MECH 6313 - Homework 5

Jonas Wagner

2021, April 14

Contents

1 Problem 1

The standard mass-spring-damper system described by

$$m\ddot{y} + \beta\dot{y} + ky = u \tag{1}$$

1.1 Part a

Problem: Design a gradient algorithm to estimate the unknown parameters m, β, and k from known inputs and outputs $u(t)$ and $y(t)$.

Solution: The system identification problem can be reformatted for a recursive estimator for constant parameters as follows:

First he spring-mass-damper dynamics can be rewritten in the following form:

$$\ddot{y} = -\frac{\beta}{m}\dot{y} - \frac{k}{m}y + \frac{1}{m}u \tag{2}$$

$$s^2Y = -\frac{\beta}{m}sY - \frac{k}{m}Y + \frac{1}{m}U \tag{3}$$

Letting $\Lambda(s) = s^2 + \lambda_1 s + \lambda_0$ to represent the charectoristic polynomial of a 2nd-order filter,

$$\frac{s^2Y}{\Lambda(s)} = \frac{-\frac{\beta}{m}sY - \frac{k}{m}Y + \frac{1}{m}U}{\Lambda(s)} \tag{4}$$

$$Y(s) = \left(\frac{1}{m}\right)\frac{1}{\Lambda(s)}U - \left(\frac{\beta}{m} - \lambda_1\right)\frac{s}{\Lambda(s)}Y - \left(\frac{k}{m} - \lambda_0\right)\frac{1}{\Lambda(s)}Y \tag{5}$$

Let

$$\theta = \begin{bmatrix} \frac{1}{m} \\ \frac{\beta}{m} - \lambda_1 \\ \frac{k}{m} - \lambda_0 \end{bmatrix} = \begin{bmatrix} b_1 \\ \bar{a}_1 \\ \bar{a}_0 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \tag{6}$$

Known measured and imputed quantities can then be defined as

$$\Psi(s) = \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{\Lambda(s)}U \\ \frac{-s}{\Lambda(s)}Y \\ \frac{-1}{\Lambda(s)}Y \end{bmatrix} \tag{7}$$

which can be used alongside the unknown parameters to define the following standard form parameter estimation problem:

$$Y(s) = \Psi(s)^T\Theta \tag{8}$$

This can then be further developed to define a gradient-based parameter estimator.

First, let $\hat{\Theta}$ be defined as the estimate of the parameters Θ and the estimation error, $\tilde{\Theta}$ of the parameter be defined as

$$\tilde{\Theta}(t) = \Theta - \hat{\Theta}(t)$$

and the convergence dynamics should be defined as

$$\dot{\tilde{\Theta}}(t) = -\dot{\hat{\Theta}}(t)$$

Subsequently the output error is can be defined in terms of the parameter error by the following:

$$\begin{aligned} e(t) &= y(t) - \hat{y}(t) && (9)\\ &= \Psi^T(t)\Theta - \Psi^T(t)\hat{\Theta}(t) && (10)\\ &= \Psi^T(t)\Big[\Theta - \hat{\Theta}(t)\Big] && (11)\\ &= \Psi^T(t)\tilde{\Theta}(t) && (12) \end{aligned}$$

A simple gradient decent algorithm can then be defined by:

$$\begin{aligned} \frac{1}{2}e^2(t) &= \frac{1}{2}e^T(t)e(t) && (13)\\ &= \frac{1}{2}\tilde{\Theta}^T(t)\Psi(t)\Psi^T(t)\tilde{\Theta} && (14) \end{aligned}$$

and since

$$\dot{\tilde{\Theta}} = -\boldsymbol{\nabla}_{\tilde{\Theta}}\left[\frac{1}{2}e^2(t)\right] \tag{15}$$

an LTV system can be defined as

$$\dot{\tilde{\Theta}} = -\Psi(t)\Psi^T(t)\tilde{\Theta} \tag{16}$$

or equivelently, since $e(t) = \Psi^T(t)\tilde{\Theta}(t)$

$$\dot{\tilde{\Theta}} = -\Psi(t)\Psi^T(t)e(t) \tag{17}$$

This algorithm can then be implemented for this specific system by defining the estimator dynamic matrice as:

$$\begin{aligned} A &= \Psi(t)\Psi^T(t)\\ B &= -\Psi(t) \end{aligned} \tag{18}$$

1.1.1 Part b

Problem: Simulate the algorithm for $m = 20$, $\beta = 0.1$, and $k = 5$ for different choices of $u(t)$ and resulting parameter convergence properties.

Solution: This was simulated in MATLAB using Simulink. The code can be found in AppendixA. The simulink model can be seen in Figure 1.

