ENAE 441 - 0101

HW05: State Estimation

Due on December 4th, 2025 at 09:30 AM

Dr. Martin, 09:30 AM

Vai Srivastava

December 3rd, 2025

Problem 1: Batch Least Squares Estimation *(30 pts.)*

Imagine there is a spacecraft flying in a circular geosynchronous orbit around the Earth. To an observer on the rotating Earth, the spacecraft location appears static. You do not know the exact semi-major axis, but you can estimate it using a range measurement taken from a radio on the ground.

$$\rho = \sqrt{x^2 + y^2 + z^2}$$

Assume you are positioned directly beneath the satellite such that range measurement can be reduced to the following form:

$$\rho = \sqrt{z^2} = z$$

For **50** consecutive nights, you go outside and point your radio antenna towards the spacecraft and collect 200 noisy range measurements. The radio antenna is not particularly precise, so it has continuous measurement noise properties $\tilde{\boldsymbol{v}}(t) \sim \mathcal{N}(0, V)$ where $V = 100\,\mathrm{m}^2$. Use this information and the measurements provided in `HW5Measurements.npy`[1] to answer the following questions:

a. Express the system in continuous time state-space form assuming a state of

$$\boldsymbol{X} = \left[x, y, z, \dot{x}, \dot{y}, \dot{z}\right]^T$$

and convert to a discrete time state-space model assuming $\Delta t = 10\,\mathrm{s}$

b. Use batch least squares algorithm[2] to estimate the spacecraft state from the first trial in `HW5Measurements.npy`. Plot the estimated position as a function of $k = \{1, \ldots, 200\}$ measurements.

c. Repeat the batch estimation process for all **50** trials included in `HW5Measurements.npy`, and plot the resulting state estimates as a function of k along side their $\pm 3\sigma$ error bounds[3][4].

d. Using all 50 trials, plot histograms for each state estimate in $\hat{\mathbf{x}}$ after $k \in \{10, 50, 200\}$ measurements. Include the sample mean and error covariance for each distribution as annotations[5]. Do these values make sense? Why?

e. Measure the average amount of time it takes to compute an estimate as a function of k. Use Python's `time` library.

Solution

Part A

In matrix form:

$$\dot{\boldsymbol{X}}(t) = A_c\,\boldsymbol{X}(t),$$

with

$$A_c = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[1] The measurements provided are formatted as a $N \times M$ array where N corresponds to the trial and M corresponds to the number of measurements per trial and the units are **meters**.

[2] Hint: When you need to take a matrix inverse for BLLS and subsequent algorithms, use `np.linalg.pinv` rather than `np.linalg.inv` to avoid singularities.

[3] Recall that the diagonal elements of the state error covariance matrix yield σ_i^2.

[4] The `matplotlib` function `plt.fill_between` can be useful for the $\pm 3\sigma$ bounds that center on the mean.

[5] Use `np.nanmean` and `np.nanstd` over all values for a particular k.

Measurement model:

$$\rho\left(t\right) = H\,\boldsymbol{X}\left(t\right) + \tilde{\boldsymbol{v}}\left(t\right), \quad \tilde{\boldsymbol{v}}\left(t\right) \sim \mathcal{N}\left(0, V\right),$$

with

$$H = \left[\begin{array}{cccccc} 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right], \quad V = 100\,\mathrm{m}^2$$

The discrete state transition is:

$$\boldsymbol{X}_{k+1} = \boldsymbol{\Phi}\boldsymbol{X}_k, \quad \boldsymbol{\Phi} = e^{A_c \Delta t}$$

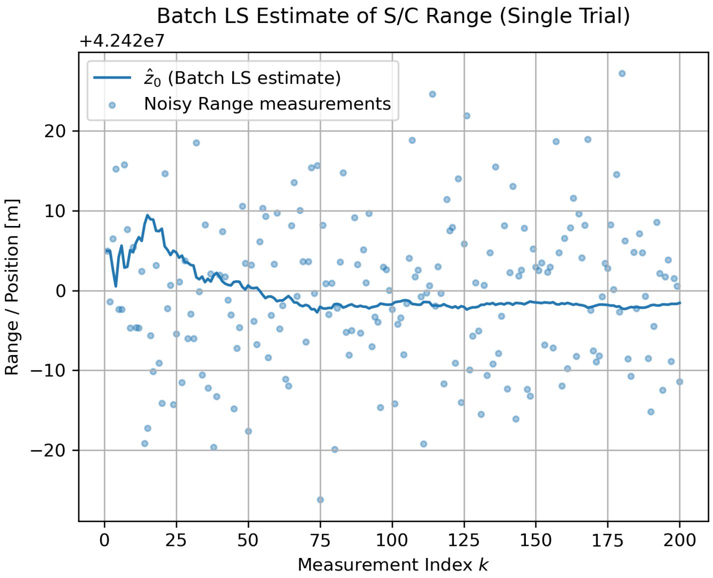
A_c has the structure of a constant-velocity model:

