Homework 3: Model Reference Adaptive Systems

Instructor: Marzia Cescon

Table of Contents

The **model reference adaptive system** (MRAS) is an example of adaptive scheme in which the specifications of performance are expressed in terms of a reference model, which gives the desired response to a command signal. Due to this property, MRAS may be regarded as an adaptive servo system. The block diagram is illustrated in Figure 1. The controller can be thought of as having two loops. The inner loop is an ordinary feedback loop consisting in the process and a controller. The outer loop is another feedback loop which adjusts the controller parameters so that the error computed as the difference between the output of the system and the output of the reference model is small. The key problem with MRAS is to determine the adjustment mechanism such that a stable system which brings the error to zero is obtained.
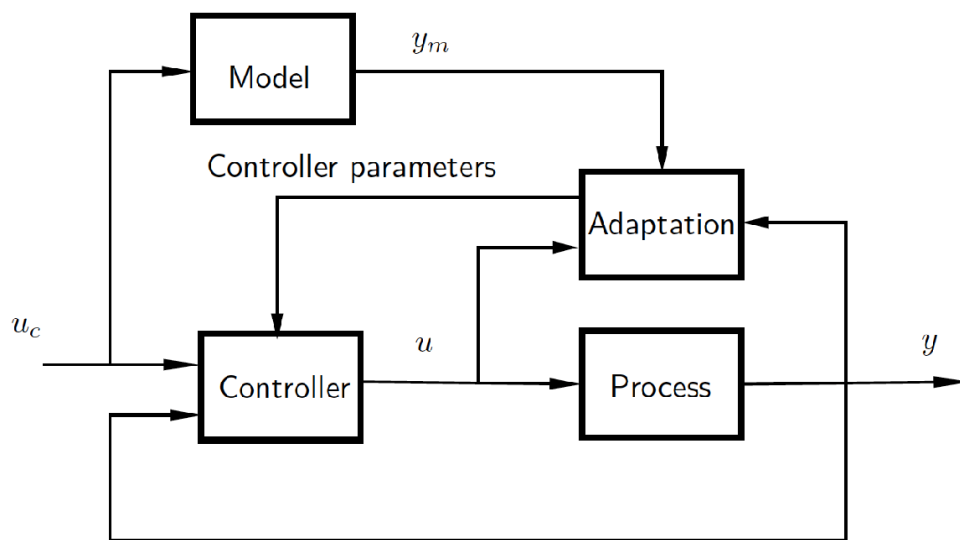

Figure 1. Block diagram of a model reference adaptive system.

This homework is a simulation exercise with the goal of providing a good introduction to MRAS. The examples are drawn from the book: "Adaptive Control", by K.J. Åström and B. Wittenmark. Two ways to obtain

mechanisms for the parameters adjustment are studied: the first one uses a gradient-based method, while the second one applies Lyapunov stability theory.

1. Adaptation of a feed-forward gain

We first consider the problem of adjusting a feedforward gain. Let us define the differential operator $p = \dfrac{d}{dt}$.

We assume that the process is a linear stable SISO process, with input-output relationship $y(t) = G(p)u(t)$, with $G(p) = \dfrac{k}{p+1}$, for some unknown parameter $k > 0$, and that the reference model is given by the expression $y_m = G_m(p)u_c(t)$, with $G_m(p) = \dfrac{k_0}{p+1}$, where $k_0 > 0$ is a known constant. We wish to construct a control law $u(t)$ for the process, which makes its output $y(t)$ behave as the output of the reference model $y_m(t)$ when controlled by the command signal $u_c(t)$. If such a control law can be established, then the process with unknown gain may be controlled with an outer feedforward law designed for the reference model.

Clearly, if k was known, the problem could be solved by a simple proportional controller. Letting $u(t) = \theta(t)u_c(t)$ with $\theta(t) = k_0/k \ \forall t$, then $y(t) = G_m(p)u(t) = G_m(p)\theta(t)u_c(t) = G_m(p)u_c(t) = y_m(t)$.

However, if the gain of the process is unknown, a way of adaptively learning the parameter $\theta(t)$ must be derived.

