

# Assignment 4

● Graded

2 Days, 19 Hours Late

## Student

Theodore Johann Wilkening

## Total Points

16 / 16 pts

### Question 1

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0.75 pts Dimensions of G are not correct.

+ 0.5 pts Your G matrix is incorrect. See solution

+ 0.5 pts Your F matrix is incorrect

### Question 2

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0.75 pts Dimensions of H are not correct

### Question 3

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0.75 pts Actually, you don't need an approximation. The calculation of the matrix exponential is exact.

+ 0.75 pts You're going a bit fast here...

### Question 4

(no title)

1 / 1 pt

✓ + 1 pt Correct

### Question 5

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 1 pt I was asking the question to avoid looking at the code :(

### Question 6

(no title)

1 / 1 pt

✓ + 1 pt Code submitted

+ 0 pts No code submitted

### Question 7

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0 pts No solution

### Question 8

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 1 pt Correct, but you're using the state after the measurement update, not after the time update

+ 0.5 pts Click here to replace this description.

+ 0 pts No solution

### Question 9

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0 pts That's not correct

+ 0 pts No solution

### Question 10

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0.75 pts Ok but you're going too fast

+ 0.75 pts you didn't finish the calculation

+ 0.5 pts Not sure what's going on

+ 0 pts No submission

### Question 11

(no title)

1 / 1 pt

✓ + 1 pt Correct

+ 0 pts No solution

### Question 12

(no title)

1 / 1 pt

+ 1 pt Correct

+ 0.5 pts Click here to replace this description.

+ 0 pts No solution

### Question 13

(no title)

1 / 1 pt

+ 1 pt Correct

+ 0 pts No solution

+ 0.75 pts Click here to replace this description.

### Question 14

(no title)

1 / 1 pt

+ 1 pt Code submitted

+ 0 pts No code submitted

### Question 15

(no title)

1 / 1 pt

+ 1 pt Correct

+ 0 pts No solution

### Question 16

(no title)

1 / 1 pt

+ 1 pt Correct

+ 0 pts No solution

+ 0.5 pts Click here to replace this description.

### Question 17

Bonus for early submission

0 / 0 pts

+ 0.3 pts Early submission

+ 0 pts Late submission

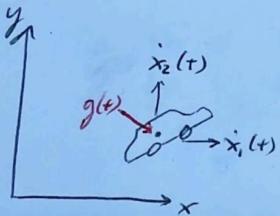
Question assigned to the following page: [1](#)

#1] Kalman Filter in Action

→ use Jupyter notebooks + python (numpy + scipy)

State Space model of a vehicle subject to unknown forces

single point w/ known mass, m, for simplicity



$$\vec{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \in \mathbb{R}^2 \quad (\text{vehicle moving in a plane})$$

$$\vec{g}(t) = \begin{bmatrix} g_1(t) \\ g_2(t) \end{bmatrix} \in \mathbb{R}^2 \quad \text{unknown force and time-varying vector}$$

$$\text{From Newtonian physics: } \ddot{g}(t) = m \frac{d^2 \vec{x}(t)}{dt^2}$$

We choose to model  $\vec{g}(t)/m$  as a 2D Gaussian white noise process with auto-correlation function  $\delta(t)$  s.t.  $R_w = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$$\frac{d^2 \vec{x}(t)}{dt^2} \triangleq \vec{w}(t) = \begin{bmatrix} w_1(t) \\ w_2(t) \end{bmatrix}$$

Set  $x_3(t) \triangleq \frac{dx_1(t)}{dt}$  and  $x_4(t) \triangleq \frac{dx_2(t)}{dt}$  to consider augmented state of car:

$$\vec{\tilde{x}}(t) \triangleq \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

Q1] Show the evolution of  $\vec{\tilde{x}}(t)$  is governed by the matrix differential eqtn:

$$\frac{d \vec{\tilde{x}}(t)}{dt} = F \vec{\tilde{x}}(t) + G \vec{w}(t) \quad (1)$$

$$\Rightarrow \frac{d \vec{\tilde{x}}(t)}{dt} = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \frac{d^2 x_1}{dt^2} \\ \frac{d^2 x_2}{dt^2} \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ w_1(t) \\ w_2(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\triangleq F} \vec{\tilde{x}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\triangleq G} \vec{w}$$

Questions assigned to the following page: [2](#) and [3](#)

Q2] position of vehicle: measured through a noisy sensor  $\vec{y} = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} \in \mathbb{R}^2$

which is a corrupted version of the true position by a 2D Gaussian white noise process  $\vec{v}(t)$  w/ auto-correlation  $\sigma^2 \delta(t) \Rightarrow R_v = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$

Show: measurement equation given by  $\vec{y} = H\vec{x} + \vec{v}(t)$  & specify  $H$ :

$$\vec{y}(t) = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}}_{\triangleq H} \vec{x}(t) + \vec{v}(t) \quad \left( = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} \right) \quad (2)$$

Simulations & filters run in discrete-time, need to sample at a period  $\Delta$ .  
Therefore only consider state of vehicle at times  $t_k \in \{k\Delta : k \in \mathbb{N}\}$

Discretization: takes some care to get a meaningful time model.

One can show the solution of (1) satisfies for  $k \geq 0$ :

$$\vec{x}(t_{k+1}) = e^{E\Delta} \vec{x}(t_k) + \underbrace{\int_0^\Delta \exp(E(\Delta-\tau)) G \vec{w}(\tau+t_k) d\tau}_{\triangleq \vec{u}(t_k)} \quad (3)$$

One can also show that the discrete-time process  $\{\vec{u}(t_k)\}_{k \geq 0}$  is Gaussian and white with a covariance matrix  $Q$  that can be computed from  $E$ ,  $G$ ,  $\Delta$  and the auto-correlation function of  $\vec{w}(t)$ . For  $\Delta \ll 1$ , one can show

$$Q = \Delta \begin{bmatrix} \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 1 & 0 \\ 0 & \frac{1}{2} & 0 & 1 \end{bmatrix} \quad (4)$$

Q3] writing  $\vec{x}_k \triangleq \vec{x}(t_k)$  and  $\vec{u}_k \triangleq \vec{u}(t_k)$  and using (3), show that the discretized state-space model when  $\Delta \ll 1$  is:

$$\vec{x}_{k+1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \vec{x}_k + \vec{u}_k \quad , \text{ where } \{\vec{u}_k\} \text{ is white with covariance } Q$$



Questions assigned to the following page: [3](#) and [5](#)

