$$
\begin{cases}
\dot{r}_x = v_x \\
\dot{r}_y = v_y \\
\dot{v}_x = 0 \\
\dot{v}_y = -g + \dfrac{\rho v_y^2}{2m\beta} \\
\Delta \dot{h} = 0 \\
\Delta \dot{\beta} = 0 \\
\Delta \dot{\rho}_0 = 0 \\
\Delta \dot{k}_p = 0
\end{cases}
$$

$$
F = \begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\dfrac{\alpha}{\hat{k}_p} & 0 & \dfrac{2\alpha}{\hat{v}_y} & 0 & -\dfrac{\alpha}{\hat{\beta}} & \dfrac{\alpha}{\hat{\rho}_0} & \dfrac{\hat{\gamma}_y}{\hat{k}_p^2}\alpha \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
\alpha = \frac{\hat{\rho}\hat{v}_y^2}{2m\hat{\beta}}
$$

$$
\phi_{k-1} = I + F\Delta t = \begin{bmatrix}
1 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & \Delta t & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -\dfrac{\alpha}{k_p} & 0 & 1 + \dfrac{2\alpha}{v_y}\Delta t & 0 & -\dfrac{\alpha}{\beta}\Delta t & \dfrac{\alpha}{\rho_0}\Delta t & \dfrac{\gamma_y}{k_p^2}\alpha \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

$$u_{k-1} = \int_{t_{k-1}}^{t_k} \phi(t_k, \tau) G(\tau) u(\tau) d\tau = -g \begin{bmatrix} 0 \\ \frac{1}{2}\Delta t \\ 0 \\ \Delta t + \frac{\alpha}{v_y}\Delta t^2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$Q_{k-1} = \int_{t_{k-1}}^{t_k} \phi(t_k, \tau) Q_s(\tau) \phi^T(t_k, \tau) d\tau$$

$$Q_s(\tau) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{s1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{s2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{s3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_{s4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{s5} \end{bmatrix}$$

$$Q_{k-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & Q_{42} & 0 & Q_{44} & 0 & Q_{46} & Q_{47} & Q_{48} \\ 0 & 0 & 0 & 0 & Q_{55} & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_{64} & 0 & Q_{66} & 0 & 0 \\ 0 & 0 & 0 & Q_{74} & 0 & 0 & Q_{77} & 0 \\ 0 & 0 & 0 & Q_{84} & 0 & 0 & 0 & Q_{88} \end{bmatrix}$$

$$Q_{42} = (\frac{1}{2}\square t^2 + \frac{2}{3}\frac{\alpha}{v_y}\Delta t^3)q_{s1}$$