1.1. The MIT rule

The most intuitive way of accomplishing the model matching, with $y(t) \to y_m(t)$ as $t \to \infty$, is to define the error between the process response and the reference model response and attempt to minimize it. Let the error be defined as a function of $\theta(t)$:

$$e(\theta, t) = y(t) - y_m(t) = G(p)\theta(t)u_c(t) - G_m(p)u_c(t).$$

In order to minimize the error, we define a quadratic loss function $J(\cdot)$: $J(\theta, t) = \dfrac{1}{2}e^2(\theta, t)$. The time derivative of $J(\cdot)$ takes the form:

$$\frac{d}{dt}J(\theta, t) = \frac{\partial}{\partial\theta}J(\theta, t) \cdot \frac{d\theta(t)}{dt} = e(\theta, t) \cdot \frac{\partial}{\partial\theta}e(\theta, t) \cdot \frac{d\theta}{dt} \quad .$$

The following relations are derived for some constant $\alpha > 0$:

$$\frac{d\theta(t)}{dt} = -\alpha\left(e(\theta, t)\frac{d\,e(\theta, t)}{dt}\right) \Rightarrow \frac{d\,J(\theta, t)}{dt} = -\alpha\,e^2\left(\partial\frac{e(\theta, t)}{\partial\theta}\right)^2 \leq 0.$$

This gradient-based method used to update the parameter θ is known as the **MIT-rule**.

Let us calculate the sensitivity derivative $\dfrac{\partial e}{\partial\theta}$:

$$\frac{\partial e(\theta, t)}{\partial\theta} = G(p)u_c(t) = \frac{k}{k_0}G_m(p)u_c(t) = \frac{k}{k_0}y_m$$

By defining $\gamma = \alpha \frac{k}{k_0}$ we obtain an adjustment law for the parameter $\theta(t)$:

$$\frac{d\theta(t)}{dt} = -\alpha \left(e(\theta, t) \frac{\partial e(\theta, t)}{\partial t} \right) = -\alpha \frac{k}{k_0} e(\theta, t) y_m(t) = -\gamma e(\theta, t) y_m(t)$$

Figure 2 depicts the block diagram implementing this type of MRAS for the adjustment of a feedforward gain applying the MIT rule with adaptation law just derived.
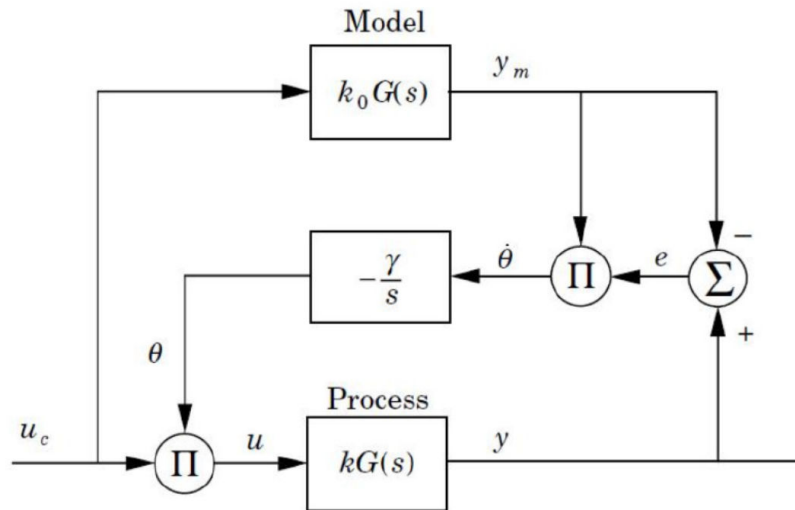

Figure 2: Block diagram of the MRAS feedforward gain adaptation using the MIT-rule synthesis.

Task 1. In order to perform the simulations to verify and experiment with the theory, you will need to build a model of the system. The Simulink library Library_1 contains a reference generator, a set of blocks describing first order process and reference models, as well as functionalities for plotting. It may be beneficial to connect the provided goto-ports to their corresponding signals - this uses the neat signal routing functionality Simulink for plotting.

1. Implement the feedback control in Figure 1 in the Task1_MIT.mdl model using Library_1 by dragging desired blocks/components.

```
open('Task1_MIT.mdl')
```

2. Run the system with $\gamma = 1$, $k = 1$, $k_0 = 2$ to verify the implementation.

```
% Parameters defining the model
k = 1;
k0 = 2;
gamma = 1;
% Simulate
sim('Task1_MIT')
```

Provide plots of the signals involved, i.e., u_c, y, y_m, e, θ. Can you briefly describe what happens during the simulation? Hint: Use the dataset logsout to plot the responses.

3

```
% Plot
HW3_plot1(logsout)
```

Enter your text here.

3. How does the rate of the adaptation change with γ? Run the simulation with different values. Provide plots and relative explanations corresponding to the tested scenarios.

   • **Trial 1:**

```
gamma = 1; % Change this value.
sim('Task1_MIT.mdl')
HW3_plot1(logsout)
```

   • **Trial 2:**

```
gamma = 1;
sim('Task1_MIT.mdl')
HW3_plot1(logsout)
```

   • **Trial 3:**

```
gamma = 1;
sim('Task1_MIT.mdl')
HW3_plot1(logsout)
```

Enter your text here.