Q3 cont'd)

$$\underline{E} \Delta = \begin{bmatrix} 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \exp(\underline{E} \Delta) = e^{\underline{E} \Delta} \Big|_{t=1} = \mathcal{L}^{-1} \left\{ (sI - \underline{E})^{-1} \right\} \Big|_{t=1}$$

$$(sI - \underline{E})^{-1} = \begin{pmatrix} s & 0 & -\frac{1}{2} & 0 \\ 0 & s & 0 & -\frac{1}{2} \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & s \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{s} & 0 & \frac{1}{2s} & 0 \\ 0 & \frac{1}{s} & 0 & \frac{1}{2s} \\ 0 & 0 & \frac{1}{s} & 0 \\ 0 & 0 & 0 & \frac{1}{s} \end{pmatrix}$$

$$\mathcal{L} \left\{ (sI - \underline{E})^{-1} \right\} = \begin{pmatrix} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow \mathcal{L} \left\{ (sI - \underline{E})^{-1} \right\} \Big|_{t=1} = \begin{pmatrix} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \exp(\underline{E} \Delta)$$

Thus the discretized state-space model from ③ when  $\Delta \ll 1$  is

$$\vec{x}_{k+1} = \begin{bmatrix} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \vec{x}_k + \vec{u}_k, \text{ where } \{\vec{u}_k\} \text{ is white noise covariance } Q \text{ given above.}$$

Q4 See jupyter notebook for plot of data.

Q5 Kalman Filter equations

From class: Time Update:

$$1) \text{ (prediction)} \quad \begin{cases} \hat{x}_{i|i-1} = \underline{F}_{i-1} \hat{x}_{i-1|i-1} \\ P_{i|i-1} = \underline{F}_{i-1} P_{i-1|i-1} \underline{F}_{i-1}^T + G Q_{i-1} G^T \end{cases}$$

2) Measurement Update:

$$\begin{cases} \hat{x}_{i|i} = \hat{x}_{i|i-1} + K_{f,i} (y_i - H_i \hat{x}_{i|i-1}) \\ K_{f,i} = P_{i|i-1} H_i^T (H_i P_{i|i-1} H_i^T + R_i)^{-1} \\ P_{i|i} = P_{i|i-1} - K_{f,i} H_i P_{i|i-1} = (I - K_{f,i} H_i) P_{i|i-1} \end{cases}$$

Question assigned to the following page: [6](#)

To write out the equations further:

1) Time Update:

$$\hat{x}_{i|i-1} = F \hat{x}_{i-1|i-1}$$

no dependence on time

$$P_{i|i-1} = F P_{i-1|i-1} F^T + G Q_{i|i-1} G^T$$

Note!

2) Measurement Update:

$$\hat{x}_{i|i} = \hat{x}_{i|i-1} + K_{f,i} u_{f,i} \quad \text{--- same as previous actually}$$

## Q6 | Implement Kalman Filter

(try both update Smt + prediction first)

State Space Model:

$$\vec{x}_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_F \vec{x}_k + G \vec{u}_k \quad G = \text{Identity}(4)$$
$$\vec{y}_k = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}}_H \vec{x}_k + \vec{v}_k$$

$$K_{f,i} = P_{i|i-1} H_i^T (H_i P_{i|i-1} H_i^T + R_i)^{-1}$$
$$(4 \times 4)(4 \times 2)(2 \times 4)(4 \times 4)(4 \times 2) + 2 \times 2$$
$$(4 \times 4 \times 4 \times 2 \times 2)$$
$$= (4 \times 2) \quad \boxed{\quad} \checkmark$$

See Jupyter Notebook for code

Question assigned to the following page: [9](#)

Markov chain ✓ (uses cond. indep.)  
Conditional Independence. ✓

Q9

Given our discrete-time probabilistic state-space model

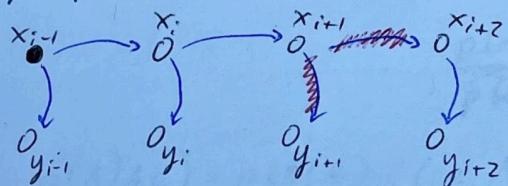
$$x_{i+1} \sim p(x_{i+1} | x_{0:i}, y_{0:i}) \text{ and } y_i \sim p(y_i | x_{0:i}, y_{0:i-1})$$

that is Markovian & satisfies the conditional independent property of measurements:

$$p(x_{i+1} | x_{0:i}, y_{0:i}) = p(x_{i+1} | x_i) \text{ and } p(y_i | x_{0:i}, y_{0:i-1}) = p(y_i | x_i)$$

Show that  $p(x_i | \cancel{x_{i+1}}, y_{0:n}) = p(x_i | x_{i+1}, y_{0:i})$

1) Sub-graph of our probabilistic state-space model:



2) Condition on  $x_{i+1}$  shown by breaking the edge coming out of  $x_{i+1}$  (in red)

This shows us visually the conditional independent relationships.

As we can see, by breaking the edges out of  $x_{i+1}$ , we have that  $y_{0:i:n}$  are no longer connected to  $x_i$  by any edge. Thus, by the Functional Dependence Graph theorem and definition given in class,

$$\boxed{p(x_i | x_{i+1}, y_{0:n}) = p(x_i | x_{i+1}, y_{0:i})}$$

□

Question assigned to the following page: [10](#)

Q10 Using Bayes rule and the properties of the state space model show that:

$$p(x_i | x_{i+1}, y_{0:n}) = \frac{p(x_{i+1} | x_i) p(x_i | y_{0:i})}{p(x_{i+1} | y_{0:i})}$$

Bayes Rule:  $P(A|B) = \frac{P(B|A) P(A)}{P(B)}$ ,  $p(A, B) = P(A|B) P(B)$

let  $A = x_i$ ,  $B = x_{i+1}$ ,  $C = y_{0:n}$

then we have:  $p(A|B, C) = \frac{p(B|A) p(A|C)}{p(B|C)}$  ~~as per derived relationship.~~

Using Bayes Rule,  $p(A|B, C) = \frac{p(BC|A) p(A)}{p(BC)}$   
~~=  $\frac{p(ABC)}{p(BC)}$  (as, but nowhere we want to go)~~

Note:  $p(B|AC) = p(B|A)$  (i.e. Markovian Property)

s.t.  $p(A|BC) = \frac{p(ABC)}{p(BC)} = \frac{* p(AC) p(B|AC)}{p(BC)} = \frac{p(A|C) p(C) p(B|AC)}{p(C) p(B|C)}$

$$p(B|AC) = \frac{p(ABC)}{p(AC)} \quad * = \text{Bayes Rule used.} \quad = \frac{p(B|AC) p(A|C)}{p(B|C)}$$

(Bayes Rule)

Using conditional  $\rightarrow = \frac{p(B|C) p(A|C)}{p(B|C)}$   
 independent property  
 of Markovian)

$$= \frac{p(x_{i+1}|x_i) p(x_i | y_{0:i})}{p(x_{i+1} | y_{0:i})} \quad \square$$

Question assigned to the following page: [11](#)