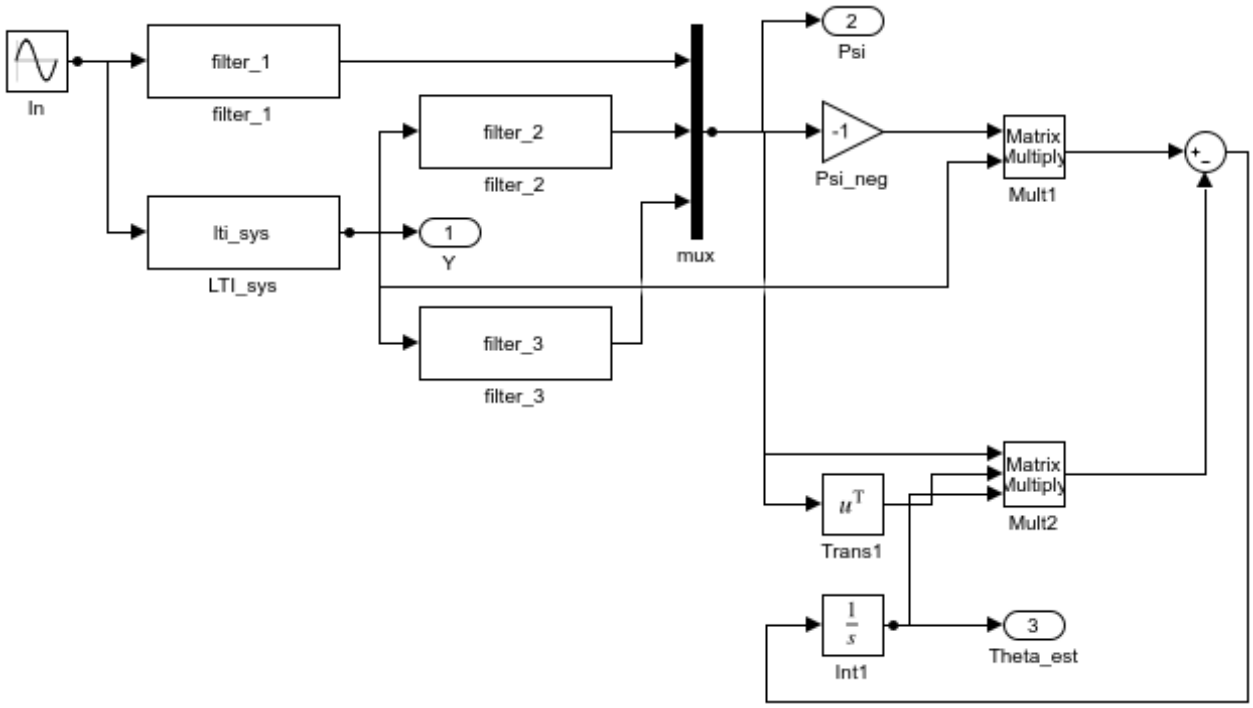

Figure 1: Simulink Model used to simulate the parameter estimation

The simulation results can be seen in Figure 2

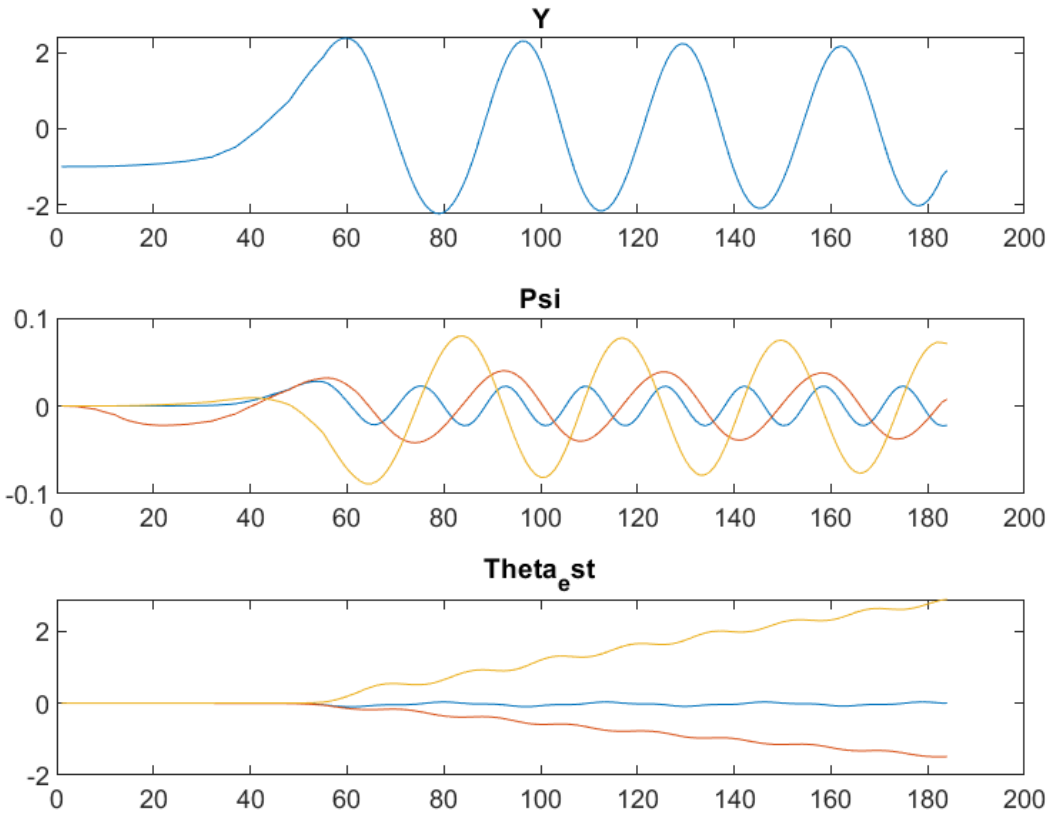

Figure 2: Simulink Results

2 Problem 2

Considering the reference model

$$\dot{y}_m = -ay_m + r(t),\ a > 0 \tag{19}$$

and the plant given as

$$\dot{y} = a^* y + b^* u,\ b^* \neq 0 \tag{20}$$

2.1 Part a

Problem: Show that a controller of form

$$u = \theta_1 y + \theta_2 r(t)$$

with gains θ_1^* and θ_2^* stabilizes the tracking error $e := y - y_m$ asymptotically to zero.