4. Does the parameter γ change the value to which θ converges?

Enter your text here.

Bonus Task: Can you find the adjustment law with another error metric, e.g. $|e(t)|$ or $e(t)^4$? Implement in Simulnk and investigate the same scenarios outlined in **Task 1**.

1. Implement your feedback law:

```
open('Task1_Bonus')
```

2. Verify the implementation:

```
gamma = 1;
sim('Task1_Bonus')
```

```
HW3_plot1(logsout)
```

3. Run the simulation with different values of γ:

- **Trial 1:**

```
gamma = 1;
sim('Task1_Bonus')
HW3_plot1(logsout)
```

- **Trial 2:**

```
gamma = 1;
sim('Task1_Bonus')
HW3_plot1(logsout)
```

- **Trial 3**   :

```
gamma = 1;
sim('Task1_Bonus')
HW3_plot1(logsout)
```

- How does the rate of the adaptation change with γ?

Enter your text here.

4. Does the parameter γ change the value to which θ converges?

Enter your text here.

5. Comment on the differences between the current adaptation law and the one studied in **Task 1**.

Enter your text here.

1.2 The Lyapunov rule

A common alternative approach to the MIT rule is the **Lyapunov rule**. In this case, the adaptation law is given by:

$$\frac{d\theta(t)}{dt} = -\gamma e(\theta, t)u_c(t).$$

5

Task 2. Change the Simulink model Task2_Lyapunov to use the Lyapunov rule instead of the MIT-rule. Investigate the same scenarios outlined in **Task 1**. Provide simulation results, comment on differences between the MIT and Lyapunov rules and explain your observations.

1. Implement the Lyapunov rule in the Task2_Lyapunov model using Library_1 by dragging desired blocks/ components.

```
open('Task2_Lyapunov')
```

2. Run the system with $\gamma = 1$, $k = 1$, $k_0 = 2$ to verify the implementation.

```
gamma = 1;
sim('Task2_Lyapunov')
```

Provide plots of the signals involved, i.e., u_c, y, y_m, e, θ. Can you briefly describe what happens during the simulation? Hint: Use the dataset `logsout` to plot the responses.

```
HW3_plot1(logsout)
```

Enter your text here.

3. How does the rate of the adaptation change with γ? Run the simulation with different values. Provide plots and relative explanations corresponding to the tested scenarios.

- **Trial 1:**

```
gamma = 1;
sim('Task2_Lyapunov')
HW3_plot1(logsout)
```

- **Trial 2:**

```
gamma = 1;
sim('Task2_Lyapunov')
HW3_plot1(logsout)
```

- **Trial 3:**

```
gamma = 1;
sim('Task2_Lyapunov')
HW3_plot1(logsout)
```

Enter your text here.

4. Does the parameter γ change the value to which θ converges?

Enter your text here.

5. Comment on differences between the MIT and Lyapunov rules, and the results obtained by using them.

Enter your text here.

2. MRAS for a first order system

Let us now consider the process being described by the model $y(t) = G(p)u(t)$, with $G(p) = \dfrac{b}{p+a}$ for some

unknown $a, b > 0$. We want to obtain a closed-loop system with behavior described by the reference model,

$y_m(t) = G_m(p)u_c(t)$, with $G_m(p) = \dfrac{b_m}{p+a_m}$ for some known parameters $a_m, b_m > 0$. This is again a problem of model

matching. Clearly, now we need to find two parameters, θ_1 and θ_2, to adapt both the gain and the pole location.

For this purpose, we consider the controller to be given by $u(t) = \theta_1 u_c(t) - \theta_2 y(t)$, yielding the closed-loop

system:

$$y(t) = \frac{b\theta_1}{p+a+b\theta_2} u_c(t).$$

If the parameters a and b were known, then choosing

$$\theta_1(t) = \frac{b_m}{b} = \theta_1^0 \quad \text{and} \quad \theta_2(t) = \frac{a_m - a}{b} = \theta_2^0 ,$$

would give $y(t) = y_m(t)$, achieving perfect model following. However, in the case of unknown parameters a, b the

gains $\theta_1(t)$ and $\theta_2(t)$ need to be calculated adaptively just as in the first part of the exercise. To this end, consider

the error $e(t) = y(t) - y_m(t)$ and define the quadratic loss function $J(\theta, t) = \dfrac{1}{2} e(t)^2$. The adaptation law is found by

minimizing $J(\theta, t)$ with respect to the adaptive gains.