Q11 Show that the joint distribution of  $x_i$  and  $x_{i+1}$  given  $y_{0:n}$  is given by

$$p(x_i, x_{i+1} | y_{0:n}) = \frac{p(x_{i+1} | x_i) p(x_i | y_{0:i}) p(x_{i+1} | y_{0:i})}{p(x_{i+1} | y_{0:i})}$$

$$\text{So, } p(x_i, x_{i+1} | y_{0:n}) = \frac{p(x_i, x_{i+1}, y_{0:n})}{p(y_{0:n})} = \frac{p(x_{i+1} | x_i, y_{0:n}) p(x_i, y_{0:n})}{p(x_{i+1} | y_{0:i})}$$

did not get us where we wanted. Looking first at results from Q9 & Q10:

$$p(x_i | x_{i+1}, y_{0:n}) = p(x_i | x_{i+1}, y_{0:i}) \quad (11-1)$$

$$p(x_i | x_{i+1}, y_{0:n}) = \frac{p(x_{i:n} | x_i) p(x_i | y_{0:i})}{p(x_{i+1} | y_{0:i})} \quad (11-2)$$

noting that 11-2 has the denominator we are looking for, we use Bayes' Theorem to get the term  $p(x_i | x_{i+1}, y_{0:n})$  from our mixed distribution. (11-3)

$$p(x_i, x_{i+1} | y_{0:n}) \quad (11-3)$$

$$\Rightarrow p(x_i, x_{i+1} | y_{0:n}) = \frac{p(x_i, x_{i+1}, y_{0:n})}{p(y_{0:n})} = \frac{p(x_i | x_{i+1}, y_{0:n}) p(x_{i+1}, y_{0:n})}{p(y_{0:n})}$$

sub in (11-2):

$$= \frac{p(x_{i+1} | x_i) p(x_i | y_{0:i}) p(x_{i+1}, y_{0:n})}{p(x_{i+1} | y_{0:i}) p(y_{0:n})}$$

use Bayes' Rule:

$$= \frac{p(x_{i+1} | x_i) p(x_i | y_{0:i}) p(x_{i+1} | y_{0:n}) p(y_{0:n})}{p(x_{i+1} | y_{0:i}) p(y_{0:n})} = \frac{p(x_{i+1} | x_i) p(x_i | y_{0:i}) p(x_{i+1} | y_{0:n})}{p(x_{i+1} | y_{0:i})}$$

□ F7

Question assigned to the following page: [12](#)

Q-12] Assume  $x_{i-1|y_{0:i-1}}$  is distributed according to Gaussian distribution

$$\mathcal{N}(\hat{x}_{i-1|0:i-1}, P_{i-1|0:i-1})$$

>Show that joint distribution of  $x_{i-1}$  and  $x_i$  given  $y_{0:i-1}$  is:

$$\mathcal{N}\left(\begin{bmatrix} \hat{x}_{i-1|0:i-1} \\ \hat{x}_{i|0:i-1} \end{bmatrix}, \begin{bmatrix} P_{i-1|0:i-1} & P_{i-1|0:i-1} F^T \\ FP_{i-1|0:i-1} & P_{i|0:i-1} \end{bmatrix}\right)$$

where  $\hat{x}_{i|0:i-1} = F\hat{x}_{i-1|0:i-1}$

$$P_{i|0:i-1} = FP_{i-1|0:i-1}F^T + Q$$

Gauss-Markov model:  $x_{i+1} = Fx_i + u_i$ ,  $y_i = Hx_i + v_i$

$$R_u = QS_{ij}, R_v = RS_{ij}, \{u_i\}, \{v_i\}$$
 are white & Gaussian

from page 8 of Lecture 18 we have the following Lemma:

[Lemma: let  $x \sim \mathcal{N}(\mu_x, \Sigma_x)$  and  $y|x \sim \mathcal{N}(Hx, R)$  (e.g.  $y = Hx + v$ )  
 Then  $\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ H\mu_x \end{bmatrix}, \begin{bmatrix} \Sigma_x & \Sigma_x H^T \\ H\Sigma_x & H\Sigma_x H^T + R \end{bmatrix}\right)$ ]

we are given the distribution of  $x_{i-1|y_{0:i-1}}$  and the state-space model  $x_{i+1} = Fx_i + u_i$ . Thus if we set  $x = x_{i-1|y_{0:i-1}}$  and  $y = x_i|y_{0:i-1}$  then we have  $y = Fx + u$ , which matches the form given in the example of the Lemma.

To prove the result to ourselves, we will actually calculate the joint distribution:

$$\text{let } \mathcal{N}(\hat{x}_{i-1|0:i-1}, P_{i-1|0:i-1}) = \mathcal{N}(\mu_x, \Sigma_x) \text{ i.e. } \mu_x \triangleq \hat{x}_{i-1|0:i-1} \\ \Sigma_x \triangleq P_{i-1|0:i-1}$$

First, we calculate the mean of  $x_i|y_{0:i-1}$ :

$$\hat{x}_{i|0:i-1} = \mathbb{E}(x_i|y_{0:i-1}) = \mathbb{E}(Fx_{i-1} + u_i|y_{0:i-1}) = F\hat{x}_{i-1|0:i-1} + \underbrace{\mathbb{E}(u_i|y_{0:i-1})}_0 = F\hat{x}_{i-1|0:i-1} = F\mu_x$$

process noise is independent of past measurements  
uncorrelated &  $\mathbb{E}(u_i) = 0$

Question assigned to the following page: [12](#)

Q12 cont'd

Covariance of  $x_i | y_{0:i-1}$

$$\begin{aligned}
 & \mathbb{E}((x_i - \mathbb{E}(x_i | y_{0:i-1}))(x_i - \mathbb{E}(x_i | y_{0:i-1}))^T | y_{0:i-1}) \\
 &= \mathbb{E}((x_i - F\mu_x)(x_i - F\mu_x)^T | y_{0:i-1}) \\
 &= \mathbb{E}(x_i x_i^T - x_i \mu_x^T F^T - F\mu_x x_i^T + F\mu_x \mu_x^T F^T | y_{0:i-1}) \\
 &= \mathbb{E}((\mathbb{E}_{x_{i-1}} + u_{i-1})(F_{x_{i-1}} + u_{i-1})^T | y_{0:i-1}) - F\mu_x \mu_x^T F^T - F\mu_x u_{i-1}^T F^T + F\mu_x u_{i-1}^T F^T \\
 &= F \cancel{\mu_x} F^T + F \mathbb{E}[x_{i-1} u_{i-1}^T | y_{0:i-1}] + \mathbb{E}[u_{i-1} x_{i-1}^T | y_{0:i-1}] F^T + \mathbb{E}[u_{i-1} u_{i-1}^T | y_{0:i-1}] \xrightarrow{\text{uncorrelated}} -F\mu_x \mu_x^T F^T \\
 &= F \mathbb{E}(x_{i-1} x_{i-1}^T | y_{0:i-1}) + F \mathbb{E}[x_{i-1} | y_{0:i-1}] \mathbb{E}[u_{i-1}^T | y_{0:i-1}]^T + \mathbb{E}[u_{i-1} | y_{0:i-1}] \mathbb{E}[x_{i-1}^T | y_{0:i-1}] F^T + Q - F\mu_x \mu_x^T F^T \\
 &\quad \xrightarrow{x_{i-1} \& u_{i-1} \text{ independent}} \xrightarrow{\{u_i\} \text{ white noise, (0-mean)}} \\
 &= F \underbrace{(\mathbb{E}(x_{i-1} x_{i-1}^T | y_{0:i-1}) - \mu_x \mu_x^T)}_{= \Sigma_x} F^T + Q \\
 &= \Sigma_x = \mathbb{E}((x_{i-1} - \mu_x)(x_{i-1} - \mu_x)^T | y_{0:i-1}) = P_{i-1 | 0:i-1} \\
 &= F P_{i-1 | 0:i-1} F^T + Q \\
 &\triangleq P_{i-1 | 0:i-1} \text{ as given in problem statement. } \square
 \end{aligned}$$

