Christopher Lum
lum@uw.edu

Lecture 07d
Trimming for Straight and Level Flight

The YouTube video entitled 'Trimming a Model of a Dynamic System Using Numerical Optimization' that covers this lecture is located at https://youtu.be/YzZI1V2mJw8.

Outline

-Trim and Equilibrium Points
-Steady State Straight and Level Trimmed Flight
-Trimming Using Matlab's Linear Analysis Tool
-Other Trim Points
-Next Steps

Trim and Equilibrium Points

In this analysis, we seek the state and control vector that will place the system in a desired configuration. In many cases, the desired configuration is an equilibrium position where the system will remain if it is not perturbed. This is also known as an equilibrium point and this has the characteristics that the state will not change if the system is at this configuration. Recall that the system dynamics are given by

$$\dot{\bar{x}} = \bar{f}(\bar{x}, \bar{u})$$

So the equilibrium points of the system can be defined as

$$\text{Equilibrium points} = \left\{ \bar{x}_o \in \Re^n, \bar{u}_o \in \Re^m \;\middle|\; \bar{f}(\bar{x}_o, \bar{u}_o) = \bar{0} \right\}$$

Note that we need both an equilibrium state, $\bar{x}_o$, and an equilibrium control vector, $\bar{u}_o$.

The next logical question would be, "How does one find an equilibrium point?" We could try solving directly

$$\bar{f}(\bar{x}, \bar{u}) = \bar{0}$$

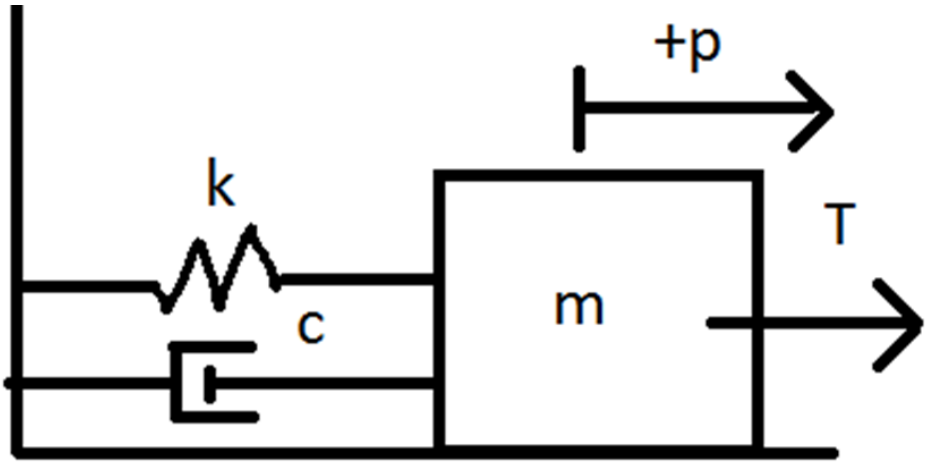
Note that this is n equations and $n + m$ variables, so in many cases, there are infinite choices.

In other situations, solving this analytical can be exceedingly difficult (see PowerPoint slides for example of EoMs for RCAM).

Many times, trim and equilibrium points are used interchangeably but we should be clear that these are not necessarily the same. Equilibrium points are ones that mathematically satisfy $\overline{f}(\overline{x}_o, \overline{u}_o) = \dot{\overline{x}} = \overline{0}$. A trim point may be a point of interest but does not necessarily yield $\dot{\overline{x}} = 0$ (for example an aircraft trimmed to a circling orbit is a trim point but the state is changing constantly)

Example: Equilibrium Point of a Spring/Mass/Damper System

Consider the standard mass/spring/damper system shown below with a constant force, T exerted upon it.



