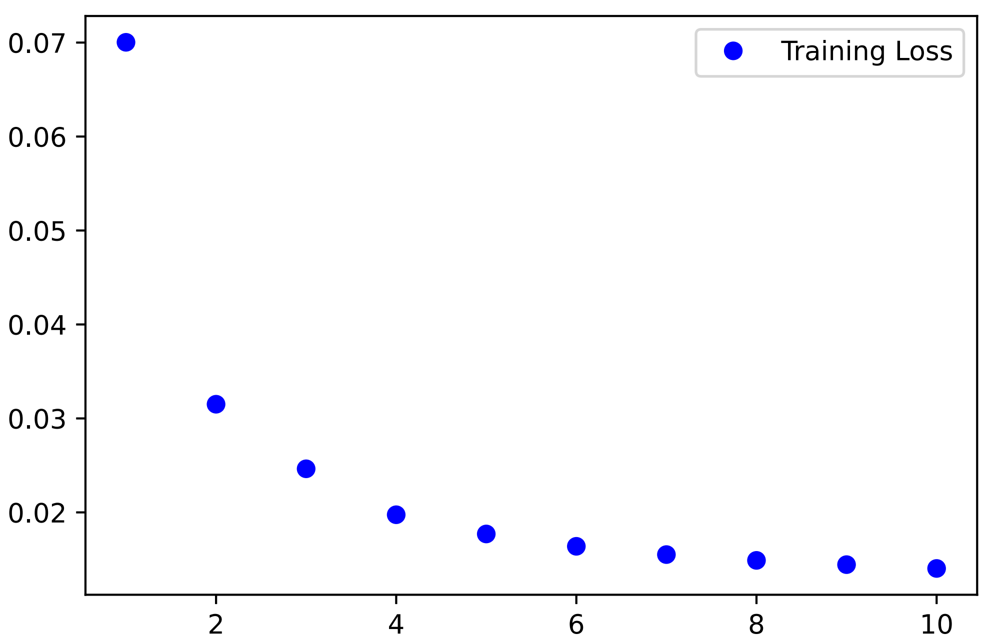
Good Example

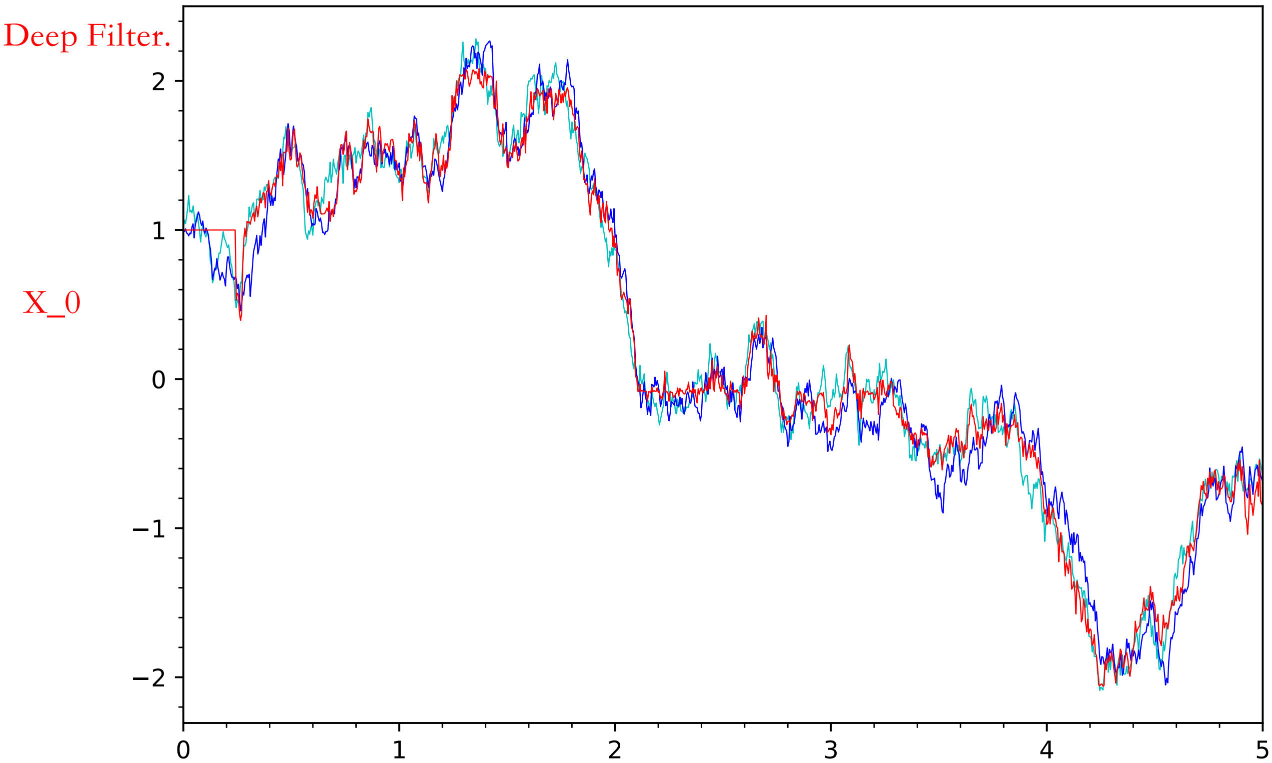
$$x_{n+1} = (I + \eta \begin{bmatrix} 0.1 & 0.5 \\ 0 & 0.1 \end{bmatrix}) x_n + \sqrt{\eta} \begin{bmatrix} 0.7 & -0.6 \\ 0 & 0.7 \end{bmatrix} u_n, \quad x_0 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