Similarly, cross-covariance of  $x_{i-1} | y_{0:i-1}$  and  $x_i | y_{0:i-1}$ ;

$$\begin{aligned}
 & \mathbb{E}((x_{i-1} - \mu_x)(x_i - F\mu_x)^T | y_{0:i-1}) = \mathbb{E}(x_{i-1} x_i^T + \mu_x \mu_x^T F^T - \mu_x x_i^T - x_{i-1} \mu_x^T F^T | y_{0:i-1}) \\
 &= \mathbb{E}(x_{i-1} (F_{x_{i-1}} + u_{i-1})^T | y_{0:i-1}) - \mu_x \mathbb{E}(F_{x_{i-1}} + u_{i-1} | y_{0:i-1})^T - \cancel{\mu_x \mu_x^T F^T} + \cancel{\mu_x \mu_x^T F^T} \\
 &= \mathbb{E}(x_{i-1} x_{i-1}^T F^T | y_{0:i-1}) + \mu_x \underbrace{\mathbb{E}[u_{i-1}^T | y_{0:i-1}]}_{\substack{\text{uncorrelated} \\ \text{white, Gaussian}}} - \mu_x ((F\mu_x)^T + \mathbb{E}[u_{i-1}^T | y_{0:i-1}])^T \\
 &\quad \xrightarrow{\substack{u_{i-1} \text{ independent of} \\ \{u_i\}, \{v_i\} \text{ white, Gaussian}}} \\
 &\quad \xrightarrow{0-\text{mean}}
 \end{aligned}$$

$$\begin{aligned}
 &= \mathbb{E}(x_{i-1} x_{i-1}^T | y_{0:i-1}) F^T + 0 - \mu_x \mu_x^T F^T = (\mathbb{E}(x_{i-1} x_{i-1}^T | y_{0:i-1}) - \mu_x \mu_x^T) F^T \\
 &= \Sigma_x F^T \quad \square
 \end{aligned}$$

Since  $R_{xy} = R_{yx}^T$ ,  $\mathbb{E}((x_{i-1} - \mu_x)(x_i - \mu_x)^T | y_{0:i-1}) = (\Sigma_x F^T)^T = F \Sigma_x$  □ 19

Question assigned to the following page: [12](#)

Therefore,

$$\begin{bmatrix} \hat{x}_{i-1|0:i-1} \\ \hat{x}_i|y_{0:i-1} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ F\mu_x \end{bmatrix}, \begin{bmatrix} \Sigma_x & \Sigma_x F^T \\ F\Sigma_x & F\Sigma_x F^T + Q \end{bmatrix}\right)$$

where  $\Sigma_x = P_{i-1|0:i-1}$ ,  $\mu_x = \hat{x}_{i-1|0:i-1}$

□

Question assigned to the following page: [13](#)

Q+3] Using the result of Q-12, show that the distribution of  $x_{i-1}$  given  $x_i$  and  $y_{0:i-1}$ , which is equal to distribution of  $x_{i-1}$  given  $x_i$  and  $y_{0:i-1}$  by Q-9, is

$$x_{i-1}|x_i, y_{0:i-1} \sim N(\tilde{x}_{i-1|0:i-1}, \tilde{P}_{i-1|0:i-1})$$

$$\text{where } \tilde{x}_{i-1|0:i-1} = \hat{x}_{i-1|0:i-1} + k_i(x_i - F\hat{x}_{i-1|0:i-1})$$

$$\tilde{P}_{i-1|0:i-1} = P_{i-1|0:i-1} - k_i(FP_{i-1|0:i-1}F^T + Q)k_i^T$$

$$k_i = P_{i-1|0:i-1}F^T(FP_{i-1|0:i-1}F^T + Q)^{-1}$$

~~mean:  $E(x_{i-1}|x_i, y_{0:i-1}) = E(x_{i-1}|x_i, y_{0:i-1})$  by Q-9 (independence)~~

$$E(x_{i-1}|x_i, y_{0:i-1}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_{i-1} p(x_{i-1}|x_i, y_{0:i-1}) dx_{i-1}$$

$$\text{note from Q-10: } p(x_{i-1}|x_i, y_{0:i-1}) = \frac{p(x_{i-1}|x_i)p(x_i|y_{0:i-1})}{p(x_{i-1}|y_{0:i-1})}$$

$$\begin{aligned} \hat{x}_{i-1|0:i-1} &= E(x_{i-1}|y_{0:i-1}) \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_{i-1} p(x_{i-1}|y_{0:i-1}) dx_{i-1} \end{aligned}$$

From Lecture #18 pg 7 if  $\begin{bmatrix} x \\ y \end{bmatrix} \sim N\left(\begin{bmatrix} M_x \\ M_y \end{bmatrix}, \begin{bmatrix} R_x & R_{xy} \\ R_{yx} & R_y \end{bmatrix}\right)$

then  $x \sim N(M_x, R_x)$ ,  $y \sim N(M_y, R_y)$  and

$$x|y \sim N(M_x + R_{xy}R_y^{-1}(y - M_y), R_x - R_{xy}R_y^{-1}R_{yx}) \quad \text{with} \quad \leftarrow$$

$$y|x \sim N(M_y + R_{yx}R_x^{-1}(x - M_x), R_y - R_{yx}R_x^{-1}R_{xy})$$

let  $x_{i-1}|y_{0:i-1} = x$  and  $x_i|y_{0:i-1} = y$  s.t.

