Solution to analysis in Home Assignment 1

QunZhang + qunz (cid)

Analysis

In this report I will present my independent analysis of the questions related to home assignment 1. I have discussed the solution with Lizi Teng, Mingxiang Zhao but I swear that the analysis written here are my own.

1 A first Kalman filter and its properties

1.1 a

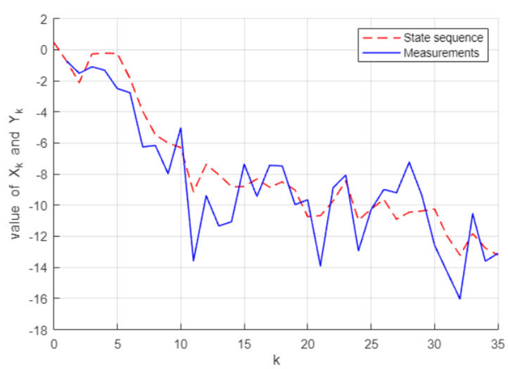

Figure 1: Comparison of states and measurement

As is shown in the figure 1, we can see that these two curves do not fit very well, because there is noise in the measurement process, but the general trend is the same, and the curve is inside $x \pm 3\sigma$. So we can say that the measurement behave according to the model.

1.2 b

As is shown in the figure 2, which contains $\hat{x}_k$, measurement and states. In the figure, the measurement and states are in the range of $\hat{x}_k \pm 3\sigma$, which represents the 99.7 % confidence interval. Therefore, if the measurements and states are consistently within this range, the filter is effectively using the available information to estimate the state of the system, and that the estimated state is likely to be accurate.
Additionally, if the estimate is reasonable and measurements and states are inside $\hat{x}_k \pm 3\sigma$, it suggests that the error covariance matrix is also accurately representing the uncertainty in the estimates.

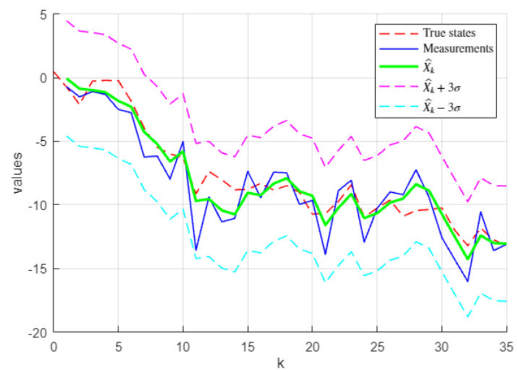

Figure 2: kalman filter, states and measurements

As is shown in the figure 3, which is the error density around zero-mean for time instances $k = [1, 2, 4, 30]$. In the figure 3, as time goes by, the filter receives more measurements and incorporates them into the state estimate, the uncertainty in the estimate decreases, and the value of σ decreases accordingly. σ keeps decreasing and remains stationary at the end, becoming stationary P. The plot in figure 3 can also verify our theory, with k goes by, the error density becomes thinner and taller, and k=1, 2, 4 gradually changes to the distribution when k=30.

