Nonlinear system model for this system is given by as follows.

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

From the definition of states, the nonlinear state

$$\dot{x}(t) = f\big(x(t)\big) + Gu + w(t)$$

where $f\big(x(t)\big) = \begin{pmatrix} v_x \\ v_y \\ 0 \\ \frac{\rho v_y{}^2}{2m\beta} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $G = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$,$u = g$,$\rho = (\overline{\rho_0} + \boldsymbol{\Delta\rho_0})\boldsymbol{e}^{-r_y/(\overline{\boldsymbol{k_p}}+\boldsymbol{\Delta k_p})}$,$\boldsymbol{\beta} = \overline{\boldsymbol{\beta}} + \boldsymbol{\Delta\beta}$**.**

From the nonlinear state model, we can get the following continuous mathematical model of the motion.

$$\dot{x}(t) = Fx(t) + Gu(t) + w(t)$$

Thus discrete-time mathematical model of motion is given by

$$x_k = \Phi_{k-1}x_{k-1} + u_{k-1} + w_{k-1}$$

where $\Phi_{k-1} = e^{F(t_k - t_{k-1})}$

We can get F as following.

$$
F(\hat{x}(t)) = \frac{\partial y(t)}{\partial x(t)}\bigg|_{x(t)=\hat{x}(t)} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -\frac{\alpha}{\hat{k}_p} & 0 & \frac{2\alpha}{\hat{v}_y} & 0 & -\frac{\alpha}{\hat{\boldsymbol{\beta}}} & \frac{\alpha}{\hat{\boldsymbol{\rho}}_{\mathbf{0}}} & \frac{\hat{r}_y\alpha}{\hat{k}_p^2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

where

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

We can get state transverse matrix

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

and

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

The measurement model for this system is given by

$$y(t) = r_m(t) + v(t)$$

,where

$$r_m(t) = \left\| \begin{matrix} r_x \\ r_y - h \end{matrix} \right\| = \sqrt{r_x^2 + \left(r_y - h\right)^2}$$

and $v(t)$ is a measurement noise.

It can be represented in discrete time range as follows.

$$r_m(k) = h_k(x_k) = \sqrt{r_x^2(k) + \left(r_y(k) - (\bar{h} + \Delta h(k))\right)^2}$$

Thus the measurement sensitivity matrix is given by

$$H_k(\hat{x}_{k-1}) = \left.\frac{\partial h_k(x_k)}{\partial x_k}\right|_{x_k = \hat{x}_{k-1}^-} = [H_1\ H_2\ 0\ 0\ H_5\ 0\ 0\ 0]$$

,where

$H_1 = \frac{\hat{r}_x^-}{\sqrt{(\hat{r}_x^-)^2 + \left(\hat{r}_y^- - \hat{h}\right)^2}}$, $H_2 = \frac{\hat{r}_y^- - \hat{h}}{\sqrt{(\hat{r}_x^-)^2 + \left(\hat{r}_y^- - \hat{h}\right)^2}}$, $H_5 = -\frac{\hat{r}_y^- - \hat{h}}{\sqrt{(\hat{r}_x^-)^2 + \left(\hat{r}_y^- - \hat{h}\right)^2}}$,$\hat{h} = \bar{h} + \Delta\hat{h}(k)$

Totally there are for steps to estimate states using EKF.

1) Initialize conditions
   Initialize the initial covariance matrix P_0.
2) Propagate the states.

$$\hat{x}_k^- = \Phi_{k-1}\hat{x}_{k-1}^+ + u_{k-1}$$
$$P_k^- = \Phi_{k-1}P_{k-1}^+\Phi_{k-1}^T + Q_{k-1}$$

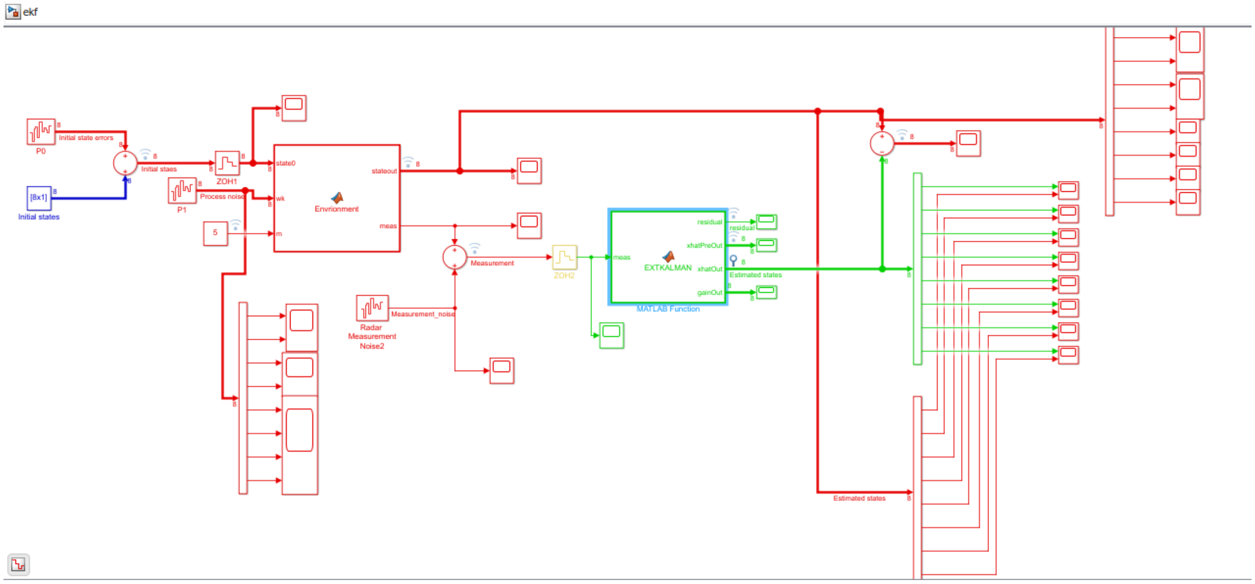
3) Update the states.

$$K_k = P_k^- H_k^T(\hat{x}_k^-)[H_k\ (\hat{x}_k^-)P_k^- H_k^T(\hat{x}_k^-) + R_k]^{-1}$$
$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - h_k(\hat{x}_k^-))$$
$$P_k^+ = [1 - K_k H_k(\hat{x}_k^-)]P_k^-$$

In the truth model, we simulate the initial state estimation errors, process noise and measurement noise.
The initial state estimation error matrix used is given as follow.

$$P_0 = \begin{bmatrix} P_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & P_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & P_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & P_5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & P_6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & P_7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_8 \end{bmatrix}$$

In the simulation we select the parameters as follows.
$P_1 = 100$,$P_2 = 1000$,$P_3 = 0.1$,$P_4 = 0.5$,$P_5 = 0.2$,$P_6 = 15$,$P_7 = 0.1$,$P_8 = 900$,$q_{s1} = 1$, $q_{s2} = 0.1$, $q_{s3} = 0.1$, $q_{s4} = 0$, $q_{s5} = 0$,$R = 4$,m=5kg