The state and control vector of the system is

$$\overline{x} = \begin{pmatrix} p \\ \dot{p} \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad \overline{u} = (T) = (u_1)$$

From Newton's 2nd law, we can write the equations of motion as

$$\dot{x}_2 = \frac{1}{m} u_1 - \frac{k}{m} x_1 - \frac{c}{m} x_2$$

The state space model of the system is given as

$$\dot{\overline{x}} = \begin{pmatrix} x_2 \\ \frac{1}{m} u_1 - \frac{k}{m} x_1 - \frac{c}{m} x_2 \end{pmatrix} = f(\overline{x}, \overline{u})$$

At this point, if we want to find the equilibrium point of this system, we would like to find the values of $\overline{x}$ and $\overline{u}$ that yield $\dot{\overline{x}} = 0$

$$\dot{\overline{x}} = f(\overline{x}, \overline{u}) = \begin{pmatrix} x_2 \\ \frac{1}{m} u_1 - \frac{k}{m} x_1 - \frac{c}{m} x_2 \end{pmatrix} = 0$$

So we see that this is two equations and two unknowns. The first equation immediately yields

$$x_2 = 0$$

Substituting this into the second equation yields

$$\frac{1}{m} u_1 - \frac{k}{m} x_1 = 0$$

$$\frac{1}{m} u_1 = \frac{k}{m} x_1$$

$$x_1 = \frac{1}{k} u_1$$

So we see that the equilibrium point is simply

$$\overline{x} = \begin{pmatrix} u_1 / k \\ 0 \end{pmatrix}$$

This makes physical sense as this simply states that the system must have a zero velocity and the spring displacement is such that the spring force balances the applied force.

End Example

In the previous example, the system was simple enough that solving for trim points can be done analytically. In many cases this is not possible as the system is too complicated and we must resort to numerical methods. The RCAM model is falls into this category.

We might be interested finding trim points. For example, what are the state and control that the aircraft must have in order to achieve straight and level flight at a certain velocity? This would typically be done by a pilot with some iterative control. As engineers, we would like a mathematical approach which can find this for us. We can do this using the optimization framework we discussed previously.

Let us now investigate the usage of numerical optimization techniques to find an equilibrium or trim point.

Steady State Straight and Level Trimmed Flight

The RCAM model has a state and control vector of

$$\overline{x} = \begin{pmatrix} u \\ v \\ w \\ p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{pmatrix} \qquad \overline{u} = \begin{pmatrix} \delta_A \\ \delta_T \\ \delta_R \\ \delta_{\text{th1}} \\ \delta_{\text{th2}} \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix}$$

The non-linear explicit model of the system is given by

$$\dot{\overline{x}} = \overline{f}(\overline{x}, \overline{u})$$

$\overline{f}(\overline{x}, \overline{u})$ is a very complicated function, but it is an explicit function of $\dot{\overline{x}}$ (in other words, we can explicitly solve the equations of motion for $\dot{\overline{x}}$).

The problem statement is

($\wp$) Find the $\overline{x}_o$ and $\overline{u}_o$ for steady state, straight and level flight ***(Eq.1)***

Note that from an optimization standpoint, there is no real distinction between a state and control because they are both parameters/variables that can be chosen/changed in an attempt to solve ($\wp$). Therefore, it becomes simpler to consider the decision vector to be the combination of state and control variables

$$\overline{z} = \begin{pmatrix} \overline{x} \\ \overline{u} \end{pmatrix} \qquad \text{(decision vector)}$$

So we can write the optimization problem as

($\wp$) Find $\overline{z}_o$ for steady state, straight and level flight

We can translate this mathematically using constraints. Given that we have 9 states and 5 controls, we have 14 total degrees of freedom. Therefore as a first attempt, we can specify 14 constraints. If the algorithm we use to solve the penalty method does is does not converge we can try removing constraints. Alternatively, if the solutions are non-deterministic and/or sensitive to the initial guess locations, we can try adding constraints to reduce the solution space.

The 14 constraints of interest are:

1-9. $\dot{\overline{x}} = 0$ (steady state flight)
10. $V_a = 85\, m/s$ (airspeed)
11. $\gamma = 0$ (straight and level flight path)
12. $x_2 = v = 0$ (no sideslip)

13. $x_7 = \phi = 0$ (no bank angle)
14. $x_9 = \psi = 0$ (heading north)

Note that constraints 1-9 make this point a equilibrium point. However, if we later add navigation equations, flying straight and level will not be an equilibrium point (as position dot is non-zero) but rather a trim point.