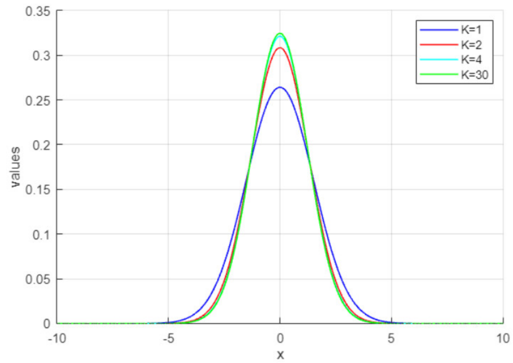

Figure 3: error density

1.3 c

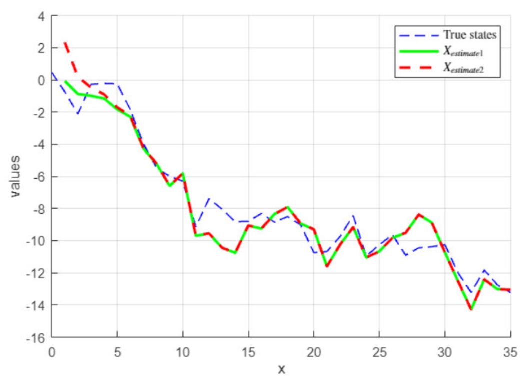

Figure 4: comparison of $\hat{X}_{k|k}$

As is shown in figure 4, with the initial assumption $x_0 \sim N(x_0; 12, 8)$, the curve is shown as red curve. According to the curve, since the prior guess is wrong, the initial error of estimate 2 is a bit large. However, with time going, more and more measurement data coming into the filter, estimate 1 curve and estimate 2 curve coincide.

This is because the Kalman filter uses a recursive algorithm to estimate the state of the system from measurements and a model of the system. As new measurements are received and incorporated into the estimate, the filter's state estimate becomes increasingly accurate and the level of uncertainty in the estimate decreases. This recursive process continues over time until the filter's estimate converges to a steady-state value.

1.4 d

The probability density function $P(x_{k|k})$ represents the uncertainty in the estimated state at time k, given all the measurements up to time k. This probability density function is obtained by combining the predicted state estimate $x_{k|k-1}$ with the measurement information y_k, using the Kalman gain K_k to weight the information from each source. The resulting estimate $x_{k|k}$ is a weighted average of the predicted state estimate and the measurement, and the error covariance $P_{k|k}$ represents the combined uncertainty in the estimate.
In the figure 5, the $P(x_{k|k-1})$ is calculated by $P(x_{k-1|k-1})$. Using the Kalman gain K_k to weight the information $x_{k|k-1}$ and y_k, we get the plot of $P(x_{k|k})$. The σ of $P(x_{k|k})$ and $P(x_{k-1|k-1})$ are similar, because it reaches to a stationary P (reach to a constant value). However, the mean is different,they represent different estimates of the state variable $x_{k|k}$ at different points in time, and these estimates are based on different information and assumptions.

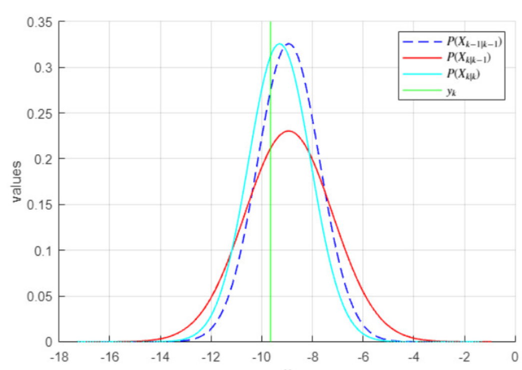

Figure 5: comparison of $P(x_{k|k})$

1.5 e

As is shown in the figure 6, where the histogram is generated from $(x_k - x_{\hat{k|k}})$ with 100000 points, and the blue curve is the analytical norm plot of $N(0, P_{N|N})$. From the plot, we can see that the blue line and histogram fits well. Such a plot suggests that the kalman filter is effectively reducing the measurement noise and providing reasonable estimates of the states x_k. Additionally, it also indicates that the covariance matrix $P_{k|k}$ is an accurate representation of the uncertainty in the estimate, and that $(x_k - x_{\hat{k|k}})$ can be modeled by a normal distribution with mean zero and covariance matrix $P_{k|k}$.