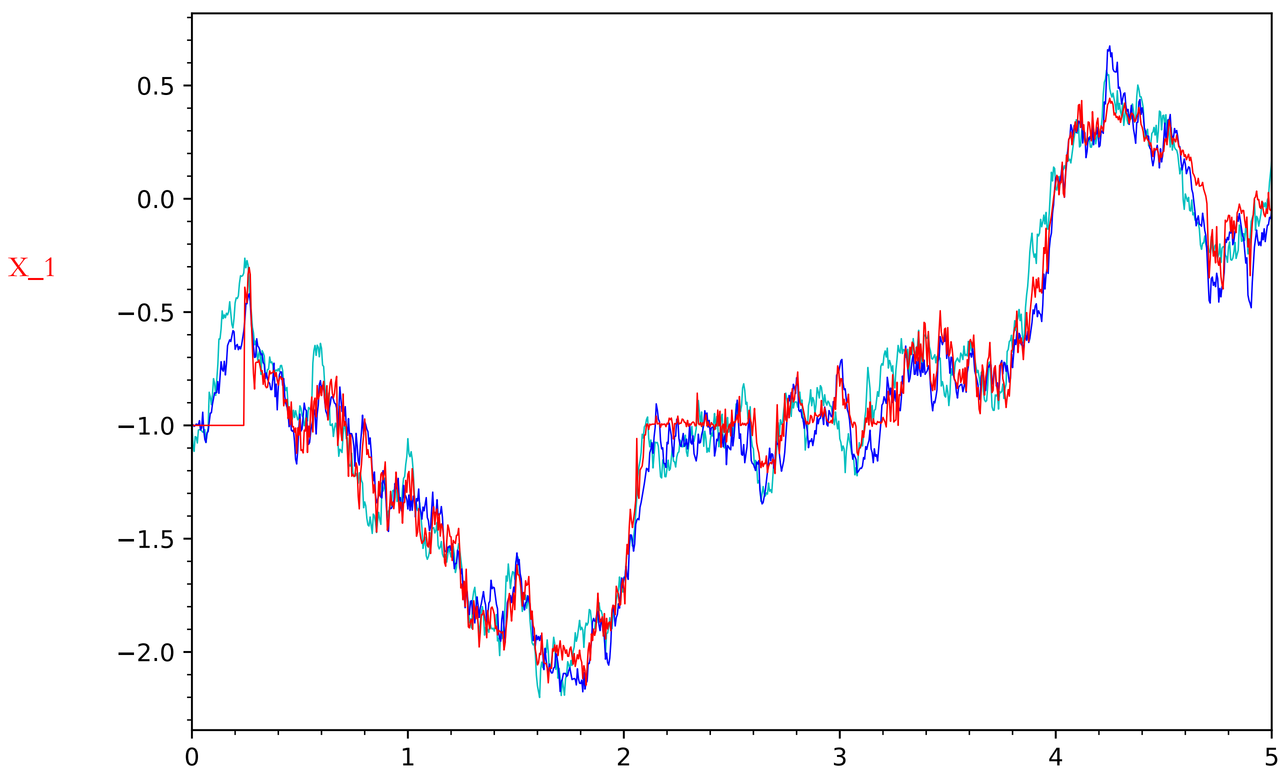
$$y_n = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_n + \sigma_0 v_n, \quad \sigma_0 = 0.5$$

```
Epoch 1/10
95200/95200 [==============================] - 37s 391us/step - loss: 0.0700 - mean_s
quared_error: 0.0700
Epoch 2/10
95200/95200 [==============================] - 37s 384us/step - loss: 0.0315 - mean_s
quared_error: 0.0315
Epoch 3/10
95200/95200 [==============================] - 37s 386us/step - loss: 0.0246 - mean_s
quared_error: 0.0246
Epoch 4/10
95200/95200 [==============================] - 37s 384us/step - loss: 0.0197 - mean_s
quared_error: 0.0197
Epoch 5/10
95200/95200 [==============================] - 36s 381us/step - loss: 0.0177 - mean_s
quared_error: 0.0177
Epoch 6/10
95200/95200 [==============================] - 36s 381us/step - loss: 0.0164 - mean_s
quared_error: 0.0164
Epoch 7/10
95200/95200 [==============================] - 36s 379us/step - loss: 0.0155 - mean_s
quared_error: 0.0155
Epoch 8/10
95200/95200 [==============================] - 36s 383us/step - loss: 0.0149 - mean_s
quared_error: 0.0149
Epoch 9/10
95200/95200 [==============================] - 38s 400us/step - loss: 0.0144 - mean_s
quared_error: 0.0144
Epoch 10/10
95200/95200 [==============================] - 39s 405us/step - loss: 0.0140 - mean_s
quared_error: 0.0140
5950/5950 [==============================] - 2s 348us/step - loss: 0.0139 - mean_squa
red_error: 0.0139
The mse of deep filtering is 1.389%
The mse of Kalman Filtering is 2.496%
The CPU consuming time is 445.51
```



The light blue is for true Monte Carlo value.
The deep blue is for Kalman Filter.
The red is for Deep Filter.