Constraints 1-11 are self-explanatory. We need to also add constraint 12 and 13 because otherwise, it would be possible to satisfy constraints 1-11 by side-slipping. Furthermore, for repeatable results, we add constraint 14 of $\psi = 0$ because otherwise, it would be possible to satisfy the previous constraints with an arbitrary ψ.

Note that at this point, this is simply a problem of solving a system of equations. We have 14 equations and 14 unknowns ($x_1, \ldots, x_9, u_1, \ldots, u_5$). The problem is that this system of equations are incredibly complex and solving it analytically may not be possible ***(show slides)***

We therefore employ the optimization penalty method for solving multiple equations simultaneously (see YouTube video entitled ***'TBD)***

The constrained optimization problem is given as (notice that there is no cost function, this is simply a problem of finding a point in the feasible set).

$$(\wp) \quad \overline{z}^* \in \arg\underset{\overline{z} \in Z}{\text{minimize}}\, f_0(\overline{z}) = \text{constant}$$

such that $f_{1-9}(\overline{z}) = \overline{f}(\overline{x}, \overline{u}) = \dot{\overline{x}} = 0$

$f_{10}(\overline{z}) = V_a - 85 = 0$

$f_{11}(\overline{z}) = \gamma = 0$

$f_{12}(\overline{z}) = x_2 = 0$

$f_{13}(\overline{z}) = x_7 = 0$

$f_{14}(\overline{z}) = x_9 = 0$

We can translate this to an unconstrained optimization problem via the penalty method. We can formulate an approximately equivalent cost function as

$$(\wp_{\text{approx}}) \quad \hat{f}_0(\overline{z}) = Q^T H Q \qquad \textbf{(Eq.1.3)}$$

where $Q^T = (\dot{x}_1 \;\; \dot{x}_2 \;\; \dot{x}_3 \;\; \dot{x}_4 \;\; \dot{x}_5 \;\; \dot{x}_6 \;\; \dot{x}_7 \;\; \dot{x}_8 \;\; \dot{x}_9 \;\; (85 - V_a) \;\; \gamma \;\; x_2 \;\; x_7 \;\; x_9)$

H = matrix which helps define cost function (a penalty matrix)

An easy formulation of H would be to use a diagonal matrix where each element on the diagonal would represent a penalty parameter to a specific penalty function.

$$H = \text{diag}([\, \alpha_1 \;\; \alpha_2 \;\; \ldots \;\; \alpha_{14}])$$

where $\alpha_i \geq 0 \; \forall \; i$

We can expand the cost function to obtain a better idea of it's behavior

$$\hat{f}_0(\bar{z}) = \alpha_1 \dot{x}_1^2 + \alpha_2 \dot{x}_2^2 + \ldots + \alpha_9 \dot{x}_9^2 + \alpha_{10}(85 - V_a)^2 + \alpha_{11} \gamma^2 + \alpha_{12} x_2^2 + \alpha_{13} x_7^2 + \alpha_{14} x_9^2$$

It is worth a minor mention that this function is a quadratic in the penalty function but not a quadratic in $\bar{z}$

From its form we can see that

i. $f_0(\bar{z}) \geq 0 \; \forall \; \bar{z}$

ii. $f_0(\bar{z}) = 0 \iff \left\{ \begin{array}{c} \dot{x}_1 = \dot{x}_2 = \ldots = \dot{x}_9 = 0 \\ V_a = 85 \\ \gamma = x_2 = x_7 = x_9 = 0 \end{array} \right\}$ (minimum is achieved when all constraints are met)

This is exactly the property we want.

Note that in Eq.1.3, $\dot{\bar{x}}$, V_a and γ are functions of the states and must be written as such.

$$\dot{\bar{x}} = f(\bar{x}, \bar{u})$$
$$V_a = \left(x_1^2 + x_2^2 + x_3^2\right)^{1/2}$$
$$\gamma = \theta - \alpha = x_8 - \text{atan}(x_3 / x_1)$$

Note that this definition of flight path angle, γ, is only valid if there is no bank angle. Later, when we discuss the navigation equations we will derive a better definition for γ.

