Online Dynamics Estimator for Adaptive UAV Control with Uncertain Model

William Grossman and Marcus Hammond
December 10, 2010

Abstract – **Small, low-cost unmanned aerial vehicles (UAVs) are more prone to actuator failure and manufacturing variability than their larger, more expensive counterparts. To build a controller that was robust to this variability and these failure modes a linear quadratic regulator (LQR) was implemented to make the aircraft follow a desired trajectory, and then at regular intervals, a discrete time continuous state and action model was fit to the state transition and input histories using least squares methods. The method was successfully implemented on several simple systems as a proof of concept, but initial attempts to apply the algorithm to a more complex aircraft model failed to explore the space thoroughly enough to generate a faithful approximation of the system dynamics.**

I. Introduction

Small, cheap UAVs have widespread potential applications in a number of fields. However, unlike their larger, more expensive cousins, they suffer from manufacturing variability and low-cost components prone to failure. This poses an interesting control problem. How does one design a controller that can account for, and even learn changes in a dynamic model of a UAV in order to tailor its control strategy to the particular platform?

This problem can be thought of as having two distinct components: calculation and implementation of an optimal control policy given the current estimate of system dynamics, and generation of a new estimate based on the input-output state transition histories recorded over some finite time.

To approximate the conditions that are encountered during actual flights using digital control, the input was updated at a frequency of 20 Hz. However, a number of simplifications from real-world applications were made, for instance the assumption that measurements were available for all state variables. In practice, this is not true, and Kalman filters or similar estimators are used to generate estimates of the state based on available measurements.

Due to time constraints and complications arising during implementation on the airplane simulator, the scope of the estimation was scaled back to trying to learn a model for only the longitudinal dynamics of the aircraft. In essence, this assumes that the estimate of the parameters influencing lateral dynamics is "good enough" and does not affect the longitudinal dynamics. For conventionally configured aircraft flying near a trim solution, this decoupling is a valid assumption [2].

II. Control Strategy

Aircraft dynamics are governed by nonlinear equations of motion. However, in this paper we are concerned with flight near cruise condition (no extreme maneuvering and far from aerodynamic stall), so we can safely use a linear model to characterize the behavior.

The continuous time linearized equations of motion are given as follows:

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (1)$$

where x(t) is the state vector at time t, and u(t) is the input at time t.

Using forward Euler integral approximation, a discrete time model can be derived:

$$\dot{x}(t) \approx \frac{x(t+1) - x(t)}{\Delta t}$$
$$x(t+1) = (I + \Delta tA)x(t) + \Delta tBu(t) \qquad (2)$$
$$x(t+1) = A_D x(t) + B_D u(t)$$

where $A_D = (I+\Delta tA)$, $B_D = \Delta tB$. These are the matrices we try to approximate, as described in the next section.

LQR Controller

General setup:

For a discrete-time linear system the linear quadratic regulator gives the control strategy that minimizes the cost function:

$$J_{LQR} = \sum_{t=0}^{H} x(t)^T Q x(t) + u(t)^T R u(t) \qquad (3)$$

where x(t) is again the state, and u(t) the input at time t. Q and R are positive semidefinite weighting matrices allowing the engineer to choose the relative penalties for error and the expense of control effort. H is a time horizon.

The case where H approaches infinity is called the infinite horizon, or steady state LQR. The u(t) that optimizes this equation is given by:

$$u(t) = -K(x(t) - x(t)_{desired}) \qquad (4)$$

where

$$K = (R + B^T P B)^{-1} B^T P A \qquad (5)$$

and P is the solution to the algebraic Ricatti equation (ARE):

$$P = Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A \qquad (6)$$

Note that normally, P is a function of time and is calculated by setting P(H) = Q and integrating the cost function backwards in time. When t is sufficiently far from H (as is always the case when H=infinity) P converges so that P(t-1) ~= P(t). This turns the difference equation into the algebraic equation (6).