Using the chain rule, we obtain: $\dfrac{d\,J(\theta, t)}{dt} = e(\theta, t)\left(\dfrac{\partial e(\theta, t)}{\partial\theta_1}\dfrac{d\theta_1}{dt} + \dfrac{\partial e(\theta, t)}{\partial\theta_2}\dfrac{d\theta_2}{dt}\right)$. To have the loss function

monotonically decreasing with time, we then choose $\alpha_1, \alpha_2 > 0$ and set the adaptation law as follows:

$$\frac{d\theta_1}{dt} = -\alpha_1 e(\theta, t)\frac{\partial e(\theta, t)}{\partial\theta_1} \qquad \frac{d\theta_2}{dt} = -\alpha_2 e(\theta, t)\frac{\partial e(\theta, t)}{\partial\theta_2}$$

With the expression of the closed-loop system, the sensitivity derivatives, i.e., the partial derivatives of the error

needed to compute the parameter time-derivatives are computed as:

$$\frac{\partial e(\theta, t)}{\partial\theta_1(t)} = \frac{b}{p+a+b\theta_2} u_2(t)$$

$$\frac{\partial e(\theta, t)}{\partial\theta_2(t)} = -\frac{b}{p+a+b\theta_2} y(t)$$

These formulations, however, cannot be used directly because they require knowledge of both a and b, which are unknown by the problem definition. One way around this, is to assume that the adaptive gains are close to their optimal values at all times. If this assumption is granted, then it is possible to use approximation in place of the unknown parameters:

$$\theta_2(t) \approx \frac{a_m - a}{b} \Leftrightarrow a + b\theta_2(t) \approx a_m$$

which implies

$$\frac{\partial e(\theta, t)}{\partial \theta_1(t)} \approx \frac{b}{p + a_m} u_c(t) \qquad \frac{\partial e(\theta, t)}{\partial \theta_2(t)} \approx \frac{b}{p + a_m} y(t).$$

2.2. The MIT-rule

Since b is unknown but positive and constant, we simply define the adaptive gains as $\gamma_i = a_i a_m / b$. The complete MRAS is then given by the feedback controller $u(t) = \theta_1 u_c(t) - \theta_2 y(t)$, where the parameters are updated by the adaptation rules obtained when using the parameter approximations in the sensitivity derivatives, yielding:

$$\frac{d\theta_1}{dt} = -\gamma_1 \frac{a_m}{p + a_m} e(t) u_c(t) \qquad \frac{d\theta_2}{dt} = \gamma_2 \frac{a_m}{p + a_m} e(t) y(t).$$

Note that the approximation may be crude if starting far away from the true parameter values θ_i^0.

Task 3. Similar to the previous exercise, we need to build a model of the system in Simulink. The Simulink library Library_2 contains a reference generator, a set of blocks describing first order (SISO) process and reference models, as well as functionalities for plotting.

1. Implement the feedback loop in the Task3_MIT model using Library_2 by dragging desired blocks/ components.

```
open('Task3_MIT')
```

2. Run with $\gamma = 1, a = 1, b = 0.5, a_m = b_m = 2$ to verify the implementation. Plot the signals involved. Can you briefly describe what happens during the simulation?

```
% Parameters defining the model
a      = 1;
b      = 0.5;
am     = 2;
bm     = 2;
gamma  = 1;
% Simulate
sim('Task3_MIT')
% Plot
HW3_plot2(logsout)
```

Enter your text here.

8

3. How does the rate of the adaptation change with γ? Run the simulation with different values. Provide plots and relative explanations corresponding to the tested scenarios.

- **Trial 1:**

```
gamma = 1;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 2:**

```
gamma = 1;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 3:**

```
gamma = 1;
sim('Task3_MIT')
HW3_plot2(logsout)
```

Enter your text here.

4. Do the parameters a and b affect the rate of adaptation?

- **Trial 1:**

```
a      = 1;
b      = 0.5;
gamma = 1;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 2:**

```
a      = 1;
b      = 0.5;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 3:**

```
a     = 1;
b     = 0.5;
sim('Task3_MIT')
HW3_plot2(logsout)
```

Enter your text here.

5. What happens when you alter the reference model? Try out a few simulations and provide results.

- **Trial 1:**

```
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 2:**

```
am    = 2;
bm    = 2;
sim('Task3_MIT')
HW3_plot2(logsout)
```

- **Trial 3:**

```
am    = 2;
bm    = 2;
sim('Task3_MIT')
HW3_plot2(logsout)
```

Enter your text here.

6. Examine the parameter plane plot, with θ_2 as a function of θ_1. To which values do the parameters converge? Is this expected from the theory?