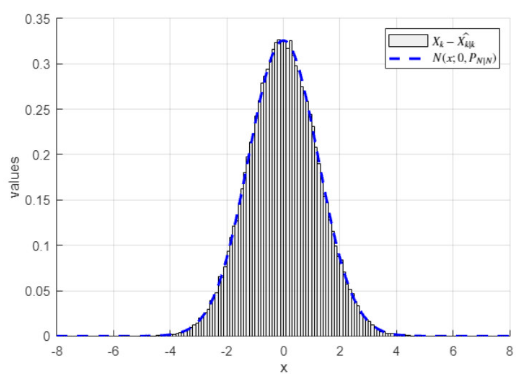

Figure 6: comparison of $P(x_{k|k})$ and $(x_k - x_{\hat{k|k}})$

The innovation $v_k = y_k - H_k x_{k|\hat{k-1}}$ should satisfy the equation shown below:

$$p\left(\mathbf{v}_k \mid \mathbf{y}_{1:k-1}\right) = \mathcal{N}\left(\mathbf{v}_k; \mathbf{0}, \mathbf{S}_k\right)$$
$$\operatorname{Cov}\left(\mathbf{v}_k, \mathbf{v}_{k-l}\right) = \begin{cases} \operatorname{Cov}\left\{\mathbf{v}_k\right\} & \text{if } I = 0 \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{1}$$

To check the correlation of the innovation properties, we can use the auto correlation function:

$$\rho(I) = \frac{\sum_{k=l+1}^{K} \mathbf{v}_k^T \mathbf{v}_{k-l}}{\sum_{\tau=1+1}^{K} \mathbf{v}_\tau^T \mathbf{v}_\tau} \tag{2}$$

As is shown in the figure 7(left), which is the consistency check of kalman filter, where $S_k^(- \frac{1}{2})V_k$ is inside the $\pm 3\sigma$ region, the kalman filter fullfill the consistency. As is shown in the figure 7(right), which is the auto correction plot of the v_k in kalman filter. For $I > 0$, $\rho(I) \approx 0$, and $\rho(I) = 0$ when $I = 0$, which confirms the correlation of our Kalman filter.

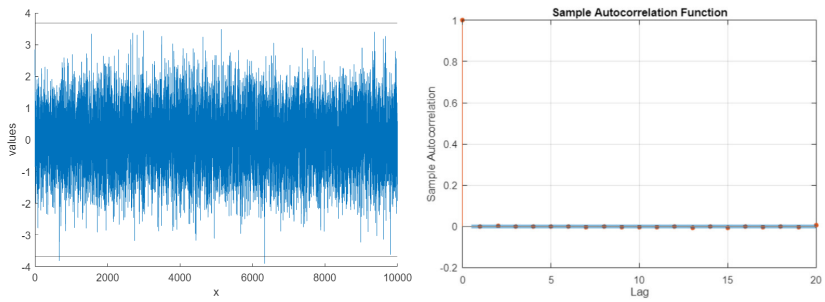

Figure 7: consistency check (left) and auto correlation of v_k (right)

2 Tuning a Kalman filter

2.1 Calibrate speed sensor model:

After loading the measurement, calculating the mean and variance of each velocity test, the μ and Σ are shown in the figure 8, 9, 10.
$\mu_1 = 0, \Sigma_1 = 2.9867, \mu_2 = 11.0264, \Sigma_2 = 3.0332, \mu_3 = 22.0453, \Sigma_3 = 3.0439.$

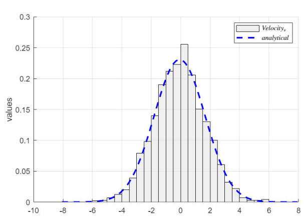

Figure 8: velocity 0

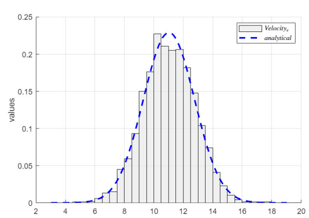

Figure 9: velocity 10

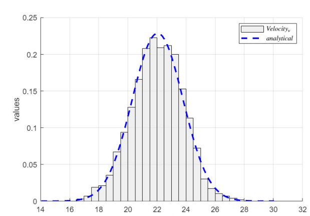

Figure 10: velocity 20

Combining the data shown above and taking average, we can get that:

$$\begin{aligned} C &= 1.1023 \\ Var[r_k^v] &= 2.4867 \end{aligned} \tag{3}$$

The distribution of r_k^v is shown below, and the histogram fits the curve generated by our analytical calculation $N \sim (0, 2.4687)$, which means that the calibrated model is behaving appropriately.