Solution: A method to achieve the calculate the tracking error is to implement a Model Reference Adaptive Controller (MRAC) that will change the dynamics of the plant into the reference model. In an ideal case, the parameters a and b will be known and a control law, $u = \theta_1^* y + \theta_2^* r(t)$, can be defined directly as follows:

$$-ay_m + r(t) = \dot{y}_m = \dot{y} = a^* y + b^*(\theta_1^* y + \theta_2^* r(t)) \tag{21}$$

$$-ay_m + r(t) = (a^* + b^*\theta_1)y + b^*\theta_2 r(t) \tag{22}$$

Clearly they dynamics are therefore equivalent if the following are true:

$$-a = (a^* + b^*\theta_1^*) \tag{23}$$

$$1 = b^*\theta_2^* \tag{24}$$

Thus these dynamics would be equivalent dynamics for gains defined as:

$$\theta_1^* = \frac{-(a + a^*)}{b^*} \tag{25}$$

$$\theta_2^* = \frac{1}{b^*} \tag{26}$$

The error dynamics for

$$e(t) = y - y_m$$

can be defined as

$$\dot{e}(t) = \frac{\mathrm{d}}{\mathrm{d}t}(y - y_m) \tag{27}$$

$$= (a^* + b^*\theta_1^*)y + b^*\theta_2^* r(t) - (-ay_m + r(t)) \tag{28}$$

$$= (a^* + b^*\frac{-(a + a^*)}{b^*})y + b^*\frac{1}{b^*}r(t) - (-ay_m + r(t)) \tag{29}$$

$$= -(a^* - a^* + a)y + r(t) - (-ay_m + r(t)) \tag{30}$$

$$= -ay - a(-y_m) = -a(y - y_m) \tag{31}$$

$$\dot{e} = -ae \tag{32}$$

which for $a > 0$ is clearly GAS.

2.2 Part b

Problem: Suppose a^* and $b*$ are unknown but the sign of b^* is known. Show that the adaptive implementation of the controller can achieve tracking when the gains are updated according to the following rule:

$$\begin{aligned}\dot{\theta}_1 &= -\text{sign}(b^*)\gamma_1 y e \\ \dot{\theta}_2 &= -\text{sign}(b^*)\gamma_2 r e\end{aligned} \tag{33}$$

with $\gamma_1, \gamma_2 > 0$.

Solution: Effective tracking occurs when the error, $e = y - y_m$, asymptotically decays to zero. For the MRAC with control law defined as

$$u = \theta_1 y + \theta_2 r$$

the asymptotic stability can be demonstrated as follows: Let

$$\tilde{\theta}_1 = \theta_1 - \theta_1^* \tag{34}$$

$$\tilde{\theta}_2 = \theta_2 - \theta_2^* \tag{35}$$

then the following equivalent gains

$$\theta_1 = \theta_1^* + \tilde{\theta}_1 \tag{36}$$

$$\theta_2 = \theta_2^* + \tilde{\theta}_2 \tag{37}$$

can be substituted into the error dynamics

$$\dot{e}(t) = \frac{\mathrm{d}}{\mathrm{d}t}(y - y_m) \tag{38}$$

$$= (a^* + b^*\theta_1)y + b^*\theta_2 r(t) - (-a y_m + r(t)) \tag{39}$$

$$= (a^* + b^*(\theta_1^* + \tilde{\theta}_1))y + b^*(\theta_2^* + \tilde{\theta}_2)r(t) - (-a y_m + r(t)) \tag{40}$$

$$= (a^* + b^*\theta_1^*)y + b^*\theta_2^* r(t) - (-a y_m + r(t)) + b^*\tilde{\theta}_1 y + b^*\tilde{\theta}_2 r(t) \tag{41}$$

$$= (a^* + b^*\frac{-(a + a^*)}{b^*})y + b^*\frac{1}{b^*}r(t) - (-a y_m + r(t))b^*\tilde{\theta}_1 y + b^*\tilde{\theta}_2 r(t) \tag{42}$$

$$= -(a^* - a^* + a)y + r(t) - (-a y_m + r(t)) + b^*\tilde{\theta}_1 y + b^*\tilde{\theta}_2 r(t) \tag{43}$$

$$= -a(y - y_m) + b^*(\tilde{\theta}_1 y + \tilde{\theta}_2 r(t)) \tag{44}$$

$$= -(a)e(t) + b^*(\tilde{\theta}_1 y + \tilde{\theta}_2 r(t)) \tag{45}$$

From this it is clear that the error dynamics are the same as the ideal plus the additional terms associated with the gain errors.

