AE4320 Assignment: Multivariate Simplex Splines

This take-home assignment consists of 4 parts. The deliverable of the assignments are a report of **maximum 20 pages** containing the theory and results of the 4 parts, and a set of Matlab or Python code files which were used to generate the results. A total of 100 points can be obtained for the assignment. In Appendix A, a brief introduction to the supplied dataset is given. In Appendix B details for the construction of the Kalman Filter are given. In Appendix C, a brief description of the supplied Matlab files is given.

Part 1: Report introduction (13 points)

You are asked to write a brief introduction (max 2 pages) for your report which should contain the following items:

1. A brief discussion on the relevance of System Identification to aircraft control & simulation. (**3 points**)
2. A brief explanation of the working principles of state estimation techniques and parameter estimation methods. Discuss these methods from the perspective of linearity of the system, knowledge of noise statistics, and model structure. (**5 points**)
3. A brief explanation of the working principles of advanced system identification methods such as neural networks and multivariate splines. Mention the role of basis functions, cost functions, and optimization methods. Name an advantage and disadvantage of both methods. (**5 points**)

Part 2: State & Parameter estimation with F-16 flight data (32 points)

You are asked first to implement a state estimation routine (Kalman Filter) to estimate a bias in the angle of attack measurements in the given dataset.

It is given that there is an unknown bias in the angle of attack (α_m) measurements caused by upwash. The angle of sideslip (β_m) measurements and velocity (V_m) are not biased. Please see **Appendix B** for details on the construction of the Kalman Filter. **The report should contain the following items:**

1. Formulate the (trivial) system equations $\dot{x}_k = f(x_k, u_k, t)$ and the (non-trivial) output equations $z = h(x_k, u_k, t)$ given the information in **Appendix B**. (**5 points**)
2. Briefly motivate which type of Kalman filter should be used in this case; the linear Kalman Filter, the Extended Kalman Filter, or the Iterated Extended Kalman Filter. (**2 points**)
3. Construct your Kalman filter, and use it to estimate $C_{\alpha_{up}}$. **Important:** Use a fixed time step of $\Delta t = 0.01$ in your integrator/continuous-to-discrete time conversion operation, otherwise the input vector cannot be used correctly. Prove that your filter has converged. (**7 points**).
4. Reconstruct α_{true} using the estimated $C_{\alpha_{up}}$, and clearly show the difference with the measured α_m. **(3 points)**

Implement an ordinary least squares estimator for a simple polynomial model structure for the reconstructed F-16 flight data and apply it on the given dataset. The report/code should contain the following items:

5. Formulate a least squares estimator, and estimate the coefficients of your polynomial model. (**5 points**)
6. Show the influence of polynomial model order on the accuracy of fit. (**5 points**)
7. Show the results from the model validation; perform both a statistical and a model-error based validation. (**5 points**)
8. Hand in the code used in this part in a separate, well documented file.

Part 3: Deriving a simplex polynomial (20 points)

You are asked to formulate and implement an estimator for the B-coefficients of a single simplex polynomial and use it to identify a model of the F-16 data using the reconstructed states:

1. Show how the basis functions of the simplex splines for general polynomial degrees are defined. Show how the barycentric coordinates are used in the basis functions. Show also how increasing the degree affects the structure of the basis polynomials. (**5 points**)
   <u>**Hint:** Don't forget the multinomial coefficient!</u>
2. Clearly show how you defined the vertex locations, and how you treated the data (e.g. pre-filtered, subdivided, etc) (**5 points**)
3. Formulate a least squares estimator for the B-coefficients of a B-form polynomial, and use it to identify models for at least two different polynomial degrees based on the F-16 data. (**5 points**)
4. Evaluate the performance of your estimator using **both** a model residual and statistical model quality analysis. Clearly show how the polynomial degree affects model quality. Compare your results with those obtained in Part 2 of this assignment. (**5 points**)
5. Hand in the code used in this part in a separate, well documented file.

Part 4: System Identification with Simplex Splines (35 points)

You are asked to formulate and implement an estimator for the B-coefficients of a complete simplex spline and use it to identify a high fidelity model of the reconstructed F-16 data:

1. Design (graphically) a complete system identification algorithm that combines state estimation techniques with an estimator for the B-coefficients of the multivariate simplex splines. (**3 points**)
2. Clearly show how you treated the data (e.g. pre-filtered, subdivided, etc) for this case. Clearly report your findings in the report. (**3 points**)
3. Formulate the Smoothness Matrix for 0^{th} order continuity for your spline model. Prove that your model really has 0^{th} order continuity. (**7 points**)
   Bonus: formulate the 1^{st} and higher continuity order smoothness matrix for your spline model, and again prove that higher order continuity is present. (**5 bonus points**)
4. Formulate a least squares (**2 points**), and a more advanced (**2 points**), estimator for the multivariate simplex splines using a triangulation consisting of at least 2 simplices. (**max 4 points**)
5. Implement the system identification framework for multivariate splines and use it to identify a multivariate spline based aerodynamic model for the F-16 given the flight data. Demonstrate the effect on the model quality of changing the polynomial degree, continuity order, and number of simplices. (**8 points**)
6. Validate your aerodynamic model using model residual and statistical model quality analysis. Clearly report your findings in the report (**10 points**)
7. Hand in the code used in this part in a separate, well documented file.