In [ ]:

Constant Velocity Model in 3D space

Measure in 3 dimension.

$$x(k+1) = Fx(k) + u(k)$$
$$x = [x, \dot{x}, y, \dot{y}, z, \dot{z}]'$$

where the transition matrix F is

$$F = \begin{bmatrix} F_1 & 0 & 0 \\ 0 & F_1 & 0 \\ 0 & 0 & F_1 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

The noise process $u(k)$ is assumed zero-mean white with covariance

$$Q = \begin{bmatrix} u_1 & 0 & 0 \\ 0 & u_1 & 0 \\ 0 & 0 & u_1 \end{bmatrix}, \quad u_1 = \begin{bmatrix} T^4/4 & T^3/2 \\ T^3/2 & T^2 \end{bmatrix} q1$$

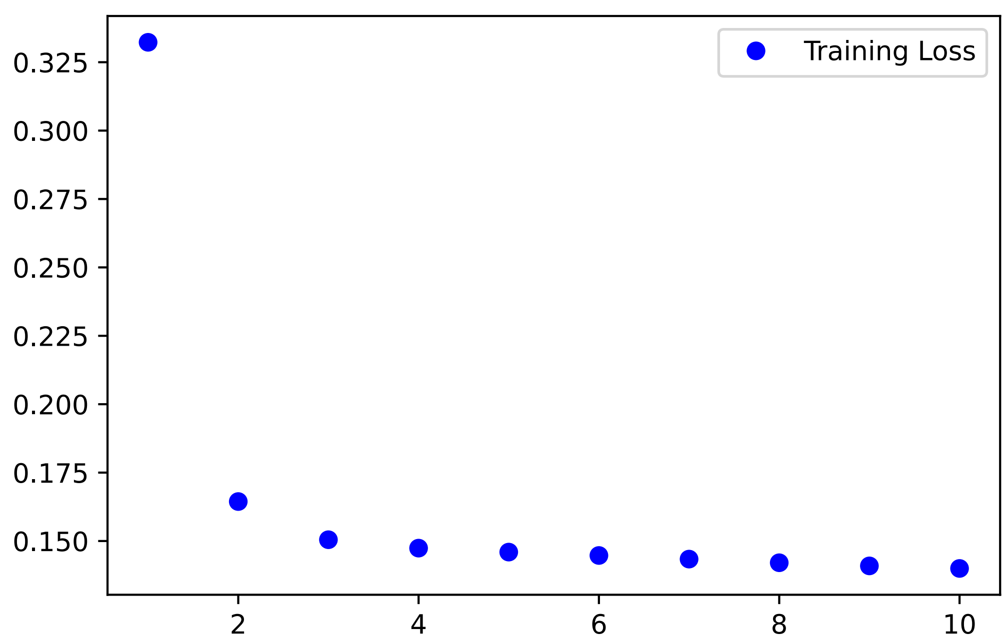
The general measurement update is

$$z_k = h(x_k) + v_k$$
$$h = \begin{bmatrix} r \\ \theta \\ \varphi \end{bmatrix} = \begin{bmatrix} \sqrt{x^2+y^2+z^2} \\ arctan(y/x) \\ arctan(z/\sqrt{x^2+y^2}) \end{bmatrix}, \quad cov(v_k) = diag(\sigma_r^2, \sigma_\theta^2, \sigma_\varphi^2)$$