$$\begin{aligned} \hat{x}_{i-1|0:i-1} &= M_x & P_{i-1|0:i-1} &= R_x & P_{i-1|0:i-1}F^T &= R_{xy} & \left. \begin{array}{l} \text{from} \\ \text{Q-12} \end{array} \right\} \\ F\hat{x}_{i-1|0:i-1} &= \hat{x}_i|0:i-1 = M_y & FP_{i-1|0:i-1}F^T + Q &= R_y & FP_{i-1|0:i-1} &= R_{yx} & \rightarrow \end{aligned}$$

Question assigned to the following page: [13](#)

$$\text{and } \mathbf{x}_{i-1} | \mathbf{x}_i, \mathbf{y}_{0:i-1} = \mathbf{x}_{i-1} | \mathbf{x}_i, \mathbf{y}_{0:i-1} = \mathbf{x} | \mathbf{y}$$

thus:

$$\begin{aligned}
 \mathbf{x}_{i-1} &= \mathbf{x} + R_{xy} R_y^{-1} (\mathbf{y} - \mathbf{M}_y) \\
 = \mathbb{E}(\mathbf{x}_{i-1} | \mathbf{x}_i, \mathbf{y}_{0:i-1}) &= \mathbf{x}_{i-1|0:i-1} + P_{i-1|0:i-1} F^T (F P_{i-1|0:i-1} F^T + Q)^{-1} (\mathbf{x}_i | \mathbf{y}_{0:i-1} - \mathbf{x}_{i|0:i-1}) \\
 = \mathbb{E}(\mathbf{x} | \mathbf{y}) &\quad \downarrow \text{Markovian} \quad \downarrow \\
 &= \mathbf{x}_{i|0:i-1} \\
 &= \mathbf{x}_{i-1|0:i-1} + K_i (\mathbf{x}_i - F \mathbf{x}_{i-1|0:i-1}) \\
 &= \tilde{\mathbf{x}}_{i-1|i-1} \quad \square
 \end{aligned}$$

Note Now for the covariance:

$$\begin{aligned}
 \text{Cov}(\mathbf{x} | \mathbf{y}) &= R_x - R_{xy} R_y^{-1} R_{yx} \\
 \Rightarrow \tilde{P}_{i-1|i-1} &= P_{i-1|0:i-1} - \underbrace{P_{i-1|0:i-1} F^T (F P_{i-1|0:i-1} F^T + Q)^{-1} F P_{i-1|0:i-1}}_{K_i} \\
 &= P_{i-1|0:i-1} - K_i F P_{i-1|0:i-1} \\
 &= P_{i-1|0:i-1} - K_i (F P_{i-1|0:i-1} F^T + Q) (F P_{i-1|0:i-1} F^T + Q)^{-1} F P_{i-1|0:i-1}
 \end{aligned}$$

$$\begin{aligned}
 \text{Note: } K_i^T &= (F P_{i-1|0:i-1} F^T + Q)^{-T} F P_{i-1|0:i-1} \\
 &= (F P_{i-1|0:i-1}^T F^T + Q^T)^{-1} F P_{i-1|0:i-1} \\
 &= (F P_{i-1|0:i-1} F^T + Q)^{-1} F P_{i-1|0:i-1}
 \end{aligned}$$

where  $P_{i+1|0:i-1} = P_{i-1|0:i-1}^T$   
 self-covariance matrices are symmetric  
 and  $Q^T = Q = R_u$  (symmetric)

$$\Rightarrow \tilde{P}_{i-1|i-1} = P_{i-1|0:i-1} - K_i (F P_{i-1|0:i-1} F^T + Q) K_i^T \quad \square$$

Question assigned to the following page: [14](#)

Q-14] Implement the Kalman Filter for smoothing

Summary:

$$\hat{x}_{i|0:i} = (\text{predict})$$

$$\hat{x}_{i|0:i-1} = F \hat{x}_{i-1|0:i-1}$$

$$P_{i|0:i-1} = F P_{i-1|0:i-1} F^T + Q$$

$$\left. \begin{aligned} \hat{x}_{i|0:i} &= \hat{x}_{i|0:i-1} + K_{f,i} (y_i - H_i \hat{x}_{i|0:i-1}) \\ K_{f,i} &= P_{i|0:i-1} H_i^T (H_i P_{i|0:i-1} H_i^T + R_i)^{-1} \\ P_{i|0:i} &= P_{i|0:i-1} - K_{f,i} H_i P_{i|0:i-1} \end{aligned} \right\} \text{Kalman Filter}$$

$$\hat{x}_{i-1|i-1} = \hat{x}_{i-1|i-1,0:i-1} = \hat{x}_{i-1|0:i-1} + K_i (x_i - F \hat{x}_{i-1|0:i-1})$$

$$K_i = P_{i-1|0:i-1} F^T (F P_{i-1|0:i-1} F^T + Q)^{-1}$$

$$\tilde{P}_{i-1|i-1} = P_{i-1|i-1} - K_i (F P_{i-1|0:i-1} F^T + Q) K_i^T$$

$$\hat{x}_{i-1|0:n} = \hat{x}_{i-1|0:i-1} + K_i (\hat{x}_{i|0:n} - F \hat{x}_{i-1|0:i-1})$$

$$P_{i-1|0:n} = K_i P_{i|0:n} K_i^T + \tilde{P}_{i-1|i-1}$$

so, in Q-6 we produced  $\hat{x}_{i|0:i}$  and  $P_{i|0:i}$ , now we will use them to go backwards

and produce:

$$K_i, \quad \hat{x}_{i-1|0:n}$$

[See code in Jupyter]

Q-15] Plot the smoothed trajectory → see Jupyter

Q-16] Compute RMS for smoothed vs. true trajectory:  $\sqrt{\sum_{i=1}^n \|\hat{x}_{i|0:n} - x_i\|^2}$

& How does it compare? [See Jupyter code]

Question assigned to the following page: [4](#)

# Table of Contents

- 1 Q-4 Trajectory + Measurements Plot
- 2 Q-5 Kalman Filter Equations
- 3 Q-6 & Q-7 Implementing + Plot the Kalman Filter
- 4 Q-8 RMS Error
  - 4.1 RMS Conclusion
- 5 Q-14 & Q-15 Implementation + Plot of Kalman Smoother
- 6 Q-16 RMS Error of Kalman Smoother
  - 6.1 RMS Conclusion