$$\boldsymbol{\Phi} = \left[\begin{array}{cc} \mathcal{I}_{3\times3} & \Delta t \mathcal{I}_{3\times3} \\ 0_{3\times3} & \mathcal{I}_{3\times3} \end{array}\right] = \left[\begin{array}{cccccc} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

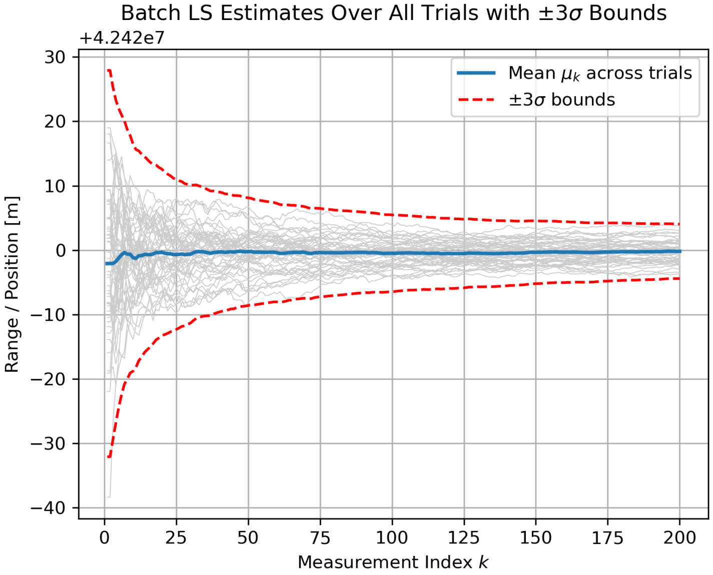
Thus, the measurement equation in discrete time is:

$$\boxed{\rho_k = H\boldsymbol{X}_k + v_k, \quad v_k \sim \mathcal{N}\left(0, V\right), \quad V = 100\,\mathrm{m}^2}$$