Parameters Setting

$x_0 = [900, 50, 950, 40, 850, 60]'$

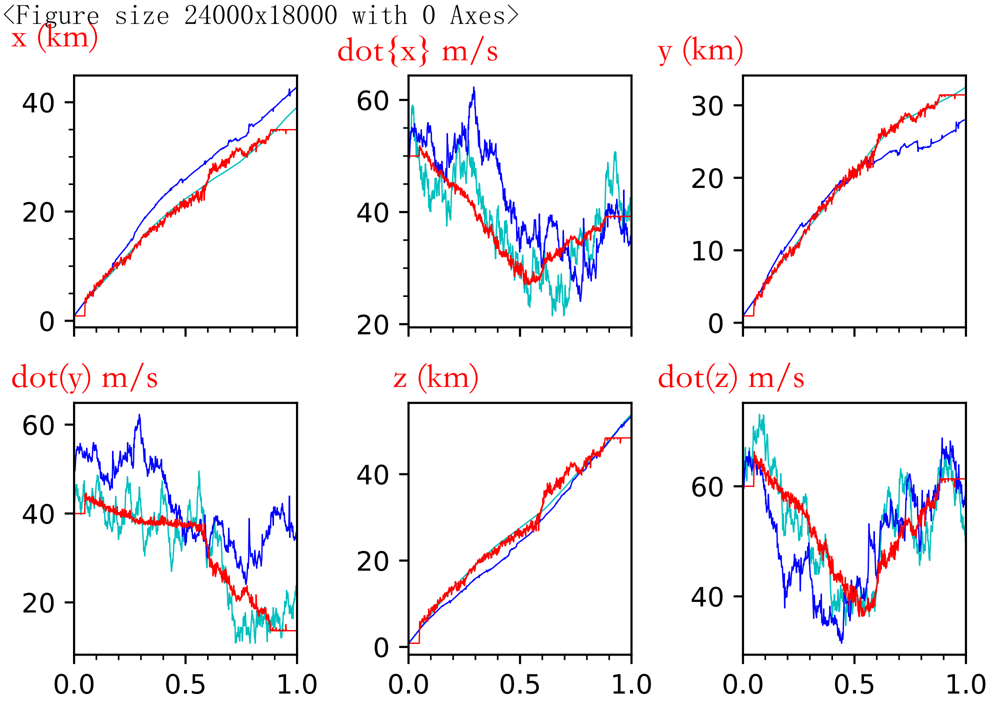
```
Epoch 1/10
47600/47600 [==============================] - 18s 388us/step - loss: 0.3323 - mean_s
quared_error: 0.3323
Epoch 2/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.1644 - mean_s
quared_error: 0.1644
Epoch 3/10
47600/47600 [==============================] - 18s 387us/step - loss: 0.1505 - mean_s
quared_error: 0.1505
Epoch 4/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.1474 - mean_s
quared_error: 0.1474
Epoch 5/10
47600/47600 [==============================] - 18s 387us/step - loss: 0.1459 - mean_s
quared_error: 0.1459
Epoch 6/10
47600/47600 [==============================] - 19s 391us/step - loss: 0.1447 - mean_s
quared_error: 0.1447
Epoch 7/10
47600/47600 [==============================] - 18s 388us/step - loss: 0.1434 - mean_s
quared_error: 0.1434
Epoch 8/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.1421 - mean_s
quared_error: 0.1421
Epoch 9/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.1409 - mean_s
quared_error: 0.1409
Epoch 10/10
47600/47600 [==============================] - 19s 390us/step - loss: 0.1400 - mean_s
quared_error: 0.1400
5950/5950 [==============================] - 2s 351us/step - loss: 0.1398 - mean_squa
red_error: 0.1398
The mse of deep filtering is 13.979%
The mse of Kalman Filtering is 39.974%
The CPU consuming time is 590.92
```



```
In [7]: import matplotlib.pyplot as plt
        %matplotlib inline
        %config InlineBackend.figure_format = 'svg'

        plt.figure(dpi = 600,figsize=[40,30])
        axis=np.linspace(0,1,N+1)
        fig,ax=plt.subplots(2,3,sharex=True)
        plt.xlim((0,1))
        ax[0][0].plot(axis, x_new[:,0]/1000,'c',axis,x_bar_new[:,0]/1000,'b',axis,df_new[:,0]/1
        000,'r',linewidth=0.5)
        ax[0][0].set_xlim((0,1))
        ax[0][0].minorticks_on()
        ax[0][1].plot(axis, x_new[:,1],'c',axis,x_bar_new[:,1],'b',axis,df_new[:,1],'r',linewid
        th=0.5)
        ax[0][1].minorticks_on()
        ax[0][2].plot(axis, x_new[:,2]/1000,'c',axis,x_bar_new[:,2]/1000,'b',axis,df_new[:,2]/1
        000,'r',linewidth=0.5)
        ax[1][0].plot(axis, x_new[:,3],'c',axis,x_bar_new[:,1],'b',axis,df_new[:,3],'r',linewid
        th=0.5)
        ax[1][1].plot(axis, x_new[:,4]/1000,'c',axis,x_bar_new[:,4]/1000,'b',axis,df_new[:,4]/1
        000,'r',linewidth=0.5)
        ax[1][2].plot(axis, x_new[:,5],'c',axis,x_bar_new[:,5],'b',axis,df_new[:,5],'r',linewid
        th=0.5)
        fig.subplots_adjust(wspace=0.5, hspace=0.3)
        plt.savefig('6dim-plot.pdf')
        plt.show()
```

<Figure size 24000x18000 with 0 Axes>




```
In [ ]:
```

Constant Velocity Model in 3D space

Measure in 4 dimension

$$x(k+1) = Fx(k) + u(k)$$
$$x = [x, \dot{x}, y, \dot{y}, z, \dot{z}]'$$

where the transition matrix F is

$$F = \begin{bmatrix} F_1 & 0 & 0 \\ 0 & F_1 & 0 \\ 0 & 0 & F_1 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

The noise process $u(k)$ is assumed zero-mean white with covariance

$$Q = \begin{bmatrix} u_1 & 0 & 0 \\ 0 & u_1 & 0 \\ 0 & 0 & u_1 \end{bmatrix}, \quad u_1 = \begin{bmatrix} T^4/4 & T^3/2 \\ T^3/2 & T^2 \end{bmatrix} q1$$