# ECE6555 HW4

Author: Teo Wilkening Due Date: 2022/11/10

## Q-4 Trajectory + Measurements Plot



```
In [1]: import numpy as np
from scipy import signal

np.random.seed(1992)
NumSteps = 201
TimeScale = np.linspace(0,10,NumSteps)
DeltaSim = np.diff(TimeScale)[0]
SigmaInput = 1
SigmaNoise = 0.5

F = np.array([[1,0,DeltaSim,0],[0,1,0,DeltaSim],[0,0,1,0],[0,0,0,1]])
Q = SigmaInput**2 * np.array([[DeltaSim**3/3,0,DeltaSim**2/2,0],
                             [0,DeltaSim**3/3,0,DeltaSim**2/2],
                             [DeltaSim**2/2,0,DeltaSim,0],
                             [0,DeltaSim**2/2,0,DeltaSim]])
H = np.array([[1,0,0,0],[0,1,0,0]])
R = SigmaNoise**2 * np.identity(2)

State = np.zeros((4,NumSteps))
NoisyMeasurements = np.zeros((2,NumSteps))

for t in np.arange(1,NumSteps):
    ProcessNoise = np.squeeze(np.matmul(np.linalg.cholesky(Q),np.random.randn(4,1)))
    State[:,t] = np.matmul(F,State[:,t-1]) + ProcessNoise
    MeasurementNoise = SigmaNoise * np.squeeze(np.random.randn(2))
    NoisyMeasurements[:,t] = np.matmul(H,State[:,t]) + MeasurementNoise
```

Questions assigned to the following page: [6](#) and [4](#)

```

StateX1 = State[0,:]
StateX2 = State[1,:]

DownSampling=2
NoisyMeasurements = NoisyMeasurements[:,::DownSampling]
MeasurementY1 = NoisyMeasurements[0,:]
MeasurementY2 = NoisyMeasurements[1,:]

```

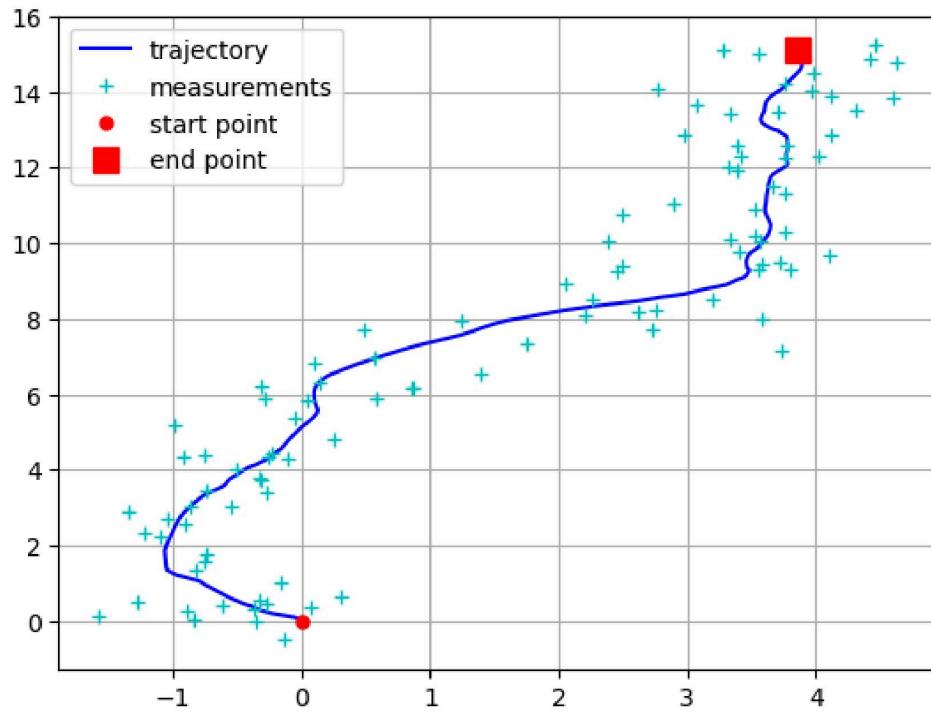
```

In [2]: import matplotlib.pyplot as plt

# plt.figure(figsize=(3,3), dpi = 180)
plt.figure(1)
plt.plot(StateX1,StateX2,'b')
plt.plot(MeasurementY1,MeasurementY2,'c+')
plt.grid(True)
plt.plot(StateX1[0],StateX2[0],'r.',markersize=10)
plt.plot(StateX1[-1],StateX2[-1],'rs',markersize=10)
plt.legend(['trajectory','measurements','start point','end point'])

plt.rcParams['figure.figsize'] = [6, 6]
plt.rcParams['figure.dpi'] = 120 # 200 e.g. is really fine, but slower
plt.show()

```



## Q-5 Kalman Filter Equations

(see written work)

## Q-6 & Q-7 Implementing + Plot the Kalman Filter

Question assigned to the following page: [6](#)



```
In [3]: # initialize all of the variables that we're going to need; F, H, G, Q, R
Delta = 0.1
Fk = np.array([[1,0,Delta,0],[0,1,0,Delta],[0,0,1,0],[0,0,0,1]])
Gk = np.identity(4)
Qk = np.array([[Delta**3/3,0,Delta**2/2,0],
               [0,Delta**3/3,0,Delta**2/2],
               [Delta**2/2,0,Delta,0],
               [0,Delta**2/2,0,Delta]])
Hk = np.array([[1,0,0,0],[0,1,0,0]])
Sigma = 0.5
Rk = Sigma**2 * np.identity(2)
```

```
In [10]: # the initial guesses of x and P
xhat_init = np.zeros((4,1))
P_init = np.identity(4)
NumSteps = len(MeasurementY1)

# initializing the matrices for computing Kalman filter state evolution over time
xhat_i_pred = np.zeros((4,NumSteps))
xhat_i_curr = np.zeros((4,NumSteps))
P_i_pred = np.zeros((NumSteps,4,4))
P_i_curr = np.zeros((NumSteps,4,4))
Kfi_curr = np.zeros((NumSteps,4,2))

# start running the Kalman Filter, using the NoisyMeasurements
for t in np.arange(0,NumSteps):
    # make the prediction update (time update)
    if t == 0:
        # use the initial guesses
        xhat_i_pred[:,t] = (Fk @ xhat_init).reshape(1,4)
        P_i_pred[t,:,:] = Fk @ P_init @ Fk.T + Gk @ Qk @ Gk.T
    elif t > 0:
        # then use the time-update (prediction) calculation
        # x i/i-1
        xhat_i_pred[:,t] = (Fk @ xhat_i_curr[:,t-1]).reshape(1,4)
        # P i/i-1
        P_i_pred[t,:,:] = Fk @ P_i_curr[t-1,:,:] @ Fk.T + Gk @ Qk @ Gk.T

    # make the measurement update
    # K f,i
    Kfi_curr[t,:,:] = P_i_pred[t,:,:] @ Hk.T @ np.linalg.inv(Hk @ P_i_pred[t,:,:] @ Hk.T + Rk)
    # P i/i
    P_i_curr[t,:,:] = (np.identity(4) - Kfi_curr[t,:,:] @ Hk) @ P_i_pred[t,:,:]
    # x i/i
    xhat_i_curr[:,t] = xhat_i_pred[:,t] + Kfi_curr[t,:,:] @ (NoisyMeasurements[:,t] -
                                                             Hk @ xhat_i_pred[:,t])

# plot the results
plt.figure()
plt.plot(StateX1,StateX2,'b')
plt.plot(MeasurementY1,MeasurementY2,'c+')
plt.grid(True)
plt.plot(StateX1[0],StateX2[0],'r.',markersize=10)
```