Simulink for EKF simulation

Matlab code for Environment

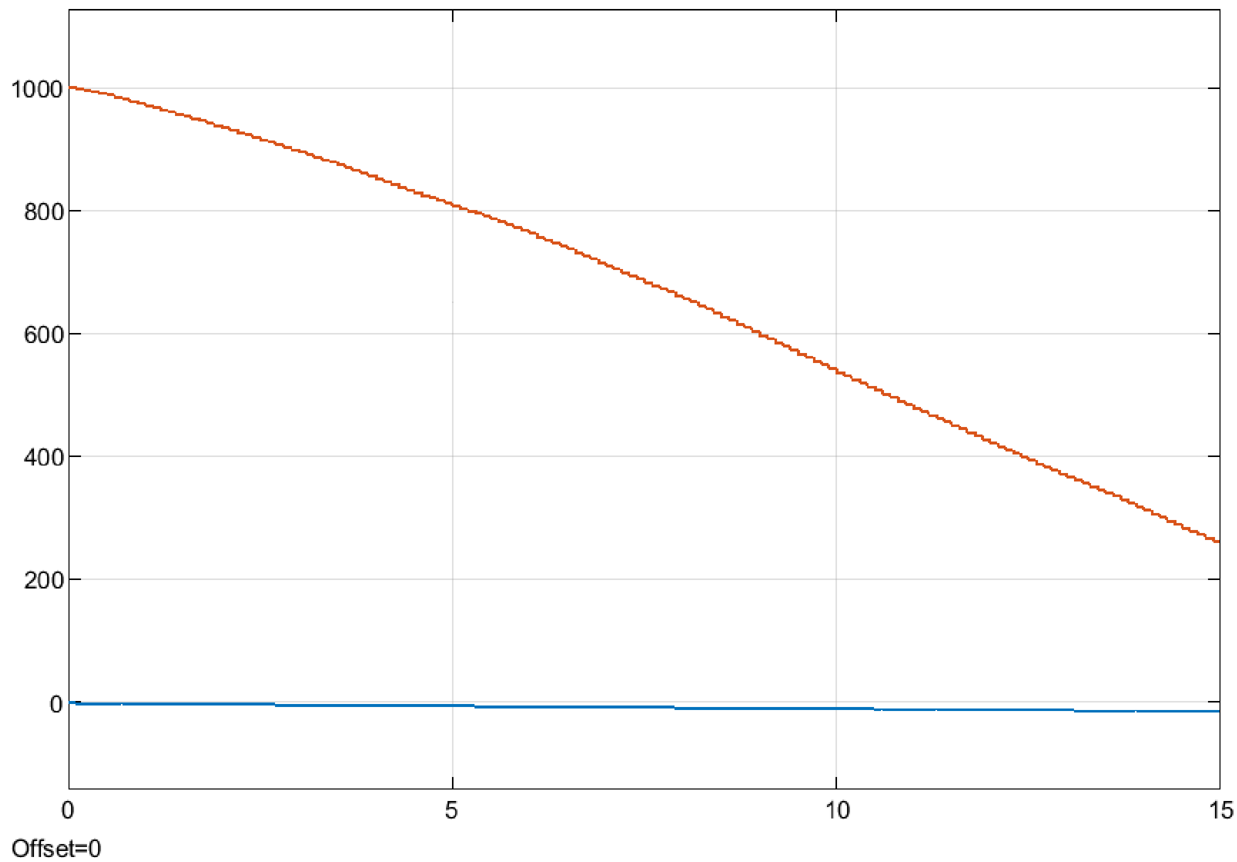
```
function [stateout, meas]  = Envrionment(state0, wk, m)

% 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;

%calculate the measurements.
y=sqrt(rx^2+(ry-h_-deltah)^2);

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

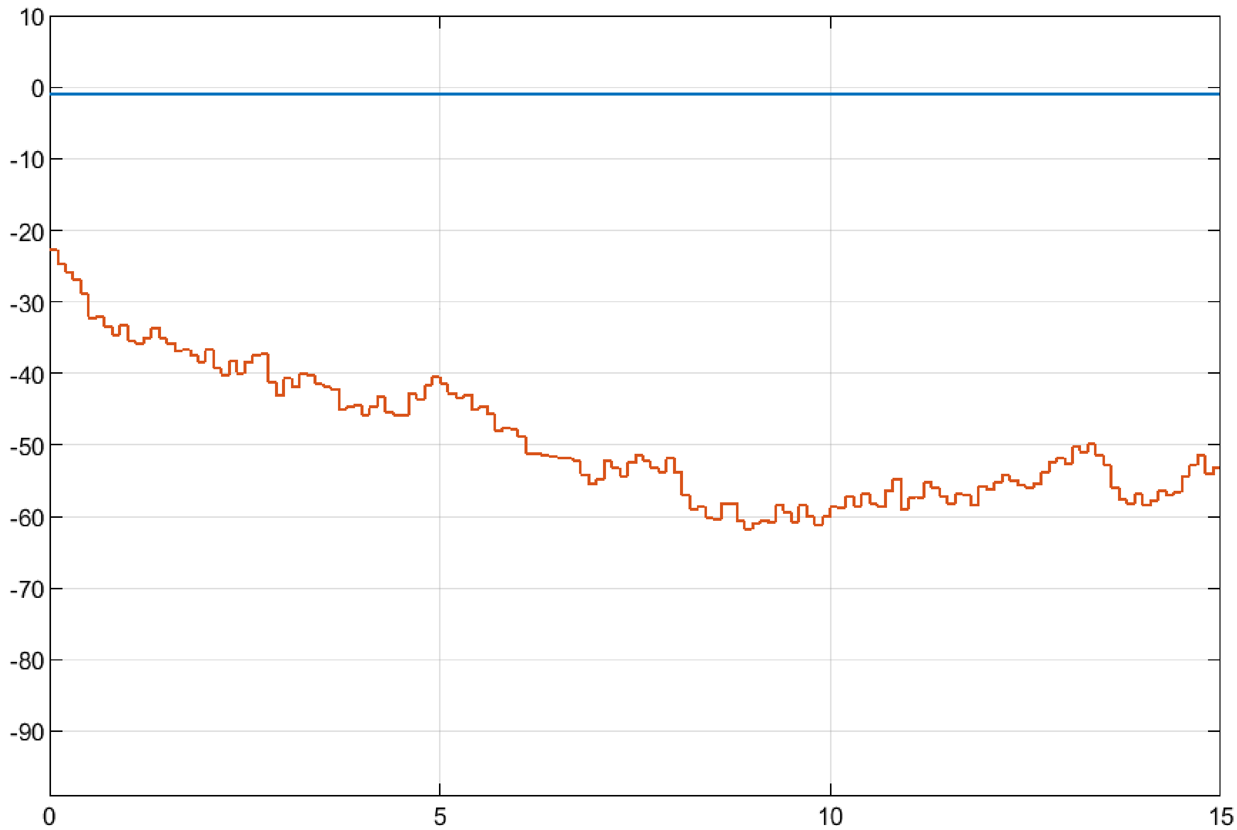
Matlab code for EKF

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

persistent P;
persistent xhat
%step1. Initialize the state
if isempty(P)
    xhat = [0; 1000; 0; 0; 0; 0; 0; 0];
    P = zeros(8);
    P(1)=100;
    P(2)=1000;

end

g=9.836;
delta_t=0.5;
m = 1;
h_ = 2;
bet_ = 150;
rho0_ = 1.225;
kp_ = 9200;
%
ry = xhat(2);
vy = xhat(4);
```