It may be tempting to penalize u_1 and u_3 (aileron and rudder) directly, but this is somewhat dangerous. For example, if the plane was not symmetric, then we would need a certain control surface deflection to achieve straight, level, northerly flight.

An example of code which evaluates the cost of a certain $\bar{x}$ and $\bar{u}$

```
function [F0] = cost_straight_level(Z)

%Extract states
X = Z(1:9);
U = Z(10:14);

%Derivatives of the states.
xdot = RCAM_model(X,U);

%Calculate Va and gamma (need to shorten to gam since gamma is a Matlab
%function).
theta = X(8);
Va = sqrt(X(1)^2 + X(2)^2 + X(3)^2);
alpha = atan2(X(3),X(1));
gam = theta - alpha;

%----------------------CONSTRAINTS-------------------------------
%    1-9.  xdot = 0        (steady flight)
%    10.   Va = 85 m/s     (airspeed)
%    11.   gam = 0         (flightpath)
%    12.   x2 = 0          (no sideslip constraints rudder and aileron)
%    13.   x7 = 0          (no bank angle constraints rudder and aileron)
%    14.   x9 = 0          (heading north)
Q = [xdot;
    85-Va;
    gam;
    X(2);
    X(7);
    X(9)];

H = diag(ones(1,14));

F0 = Q'*H*Q;
```

So the general flow of code should be

if starting from scratch

initialize guess of Z_guess = $\begin{pmatrix} \overline{x}_o \\ \overline{u}_o \end{pmatrix}$

else

load old solution

initialize guess of Z_guess = $\begin{pmatrix} \overline{x}_{\text{old}} \\ \overline{u}_{\text{old}} \end{pmatrix}$

end

$[Z^*, f_0]$ = fminsearch(cost function, initial guess, parameters)

check if solution satisfies constraints

save optimal solution

This yields a solution of

```
f0(X,u) = 9.0194e-20 (Final Cost)

---------------TRIM VALUES---------------------
x1 (u)           = 84.9905 m/s
x2 (v)           = 1.2629e-10 m/s
x3 (w)           = 1.2713 m/s
x4 (p)           = -2.0247e-10 rad/s
x5 (q)           = -3.639e-11 rad/s
x6 (r)           = 9.1864e-11 rad/s
x7 (phi)         = 6.7443e-11 rad
x8 (theta)       = 0.014957 rad
x9 (psi)         = 1.3425e-10 rad
u1 (aileron)     = 9.363e-08 deg
u2 (stabilizer) = -10.1991 deg
u3 (rudder)      = 1.8432e-07 deg
u4 (throttle 1) = 0.082083
u5 (throttle 2) = 0.082083
-----------------------------------------------
```

We can check that this satisfies the constraints

```
---------------VERIFY CONSTRAINTS--------------
xdot1            = -4.702le-12 m/s^2
xdot2            = -2.3338e-11 m/s^2
xdot3            = 1.7341e-11 m/s^2
xdot4            = 1.7692e-11 rad/s^2
xdot5            = -7.773le-13 rad/s^2
xdot6            = 1.3921e-11 rad/s^2
xdot7            = -2.011e-10 rad/s
xdot8            = -3.639e-11 rad/s
xdot9            = 9.1874e-11 rad/s
Va               = 85 m/s
gamma            = 5.3575e-10 deg
x2 (v)           = 1.2629e-10 m/s
x7 (phi)         = 6.7443e-11 rad
x9 (psi)         = 1.3425e-10 rad
-----------------------------------------------
```