The Ricatti equation is solved iteratively, and then the calculated value of P is used to produce the control gain matrix K. Again, we have assumed access to the full state of the system, which is not, in general, the case. However, the purpose of this project was to implement and test a machine learning algorithm, as opposed to designing a state estimator. If the algorithm proved successful on the idealized system, we could then introduce these other complications and test it further. If it did not work for the ideal case, it certainly would not work with measurement and process errors.

Choosing Weighting Matrices

The weighting matrices were chosen based on Bryson's rule [3]:

$$Q = \begin{bmatrix} \frac{1}{(s_{1\max} - s_{1des})^2} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{(s_{n\max} - s_{ndes})^2} \end{bmatrix} \qquad (6)$$

$$R = \begin{bmatrix} 1/u_{1\max}^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1/u_{n\max}^2 \end{bmatrix} \qquad (7)$$

Where $s_{i\ max}$-$s_{i\ des}$ and $u_{i\ max}$ are the maximum allowable errors off of nominal and input magnitude, respectively. It is important to choose these errors carefully, as the parameters in the dynamics matrix operate on very different scales. A 1 m/s error in velocity is tolerable, whereas a 1 radian error in pitch angle is not. Similarly, the aircraft actuators are limited in their available travel and even further limited by the tolerances of the linearized model.

III. Parameter Estimation

After letting the initial control policy fly the airplane for a set time, the control inputs and state history can be used to compute a least squares linear model of the aircraft dynamics. This new linear model will account for any changes in the dynamics model due to actuator

$$\begin{bmatrix} x_1(2) \\ x_2(2) \\ \vdots \\ x_n(2) \\ \vdots \\ x_1(\tau+1) \\ x_2(\tau+1) \\ \vdots \\ x_n(\tau+1) \end{bmatrix} = \begin{bmatrix} diag(x_1(1)) & \cdots & 0 & diag(u_1(1)) & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & diag(x_n(1)) & 0 & \cdots & diag(u_m(1)) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ diag(x_1(\tau)) & \cdots & 0 & diag(u_1(\tau)) & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & diag(x_n(\tau)) & 0 & \cdots & diag(u_m(\tau)) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

a_1 through a_n and b_1 through b_m are the column vectors of the A_d and B_d matrix (respectively) of the linear aircraft model. $x_n(\tau)$ is the n^{th} state of the aircraft at time τ.

failure, or airframe modification. In practice, we found the condition number of the least squares matrix to be very high, giving low credibility to the estimated linear model of the airplane. We hypothesized that this was due to the relatively stable flight of the aircraft, limiting the state and control input data to a very small range. By adding control inputs that disturbed the planes flight path, we were able to significantly reduce the condition number of the least squares matrix and obtain a higher fidelity model of the aircraft. However, the condition number was still much higher than we would have liked.

To perform the least squares estimate, we rearranged equation 2 in order to solve for A_d and B_d. At the top of this page, we rearranged equation 2 to a form that can be used to solve for A_d and B_d.

IV. Simulation

For our control policy validation and Aircraft dynamics estimation, we used a 6 degree-of-freedom model of a light business jet. This model was initially built by Robert Stengal for use in his textbook on flight dynamics [2]. The model uses wind tunnel test data from various flight conditions and summarizes these values into linear and quadratic coefficients used in the dynamics equations. We used this model because we wanted a realistic simulator to test our control system. Although the system modeled all six degrees of freedom of an airplane, out controller only used 2 of them.

V. Results

a) Proof of concept on simple systems

Airplanes are governed by rather complex nonlinear equations. Even the linearized equations result in an 8th order system with three oscillatory modes and two real modes (one of which is typically unstable) [2]. Rather than immediately try to test the viability of the learning algorithm on this system, an incremental approach was taken. The algorithm was applied to a second order system, and then a 4th order system before it was implemented on the airplane.

2nd order system: Pendulum with one input

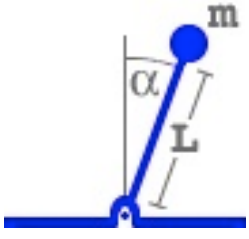

Figure 1: Inverted pendulum on a fixed base

To start, the algorithm was told that it had two states and one input (torque applied at the pivot), then seeded with a random estimate of the A and B matrices in the linearized model. Based on this estimate it formed a gain matrix K and began to run the nonlinear simulator. Generally it would perform poorly on the first trial, as its estimate of the system dynamics was random. This poor performance gave it insight into the actual dynamics, however, and the second attempt would successfully drive the system to zero. Initially, the system would be driven to zero so quickly that the third estimate would have very little information and could not generate a good system approximation. Thus emerged a pattern in which odd trials were unstable and even trials were stable. A heuristic solution was implemented in which the system estimate would only be updated if the condition number of the matrix to be inverted in the least squares procedure was below a certain (albeit quite high) threshold. Loosely, this corresponds to some information threshold in the input-output pairs of the system.

To further test the robustness of the algorithm, the magnitude and direction of gravity was changed at random intervals, forcing the algorithm to re-estimate its parameters and update its control strategy.

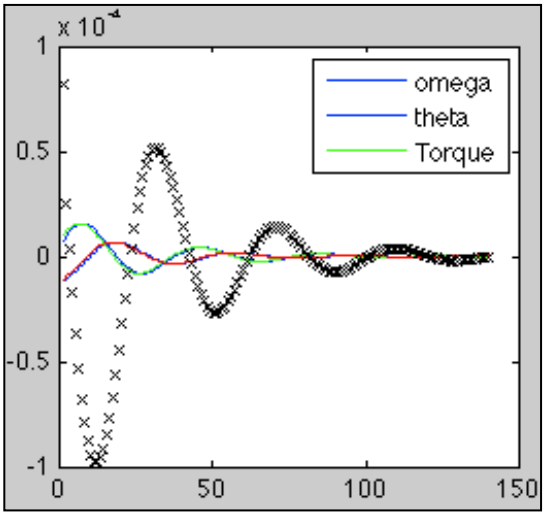

Figure 2: Pendulum stabilizes

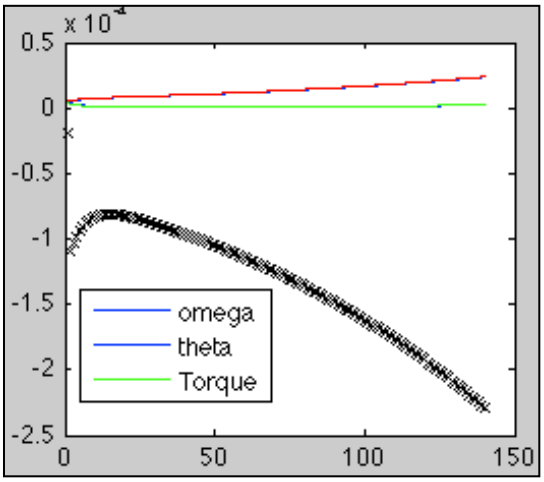

Figure 3: Gravity flips sign. System destabilizes

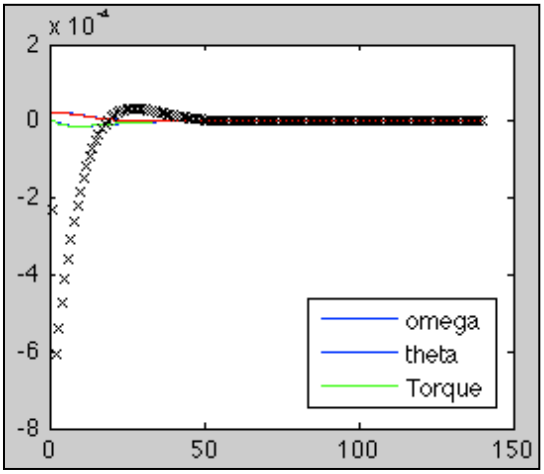

Figure 4: Algorithm learns new parameters, re-stabilizes

4th order system: Pendulum on cart with one input

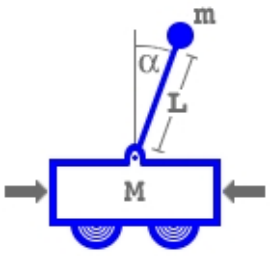

Figure 5: Inverted pendulum on moving cart

In the second setup (figure 5), the pendulum is mounted on a moving base and the single input is a force that can be applied to the left or right. This adds two more states to the state vector, making it a 4th order system with a single input. Again, the system was robust to variations in magnitude and direction of gravity, although without the addition of integral control, the system could find

equilibrium at nonzero position error. This was observed to happen when gravity was inverted, effectively turning the system from a normal pendulum on a cart into an inverted pendulum on a cart. The controller would "chase" after the pendulum, trying to position the cart under the mass to keep it from tipping over, and eventually settle on a position far from the origin.

(Plots for this system resemble the second order one and are omitted for brevity)

B) Implementation on full nonlinear Aircraft simulator

The positive results obtained on the simpler test cases did not extend to the aircraft model. The condition number of the matrix to be inverted was on the order of 10^6 in the best of cases, and no good estimate of system dynamics could be extracted from the data. Adding noise to the inputs after calculating the LQR controller resulted in an improvement in this condition number of several orders of magnitude, but still did not yield a good estimate of the dynamics. The LQR controller did not explore the state space extensively enough to generate a good picture of the aircraft dynamic parameters.

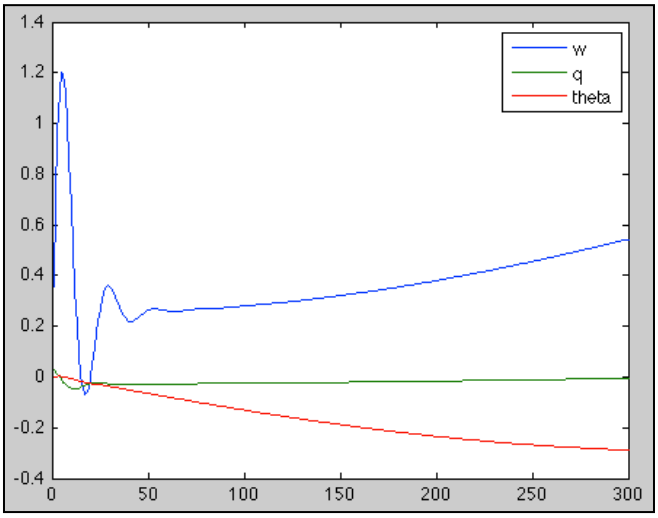

Figure 6: Aircraft response to LQR control based on initial estimate of system dynamics

VI. Conclusions

Although the initial implementation of the algorithm worked well for the inverted pendulum, it performed poorly on the aircraft simulator due to the high condition number of the matrix used to calculate the dynamics equations. Even with pseudo-random additions to the control inputs to promote state space exploration, the condition number was still too high to give a good dynamics model. A potential solution to this problem might be to assume a prior on the dynamics model and incorporate this information into the regression calculation. This has been done successfully in similar contexts [1].

VII. References

[1] Abbeel, P., Quigley, M., & Ng, A. *Using Inaccurate Models in Reinforcement Learning*

[2] Bryson, Arthur. *Control of Spacecraft and Aircraft*, Princeton University Press,1993

[3] Bryson, A.E. and Ho, Y.C. *Applied Optimal Control*. Wiley New York, 1975.

[4] Ng, A. *CS 229 Lecture Notes: Reinforcement Learning* 2010

[5] Stengal, Robert. *Flight Dynamics*. Princeton University Press, 2006.