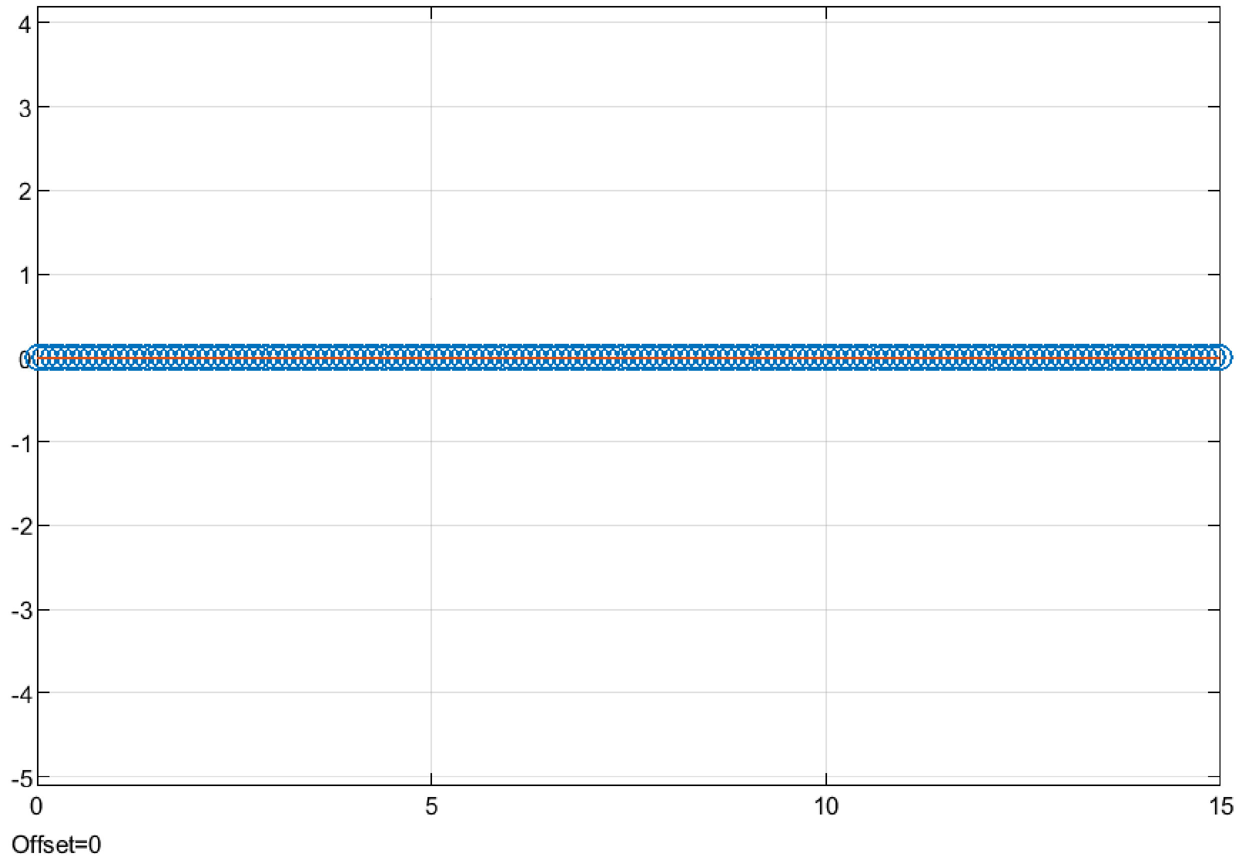
```
deltabet = xhat(6);
deltarho = xhat(7);
deltakp = xhat(8);
%
bet=bet_+deltabet;
rho0=rho0_+deltarho;
kp=kp_+deltakp;
%
alpha = rho0*exp(-ry/kp)*vy*vy/(2*m*bet);
%
%  Compute Phi, Q, and R
Phi = [1 0 delta_t 0 0 0 0 0; 0 1 0 delta_t 0 0 0 0; 0 0 1 0 0 0 0 0; 0 -
alpha*delta_t/kp 0 1+rho0*exp(-ry/kp)*vy/(m*bet)*delta_t 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 0 0 10 0 0 0 0 ]);
R =  diag(4);

%Step 2. 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;
P = Phi*P*Phi' + Q;
% output the prio estimates.
xhatPreOut = xhat;
rx_pre=xhat(1);
ry_pre=xhat(2);
h_pre=h_+xhat(5);

%Step 3: Update the states
%  a). Compute observation estimates:
rm=sqrt(rx_pre^2+(ry_pre-h_pre)^2);
H1=rx_pre/rm;
H2=(ry_pre-h_pre)/rm;
H5=-H2;
H = [H1 H2 0 0 H5 0 0 0];
b = 0;

yhat = H*xhat + b;

% b).  Compute residual (Estimation Error)
residual = meas - yhat;

% c). Compute Kalman Gain:
W = P*H'*inv(H*P*H'+ R);

% d). Update estimate
xhat = xhat + W*residual;

% e). Update Covariance Matrix
P = (eye(8)-W*H)*P*(eye(8)-W*H)' + W*R*W';
%output post estimates.
xhatOut = xhat;
gainOut = W;
```



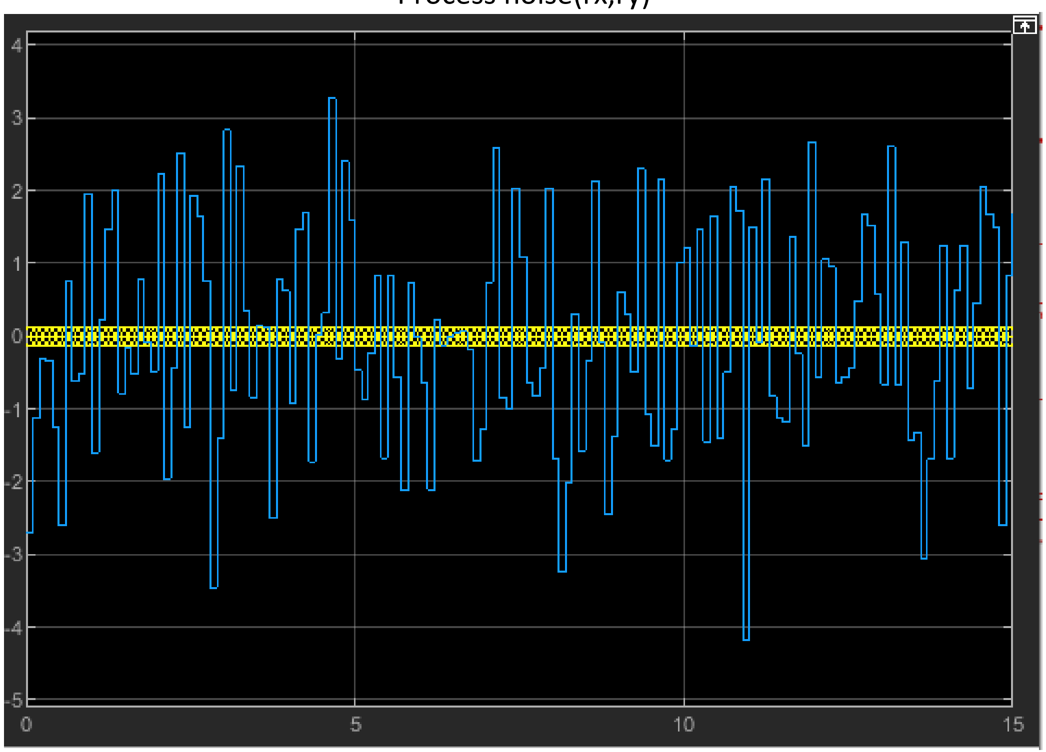
True Position(rx:yello,ry:blue)