The general measurement update is

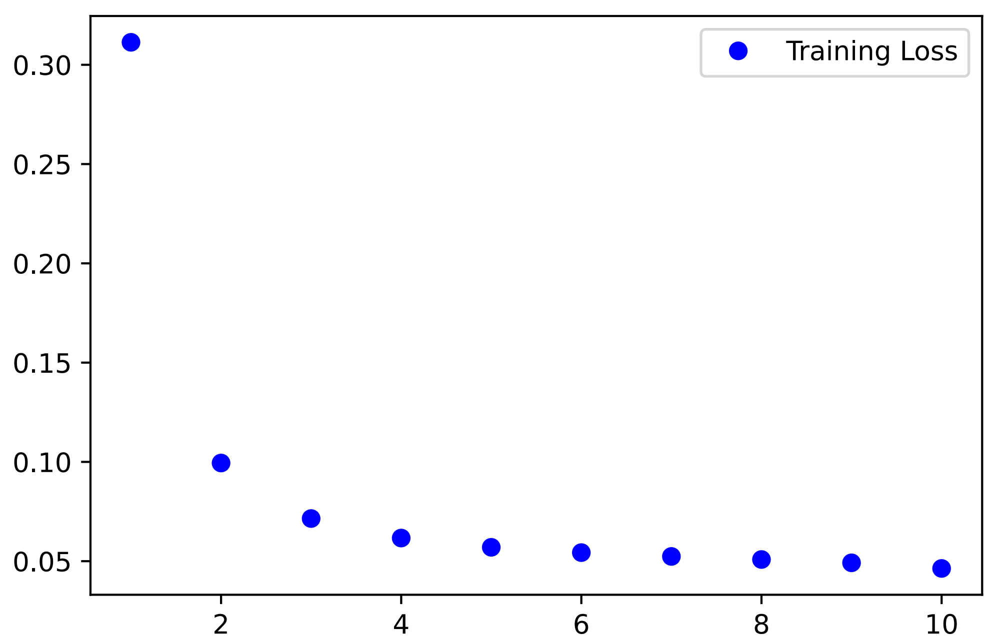
$$z_k = h(x_k) + v_k$$

From [JPLY], we take range-rate measurement as

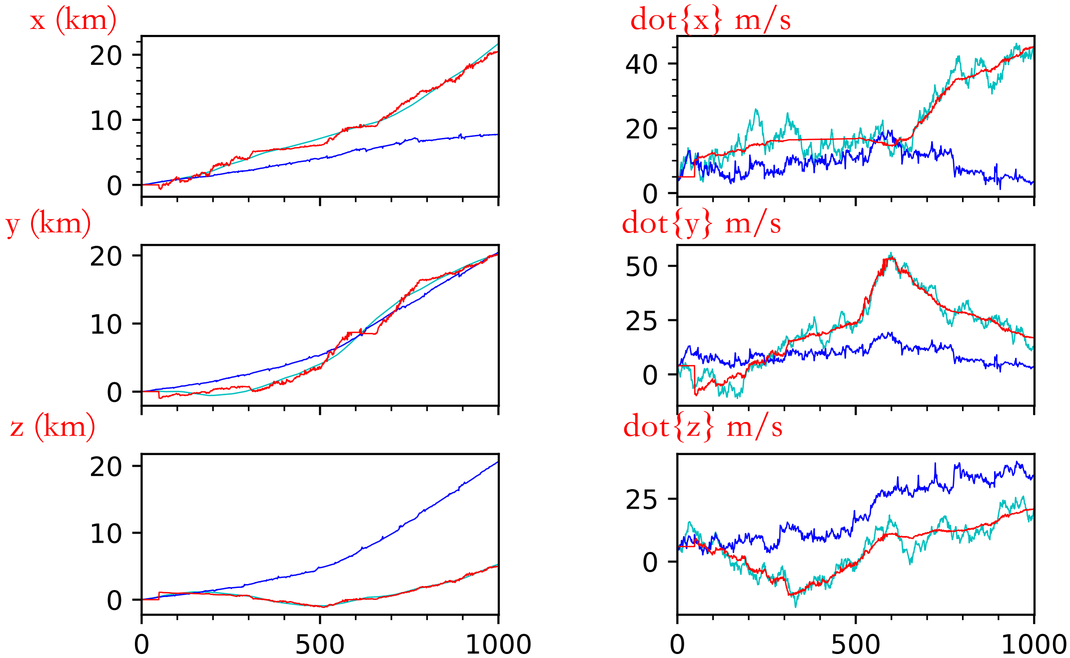
$$h = \begin{bmatrix} r \\ \theta \\ \varphi \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \sqrt{x^2 + y^2 + z^2} \\ arctan(y/x) \\ arctan(z/\sqrt{x^2 + y^2}) \\ (x\dot{x} + y\dot{y} + z\dot{z})/\sqrt{x^2 + y^2 + z^2} \end{bmatrix}$$

v_k is assumed to be zero-mean white with variance $\sigma_r^2, \sigma_\theta^2, \sigma_\varphi^2, \sigma_{\dot{r}}^2$ and $\rho(r, \dot{r}) = \rho$.

```
Epoch 1/10
47600/47600 [==============================] - 18s 382us/step - loss: 0.3113 - mean_s
quared_error: 0.3113
Epoch 2/10
47600/47600 [==============================] - 18s 388us/step - loss: 0.0994 - mean_s
quared_error: 0.0994
Epoch 3/10
47600/47600 [==============================] - 18s 383us/step - loss: 0.0715 - mean_s
quared_error: 0.0715
Epoch 4/10
47600/47600 [==============================] - 18s 384us/step - loss: 0.0616 - mean_s
quared_error: 0.0616
Epoch 5/10
47600/47600 [==============================] - 19s 390us/step - loss: 0.0570 - mean_s
quared_error: 0.0570
Epoch 6/10
47600/47600 [==============================] - 18s 386us/step - loss: 0.0543 - mean_s
quared_error: 0.0543
Epoch 7/10
47600/47600 [==============================] - 18s 386us/step - loss: 0.0524 - mean_s
quared_error: 0.0524
Epoch 8/10
47600/47600 [==============================] - 18s 385us/step - loss: 0.0508 - mean_s
quared_error: 0.0508
Epoch 9/10
47600/47600 [==============================] - 18s 386us/step - loss: 0.0492 - mean_s
quared_error: 0.0492
Epoch 10/10
47600/47600 [==============================] - 18s 386us/step - loss: 0.0463 - mean_s
quared_error: 0.0463
5950/5950 [==============================] - 2s 339us/step - loss: 0.0454 - mean_squa
red_error: 0.0454
The mse of deep filtering is 4.536%
The mse of Kalman Filtering is 603.455%
The CPU consuming time is 942.72
```



```
&lt;Figure size 24000x18000 with 0 Axes&gt;
```




```
In [ ]:
```

Measure in 6 dimension.