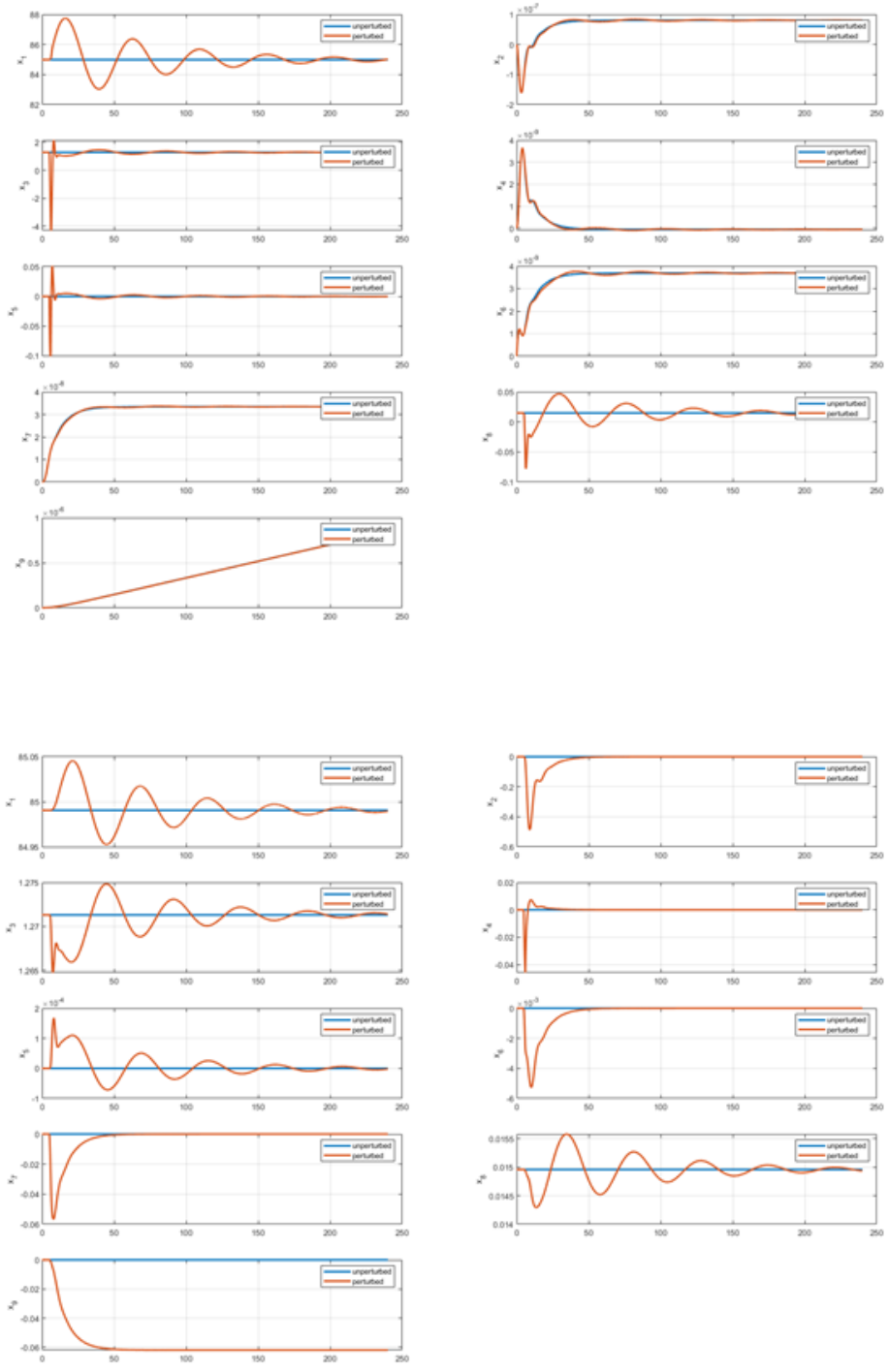
As can be seen, this does indeed satisfy all the constraints.

Notice that there are many small numbers that are effectively zero. It may be useful to set these to exactly zero.

```
---------------TRIM VALUES---------------------
x1 (u)           = 84.9905 m/s
x2 (v)           = 0 m/s
x3 (w)           = 1.2713 m/s
x4 (p)           = 0 rad/s
x5 (q)           = 0 rad/s
x6 (r)           = 0 rad/s
x7 (phi)         = 0 rad
x8 (theta)       = 0.014957 rad
x9 (psi)         = 0 rad
u1 (aileron)     = 0 deg
u2 (stabilizer) = -10.1991 deg
u3 (rudder)      = 0 deg
u4 (throttle 1) = 0.082083
u5 (throttle 2) = 0.082083
-----------------------------------------------
```

This is our solution for trimmed straight and level flight.