Therefore effective tracking can be achieved if the combined system defined as follows by combining (45) with the time derivatives of gain errors (equivalent to (33)):

$$\begin{aligned}\dot{e}(t) &= -(a)e + b^*(\tilde{\theta}_1 y + \tilde{\theta}_2 r(t)) \\ \dot{\tilde{\theta}}_1 &= -\tilde{\gamma}_1\ y\ e \\ \dot{\tilde{\theta}}_2 &= -\tilde{\gamma}_2\ r\ e\end{aligned} \tag{46}$$

where $\text{sign}(\tilde{\gamma}_i) = \text{sign}(b^*)$ and $|\tilde{\gamma}_i| = |\gamma_i|$ for $i = 1, 2$.

2.2.1 Error Dynamics Origin Stability

Upon analysis of (46), it is clear that there is an equilibrium point when

$$e = \dot{\tilde{\theta}}_1 = \dot{\tilde{\theta}}_2 = 0$$

and since an effective tracker exists at that equilibrium point, if the error dynamics are GAS about that point then it is an effective tracker.

The linearized system matrices about the equivalent origin is given as

$$A = \begin{bmatrix} -a & b^*y & b^*r \\ -\tilde{\gamma}_1\ y & 0 & 0 \\ -\tilde{\gamma}_2\ r & 0 & 0 \end{bmatrix}\Bigg|_{y=r=0} = \begin{bmatrix} -a & & \\ & 0 & \\ & & 0 \end{bmatrix} \tag{47}$$

with $x = \begin{bmatrix} e \\ \tilde{\theta}_1 \\ \tilde{\theta}_2 \end{bmatrix}$ and time-varying parameters y and $r(t)$.

It is clear the linearzed system is at least marginally zero so any tracking errors caused will at least be bounded locally. It is also clear that if the dynamic gain component is ignored, the error will asymptotically approach zero as demonstrated by (32).This isn't enough to demonstrate GAS though.

2.2.2 Error Lyapnov Equation

Using a lyapnov function

$$x^T P x$$

for $P = \frac{1}{2} I_3$ the error dynamics at the origin can be better analyzed.

In this case the a similarly structured time varying system system will be defined as

$$\begin{aligned} \dot{e} &= -ae + by(t)\theta_1 + br(t)\theta_2 \\ \dot{\theta_1} &= -c_1 y(t) e \\ \dot{\theta_2} &= -c_2 r(t) e \end{aligned} \tag{48}$$

with time varying parameters being the signals $y(t)$ and $r(t)$, along with scaler constants a, b, c_1, c_2.

For a state vector $x = \begin{bmatrix} e \\ \theta_1 \\ \theta_2 \end{bmatrix}$, the LTV state-space dynamics can be defined by:

$$A = \begin{bmatrix} -a & by(t) & br(t) \\ -c_1 y(t) & 0 & 0 \\ -c_2 r(t) & 0 & 0 \end{bmatrix} \tag{49}$$

The stability of these dynamics can be tested for the same marginal stability as shown previously by applying Lyopnov methods with $P = \frac{1}{2} I_3$.

$$V(e, \theta_1, \theta_2) = x^T P x = \frac{1}{2} e^2 + \frac{1}{2}\theta_1^2 + \frac{1}{2}\theta_2^2 \tag{50}$$

$$\dot{V}(e, \theta_1, \theta_2) = 2 P x^T \dot{x} = e\dot{e} + \theta_1\dot{\theta}_1 + \theta_2\dot{\theta}_2 \tag{51}$$

(52)

$$\dot{V} = x^T(2\left(\frac{1}{2}\right)I_3)\dot{x} = e(-ae + by(t)\theta_1 + br(t)\theta_2) + \theta_1(-c_1y(t)e) + \theta_2(-c_2r(t)e) \tag{53}$$

$$= -ae^2 + (b - c_1)(y(t))(\theta_1 e) + (b - c_1)(r(t))(\theta_2 e) \tag{54}$$

$$= -x^T \begin{bmatrix} a & & \\ & 0 & \\ & & 0 \end{bmatrix} x + (b - c_1)(y(t))(\theta_1 e) + (b - c_1)(r(t))(\theta_2 e) \tag{55}$$

from this the observation, once again, of marginal stability (at the origin) is obvious. For design of the tracking error itself, the $Q = \begin{bmatrix} a & & \\ & 0 & \\ & & 0 \end{bmatrix}$ matrix can be used to generate an observation matrix C.

