ASEN 5044 Statistical Estimation for Dynamical Systems
Fall 2017
Final Project Assignment - Part 2

Out: Thursday 12/07/2017 (posted on D2L)

Due: Thursday 12/21/2017, 4:59 pm

(***GROUP SUBMISSION* TO D2L DROPBOX!!!**)

You are to submit solutions as a project group and will receive a group grade (out of 130 total points, not counting the bonus challenge problems). You need not prepare a formal report for this assignment – you may treat this as if this were just another (long) homework assignment, ***but as always show all your work and explain your reasoning****. Remember that good basic programming practices will save you a lot of time/effort, and neatness, completeness and clarity in your submission will go a long way to helping your grade.* ***Good luck!!***

Part I. ***BASIC SYSTEM ANALYSIS***: [30 pts] Report your results for HW 8 Problem 2 parts (a)-(c) here. Note that this is your chance to rectify any mistakes/oversights or misunderstandings from HW 8. Unlike on HW 8, your answers for all parts will be graded this time, so be sure to provide clear, concise, and comprehensive answers to all parts.

Part II. ***NONLINEAR FILTERING***:

1. [40 pts] Implement and tune a linearized KF using the specified nominal state trajectory along with the control input values and covariance matrix values posted on D2L for the DT nonlinear AWGN process noise and measurement noise for your selected system. Note: if these are not posted yet by the time you read this, they will be posted very soon – in the meantime, assume physically sensible 'placeholder' values for the control inputs and covariance matrices to proceed with filter code development until these are posted. Use NEES and NIS chi-square tests based on Monte Carlo truth model test (TMT) simulations to tune and validate your linearized KF's performance (note: be sure to explain how the relevant variables in each test can be adapted to the linearized KF). Choose a sufficiently large number of Monte Carlo runs and sample trajectory simulation length to perform the tests, and choose the α value for running each test (provide some justification for your choices).

Explain how you tuned your linearized KF's guess of the process noise, and provide appropriate plots/illustrations to show that your filter is working properly, namely:

i. Plots for a single 'typical' simulation instance, showing the noisy simulated ground truth states, noisy simulated data, and resulting linearized KF state estimation errors.

ii. Plots of the NEES test statistic points from all Monte Carlo simulations vs. time, comparing the resulting averages to computed upper/lower bounds for the NEES chi-square test.

iii. Plots of the NIS test statistic points from all Monte Carlo simulations vs. time, comparing the resulting averages to computed upper/lower bounds for the NIS chi-square test.

Note that:

- to do TMT, you will have to generate multiple sets of simulated ground truth trajectories using the nonlinear dynamics models (with process noise included along the same nominal trajectory) and corresponding measurements (with measurement noise included) using the nonlinear measurement models. The simulated measurements will be used as input to your linearized KF to produce state estimates and predicted measurements for the NEES and NIS tests, respectively.

- for each test trajectory sample in the NEES and NIS tests, you should initialize the filter with exactly the same initial perturbation state estimate $\hat{\delta x}^+(0)$ and covariance $P^+(0)$, and use these to randomly instantiate the ground truth state $x(0)$ at the start of each full nonlinear trajectory simulation for truth model testing. So, this means will need to come up with reasonable values for $\hat{\delta x}^+(0)$ and $P^+(0)$ to tune and test against (you actually may want to check that your filter works consistently well for multiple values of $\hat{\delta x}^+(0)$ and $P^+(0)$, but you only need to report one set of values that you think are representative for your application).

- your linearized KF will need to be tuned with a certain 'guessed' process noise covariance Q_{KF}, which in general will not be the same as the actual process noise in the full nonlinear TMT simulation (in reality, the latter would be unknown and thus 'hidden' from your filter).

Hint: The example code posted for NEES and NIS tests for linear KFs in the lecture notes might be useful to look at. Keep in mind that since you are dealing with linearized models to examine the δx states, your full state estimates and recorded measurements (and hence state estimation errors and full measurement innovations) need to account for the contribution of the nominal state trajectory that you are linearizing about at each time step.