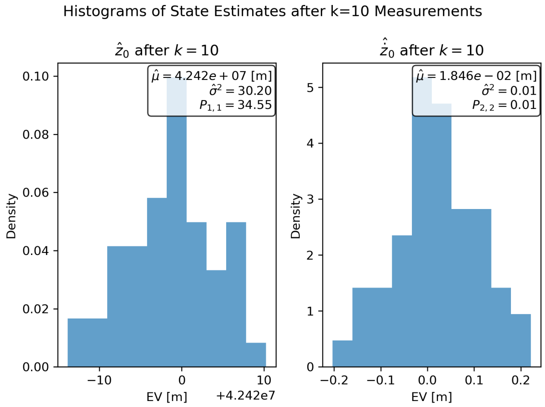
Part B

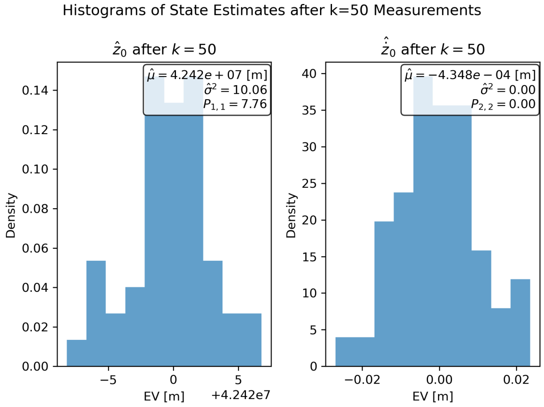


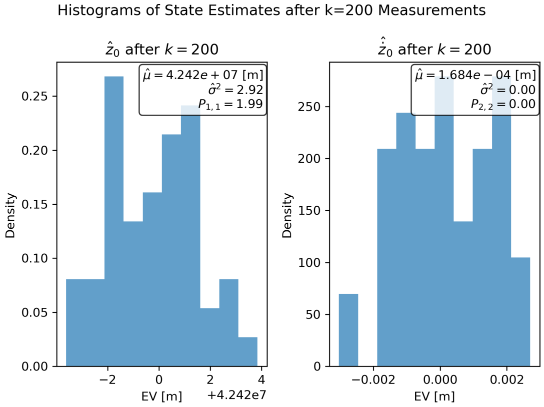
Part C



Part D







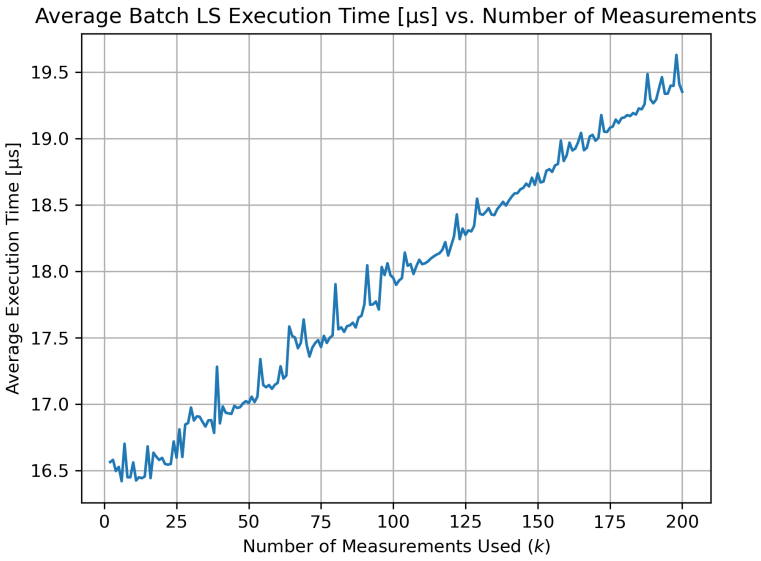
Problem 1 continued on next page...

```
For each k ∈ {10, 50, 200}, the histograms show the distribution of the
batch least-squares state estimates across the 50 trials.

As k increases, the histograms for ẑ₀ become narrower and more tightly
clustered around a common mean. The sample variances of ẑ₀ decrease
and approach the corresponding diagonal entries of the analytical
covariance Pₖ = (Γₖᵀ R⁻¹ Γₖ)⁻¹, which is expected for a linear
unbiased least-squares estimator driven by independent Gaussian noise.

The estimates of ẑ̇₀ are centered close to zero (since the spacecraft
is effectively stationary in the chosen frame), and their spread also
shrinks with increasing k, reflecting the fact that longer time spans
give more leverage to estimate the slope in the constant-velocity model.
```

Part E



Code

See the Python code for this problem.

Problem 2: Recursive Least Squares Estimation *(30 pts.)*

a. Using the same dataset, implement the recursive least squares algorithm to estimate $\hat{\mathbf{x}}$. Use an initial estimate of:

$$\hat{\mathbf{x}}_0 = \left[0\,\mathrm{km}, 0\,\mathrm{km}, 42 \times 10^3\,\mathrm{km}, 0\,\frac{\mathrm{km}}{\mathrm{s}}, 0\,\frac{\mathrm{km}}{\mathrm{s}}, 0\,\frac{\mathrm{km}}{\mathrm{s}}\right]$$

and a covariance of:

$$P_0 = \begin{bmatrix} 50 \cdot \mathbb{I}_{3\times 3}\mathrm{km}^2 & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & 1 \cdot \mathbb{I}_{3\times 3}\frac{\mathrm{km}^2}{\mathrm{s}^2} \end{bmatrix}$$

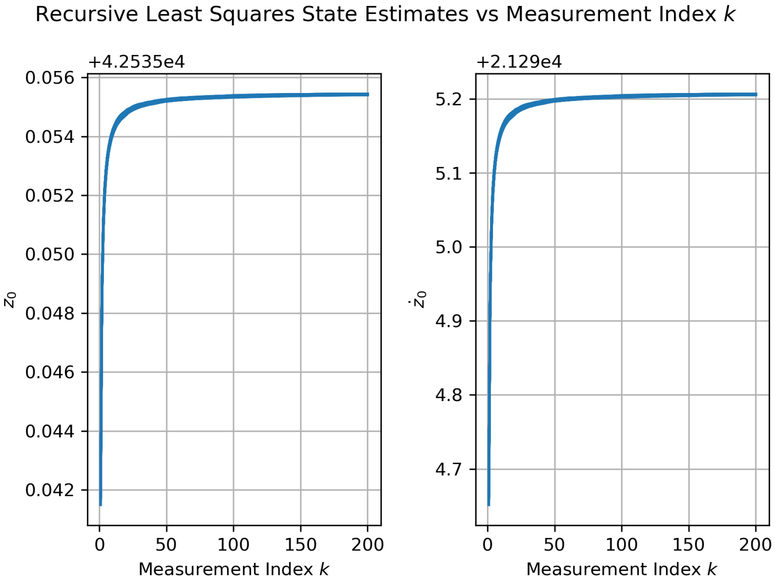
Plot the state estimate for the 50 trials as a function of k.