Since $\dot{V} = -x^TQx = -x^TC^TCx$, C can be found as

$$Q = C^TC \tag{56}$$

$$\begin{bmatrix} a & & \\ & 0 & \\ & & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{a} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} \sqrt{a} & 0 & 0 \end{bmatrix} \tag{57}$$

therefore,

$$C = \begin{bmatrix} \sqrt{a} & 0 & 0 \end{bmatrix} \tag{58}$$

2.2.3 Universal Observability of (A+LC,C)

Uniform observability (UO) of the pair

$$(A(t), C(t))$$

can then be assured by first confirming UO of

$$A(t) + L(t)C(t), C(t)$$

by defining

$$L(t) = \begin{bmatrix} \sqrt{a} \\ \left(\frac{1}{\sqrt{a}}\right)y(t) \\ \left(\frac{1}{\sqrt{a}}\right)r(t) \end{bmatrix} \tag{59}$$

and then calculating

$$A(t) + L(t)C(t) = \begin{bmatrix} -a & by(t) & br(t) \\ -c_1y(t) & 0 & 0 \\ -c_2r(t) & 0 & 0 \end{bmatrix} + \begin{bmatrix} \sqrt{a} \\ \left(\frac{c_1}{\sqrt{a}}\right)y(t) \\ \left(\frac{c_2}{\sqrt{a}}\right)r(t) \end{bmatrix} \begin{bmatrix} \sqrt{a} & 0 & 0 \end{bmatrix} \tag{60}$$

$$= \begin{bmatrix} -a & by(t) & br(t) \\ -c_1 y(t) & 0 & 0 \\ -c_2 r(t) & 0 & 0 \end{bmatrix} + \begin{bmatrix} a & 0 & 0 \\ c_1 y(t) & 0 & 0 \\ c_2 r(t) & 0 & 0 \end{bmatrix} \tag{61}$$

$$A(t) + L(t)C(t) = \begin{bmatrix} 0 & by(t) & br(t) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{62}$$

The pair $(A(t) + L(t)C(t), C(t))$ is clearly observable as the output of a system governed by these dynamics would output

$$y = \sqrt{a}\ b\ (\theta_1 y(t) + \theta_2 r(t)) \tag{63}$$

which, assuming $y(t)$ and $r(t)$ is known, can be used to calculate all of the states.

This uniform stability can be confirmed for the pair $(A(t) + L(t)C(t), C(t))$ by recognizing that UO for the system:

$$\begin{aligned} \dot{x} &= (A(t) + L(t)C(t)) = \begin{bmatrix} 0 & by(t) & br(t) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} x \\ \hat{y} &= Cx = \begin{bmatrix} \sqrt{a} & 0 & 0 \end{bmatrix} x \end{aligned} \tag{64}$$

is the same as stating UO of the system:

$$\begin{aligned} \dot{x}_{2,3} &= \mathbf{0}_2 x \\ \hat{y}_{2,3} &= \sqrt{a} b \begin{bmatrix} y(t) & r(t) \end{bmatrix} x_{2,3} \end{aligned} \tag{65}$$

and this single C matrix can be used to prove UO.

UO (and UAS) if $\exists \delta,\ \alpha > 0$ s.t.

$$\int_{t_0}^{t_0+\delta} CC^T \, \mathrm{d}t \geq \alpha \mathbf{I}_2 \ \forall t_0 \tag{66}$$

$$\int_{t_0}^{t_0+\delta} (\sqrt{a}b)^2 \begin{bmatrix} y(t) & r(t) \end{bmatrix} \begin{bmatrix} y(t) \\ r(t) \end{bmatrix} \mathrm{d}t \geq \alpha \mathbf{I}_2 \ \forall t_0 \tag{67}$$

$$ab^2 \int_{t_0}^{t_0+\delta} (y(t))^2 + (r(t))^2 \, \mathrm{d}t \geq \alpha \mathbf{I}_2 \ \forall t_0 \tag{68}$$

which is clearly true for $y(t)$ and $r(t)$ that are persistently excited.

From this it can be concluded that the system is UAS. This can then be applied to to the original system noting that $c_i = \text{sign}(b^*)\gamma_i \ \forall i = 1, 2$.

2.3 Part c

Problem: Provide conditions that also guarantee that $\theta_1(t) \to \theta_1^*$ and $\theta_2(t) \to \theta_2^*$ as $t \to \infty$.

Solution:

From the definition of the ideal control, $u = \theta_1^* y + \theta_2^* r$, it is known that if $\theta_1(t) \to \theta_1^*$ and $\theta_2(t) \to \theta_2^*$ then the MRAC will effectively transform the plant dynamics to that of the reference model.

For the dynamics of the gains, it can be seen from (55) that for GAS, the following must be true:

$$\dot{V} = -ae^2 + (b - \text{sign}(b^*)\gamma_1)(y(t))(\tilde{\theta}_1 e) + (b - \text{sign}(b^*)\gamma_2)(r(t))(\tilde{\theta}_2 e) \prec 0 \tag{69}$$

Originally the terms that were not obviously $\preceq 0$ were ignored as they will at a minimum remain bounded. In order to also insure $\theta_1(t) \to \theta_1^*$ and $\theta_2(t) \to \theta_2^*$, the two other terms must be $\prec 0$.

From (68) it is known that the system is UO and therefore UAS regardless of the system parameters or tun-able γ constants, but this is guaranteed when it is known $y(t)$ and $r(t)$ are persistently excited.

3 Problem 3

A simplified model of an axial compressor, used in jet engine control studies, is given by the following second order system:

$$\begin{aligned}\dot{\phi} &= -\frac{3}{2}\phi^2 - \frac{1}{2}\phi^3 - \psi \\ \dot{\psi} &= \frac{1}{\beta^2}(\phi + 1 - u)\end{aligned} \tag{70}$$

This model captures the main surge instability between the mass flow and the pressure rise. Here, ϕ and ψ are deviations of the mass flow and the pressure rise from their set points, the control input u is the flow through the throttle, and β is a positive constant.