We can verify that this is a valid solution by initializing our Simulink model of the aircraft with this initial condition and constant control surface deflections and verify that it flies straight and level. Below are responses to perturbations in u_2 and u_1, respectively



Trimming Using Matlab's Linear Analysis Tool

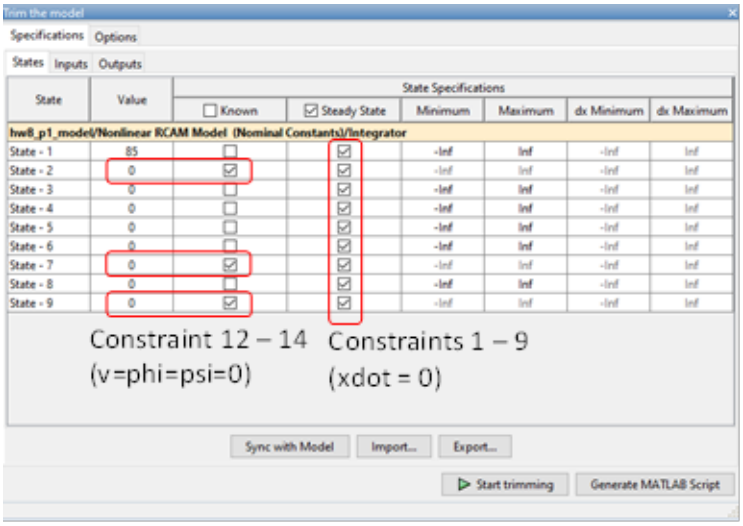
We can also use Matlab's Linear Analysis Tool to find a trim point. See YouTube video entitled 'Trimming a Simulink Model Using the Linear Analysis Tool' (https://youtu.be/kypswO4RLkk)

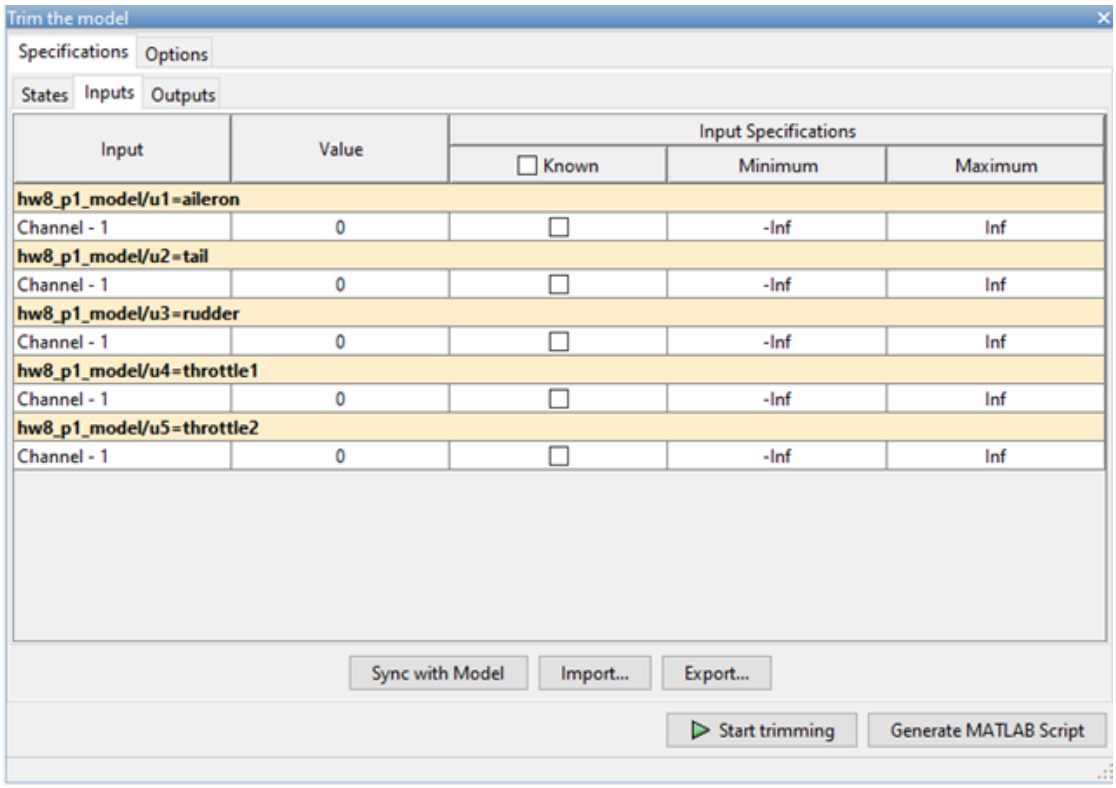
We first build the appropriate Simulink model (show model).