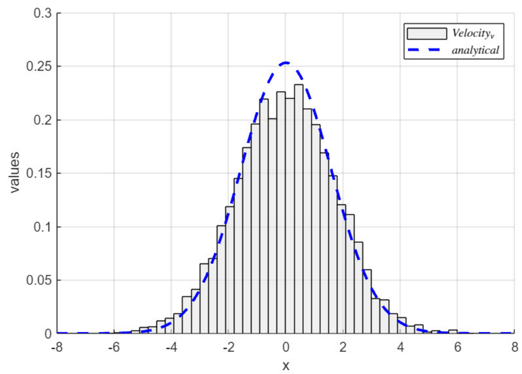

Figure 11: distribution of r_k^v

2.2 Fusing sensors with different rates

As is shown in the figure 11, which is the sensor, due to the different sampling time, the data in the position sensor is fewer than what we get in the velocity.

```
SensorData = 2×2000
10^4 ×
   0.0002      NaN   0.0000      NaN   0.0000      NaN  -0.0003      NaN   0.0002      NaN   0.0000      NaN   0.0003
   0.0001   0.0002  -0.0001   0.0000  -0.0001  -0.0001  -0.0001  -0.0001  -0.0000   0.0003  -0.0000  -0.0000  -0.0001
```

Figure 12: sensor data

The efficient way to approach such a problem is to let the sample time of velocity and position sensor be same. Put the codes shown below into the kalman filter to remove those NAN data column, then can we adapt the general Kalman filter equations in the prediction and update steps to account for the sensor data.

```
%%%%%%%%%%to detect NAN%%%%%%%%%%%%
k=length(Y);
newData=[];
for i=1:k
    if ~isnan(Y(:,i))%remove NAN data
        newData=[newData,Y(:,i)];
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Y=newData;
N = size(Y,2);
m = size(Y,1);
```

Figure 13: change of kalman filter

2.3 Motion model selection and tuning

In this section, we can choose two kinds of motion model: Constant velocity and Constant acceleration, where the equation is shown below:

$$\begin{aligned} \textbf{Constant velocity:} X = \begin{bmatrix} p \\ v \end{bmatrix}, A_{k-1} = \begin{bmatrix} I_n & TI_n \\ 0_n & I_n \end{bmatrix} \\ \textbf{Constant acceleration:} X = \begin{bmatrix} p \\ v \\ a \end{bmatrix}, A_{k-1} = \begin{bmatrix} I_n & TI_n & T^2/2I_n \\ 0_n & I_n & TI_n \\ 0_n & 0_n & I_n \end{bmatrix} \end{aligned} \tag{4}$$

position and velocity of CV model with Q=1 Taking the initial condition $x_{cv} = [1;1]; P_{cv} = [1\ 0; 0\ 1]; x_{ca} = [1;1;1]; P_{ca} = [1\ 0\ 0; 0\ 1\ 0; 0\ 0\ 1]; Q_{cv} = [0\ 0; 0\ 0.1]; Q_{ca} = [0\ 0\ 0; 0\ 0\ 0; 0\ 0\ 0.1]$ into account, the plot of Constant velocity model and Constant acceleration are shown below:

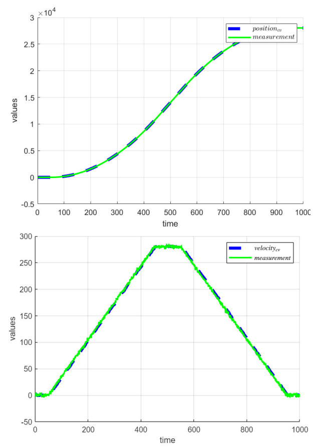

Figure 14: position and velocity of CV model with Q=0.1

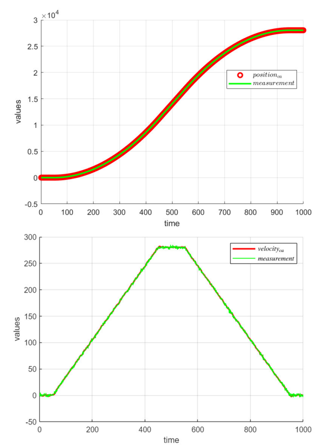

Figure 15: position and velocity of CA model with Q=0.1

Taking the initial condition $Q_{cv} = [0\ 0; 0\ 10]; Q_{ca} = [0\ 0\ 0; 0\ 0\ 0; 0\ 0\ 10]$ into account, plot of Constant velocity model and Constant acceleration are shown below:

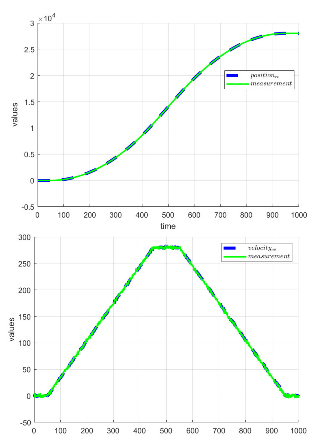

Figure 16: position and velocity of CV model with Q=10

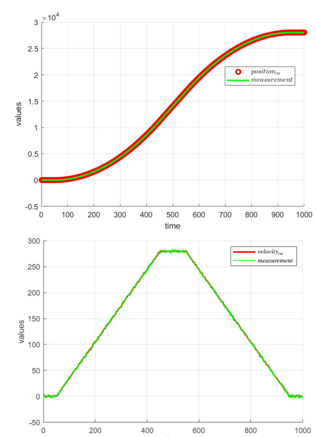

Figure 17: position and velocity of CA model with Q=10

Taking the initial condition $Q_{cv} = [0\ 0; 0\ 1]; Q_{ca} = [0\ 0\ 0; 0\ 0\ 0; 0\ 0\ 1]$ into account, the plot of Constant velocity model and Constant acceleration are shown below:

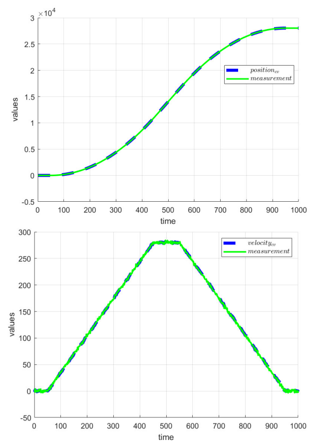

Figure 18: position and velocity of CV model with Q=1

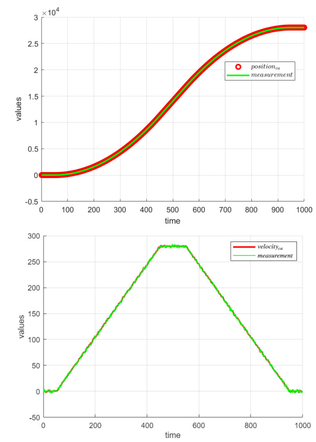

Figure 19: position and velocity of CA model with Q=1

Taking the initial condition $x_{cv} = [100; 100]; P_{cv} = [1\ \ 0; 0\ \ 1]; x_{ca} = [100; 100; 100]; P_{ca} = [1\,0\,0; 0\,1\,0; 0\,0\,1]; Q_{cv} = [0\,0; 0\,1]; Q_{ca} = [0\,0\,0; 0\,0\,0; 0\,0\,1]$ into account, the plots are shown below:

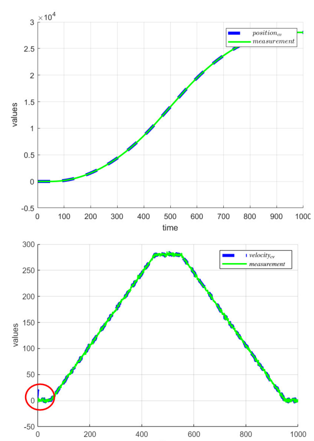

Figure 20: position and velocity of CV model with x_0=10

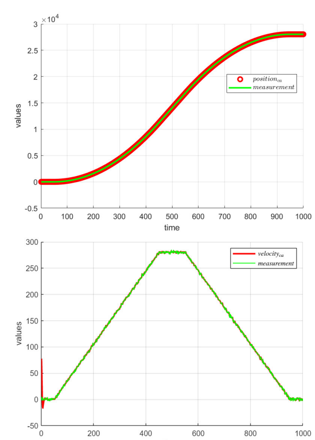

Figure 21: position and velocity of CA model with x_0=10

According to the plot shown above, we can know that: if the process is low, the kalman filter tend to trust the prediction generated from the motion model(figure 14,15). However, if the model chosen is wrong, our kalman filter will not be so correct (CV model fits bad (obvious shift shown in the figure 14 of CV model), however the CA model fits well).
However, if the process noise is larger than our process noise, we tend to trust our measurement rather than our prediction(figure 16,17).It fits very well, but it ignore the data come from the motion process. So, we choose Q=1 as our process noise,as is shown in figure 18,19, which is more accurate. However, the initial condition may also affect our result, as is shown in the figure 20, 21. Initially, the curve changed a lot, but with the addition of measured values, the wrong prior was gradually corrected and fitted with the curve.
Thus, we choose parameters show below to work for our model:
$x_{cv} = [1; 1]; P_{cv} = [1\ 0; 0\ 1]; x_{ca} = [1; 1; 1]; P_{ca} = [1\ 0\ 0; 0\ 1\ 0; 0\ 0\ 1];$
$Q_{cv} = [0\ 0; 0\ 1]; Q_{ca} = [0\ 0\ 0; 0\ 0\ 0; 0\ 0\ 1]$

2.4 Which model do you think fits better

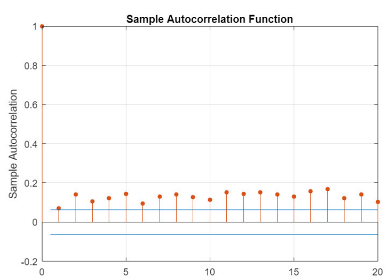

Figure 22: correlation of CV position Q=1

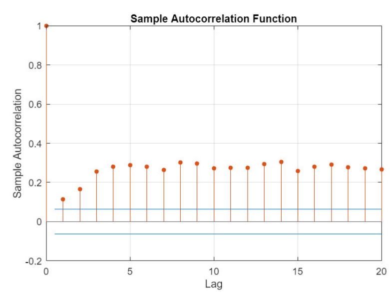

Figure 23: correlation of CV velocity Q=1

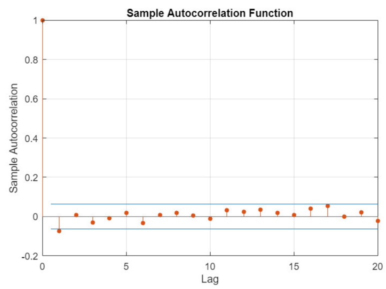

Figure 24: correlation of CA position Q=1

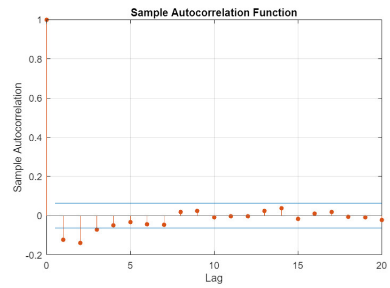

Figure 25: correlation of CA velocity Q=1

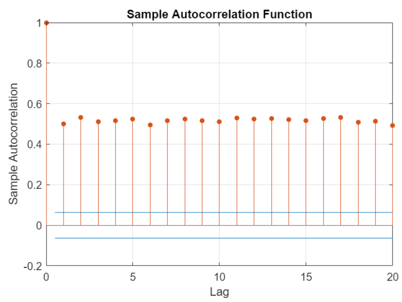

Figure 26: correlation of CV position Q=0.1

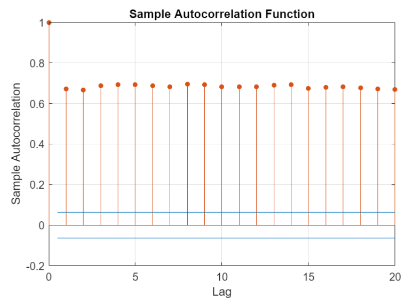

Figure 27: correlation of CV velocity Q=0.1

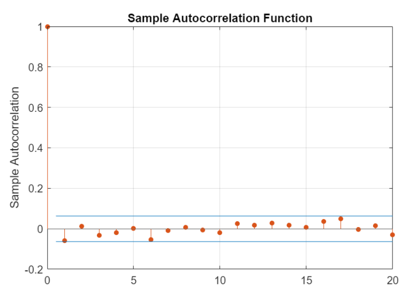

Figure 28: correlation of CA position Q=0.1

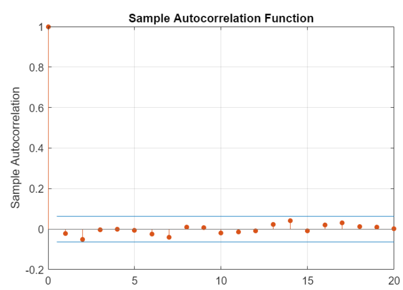

Figure 29: correlation of CA velocity Q=0.1

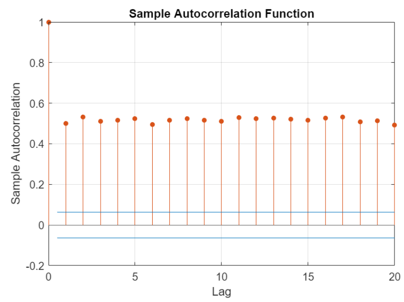

Figure 30: correlation of CV position Q=10

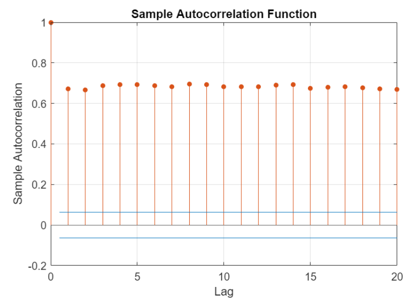

Figure 31: correlation of CV velocity Q=10

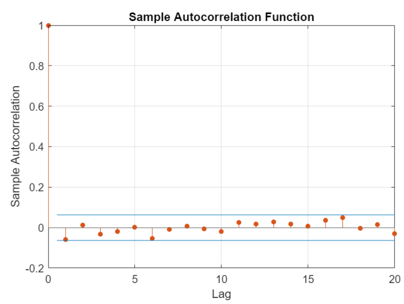

Figure 32: correlation of CA position Q=10

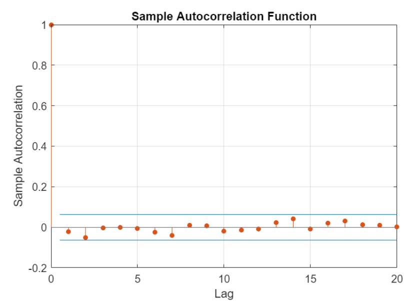

Figure 33: correlation of CA velocity Q=10

According to the plot shown above, we can see that the v_k of CV model are more related than the CA model, which means that the CA model performs better than the CA model-the kalman filter can deal with the uncertainty perfectly.

Additionally, since the car in this problem is doing variable acceleration, it is better to choose CA model as the motion model, even through it is more complex, computationally intensive.

Comparison of CV model and CA model:

Constant Velocity Model:

Pros: Simple, computationally efficient, good for smooth and continuous motion.

Cons: Limited to constant velocity, not suitable for sudden accelerations or

changes in direction.
Constant Acceleration Model:
Pros: Can handle changes in acceleration, useful for accelerating/decelerating objects.
Cons: More complex, computationally intensive, more prone to over fitting.