3.1 Part a

Problem: Use backstopping to obtain a control law to stabilize the origin.

Solution: This problem can be written in a more standard form of

$$\begin{aligned}\dot{x_1} &= f(x_1) + g(x_1)x_2 \\ \dot{x_2} &= f_a(x_1, x_2) + g_a(x_1, x_2)u\end{aligned} \tag{71}$$

where

$$f(x_1) = -\frac{3}{2}x_1^2 - \frac{1}{2}x_1^3 \tag{72}$$

$$g(x_1) = -1$$

$$f_a(x_1, x_2) = \frac{1}{\beta^2}(x_1 + 1) \tag{73}$$

$$g_a(x_1, x_2) = -1$$

and the state variables and inputs are defined such that $x_1 = \phi$, $x_2 = \psi$, and $u = u$.

First looking at the x_1 state equation independently and considering the x_2 state as an input, z_1, a proposed lyapnov equation

$$V_1(x_1) = \frac{1}{2}x_1^2$$

can be used to develop a virtual control signal $\alpha(x_1)$.

$$\dot{V}(x_1) = \frac{\mathrm{d}V}{\mathrm{d}x_1}\dot{x_1} \tag{74}$$

$$= x_1(f(x_1) + g(x_1)\alpha(x_1)) \tag{75}$$

$$= x_1\left(-\frac{3}{2}x_1^2 - \frac{1}{2}x_1^3 - \alpha(x_1)\right) \tag{76}$$

$$= -\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4 - x_1\alpha(x_1) \tag{77}$$

Since $-\frac{3}{2}x_1^3$ is not negative definite, $\alpha(x_1)$ can be defined as

$$\alpha(x_1) = -\frac{3}{2}x_1^2 + K_1x_1,\ K_1 > 0$$

and substituted in to cancel it out and add an additional asymptotically decreasing term

$$\dot{V}(x_1) = -\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4 - x_1\left(-\frac{3}{2}x_1^3 + Kx_1\right) \tag{78}$$

$$= -\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4 + \frac{3}{2}x_1^3 - K_1x_1^2,\ K_1 > 0 \tag{79}$$

$$= -\frac{1}{2}x_1^4 - K_1x_1^2 \leq -W_1(x_1) < 0,\ K_1 > 0 \tag{80}$$

Clearly this is negative definite, so the virtual control signal of $\alpha(x_1) = -\frac{3}{2}x_1^2 + K_1x_1$ is acceptable.

Next, define the additional state equation

$$z_2 = x_2 - \alpha(x_1) \tag{81}$$

$$\dot{z_2} = \dot{x_2} - \dot{\alpha} \tag{82}$$

$$\tag{83}$$

Next, an augmented Lyapnov function:

$$V_a(x_1, z_2) = V_1(x_1) + \frac{1}{2}z_2^2 \tag{84}$$

$$= \frac{1}{2}x_1^2 + \frac{1}{2}z_2^2 \tag{85}$$

stability can then be guaranteed as follows:

$$\dot{V}_a(x_1, z_2) = \dot{V}_1 + z_2\dot{z_2} \tag{86}$$

$$= \dot{V}_1 + z_2(\dot{x_2} - \dot{\alpha}) \tag{87}$$

$$= \dot{V}_1 + z_2((\frac{1}{\beta^2}(x_1 + 1 - u)) - \dot{\alpha}) \tag{88}$$

since $\dot{V}_1 \leq -W_1(x_1)$ this component is not problematic. Thus, setting

$$u = -x_1 - \dot{\alpha} + \beta^2 k_2 z_2,\ k_2 > 0$$

can be used to generate the following

$$\dot{V}_a = \dot{v_1} - z_2(-k_2z_2) \tag{89}$$

$$= -W_1(x_1) - k_2z_2^2 < 0 \tag{90}$$

Which means a control signal of

$$u = -x_1 - \dot{\alpha} + \beta^2 k_2(x_2 - \alpha(x_1)),\ k_2 > 0 \tag{91}$$

can be implemented in a static nonlinear controller to ensure GAS at the origin.

3.1.1 Part b

Problem: Use Sontag's Formula and the Control Lyapunov Function obtained previously to obtain an alternative control law.

Solution: The Control Lyapunov Function (CLF) that comes from the previous problem was calculated as

$$\frac{\mathrm{d}V}{\mathrm{d}x}f(x) + \frac{\mathrm{d}V}{\mathrm{d}x}g(x)\alpha(x) < 0 \tag{92}$$

$$(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4) + (-x_1\alpha(x_1)) < 0 \tag{93}$$

since

$$\frac{\mathrm{d}V}{\mathrm{d}x}f(x) + \frac{\mathrm{d}V}{\mathrm{d}x}g(x)\alpha(x) = L_fV(x) + L_gV(x)\alpha < 0$$

the CLF components are given as

$$L_fV(x) = -\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4 \tag{94}$$

$$L_gV(x) = -x_1 \tag{95}$$

This can then be entered into Sontag's Formula to attain a virtual control function of

$$\alpha(x) = \begin{cases} 0, & L_gV(x) = 0 \\ -\dfrac{L_fV(x) + \sqrt{(L_fV(x))^2 + (L_gV(x))^2}}{L_gV(x)}, & \text{otherwise} \end{cases} \tag{96}$$

$$= \begin{cases} 0, & -x_1 = 0 \\ -\dfrac{(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4) + \sqrt{(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4)^2 + (-x_1)^2}}{-x_1}, & \text{otherwise} \end{cases} \tag{97}$$