2. [40 pts] Implement and tune an extended KF (EKF) using the specified nominal state trajectory along with the control input values and covariance matrix values posted on D2L for the DT nonlinear process noise and measurement noise for your selected system. Note: as with #1: if these are not posted yet by the time you read this, they will be posted very soon – in the meantime, assume physically sensible 'placeholder' values for the control inputs and covariance matrices to proceed with filter code development until these are posted. Use NEES and NIS chi-square tests based on Monte Carlo truth model test (TMT) simulations to tune and validate your EKF's performance (be sure to explain how the relevant variables in each test can be adapted to the EKF). Choose a sufficiently large number of Monte Carlo runs and sample trajectory simulation length to perform the tests, and choose the α value for running each test (provide some justification for your choices).

Explain how you tuned your EKF's guess of the process noise, and provide appropriate plots/illustrations to show that your filter is working properly, namely:

i. Plots for a single 'typical' simulation instance, showing the noisy simulated ground truth states, noisy simulated data, and resulting EKF state estimation errors for each state (with 2σ bounds).

ii. Plots of the NEES test statistic points from all Monte Carlo simulations vs. time, comparing the resulting averages to computed upper/lower bounds for the NEES chi-square test.

iii. Plots of the NIS test statistic points from all Monte Carlo simulations vs. time, comparing the resulting averages to computed upper/lower bounds for the NIS chi-square test.

Hint: Many of the same notes/hints given above for the linearized KF apply here for the EKF, with some obvious changes, e.g. you must pick and consistently initialize with $\hat{x}^+(0)$ for the total state instead of just the perturbation state, and account for the fact that you are estimating the total state and looking at the total measurement innovation for NEES and NIS assessments in the EKF (i.e. not just state/measurement perturbations).

3. [20 pts] Implement your linearized KF and EKF to estimate the state trajectory for the observation data log posted for your system on D2L. Turn in plots of the estimated states and 2σ (make sure these are legible/readable). Compare your results – do you think the linearized KF or EKF does a better job (justify)? (Data for this part will be posted soon for each system – in the meantime, your group and other groups in the class working on the same system can consider swapping simulated sensor data logs that you each generate, in order to cross-test your filters and assess how well your linearized KFs and EKFs get to the actual ground truth states.)

Challenge Question *Extra credit - partial credit only applies if you submit good work.* ***You can submit this as part your project assignment writeup.***

C16. (10 pts) Write a haiku about estimation. Alternative forms of short poetry/verse are also acceptable. [1]

C15. (25 pts) Select and implement **one** of the following alternative KF techniques in place of the 'vanilla' linearized KF and/or EKF for your system (locate and read the corresponding portions of Chapters 6, 7, 10, 13, 14 or 15 in the Simon textbook for the theory and details):

- the square root KF (adapt to either the linearized KF or EKF);
- the information filter (adapt to either the linearized KF or EKF);
- the reduced order KF (adapt to either the linearized KF or EKF for some desired subset of states that you come up with);
- the colored/correlated noise KF (adapt to either the linearized KF or EKF for truly correlated/colored noise processes);
- the constrained KF (adapt to either the linearized KF or EKF for some set of constraints that you come up with);
- the iterated EKF (direct alternative to either the linearized KF or the EKF);

[1] must be G-rated/family-friendly, so no limericks about a man from Nantucket...

- the unscented KF (UKF, aka the sigma point filter or SPF – another direct alternative to both the linearized KF and the EKF);
- the particle filter (PF – yet another direct alternative to both the linearized KF and the EKF).

You need not perform NEES/NIS tests for whatever approach you decide to implement, but do provide estimation error plots (with 2σ bounds) for a 'typical' problem to show your implementation works. Compare the results to those obtained using the standard linearized KF and EKF and comment on the strengths/weaknesses of the alternative approach you implemented (i.e. where/why might it actually be useful?).