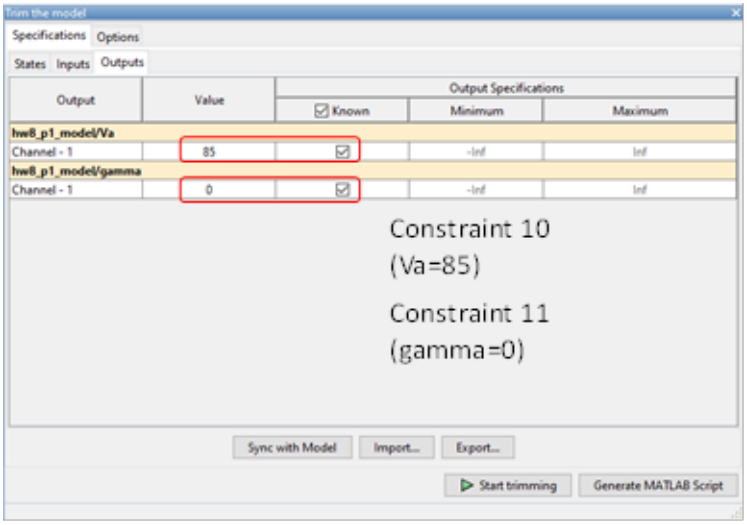
Simulink > Analysis > Control Design > Linear Analysis Tool

Operating Point > Trim Model

We then apply appropriate constraints in the tool







This yields a successful trim solution

Trim progress viewer

```
Optimizing to solve for all desired dx/dt=0, x(k+1)-x(k)=0, and y=ydes.

(Maximum Error) Block
-----------------------------------------------------------
(1.53353e+00) hw8_p1_model/Nonlinear RCAM Model (Nominal Constants)/Integrator
(1.11842e-02) hw8_p1_model/Nonlinear RCAM Model (Nominal Constants)/Integrator
(1.35176e-07) hw8_p1_model/Nonlinear RCAM Model (Nominal Constants)/Integrator
(1.35176e-07) hw8_p1_model/Nonlinear RCAM Model (Nominal Constants)/Integrator

Operating point specifications were successfully met.

An operating point op_trim1 has been created.
```

The values are

Edit: op_trim1

Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
hw8_p1_model/Nonlinear RCAM Model (Nominal Constants)/Integrator				
State - 1	[-Inf , Inf]	84.9905	0	5.2556e-09
State - 2	0	0	0	1.1402e-16
State - 3	[-Inf , Inf]	1.2713	0	-8.9445e-08
State - 4	[-Inf , Inf]	6.8153e-20	0	9.3325e-19
State - 5	[-Inf , Inf]	-9.8545e-22	0	1.3518e-07
State - 6	[-Inf , Inf]	2.8939e-19	0	1.3672e-17
State - 7	0	0	0	7.2482e-20
State - 8	[-Inf , Inf]	0.014957	0	-9.8545e-22
State - 9	0	0	0	2.8943e-19