Constant Velocity Model in 3D space

$$x(k+1) = Fx(k) + u(k)$$
$$x = [x, \dot{x}, y, \dot{y}, z, \dot{z}]'$$

where the transition matrix F is

$$F = \begin{bmatrix} F_1 & 0 & 0 \\ 0 & F_1 & 0 \\ 0 & 0 & F_1 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

The noise process $u(k)$ is assumed zero-mean white with covariance

$$Q = \begin{bmatrix} u_1 & 0 & 0 \\ 0 & u_1 & 0 \\ 0 & 0 & u_1 \end{bmatrix}, \quad u_1 = \begin{bmatrix} T^4/4 & T^3/2 \\ T^3/2 & T^2 \end{bmatrix} q1$$

The general measurement update is

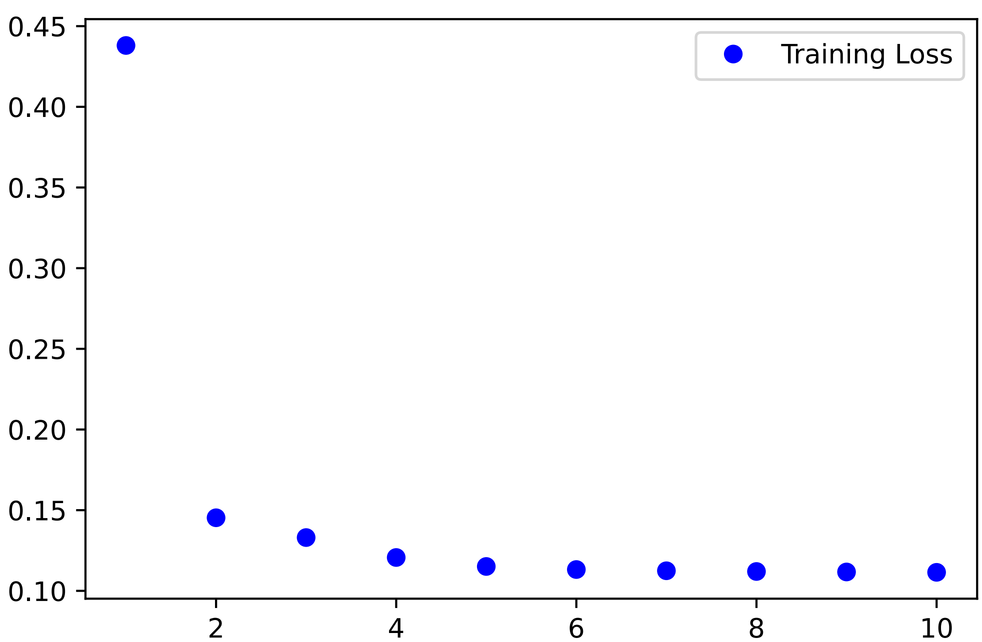
$$z_k = h(x_k) + v_k$$

$$h = \begin{bmatrix} r_1 \\ \theta_1 \\ \varphi_1 \\ r_2 \\ \theta_2 \\ \varphi_2 \end{bmatrix} = \begin{bmatrix} \sqrt{x^2+y^2+z^2} \\ arctan(y/x) \\ arctan(z/\sqrt{x^2+y^2}) \\ \sqrt{x^2+y^2+z^2} \\ arctan(y/x) \\ arctan(z/\sqrt{x^2+y^2}) \end{bmatrix}, \quad cov(v_k) = diag(\sigma_{r1}^2, \sigma_{\theta 1}^2, \sigma_{\varphi}^2, \sigma_{r2}^2, \sigma_{\theta 2}^2, \sigma_{\varphi 2}^2)$$

