x = cell(1,K_final);                     % variable used to collect measurements

% Initialization
xhat_m{1}(1,1) = 9000;
xhat_m{1}(2,1) = 0;
x_pure{1}(1,1) = 10000;
x_pure{1}(2,1) = 0;
x{1}(1,1) = 10000;
x{1}(2,1) = 0;
z{1} = H*xhat_m{1}+v{1};

K = cell(1, K_final);
P_m = cell(1, K_final);
P_p = cell(1, K_final);
alpha = 1e6;
P_m{1} = alpha*eye(n, n);

%% Collecting Measurements with Process Disturbance
for k = 1 : K_final-1
    x_pure{k+1} = phi*x_pure{k}+B*(g);
    x{k+1} = phi*x{k}+B*(g)+w{k};
    z{k+1} = H*x{k} + v{k};
end

%% Kalman Algorithm
for k = 1 : K_final
    K{k} = P_m{k}*H'*inv(H*P_m{k}*H'+R);
    xhat_p{k} = xhat_m{k}+K{k}*(z{k}-H*xhat_m{k});
    P_p{k} = (In - K{k}*H)*P_m{k};
    P_m{k+1} = phi*P_p{k}*phi'+Q;
    xhat_m{k+1} = phi*xhat_p{k}+B*(g);
end

%% Extracting Data
State1 = cell2mat(x_pure);
pos1 = State1(1,:);
vel1 = State1(2,:);
State2 = cell2mat(xhat_m);
pos2 = State2(1,1:end-1);
vel2 = State2(2,1:end-1);
measure = cell2mat(z);
measure = measure(1,:);
State3 = cell2mat(xhat_p);
pos3 = State3(1,:);
vel3 = State3(2,:);

%% Plotting
hh = figure;
hold all
plot(t, pos1 ,'LineWidth', 2, 'color', 'black')
plot(t, pos2, 'LineWidth', 2, 'color', 'blue')
plot(t, measure, '--*', 'LineWidth', 2, 'color', 'red')
plot(t, pos3, '--', 'LineWidth', 2, 'color', 'blue')
xlabel('time')
ylabel('Height h')
xlim([0 K_final])
grid on
legend('Height without disturbance', 'Height with disturbance', 'Height measurement with ...
```

```
        noise', 'Kalman filtered height')
print(hh, '-dpng', '-r1000', 'pos')
print(hh, '-depsc', 'pos')

hh = figure;
hold all
plot(t, vel1,'LineWidth', 2, 'color', 'black')
plot(t, vel2,'LineWidth', 2, 'color', 'blue')
plot(t, vel3 ,'--','LineWidth', 2, 'color', 'blue')
xlabel('time')
ylabel('Falling rate h^o')
xlim([0 K_final])
grid on
legend('Velocity without disturbance', 'Velocity with disturbance', 'Kalman filter ...
    velocity output')
print(hh, '-dpng', '-r1000', 'vel')
print(hh, '-depsc', 'vel')
```

## References

1. Brown, Robert Grover., Patrick Y. C. Hwang, and Robert Grover. Brown. Introduction to Random Signals and Applied Kalman Filtering. New York: J. Wiley, 1992. Print