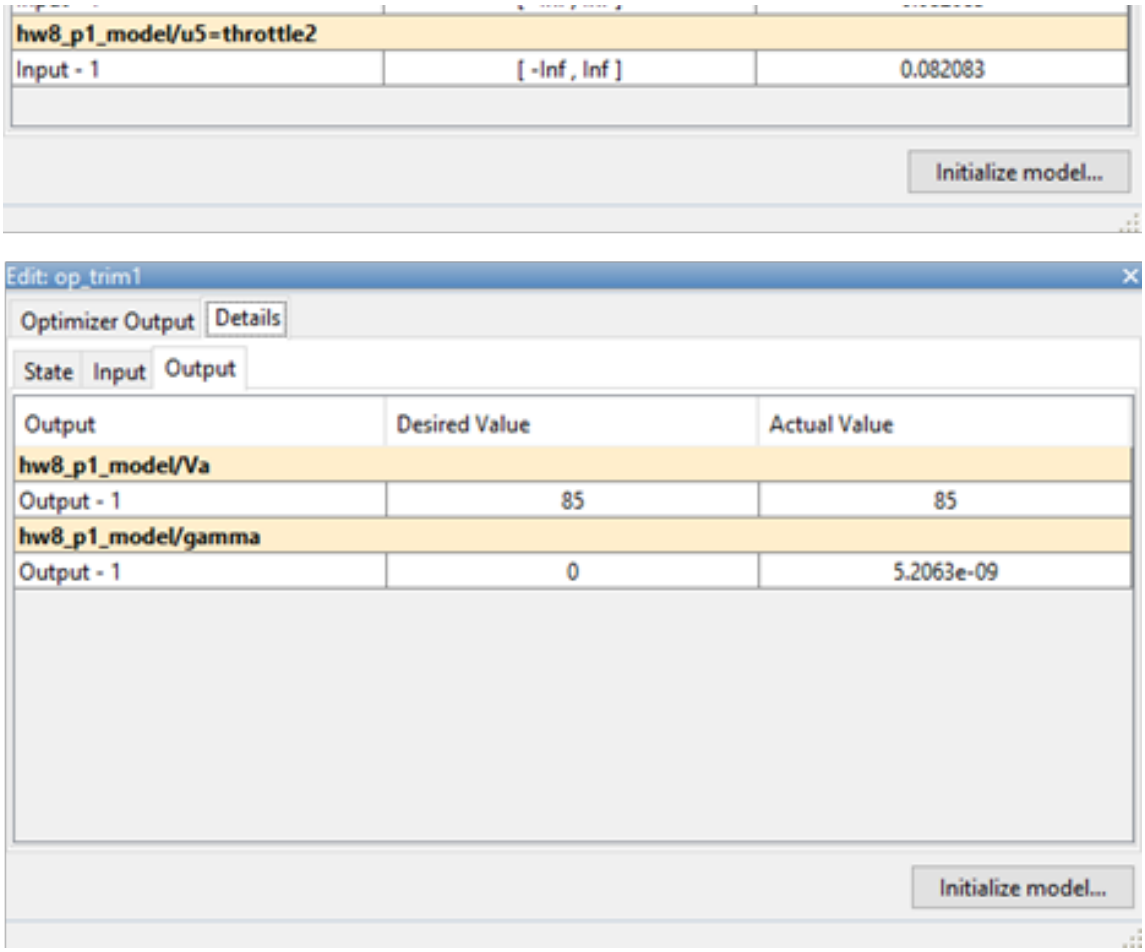
Initialize model...

Edit: op_trim1

Optimizer Output | Details

State | Input | Output

Input	Desired Value	Actual Value
hw8_p1_model/u1=aileron		
Input - 1	[-Inf , Inf]	2.4339e-17
hw8_p1_model/u2=tail		
Input - 1	[-Inf , Inf]	-0.17801
hw8_p1_model/u3=rudder		
Input - 1	[-Inf , Inf]	6.0199e-17
hw8_p1_model/u4=throttle1		
Input - 1	[-Inf , Inf]	0.082083



hw8_p1_model/u5=throttle2		
Input - 1	[-Inf , Inf]	0.082083

Initialize model...

Edit: op_trim1

Optimizer Output | Details

State | Input | Output

Output	Desired Value	Actual Value
hw8_p1_model/Va		
Output - 1	85	85
hw8_p1_model/gamma		
Output - 1	0	5.2063e-09

Initialize model...

Which is the same solution we obtained previously.

Other Trim Points

This technique can be applied to find other trim points (for example we will later investigate a steady state coordinated turn).

Next Steps

Once we have the equilibrium or trim point, we can analyze the system's stability about this point by linearizing about this point (next lecture).