b. Plot the sample mean state estimate μ_k across all trials and the $\pm 3\sigma$ error bounds around the estimate. Does this mirror the same plot in Problem 1, Part D? Why or why not?

c. Measure/plot the average amount of time it takes to perform each update from $k = \{1, \ldots, 200\}$ and compare against the batch least squares estimator. Explain any differences.

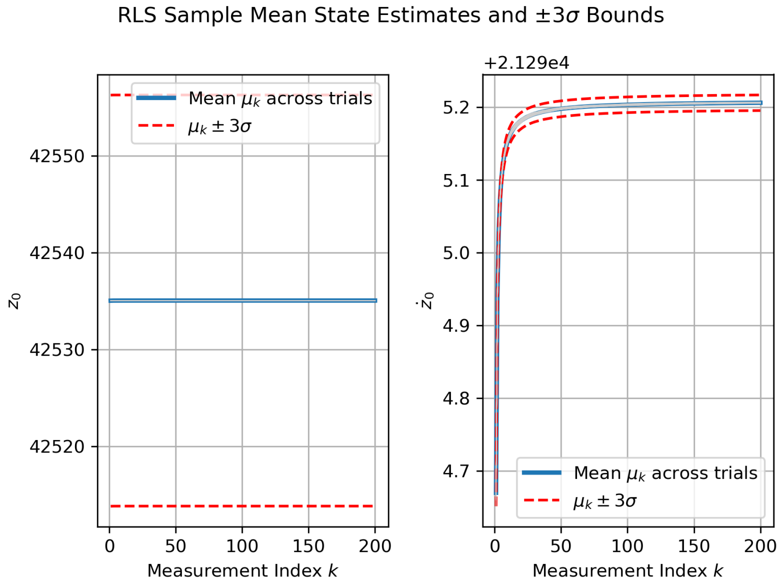
Solution

Part A



Problem 2 continued on next page...

Part B

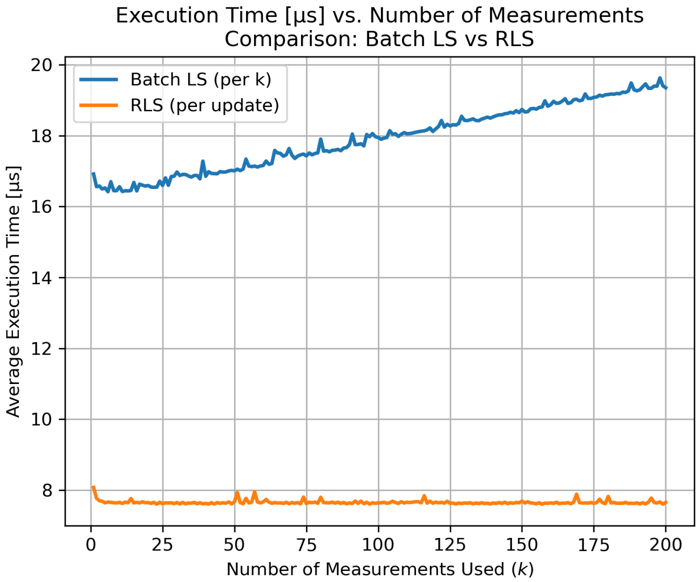


```
The plots show the sample mean state estimates μₖ across all 50 trials
together with ±3σ envelopes derived from the RLS covariance Pₖ.

Only the z-related states (z₀ and ż₀) change significantly with k,
because the measurements depend only on z. The x, y, ẋ and ẏ
components remain essentially fixed at their prior means and covariances,
reflecting the fact that they are unobservable in this simplified setup.

For the observable components, the behavior mirrors the batch least squares
results from Problem 1: as k increases, the mean converges toward the true
value and the ±3σ bounds shrink. This happens because recursive least
squares is algebraically equivalent to batch least squares for a linear
Gaussian model: RLS simply builds the same normal-equation solution one
measurement at a time, using the prior (x̂₀, P₀) as an initial condition.
Differences between the RLS and batch plots at small k come from the way
the prior is incorporated and from finite-precision numerical effects,
but they converge as more data are assimilated.
```