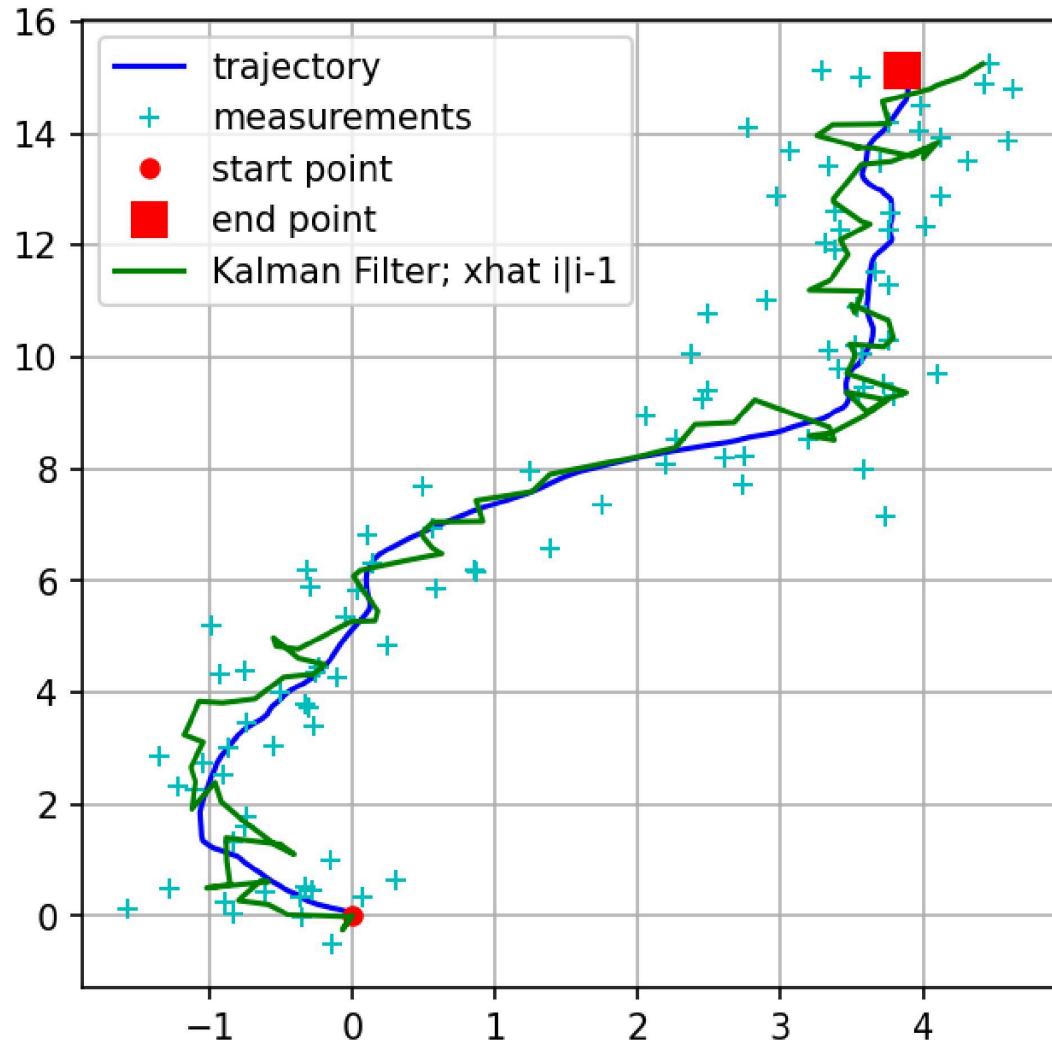
Questions assigned to the following page: [6](#), [8](#), and [7](#)

```

plt.plot(StateX1[-1],StateX2[-1],'rs',markersize=10)
plt.plot(xhat_i_pred[0,:],xhat_i_pred[1,:],'g')
plt.legend(['trajectory','measurements','start point',
           'end point','Kalman Filter; xhat i|i-1'])

plt.rcParams['figure.figsize'] = [4, 4]
plt.rcParams['figure.dpi'] = 120 # 200 e.g. is really fine, but slower
plt.show()

```



## Q-8 RMS Error



```

In [5]: StateX = State[:,::DownSampling][0:2,:]
error_measure = NoisyMeasurements - StateX
rms_measure = np.linalg.norm( np.linalg.norm( error_measure, ord=2, axis=0 ) ,ord=2)
error_kalman_f = xhat_i_pred[0:2,:] - StateX
rms_kalman_f = np.linalg.norm( np.linalg.norm( error_kalman_f, ord=2, axis=0), ord=2)

```

Questions assigned to the following page: [8](#) and [14](#)

```
print(f'''RMS of the noisy measurements: {rms_measure}''')
print(f'''RMS of the Kalman filter estimates: {rms_kalman_f}'''')

RMS of the noisy measurements: 7.085174167758084
RMS of the Kalman filter estimates: 3.889252921005451
```

## RMS Conclusion

Based on the above RMS calculation, the RMS of the Kalman filter is less than the RMS of the Noisy Measurements. This is indeed expected since we are using the Kalman filter to "filter" out some of the effects of the noise on our measurements.

## Q-14 & Q-15 Implementation + Plot of Kalman Smoother



```
In [12]: # Recall the initial setup for computing Kalman filter
# xhat_i_pred = np.zeros((4,NumSteps))           # xhat i|i-1
# xhat_i_curr = np.zeros((4,NumSteps))            # xhat i|i
# P_i_pred = np.zeros((NumSteps,4,4))             # P i|i-1
# P_i_curr = np.zeros((NumSteps,4,4))              # P i|i
# Kfi_curr = np.zeros((NumSteps,4,2))              # K f,i

# initialize new variables to hold the Kalman Smoother data
xhat_i_min_1_given_n = np.zeros((4,NumSteps))    # xhat i-1|0:n
Ki = np.zeros((NumSteps,4,4))                      # K_i

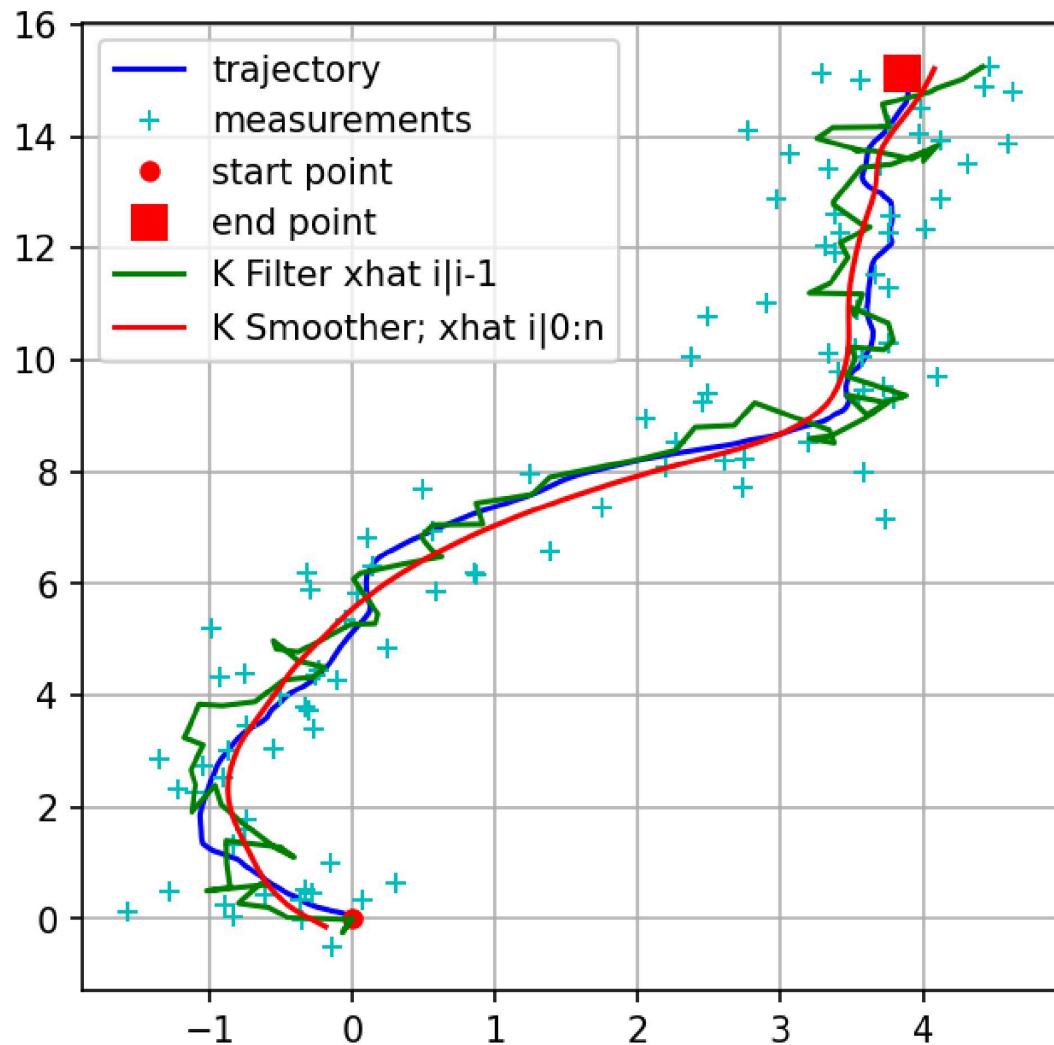
## Implement the Kalman Smoothing algorithm, given the Kalman Filtered data
# for xhat n|0:n, use the last step of the Kalman Filter
xhat_i_min_1_given_n[:,NumSteps-1] = xhat_i_curr[:,NumSteps-1]

# now for each time step compute the Kalman Smoothed state evolution
for t in np.arange(NumSteps-1,0,-1):
    # K_i
    Ki[t,:,:] = P_i_curr[t-1,:,:] @ Fk.T @ np.linalg.inv(Fk @ P_i_curr[t-1,:,:] @
                                                          Fk.T + Qk)
    # xhat i-1|0:n
    xhat_i_min_1_given_n[:,t-1] = (xhat_i_curr[:,t-1] +
                                    Ki[t,:,:] @ (xhat_i_min_1_given_n[:,t] -
                                                 Fk @ xhat_i_curr[:,t-1]))

# plot the results
plt.figure()
plt.plot(StateX1,StateX2,'b')
plt.plot(MeasurementY1,MeasurementY2,'c+')
plt.grid(True)
plt.plot(StateX1[0],StateX2[0],'r.',markersize=10)
plt.plot(StateX1[-1],StateX2[-1],'rs',markersize=10)
plt.plot(xhat_i_pred[0,:],xhat_i_pred[1,:],'g')
plt.plot(xhat_i_min_1_given_n[0,:],xhat_i_min_1_given_n[1,:],'r')
plt.legend(['trajectory','measurements','start point','end point',
           'K Filter xhat i|i-1','K Smoother; xhat i|0:n'])
```

Questions assigned to the following page: [16](#), [15](#), and [14](#)

```
plt.rcParams['figure.figsize'] = [6,6]
plt.rcParams['figure.dpi'] = 150 # 200 e.g. is really fine, but slower
plt.show()
```



## Q-16 RMS Error of Kalman Smoother



```
In [14]: StateX = State[:,::DownSampling][0:2,:]
error_measure = NoisyMeasurements - StateX
rms_measure = np.linalg.norm( np.linalg.norm( error_measure, ord=2, axis=0 ) ,ord=2)
error_kalman_f = xhat_i_pred[0:2,:] - StateX
rms_kalman_f = np.linalg.norm( np.linalg.norm( error_kalman_f, ord=2, axis=0), ord=2)
error_kalman_f_curr = xhat_i_curr[0:2,:] - StateX
rms_kalman_f_curr = np.linalg.norm( np.linalg.norm( error_kalman_f_curr,
                                                 ord=2, axis=0), ord=2)
error_kalman_s = xhat_i_min_1_given_n[0:2,:] - StateX
rms_kalman_s = np.linalg.norm( np.linalg.norm( error_kalman_s, ord=2, axis=0), ord=2)
```

Question assigned to the following page: [16](#)

```
print(f'''RMS of the noisy measurements: {rms_measure}''')
print(f'''RMS of the Kalman Filter xhat i|i-1 estimates: {rms_kalman_f}''')
print(f'''RMS of the Kalman Filter xhat i|i estimates: {rms_kalman_f_curr}''')
print(f'''RMS of the Kalman Smoother xhat i|0:n estimates: {rms_kalman_s}''')
print(f'''RMS Kalman Smoother/RMS Kalman Filter: {rms_kalman_s/rms_kalman_f}'''')
```

```
RMS of the noisy measurements: 7.085174167758084
RMS of the Kalman Filter xhat i|i-1 estimates: 3.889252921005451
RMS of the Kalman Filter xhat i|i estimates: 3.3022335707201216
RMS of the Kalman Smoother xhat i|0:n estimates: 2.2990220257280076
RMS Kalman Smoother/RMS Kalman Filter: 0.5911217584516627
```

## RMS Conclusion

Based on the above RMS calculation, the RMS of the Kalman Smoother is ~41% less than the RMS of the Kalman Filter. This is good news since we expect our RMS to improve when using all of the measurements for each step.