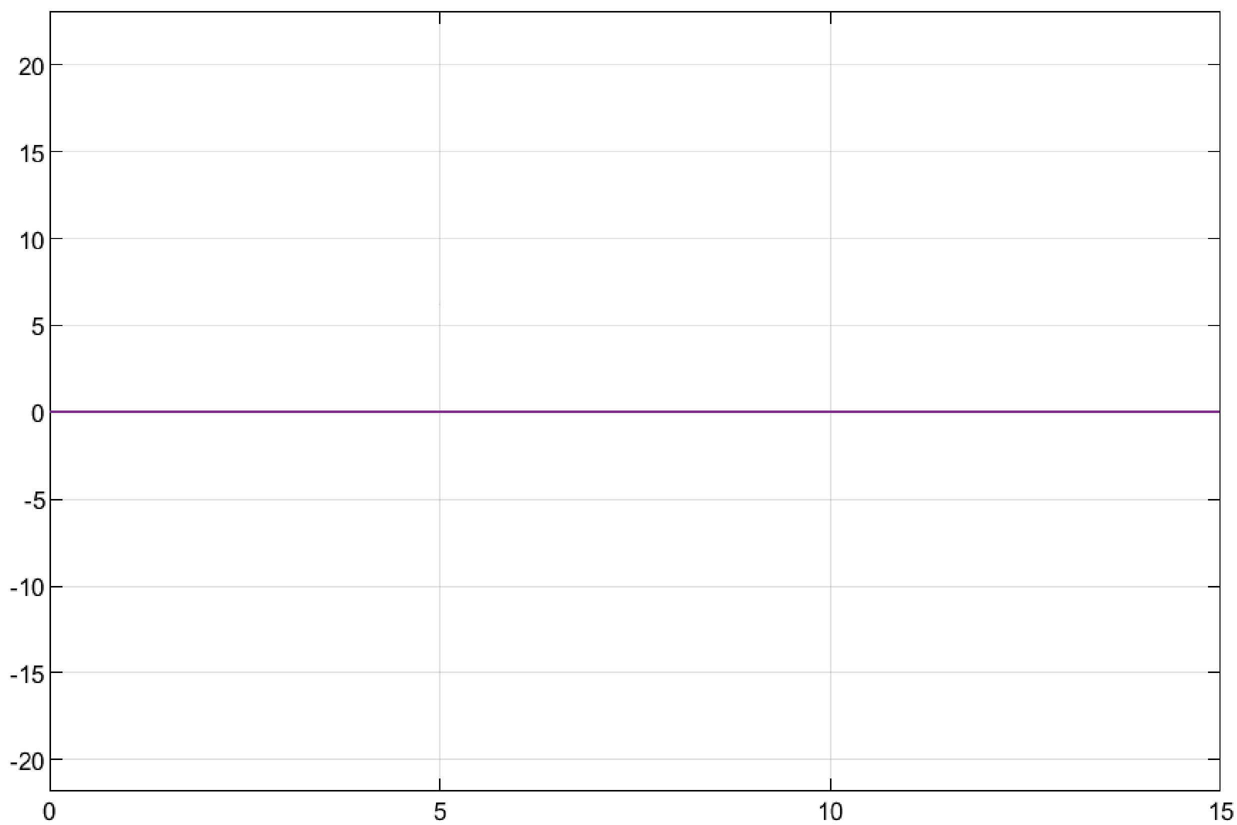
|True Velocity(blue:vx red:vy)



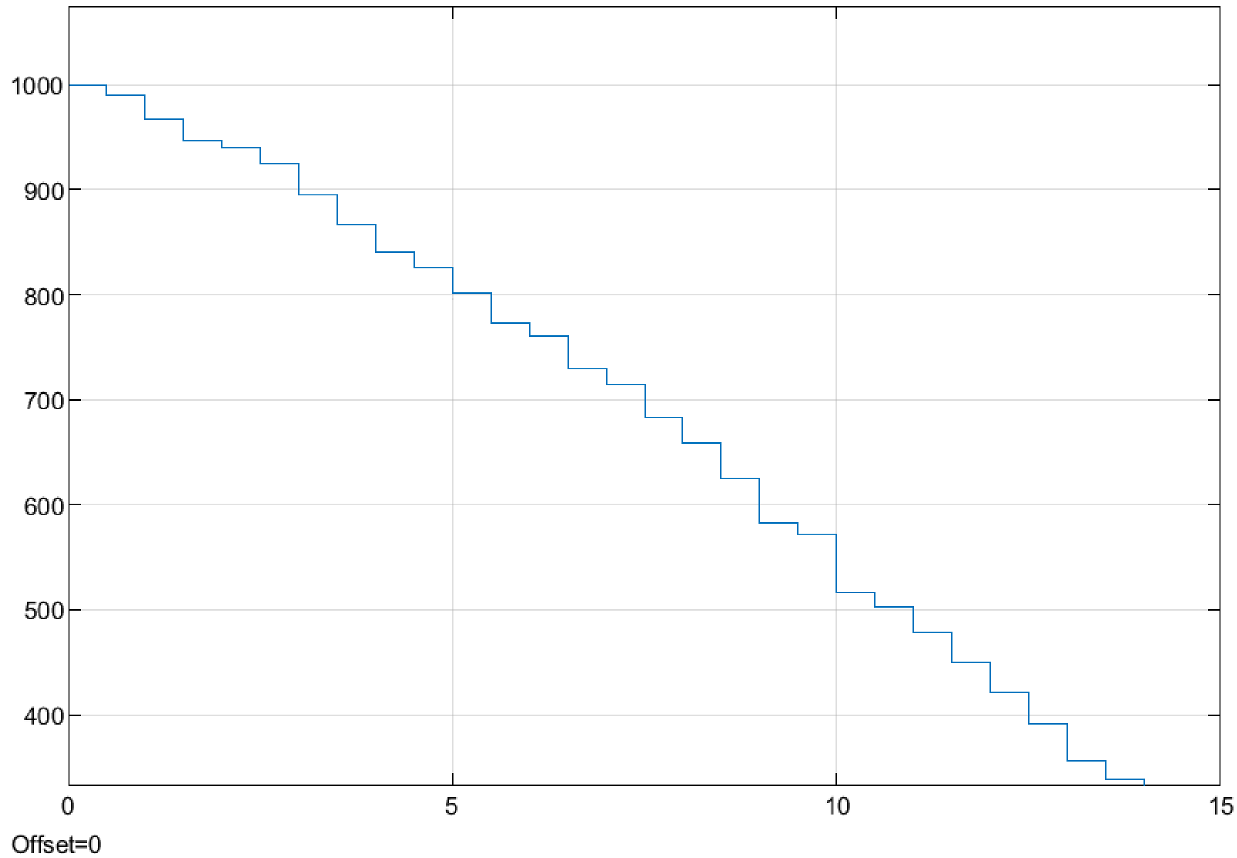
Process noise(rx,ry)



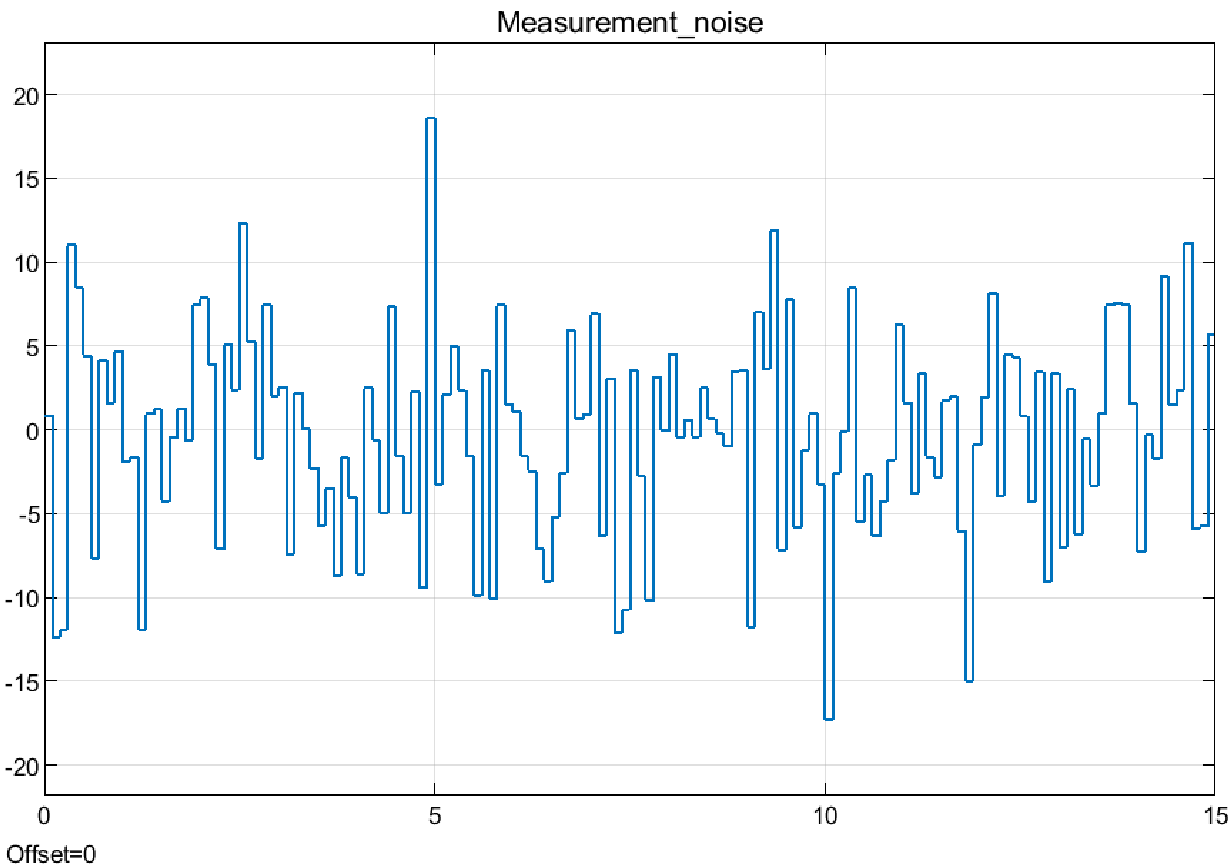
Process noise(vx:yellow, vy:blue)