Part C



```
The batch least squares implementation recomputes the normal equations
using all k measurements at each step. As a result, its computational
cost per update grows roughly linearly with k, and the measured average
execution time increases as more measurements are included.

In contrast, the recursive least squares implementation updates the
estimate and covariance using only the new measurement and the previous
state ($\hat{x}_{k-1}$, $P_{k-1}$). Each RLS update has essentially constant cost
O($n^2$) in the state dimension n and does not depend on k, so the
measured execution time per update remains nearly flat as k increases.

For this problem the absolute differences in timing are small because
both the measurement dimension and state dimension are modest. However,
the scaling behavior is fundamentally different: for long data records
or higher-dimensional states, RLS is significantly more efficient than
repeatedly solving the batch least squares problem from scratch.
```

Code

See the Python code for this problem.

Problem 3: Kalman Filtering *(40 pts.)*

Imagine you are floating in interplanetary space and observing a spacecraft traveling at some undetermined position awway from you. The spacecraft has the state $\boldsymbol{x} = \left[x, y, z, \dot{x}, \dot{y}, \dot{z}\right]^T$. The spacecraft is not under the influence of gravity, rather its motion is only driven by its current velocity vector and some gaussian white noise acting on the acceleration terms — i.e.

$$\ddot{x}\left(t\right) = w_1\left(t\right) \tag{1}$$

$$\ddot{y}\left(t\right) = w_2\left(t\right) \tag{2}$$

$$\ddot{z}\left(t\right) = w_3\left(t\right) \tag{3}$$

where

$$\boldsymbol{w}\left(t\right) \sim \mathcal{N}\left(\boldsymbol{0}, W\right), \quad W = \begin{bmatrix} 10^{-5} & 0 & 0 \\ 0 & 10^{-5} & 0 \\ 0 & 0 & 10^{-5} \end{bmatrix} \frac{\text{km}^2}{\text{s}^4}$$

You have a magical sensor that can produce noisy measurements of the spacecraft's current position, but it is unable to resolve any information about its velocity — i.e.

$$\boldsymbol{Y}\left(t\right) = \left[x, y, z\right]^T + \boldsymbol{v}\left(t\right), \quad \boldsymbol{v}\left(t\right) \sim \mathcal{N}\left(0, V\right)$$

where the noise associated with the sensor is characterized by the continuous time process noise matrix

$$V = \begin{bmatrix} 10^3 & 0 & 0 \\ 0 & 10^3 & 0 \\ 0 & 0 & 10^3 \end{bmatrix} \text{km}^2$$

Using your expert insight, you have a rough order of magnitude estimate for the spacecraft's initial state as

$$\hat{\mathbf{x}}_0 = \left[0\,\text{km}, 0\,\text{km}, 500\,\text{km}, 0.01\,\tfrac{\text{km}}{\text{s}}, 0\,\tfrac{\text{km}}{\text{s}}, 0.01\,\tfrac{\text{km}}{\text{s}}\right]$$

and a covariance of

$$P_0 = \begin{bmatrix} 50 \cdot \mathbb{I}_{3\times3}\text{km}^2 & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & 0.1 \cdot \mathbb{I}_{3\times3}\frac{\text{km}^2}{\text{s}^2} \end{bmatrix}$$

Using this information, answer the following questions:

a. Express this system in a discrete time state-space model. Specify $A, C, F, H, Q,$ and R.

b. Implement a Kalman filter to estimate the position and velocity of the spacecraft using measurements provided in `HW5Measurements-P3.npy`. Report your best estimate of the state at the final time-step and its associated error covariance matrix.

c. Plot the pure prediction estimates of the state and ±3σ error bounds over time — i.e. $\hat{\mu}_k^-$ and P_k^-.

d. Plot the measurement corrected estimates of the state and ±3σ error bounds as a function of time — i.e. $\hat{\mu}_k^+$ and P_k^+.

e. Explain the differences between the plots of $\left(\hat{\mu}_k^-, P_k^-\right)$ and $\left(\hat{\mu}_k^+, P_k^+\right)$.

f. Using the history of $\hat{\mu}_k^+$, plot the measurement residuals over time

$$\delta\boldsymbol{y} = \boldsymbol{y}_k - H_k\hat{\mu}_k^+$$

Do these match expectation?

Solution

Part A

Answer

Part B

```
x_final = 0
P_final = 2
```

Part C

Part D

Problem 3 continued on next page...

Part E

```
Write your answer here.
Write your answer here.
```

Part F

```
Write your answer here.
Write your answer here.
```

Problem 3 continued on next page...

Code

See the Python code for this problem.