```
% Parameters defining the model
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
gamma = 1;
% Simulate
sim('Task3_MIT')
```

```
% Plot
HW3_plot2(logsout)
```

Enter your text here.

2.3 The Lyapunov Rule

Let us now use conventional Lyapunov theory. To this end, we write the error dynamics:

$$\frac{d\,e(t)}{dt} = -a_m e(t) - (b\theta_2(t) + a - a_m)y(t) + (b\theta_1 - b_m)u_c(t).$$

The objective is to construct a parameter adjustment mechanism such that the error $y(t) - y_m(t)$ is driven to zero. This is achieved when the adaptive parameters approach the values which can guarantee perfect model following. Introduce the following Lyapunov function, dropping time dependance for notational convenience:

$$V(e,\theta) = \frac{1}{2}\left(e^2 + \frac{1}{b\gamma}(b\theta_2(t) + a - a_m)^2 + \frac{1}{b\gamma}\left(b\theta_1 - b_m\right)^2\right)$$

Applying the chain rule, the time derivative of the Lyapunov function is obtained:

$$\frac{d\,V}{dt} = -\alpha_m e^2 + \frac{1}{\gamma}(b\theta_2(t) + \alpha - \alpha_m)\left(\frac{d\theta_2}{dt} - \gamma y e\right) + \frac{1}{\gamma}(b\theta_1 - b_m)\left(\frac{d\theta_1}{dt} + \gamma u_c e\right)$$

This function is zero when $e(t) = 0$ and the parameters assuming the correct value for model for perfect model following. The time derivative is:

$$\frac{d\,V}{dt} = -a_m e^2 + \frac{1}{\gamma}(b\,\theta_2 + a - a_m) \cdot \left(\frac{d\,\theta_2}{dt} - \gamma\,y\,e\right) + \frac{1}{\gamma}(b\theta_1 - b_m) \cdot \left(\frac{d\theta_1}{dt} + \gamma\,u_c\,e\right)$$

If the parameters are updated as

$$\frac{d\theta_1(t)}{dt} = -\gamma u_c(t)e(t) \qquad \frac{d\theta_2(t)}{dt} = \gamma y(t)e(t)$$

we get a derivative of the Lyapunov function which is negative semidefinite, i.e., $\dfrac{d\,V}{dt} = -a_m e(t)^2 \le 0$, implying that $e(t) \to 0$.

Task 4. Similarly to the previous exercise, we investigate the system in Simulink.

1. Implement the feedback loop in the Task4_Lyapunov model using Library_2 by dragging desired blocks/components.

```
open('Task4_Lyapunov')
```

2. Run with $\gamma = 1, a = 1, b = 0.5, a_m = b_m = 2$ to verify the implementation. Plot the signals involved. Can you briefly describe what happens during the simulation?

```
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

Enter your text here.

3. How does this feedback law based on Lyapunov theory differ from that of the MIT-rule? Explain. Try out a few simulations and provide results.

- **Trial 1:**

```
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
gamma = 1;
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

- **Trial 2:**

```
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
gamma = 1;
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

- **Trial 3:**

```
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
gamma = 1;
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

- **Trial 4:**

```
a     = 1;
b     = 0.5;
am    = 2;
bm    = 2;
gamma = 1;
sim('Task4_Lyapunov')
```

```
HW3_plot2(logsout)
```

- **Trial 5:**

```
a       = 1;
b       = 0.5;
am      = 2;
bm      = 2;
gamma = 1;
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

- **Trial 6:**

```
a       = 1;
b       = 0.5;
am      = 2;
bm      = 2;
gamma = 1;
sim('Task4_Lyapunov')
HW3_plot2(logsout)
```

Enter your text here.

4. Which would you implement in practice? Which guarantees stability?

Enter your text here.

3. Quadrotor Control Using Model Reference Adaptive Control

In the previous examples, we learned about MRAS to closely match the system and the reference model outputs. In this example we will learn how to control a quadrotor with unknown dynamics and external disturbances using model reference adaptive control (MRAC) to track an ideal reference model.The six degree-of-freedom (DOF) quadrotor dynamics are modeled in Simulink® and Simscape, the MRAC controller is implemented using the Model Reference Adaptive Control block.

Open the MATLAB example. Run the example, simulate the model, then answer the following question.

```
openExample('slcontrol/QuadrotorControlModelReferenceAdaptiveControlExample')
```

Task 5. Change the initial feedback gain and feedforward gain to produce a better tracking performance. What do you observe?

Enter your text here.