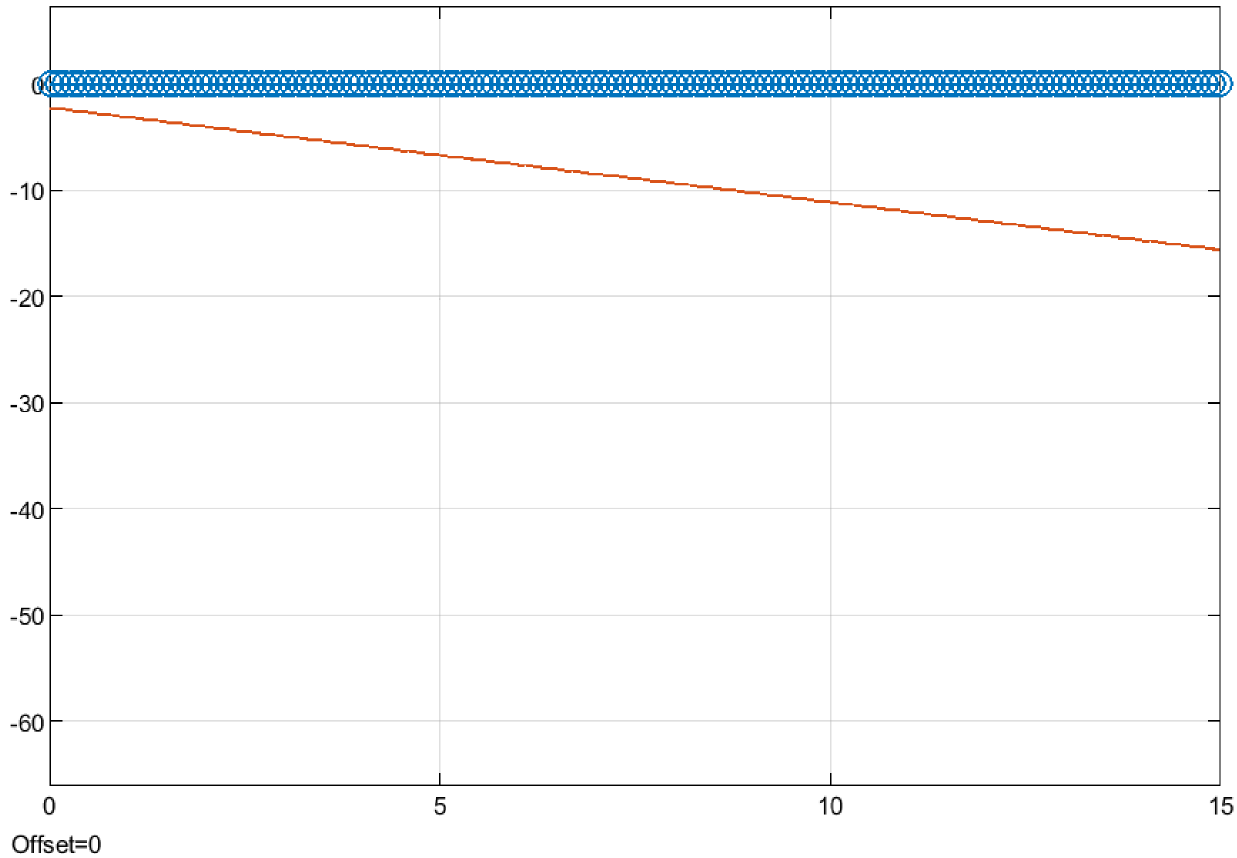
Process noise(others)