$$Q_{44} = (1 + \frac{2\alpha}{v_y}\Delta t)\Delta t \cdot q_{s1} + \ldots$$

$$Q_{46} = -\frac{\alpha}{2\beta}q_{s3}\Delta t^2$$

$$Q_{47} = -\frac{\alpha}{2\rho_0}q_{s4}\Delta t^2$$

$$Q_{48} = -\frac{\gamma_y \alpha}{2k_p^2}\Delta t^2$$

$$Q_{55} = q_{s2}\Delta t$$

$$Q_{64} = -\frac{\alpha}{2\beta}\Delta t^2$$

$$q_{66} = q_{s3}\Delta t$$

$$q_{74} = \frac{\alpha}{2\rho_0}q_{s4}\Delta t^2$$

$$q_{77} = q_{s4}\Delta t$$

$$q_{84} = \frac{\gamma_y \alpha}{k_p^2}q_{s5}\Delta t^2$$

$$q_{88} = q_{55}\Delta t$$

Matlab code for Environment

```
function [stateout, meas]  = Envrionment(state0, wk, m)

% Initialization
deltat = 0.1;
g=9.836;
persistent state; %variable to save states of environment

if isempty(state)
    state = state0;     %initialize the state using state0 from input.
end

h_ = 2;
bet_ = 150;
rho0_ = 1.225;
kp_ = 9200;
%
ry = state(2);
vy = state(4);
deltah = state(5);
deltabet = state(6);
```

```
deltarho = state(7);
deltakp = state(8);
%
h=h_+deltah;
bet=bet_+deltabet;
rho0=rho0_+deltarho;
kp=kp_+deltakp;
%
alpha = rho0*exp(-ry/kp)*vy*vy/(2*m*bet);
% calculate the state transverse matrix. This is calculated using equation
above and also used in EKF.
phi = [1 0 deltat 0 0 0 0 0; 0 1 0 deltat 0 0 0 0; 0 0 1 0 0 0 0 0; 0 -
alpha*deltat/kp 0 1+rho0*exp(-ry/kp)*vy/(m*bet)*deltat 0 -alpha/bet*deltat
alpha/rho0*deltat ry*alpha*deltat/(kp*kp);
    0 0 0 0 1 0 0 0; 0 0 0 0 0 1 0 0; 0 0 0 0 0 0 1 0; 0 0 0 0 0 0 0 1];
u = [0; -g*deltat*deltat/2; 0; -g*deltat*(1+rho0*exp(-ry/kp)*vy/(2*m*bet)); 0;
0; 0; 0];
%update state and mix with process noise wk which is given by input.You can
set the covariance for process noise in the Simulink.
state = phi*state + u + wk;

% Measurement matrix to be used to output measurements.
H = [1 0 0 0 0 0 0 0; 0 1 0 0 0 0 0 0];
b = [0; -h];
%calculate the measurements.
y = H*state + b;

meas = y;
%output the updated state
stateout = state;
```

Matlab code for EKF

```
function [residual,xhatPreOut, xhatOut] = EXTKALMAN(meas)

persistent P;
persistent xhat

if isempty(P)
    xhat = [0; 1000; 0; 0; 0; 0; 0; 0];
    P = zeros(8);
    P(2)=10;
    P(4)=1;
end

g=9.836;
delta_t=0.5;
m = 1;
h_ = 2;
bet_ = 150;
rho0_ = 1.225;
kp_ = 9200;
%get the states of last step to calculate the state transverse matrix.
ry = xhat(2);
vy = xhat(4);
deltah = xhat(5);
```

```
deltabet = xhat(6);
deltarho = xhat(7);
deltakp = xhat(8);
%
h=h_+deltah;
bet=bet_+deltabet;
rho0=rho0_+deltarho;
kp=kp_+deltakp;
%
alpha = rho0*exp(-ry/kp)*vy*vy/(2*m*bet);
%
% 1. Compute Phi, Q, and R
% calculate phi from the equation above. It is same used in environment.
Phi = [1 0 delta_t 0 0 0 0 0; 0 1 0 delta_t 0 0 0 0; 0 0 1 0 0 0 0 0; 0 -
alpha*delta_t/kp 0 1+rho0*exp(-ry/kp)*vy/(m*bet)*delta_t 0 -alpha/bet*delta_t
alpha/rho0*delta_t ry*alpha*delta_t/(kp*kp);
    0 0 0 0 1 0 0 0; 0 0 0 0 0 1 0 0; 0 0 0 0 0 0 1 0; 0 0 0 0 0 0 0 1];
Q =  diag([0 10 0 1 0 0 0 0 ]);
R =  diag([1 4]);

% 2. Propagate the covariance matrix:
```

$$\mathbf{P}_k^- = \Phi_{k-1}\mathbf{P}_{k-1}^+\Phi_{k-1}^T + \mathbf{Q}_{k-1}$$

```
P = Phi*P*Phi' + Q;

% 3. Propagate the track estimate::
u = [0; -g*delta_t*delta_t/2; 0; -g*delta_t*(1+rho0*exp(-ry/kp)*vy/(2*m*bet));
0; 0; 0; 0];

xhat = Phi*xhat+u;
%output the prio estimates.
xhatPreOut = xhat;
% 4 a). Compute observation estimates:
H = [1 0 0 0 0 0 0 0; 0 1 0 0 0 0 0 0];
b = [0; -h];
```

$\mathbf{h}_k(\hat{\mathbf{x}}_k^-)$

```
yhat = H*xhat + b;

% 4 c).  Compute residual (Estimation Error)
```

$\mathbf{y}_k - \mathbf{h}_k(\hat{\mathbf{x}}_k^-)$

```
residual = meas - yhat;

% 5. Compute Kalman Gain:
```