This therefore provides that while the system is not stabalized at the origin, the virtual control signal $\alpha(x_1)$ is given as:

$$\alpha(x_1) = \frac{(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4) + \sqrt{(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4)^2 + (x_1)^2}}{x_1} \tag{98}$$

$$= \frac{(-\frac{3}{2}x_1^3 - \frac{1}{2}x_1^4) + \sqrt{\frac{9}{2}x_1^6 + \frac{6}{4}x_1^5 + \frac{1}{4}x_1^8 + x_1^2}}{x_1} \tag{99}$$

$$= -\frac{3}{2}x_1^2 - \frac{1}{2}x_1^3 + \sqrt{\frac{9}{2}x_1^4 + \frac{6}{4}x_1^3 + \frac{1}{4}x_1^6 + 1} \tag{100}$$

This can then also be implemented into the control signal derived in the previous problem as

$$u = -x_1 - \dot{\alpha} + \beta^2 k_2(x_2 - \alpha(x_1)),\ k_2 > 0 \tag{101}$$

A MATLAB Code:

All code I write in this course can be found on my GitHub repository:
https://github.com/jonaswagner2826/MECH6313

Script 1: MECH6313_HW5

```
% MECH 6313 - HW5
% Jonas Wagner
% 2021-04-08
%

% Only Problem 1b
clear
close all

%% Settings
generateModel = true;
openModel = false;
simulateModel = true;
plotResults = true;

% Name of the simulink model
[cfolder,~,~] = fileparts(mfilename('fullpath'));
subfolder = ''; %include / at end of subfolder
fname = 'pblm1b';


%% System Definitions

% Physical Parameters
m = 20;
b = 0.1;
k = 5;

% Matrix Definitions
A = [-b/m, -k/m;
     1, 0];
B = [1/m;
     0];
C = [0, 1];
D = 0;
lti_sys = ss(A,B,C,D);
x0 = [1;
     -1];
```

```
39
40 % Filter TF
41 lambda_0 = 20;
42 lambda_1 = 2 * 1 * lambda_0;
43 filter_1 = tf([1], [1, lambda_1, lambda_0]);
44 % filter_1 = [filter_1; filter_1];
45 filter_2 = tf([1, 0], [1, lambda_1, lambda_0]);
46 % filter_2 = [filter_2; filter_2];
47 filter_3 = - filter_1;
48
49
50 if generateModel
51 %% Simulink Creation In Code
52 % Simulink Settings ----------------------
53 % Get the current configuration
54 cfg = Simulink.fileGenControl('getConfig');
55 % Changes Code Save Location
56 cfg.CacheFolder = [pwd, '\', subfolder];
57 cfg.CodeGenFolder = [pwd, '\', subfolder];
58 cfg.CodeGenFolderStructure = 'TargetEnvironmentSubfolder';
59 % Apply new Config
60 Simulink.fileGenControl('setConfig', 'config', cfg, 'keepPreviousPath',true, 'createDir',
       true);
61
62 % Check if the file already exists and delete it if it does
63 if exist(fname,'file') == 4
64     % If it does then check whether it's open
65     if bdIsLoaded(fname)
66         % If it is then close it (without saving!)
67         close_system(fname,0)
68     end
69     % delete the file
70     delete([fname,'.slx']);
71 end
72
73 % Create Simulink Model
74 new_system;
75
76 % Create Simiple Input
77 add_block('simulink/Sources/Sine Wave', [gcs, '/In']);
78
79 % Plant
80 add_block('cstblocks/LTI System', [gcs, '/LTI_sys'],...
```

```
    'sys','lti_sys',...
    'IC', 'x0');
u = add_line(gcs, 'In/1', 'LTI_sys/1');
set_param(u, 'Name', 'u')


% Filter 1
add_block('cstblocks/LTI System', [gcs, '/filter_1'],...
    'sys','filter_1');
add_line(gcs, 'In/1', 'filter_1/1');


% Filter 2
add_block('cstblocks/LTI System', [gcs, '/filter_2'],...
    'sys','filter_2');
add_line(gcs, 'LTI_sys/1', 'filter_2/1');


% Filter 1
add_block('cstblocks/LTI System', [gcs, '/filter_3'],...
    'sys','filter_3');
add_line(gcs, 'LTI_sys/1', 'filter_3/1');


% mux
add_block('simulink/Commonly Used Blocks/Mux', [gcs,'/mux'],...
    'inputs', '3');
psi_1 = add_line(gcs, 'filter_1/1', 'mux/1');
psi_2 = add_line(gcs, 'filter_2/1', 'mux/2');
psi_3 = add_line(gcs, 'filter_3/1', 'mux/3');
set_param(psi_1, 'Name', 'psi_1')
set_param(psi_2, 'Name', 'psi_2')
set_param(psi_3, 'Name', 'psi_3')


% Parameter Estimator
% -------------------------
% Negative Gain Block
add_block('simulink/Commonly Used Blocks/Gain', [gcs,'/Psi_neg'],...
    'gain', '-1');
Psi = add_line(gcs, 'mux/1', 'Psi_neg/1');
set_param(Psi, 'Name', 'Psi')

```

```
124
125
126 % Product Block
127 add_block('simulink/Math Operations/Product', [gcs, '/Mult1'],...
128     'inputs', '**',...
129     'Multiplication', 'Matrix(*)');
130 Psi_neg = add_line(gcs, 'Psi_neg/1', 'Mult1/1');
131 set_param(Psi_neg, 'Name', 'Psi_neg')
132 Y = add_line(gcs, 'LTI_sys/1', 'Mult1/2');
133 set_param(Y, 'Name', 'Y')
134
135
136 % Create Sum block
137 add_block('simulink/Commonly Used Blocks/Sum', [gcs, '/Sum1'],...
138     'inputs', '|+-');
139 theta_m = add_line(gcs, 'Mult1/1', 'Sum1/1');
140 set_param(theta_m, 'Name', 'theta_m')
141
142 % Creates Integrator
143 add_block('simulink/Commonly Used Blocks/Integrator', [gcs, '/Int1']);
144 theta_error = add_line(gcs, 'Sum1/1', 'Int1/1');
145 set_param(theta_error, 'Name', 'theta_error');
146
147 % Create Transpose
148 add_block('simulink/Math Operations/Math Function', [gcs, '/Trans1'],...
149     'Function', 'transpose')
150 add_line(gcs, 'mux/1', 'Trans1/1')
151
152 % Product Block (feedback)
153 add_block('simulink/Math Operations/Product', [gcs, '/Mult2'],...
154     'inputs', '***',...
155     'Multiplication', 'Matrix(*)');
156 Psi = add_line(gcs, 'mux/1', 'Mult2/1');
157 set_param(Psi, 'Name', 'Psi')
158 add_line(gcs, 'Trans1/1', 'Mult2/2');
159 % add_param(Psi_T, 'Name', 'Psi_T')
160 theta_est = add_line(gcs, 'Int1/1', 'Mult2/3');
161 set_param(theta_est, 'Name', 'theta_est')
162
163 add_line(gcs, 'Mult2/1', 'Sum1/2')
164
165
166 % Create Simple Scope/Output
```

```
add_block('simulink/Sinks/Out1', [gcs, '/Y']);
y = add_line(gcs, 'LTI_sys/1', 'Y/1');
set_param(y, 'Name', 'y');
add_block('simulink/Sinks/Out1', [gcs, '/Psi']);
Psi = add_line(gcs, 'mux/1', 'Psi/1');
set_param(Psi, 'Name', 'Psi');
add_block('simulink/Sinks/Out1', [gcs, '/Theta_est']);
Theta_est = add_line(gcs, 'Int1/1', 'Theta_est/1');
set_param(Theta_est, 'Name', 'Theta_est');



% https://www.mathworks.com/help/simulink/ug/creating-an-example-model-that-uses-a-matlab
    -function-block.html
% S-block implimentation... ran out of time for implimenatation
% add_block('simulink/User-Defined Functions/Level-2 MATLAB S-Function', [gcs, '/Est
    '],...
% 'FunctionName', 'param_est')
% add_line(gcs, 'Mult1/1', 'Est/1');
% add_line(gcs, 'LTI_sys/1', 'Est/2');


% Auto Arrange
Simulink.BlockDiagram.arrangeSystem(gcs) %Auto Arrange

%% Save and Open System
save_system(gcs,[cfolder, '\', subfolder, fname]);
print(['-s', gcs], '-dpng',... % Print model to figure
    [cfolder, '\' subfolder, 'fig\', 'sim_model_', fname, '.png'])

end
if openModel
    open(fname); % Don't need to open to run
end

if simulateModel
%% Simulate System
simConfig.SaveState = 'on';
simOut = sim(fname, 'SaveState', 'on', 'StartTime', '0', 'StopTime', '50');

% Sim Data
Y_out = simOut.yout{1}.Values;
Psi_out = simOut.yout{2}.Values;
```

```
Theta_out = simOut.yout{3}.Values;
Xout = simOut.xout{1}.Values.Data; %Only works by grabbing states of first block (LTI_sys
    )
end

if plotResults
%% Plot Results

fig = figure;
subplot(3,1,1)
plot(Y_out.Data(:,1))
title('Y')


subplot(3,1,2)
plot(Psi_out.Data(:,1))
hold on
plot(Psi_out.Data(:,2))
hold on
plot(Psi_out.Data(:,3))
title('Psi')

subplot(3,1,3)
plot(Theta_out.Data(:,1))
hold on
plot(Theta_out.Data(:,2))
hold on
plot(Theta_out.Data(:,3))
title('Theta_est')

% fig = figure;
% plot(Xout(:,1))
% hold on
% plot(Xout(:,2))
% legend('X_1', 'X_2')
% title('Saturated Double Integration Response while stabalized')
saveas(fig, [cfolder, '\',subfolder, 'fig\', fname,'_results.png'])
close all
end
```