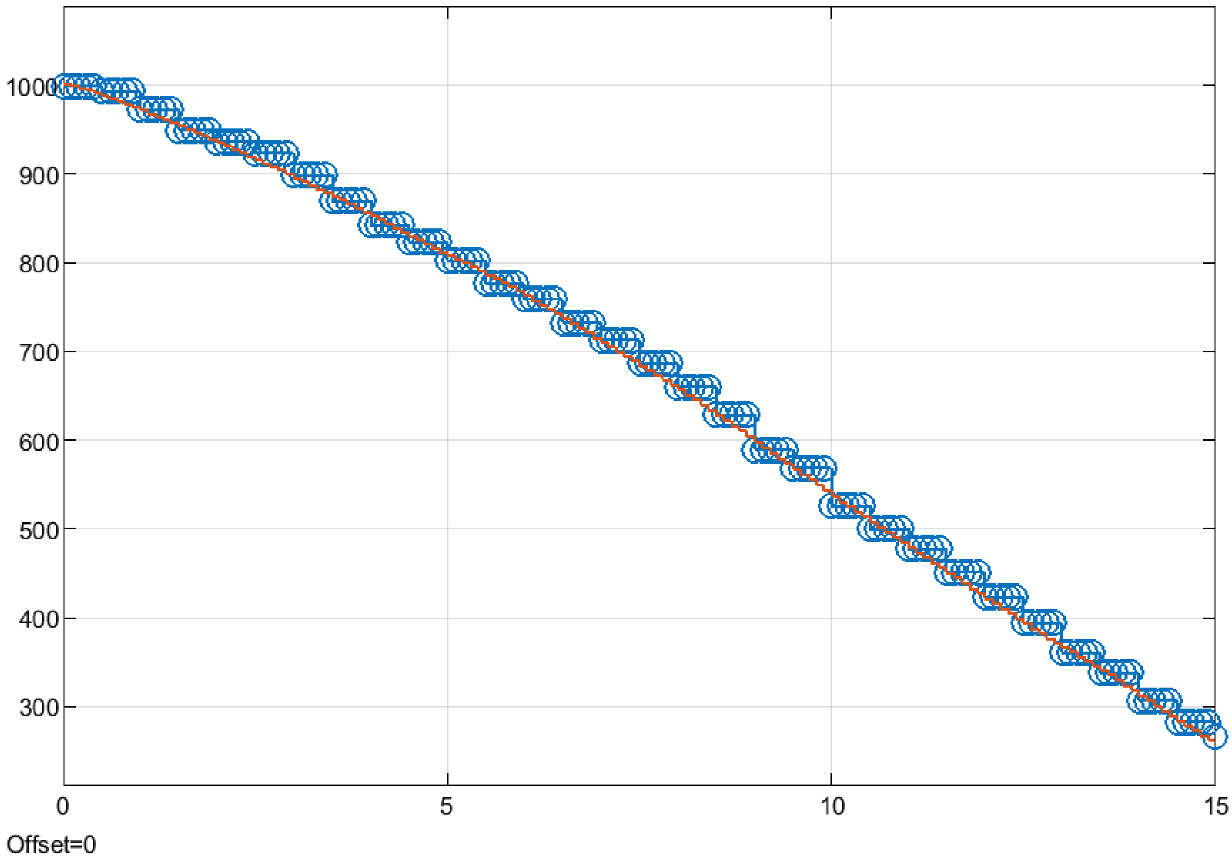
Measurement



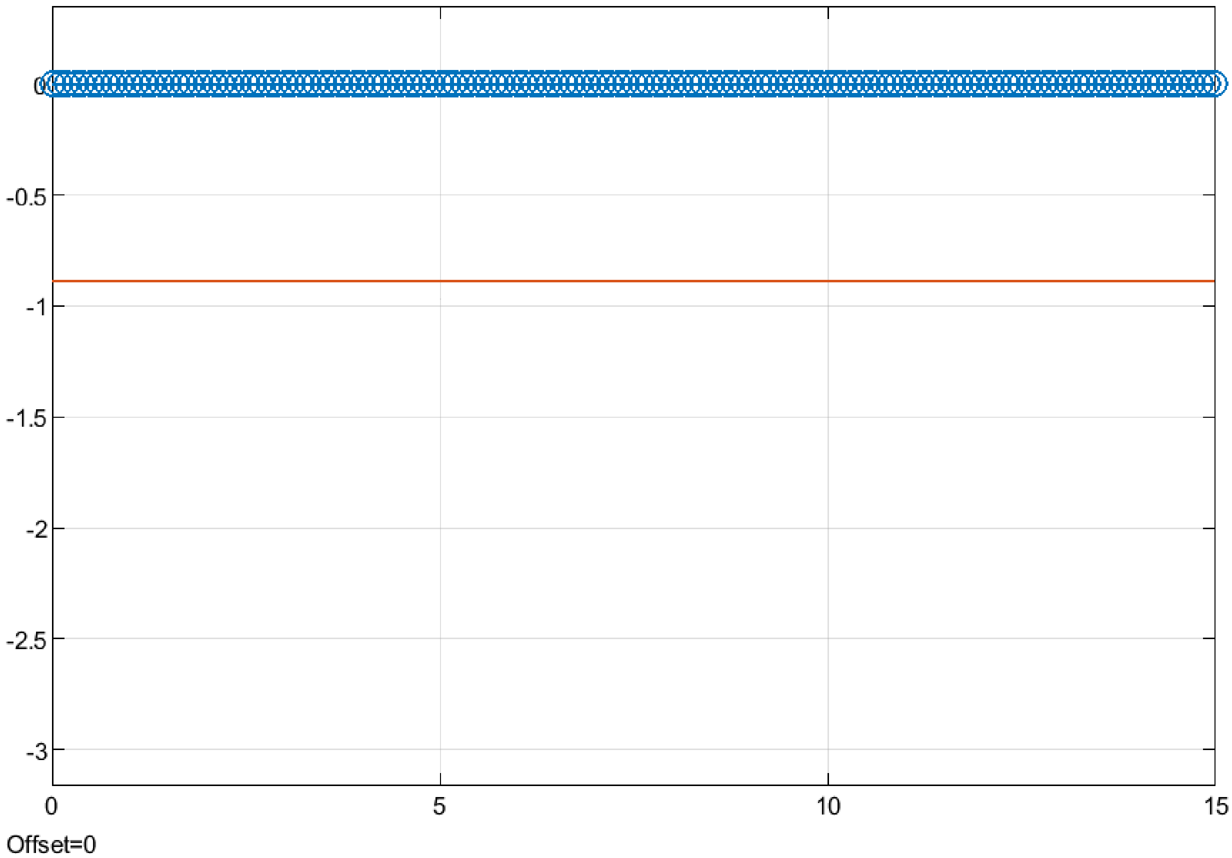
Measurement noise



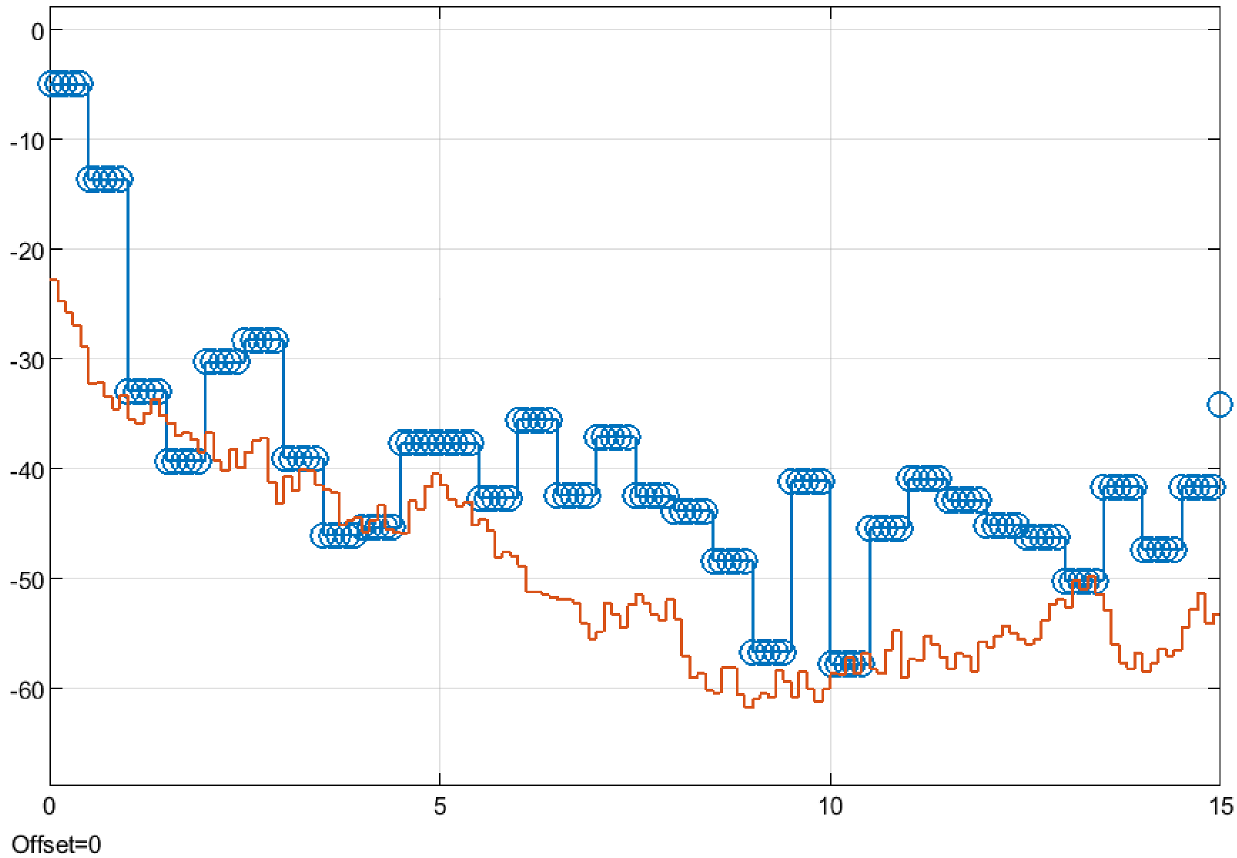
State and estimation of rx



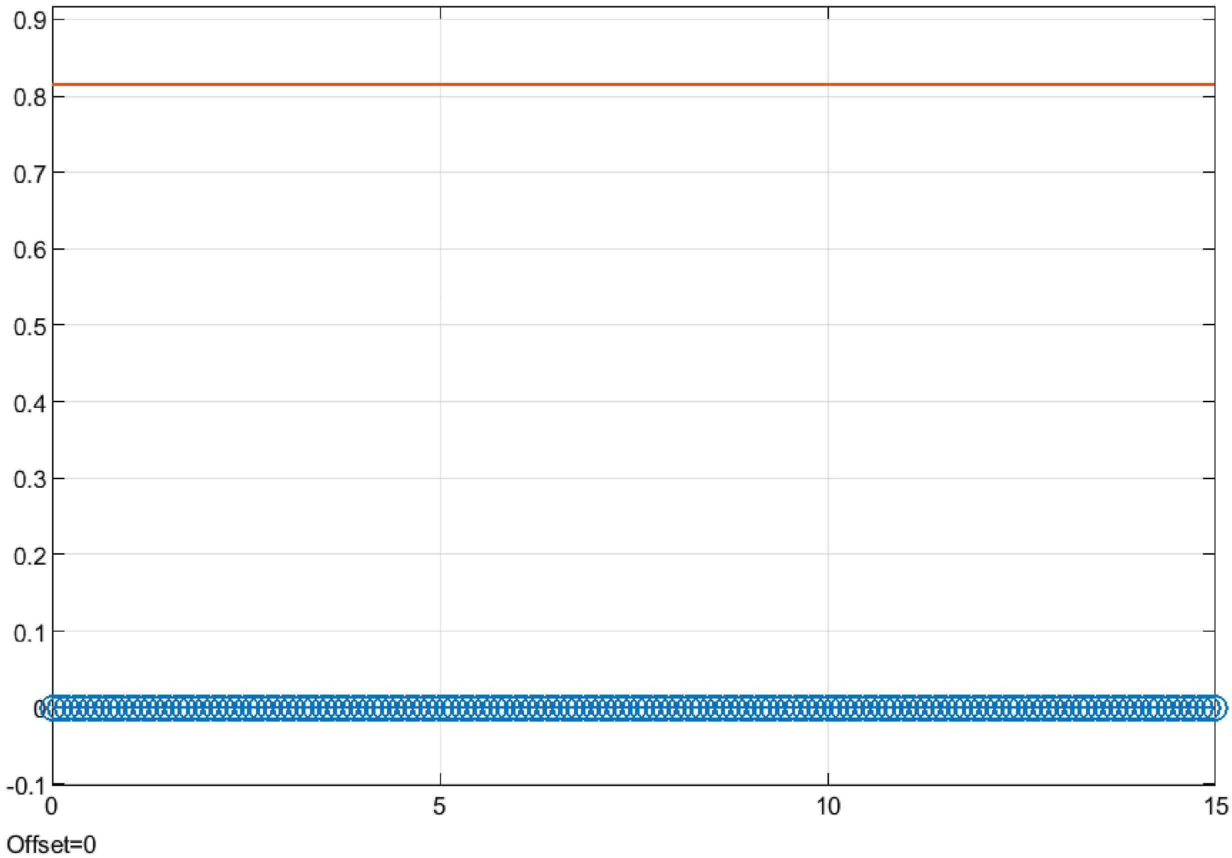