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\hat{\mathbf{x}}_k^-)\left[\mathbf{H}_k(\hat{\mathbf{x}}_k^-)\mathbf{P}_k^-\mathbf{H}_k^T(\hat{\mathbf{x}}_k^-) + \mathbf{R}_k\right]^{-1}$$

```
W = P*H'*inv(H*P*H'+ R);
%
% % 6. Update post estimates
```

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{h}_k(\hat{\mathbf{x}}_k^-)\right)$$

```
xhat = xhat + W*residual;
%
% % 7. Update Covariance Matrix
```

$$\mathbf{P}_k^+ = \left[\mathbf{I} - \mathbf{K}_k\mathbf{H}_k(\hat{\mathbf{x}}_k^-)\right]\mathbf{P}_k^-$$

```
P = (eye(8)-W*H)*P*(eye(8)-W*H)' + W*R*W';
```

```
 %output the post estimates.
xhatOut = xhat;
```

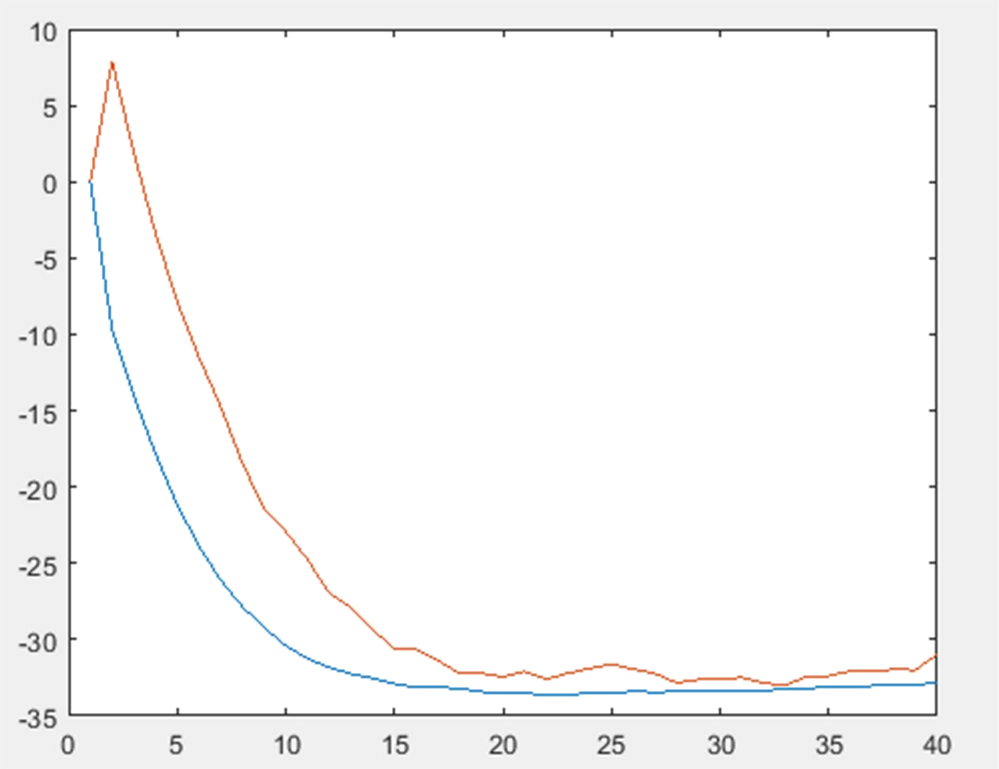

Fig1. ± standard deviation for v_y

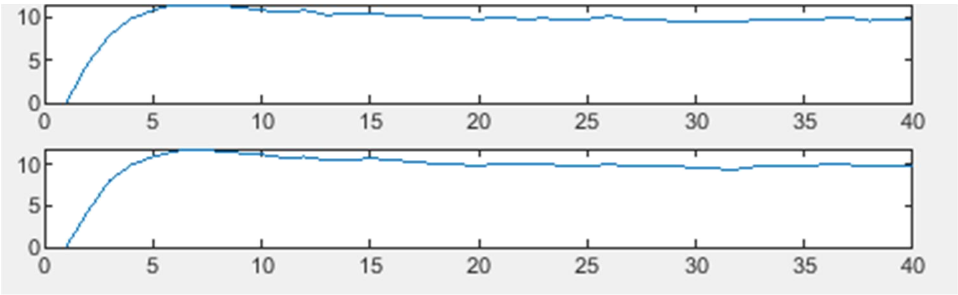

Fig2.covariance for post and prio estimates for v_y

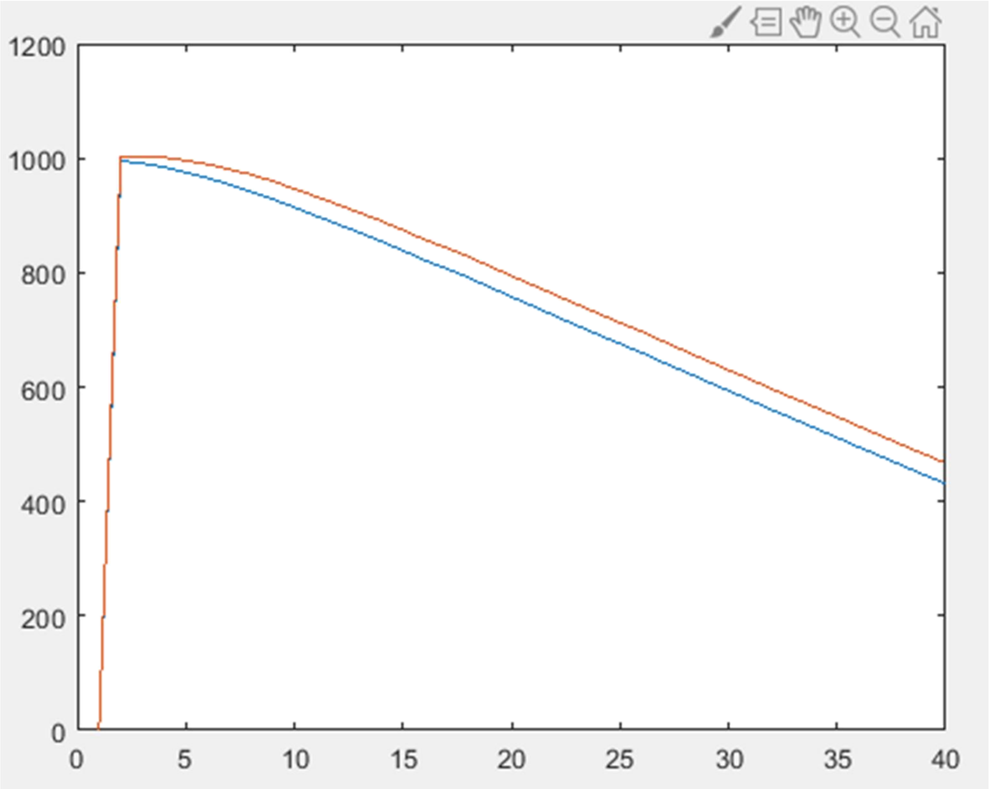

Fig3 ± standard deviation for r_y

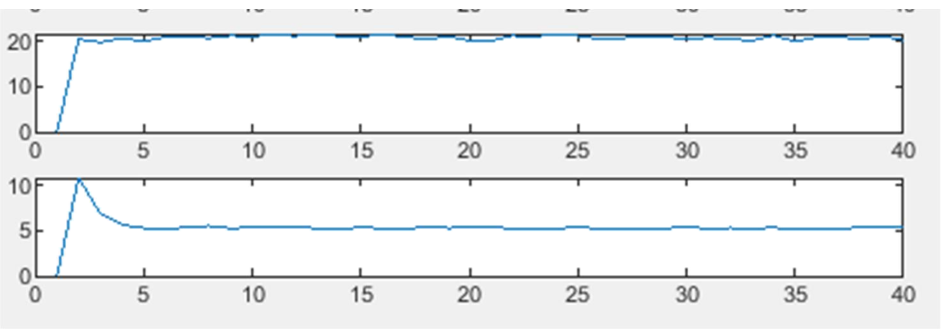

Fig4.covariance for post and prio estimates for r_y