Appendix A: Introduction to the F16 dataset.

The file 'F16traindata_CMabV_2019.mat' contains the data in the Matlab MAT format , 'F16traindata_CMabV_2019.csv' contains the data in CSV format, which is more useful when using e.g. Python. The data is loaded from disk with Matlab using the command:

```
load('F16traindata_CMabV_2019.mat', 'Cm', 'Z_k', 'U_k');
```

The vector 'Cm' contains the measurement data of the pitching moment coefficient: $C_m \in \mathbb{R}^{N\times 1}$ with 'N' the total number of measurements. The matrix 'Z_k' contains the measurement locations (states), where Z_k contains the following data: $Z_k = \begin{bmatrix} \alpha_m & \beta_m & V_m \end{bmatrix} \in \mathbb{R}^{N\times 3}$. The matrix 'U_k' contains perfect linear accelerometer measurements that can be used in your Kalman filter. It contains the following data: $U_k = \begin{bmatrix} \dot{u} & \dot{v} & \dot{w} \end{bmatrix} \in \mathbb{R}^{N\times 3}$, with $\dot{u}, \dot{v}, \dot{w}$ the perfect (no biases, noise-free) time derivatives of the body velocities u, v, w. **Please do not confuse the longitudinal velocity 'u' with the system input U_k !**

When plotting the data, which can be done with the supplied file 'F16_PlotData.m' you will find that α_m, β_m, V_m are contaminated with zero mean white noise. In addition α_m is biased due to upwash at the location of the angle of attack vane, see **Appendix B**. The measurements made on Cm are also contaminated with noise.

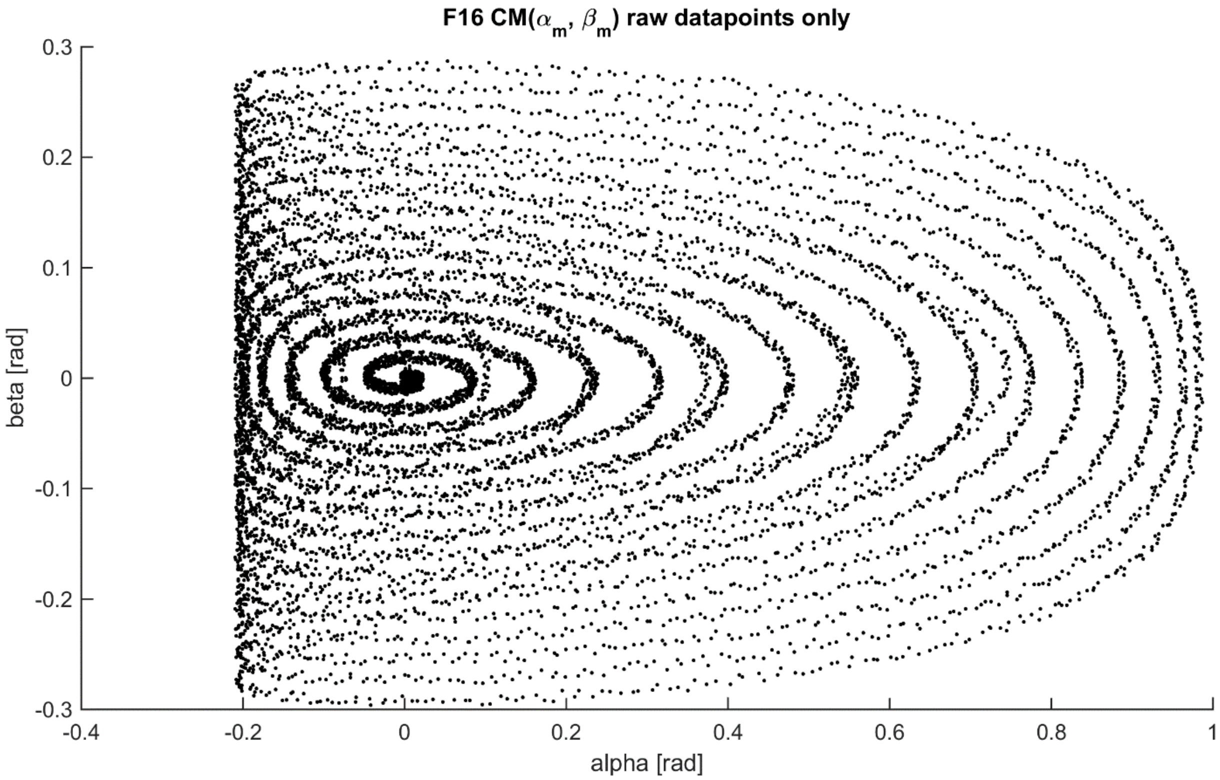

Figure 1: Locations in the angle of attack and angle of sideslip domain at which measurements of Cm are available.