State and estimation of ry



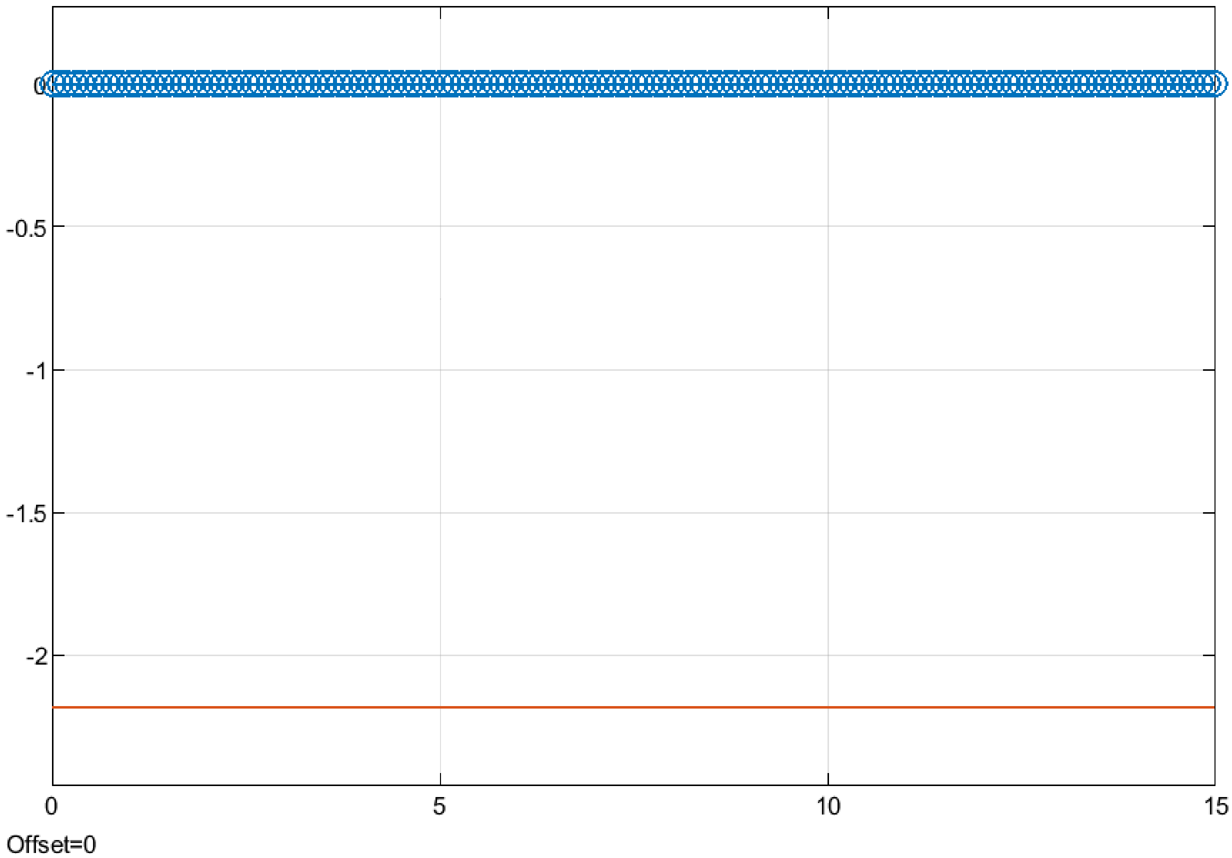
State and estimation of vx



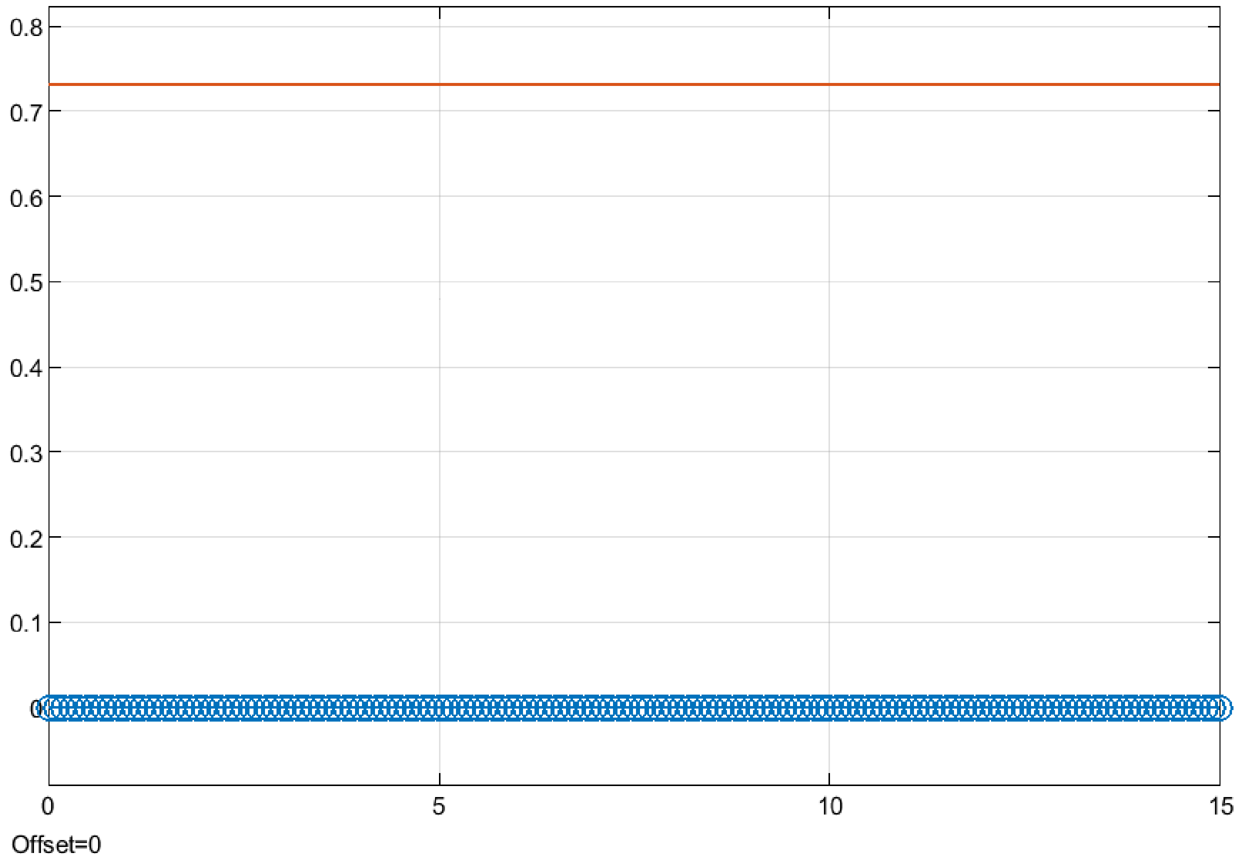
State and estimation of vy



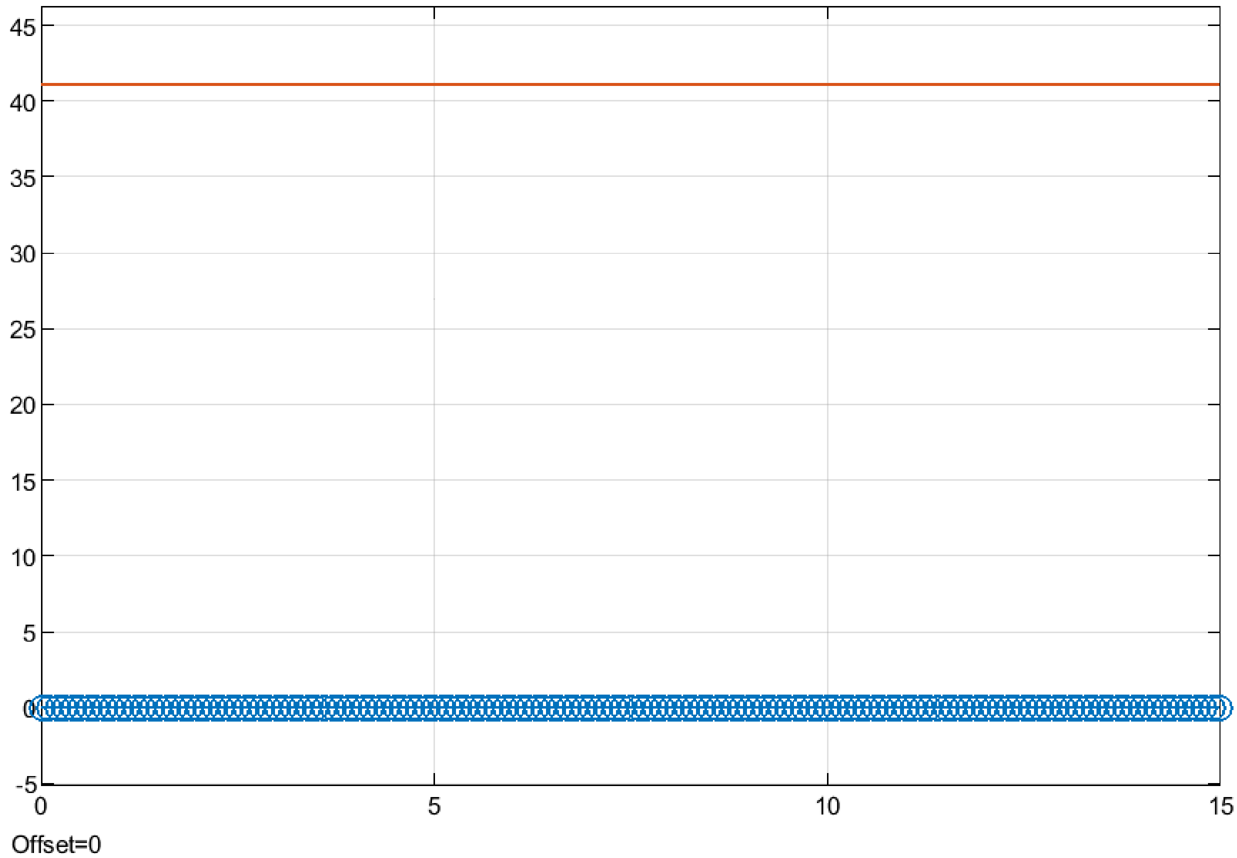