Parameters Setting

$x_0 = [900, 50, 950, 40, 850, 60]'$

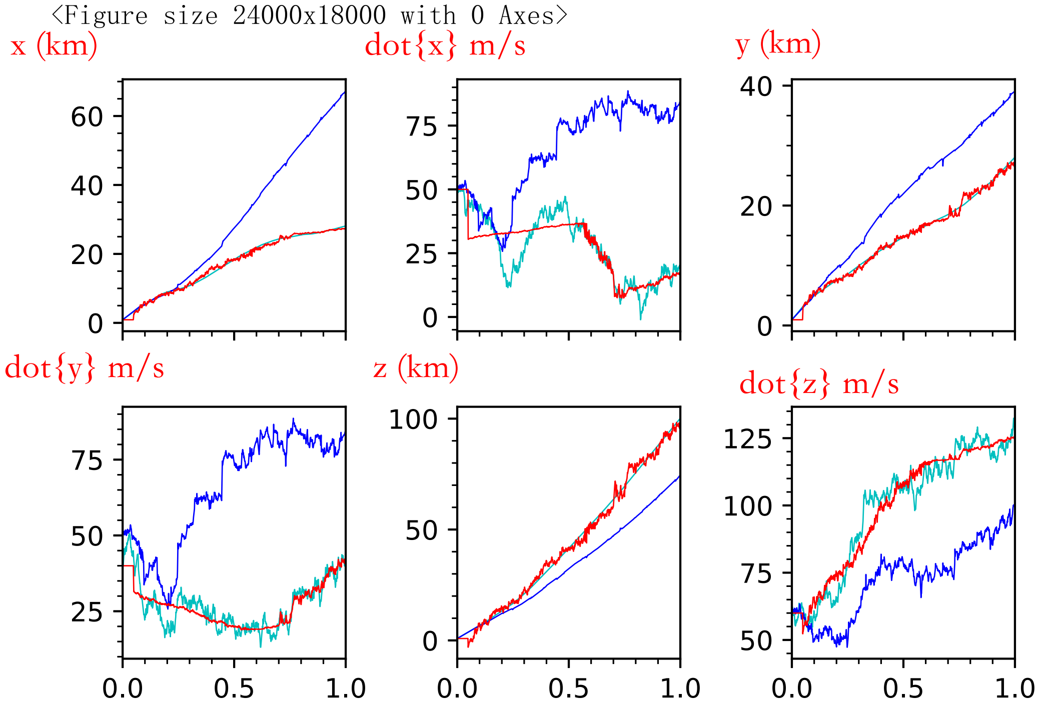
```
Epoch 1/10
47600/47600 [==============================] - 21s 437us/step - loss: 0.4380 - mean_s
quared_error: 0.4380
Epoch 2/10
47600/47600 [==============================] - 21s 436us/step - loss: 0.1453 - mean_s
quared_error: 0.1453
Epoch 3/10
47600/47600 [==============================] - 21s 435us/step - loss: 0.1330 - mean_s
quared_error: 0.1330
Epoch 4/10
47600/47600 [==============================] - 21s 434us/step - loss: 0.1207 - mean_s
quared_error: 0.1207
Epoch 5/10
47600/47600 [==============================] - 21s 435us/step - loss: 0.1151 - mean_s
quared_error: 0.1151
Epoch 6/10
47600/47600 [==============================] - 21s 435us/step - loss: 0.1132 - mean_s
quared_error: 0.1132
Epoch 7/10
47600/47600 [==============================] - 21s 433us/step - loss: 0.1125 - mean_s
quared_error: 0.1125
Epoch 8/10
47600/47600 [==============================] - 21s 433us/step - loss: 0.1120 - mean_s
quared_error: 0.1120
Epoch 9/10
47600/47600 [==============================] - 21s 434us/step - loss: 0.1117 - mean_s
quared_error: 0.1117
Epoch 10/10
47600/47600 [==============================] - 21s 433us/step - loss: 0.1115 - mean_s
quared_error: 0.1115
5950/5950 [==============================] - 2s 400us/step - loss: 0.1113 - mean_squa
red_error: 0.1113
The mse of deep filtering is 11.135%
The mse of Kalman Filtering is 505.838%
The CPU consuming time is 951.54
```



In [6]:
```
# import matplotlib.pyplot as plt
%matplotlib inline
%config InlineBackend.figure_format = 'svg'

plt.figure(dpi = 600,figsize=[40,30])
axis=np.linspace(0,1,N+1)
fig,ax=plt.subplots(2,3,sharex=True)
plt.xlim((0,1))
ax[0][0].plot(axis, x_new[:,0]/1000,'c',axis,x_bar_new[:,0]/1000,'b',axis,df_new[:,0]/1
000,'r',linewidth=0.5)
ax[0][0].set_xlim((0,1))
ax[0][0].minorticks_on()
ax[0][1].plot(axis, x_new[:,1],'c',axis,x_bar_new[:,1],'b',axis,df_new[:,1],'r',linewid
th=0.5)
ax[0][1].minorticks_on()
ax[0][2].plot(axis, x_new[:,2]/1000,'c',axis,x_bar_new[:,2]/1000,'b',axis,df_new[:,2]/1
000,'r',linewidth=0.5)
ax[0][2].minorticks_on()
ax[1][0].plot(axis, x_new[:,3],'c',axis,x_bar_new[:,1],'b',axis,df_new[:,3],'r',linewid
th=0.5)
ax[1][0].minorticks_on()
ax[1][1].plot(axis, x_new[:,4]/1000,'c',axis,x_bar_new[:,4]/1000,'b',axis,df_new[:,4]/1
000,'r',linewidth=0.5)
ax[1][1].minorticks_on()
ax[1][2].plot(axis, x_new[:,5],'c',axis,x_bar_new[:,5],'b',axis,df_new[:,5],'r',linewid
th=0.5)
ax[1][2].minorticks_on()
fig.subplots_adjust(wspace=0.5, hspace=0.3)
plt.savefig('6dim-plot.pdf')
plt.show()
```

<Figure size 24000x18000 with 0 Axes>



In [ ]:

Constant Acceleration Model in 3D space(9-dim)

Measure in 4 dimension

$$x_{n+1} = Fx_n + u_n, \quad x_n = [x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}]'$$

where the transistion matrix is

$$F = \begin{bmatrix} F_2 & 0 & 0 \\ 0 & F_2 & 0 \\ 0 & 0 & F_2 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}$$

u_n is zero mean Gaussian with covariance

$$Q = \begin{bmatrix} u_2 & 0 & 0 \\ 0 & u_2 & 0 \\ 0 & 0 & u_2 \end{bmatrix}, \quad u_2 = \begin{bmatrix} T^4/4 & T^3/2 & T^2/2 \\ T^3/2 & T^2 & T \\ T^2 & T & 1 \end{bmatrix} q_2$$

Measurement Updata, we take the form from Li's book

$$z_n = h(x_n) + v_n$$

where

$$h = \begin{bmatrix} r \\ a1 \\ a2 \\ e \end{bmatrix} = \begin{bmatrix} \sqrt{x^2 + y^2 + z^2} \\ arctan(y/x) \\ arctan(y/x) \\ arctan(z/\sqrt{x^2 + y^2}) \end{bmatrix}$$