Appendix B: Constructing the Kalman filter

In the construction of your Kalman filter, use the following state vector:

$$x_k = \begin{bmatrix} u & v & w & C_{\alpha_{up}} \end{bmatrix}^T$$

As measurement vector use:

$$z_k = \begin{bmatrix} \alpha_m & \beta_m & V_m \end{bmatrix}^T$$

As input vector use the perfect (**unbiased, noise-free**) linear accelerometer values, which are sampled with $\Delta t = 0.01$:

$$u_k = \begin{bmatrix} \dot{u} & \dot{v} & \dot{w} \end{bmatrix}$$

It is given that there is an unknown bias in the angle of attack (α_m) measurements caused by upwash at the location of the angle of attack vane. The angle of sideslip (β_m) measurements and velocity (V_m) are not biased. It is given that the **measured** angle of attack (α_m,), angle of sideslip (β_m), and velocity (V_m) are related to their **true** values (α_{true}, β_{true}, V_{true}) by the following formulas:

$$\alpha_m = \alpha_{true}(1 + C_{\alpha_{up}}) + v_\alpha$$

$$\beta_m = \beta_{true} + v_\beta$$

$$V_m = V_{true} + v_V$$

With $C_{\alpha_{up}}$ an unknown upwash coefficient that must be estimated, and with v_α, v_β, and v_V white noise sequences. The true angle of attack, angle of sideslip, and velocity are related to the system states as follows:

$$\alpha_{true} = \tan^{-1}\left(\frac{w}{u}\right)$$

$$\beta_{true} = \tan^{-1}\left(\frac{v}{\sqrt{u^2 + w^2}}\right)$$

$$V_{true} = \sqrt{u^2 + v^2 + w^2}$$

Substituting these expressions in the equations for α_m, β_m, and V_m will give you the observation equations $h(x,u,t) + \begin{bmatrix} v_\alpha & v_\beta & v_V \end{bmatrix}^T$.

You can use the following statistics for the process noise:

$$E\{\underline{w}(t_i)\} = 0; \quad E\{\underline{w}(t_i)\underline{w}^T(t_i)\} = Q\delta_{ij}; \qquad Q = diag(\sigma_{w_u}^2, \sigma_{w_v}^2, \sigma_{w_w}^2, \sigma_{w_C}^2)$$

$$\sigma_{w_u} = 1e-3, \sigma_{w_v} = 1\text{e}-3, \sigma_{w_w} = 1\text{e}-3, \sigma_{w_C} = 0$$

And the following sensor noise statistics:

$$E\{\underline{v}(t_i)\} = 0; \quad E\{\underline{v}(t_i)\underline{v}^T(t_j)\} = R\delta_{ij}; \quad R = diag\left(\sigma_{v_\alpha}^2, \sigma_{v_\beta}^2, \sigma_{v_V}^2\right)$$

$$\sigma_{v_\alpha} = 0.035, \sigma_{v_\beta} = 0.013, \sigma_{v_V} = 0.110$$

Make a smart choice for initializing your diagonal state prediction covariance matrix $P(0|0)$, as it will determine how fast your filter will converge depending on how close your initial guess for the state vector is to the real state vector.

Hint: Use a fixed time step of $\Delta t = 0.01$ in your (Runge-Kutta) integrator and 'c2d' operations! **Hint2:** Note that the system dynamics equations $f(x,u,t)$ are trivial in the sense that they are fully described by the system input.

Appendix C: Description of Matlab Files

<F16traindata_CMabV_2019.mat>
The F-16 aerodynamic dataset used in this year's multivariate spline assignment.

<load_f16data2019.m>
Use this script in Matlab to load the F-16 data from disk.

<F16_PlotData.m>
This script shows how to interpret and plot the F-16 data in the data file.

<bsplinen_cart2bary.m>
This script transforms a global Cartesian coordinate to a local barycentric coordinate relative to a given simplex. Note that you don't need to use this function if you use the Matlab built-in TSEARCHN function.

<bsplinen_bary2cart.m>
This script transforms a local barycentric coordinate relative to a given simplex back to global Cartesian coordinates.