State and estimation of delta_h



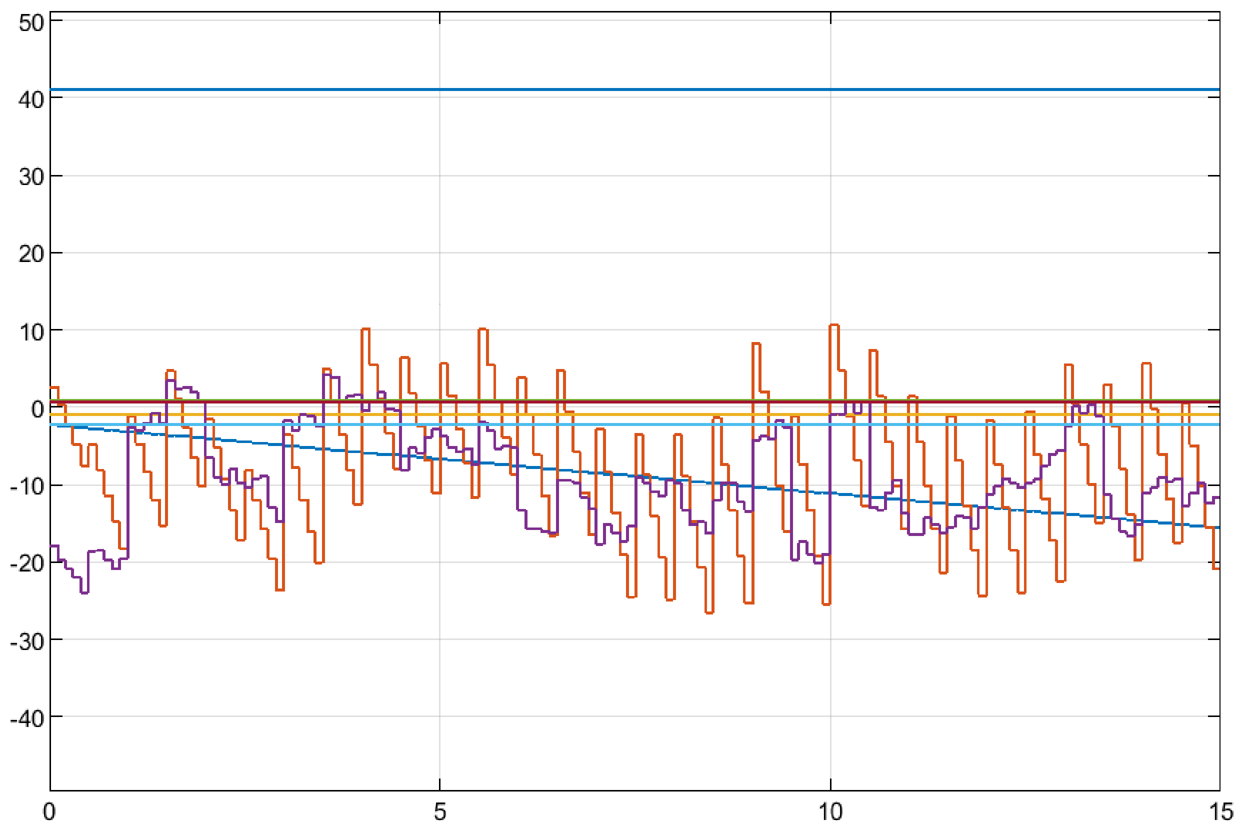
State and estimation of delta_beta



State and estimation of delta_rho0



State and estimation of delta_kp



State estimation errors