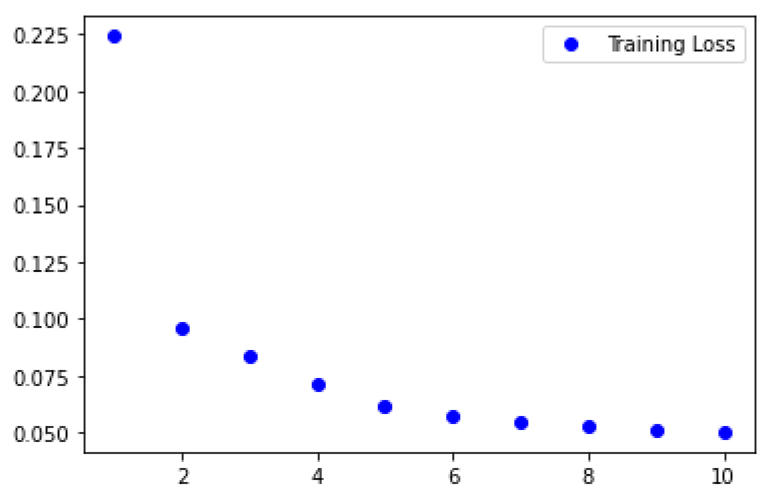
v_n is zero mean Gaussian with covariance

$$Q = diag(\sigma_r^2, \sigma_{a1}^2, \sigma_{a2}^2, \sigma_e^2)$$

Parameters Setting

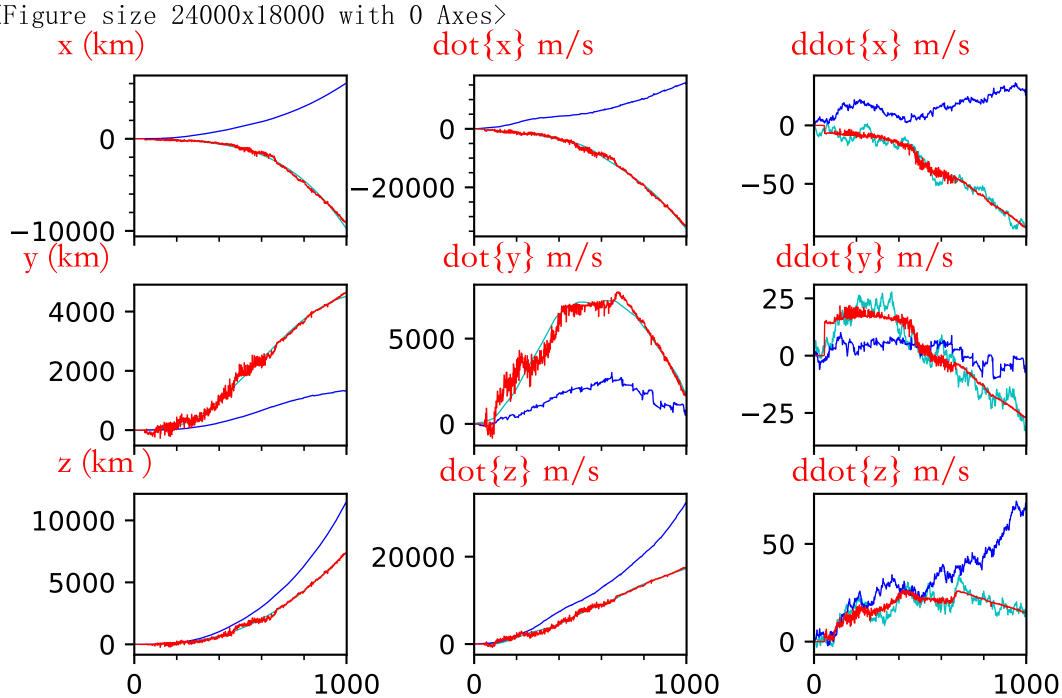
$T = 1, q_2 = 1, (\sigma_r, \sigma_{a1}, \sigma_{a2}, \sigma_e) = (20, 7, 2, 2).$

```
Epoch 1/10
47600/47600 [==============================] - 19s 396us/step - loss: 0.2242 - mean_s
quared_error: 0.2242
Epoch 2/10
47600/47600 [==============================] - 19s 397us/step - loss: 0.0961 - mean_s
quared_error: 0.0961
Epoch 3/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.0837 - mean_s
quared_error: 0.0837
Epoch 4/10
47600/47600 [==============================] - 19s 395us/step - loss: 0.0714 - mean_s
quared_error: 0.0714
Epoch 5/10
47600/47600 [==============================] - 19s 392us/step - loss: 0.0617 - mean_s
quared_error: 0.0617
Epoch 6/10
47600/47600 [==============================] - 19s 392us/step - loss: 0.0570 - mean_s
quared_error: 0.0570
Epoch 7/10
47600/47600 [==============================] - 19s 390us/step - loss: 0.0545 - mean_s
quared_error: 0.0545
Epoch 8/10
47600/47600 [==============================] - 19s 390us/step - loss: 0.0528 - mean_s
quared_error: 0.0528
Epoch 9/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.0515 - mean_s
quared_error: 0.0515
Epoch 10/10
47600/47600 [==============================] - 19s 389us/step - loss: 0.0503 - mean_s
quared_error: 0.0503
5950/5950 [==============================] - 2s 344us/step - loss: 0.0497 - mean_squa
red_error: 0.0497
The mse of deep filtering is 4.968%
The mse of Kalman Filtering is 483896195876064.875%
The CPU consuming time is 619.08
```



```
fig.subplots_adjust(wspace=0.6, hspace=0.3)
plt.savefig('6dim-plot.pdf')
plt.show()
```

<Figure size 24000x18000 with 0 Axes>



In [ ]: