Predictive Control
Laboratory Experiment 2
Direct Adaptive Control Structures

Department of Automatic Control
Lund Institute of Technology

Introduction

The purpose of this laboratory exercise is to design and study different types of direct adaptive controllers. The lab consists of a theoretical part concerning a simple first order process, and an experimental part concerning a second order DC servo. The purpose is give you a better understanding of how the adaptive self-tuning regulators (STR) can be formulated and leveraged to attenuate load disturbances with certain frequency domain characteristics.

In the first part of the lab we consider a simple first order system and derive a PI controller, minimum-variance controller and direct adaptive controller (see Section 1). In second part of the lab a DC-servo is controlled using a standard PID controller, as well as a direct self-tuning regulator. Here we will derive an STR with an integrator to attenuate constant load disturbances (see Section 2). In the final part of the lab, iterative learning controll (ILC) will be studied to improve reference following with PI control (see Section 3).

The laboratory session consists of a set of preparatory exercises which are to be solved before the session, {***1, 2, 3, 4, 7, 8, 11***}. The remaining 6 exercises should be solved during the laboratory session. Some of the preparatory exercises require Simulink files which can be retreived from the course homepage `http://www.control.lth.se/course/FRTN15/`.

Helpful Matlab Commands

help Matlab help, try for example *help control*

tf Create transfer function model.

bode Plot Bode diagram.

margin Plot Bode diagram with gain- and phase margins.

c2d Convert continuous-time system to discrete-time.

1. Disturbances and Minimum-variance Control

The purpose of this section is to illustrate some properties of a direct adaptive controller on a control problem with time-varying high-frequency disturbances. The difficulty is that the frequency of the disturbance can be quite close to the the bandwidth of the controller. With a traditional PI controller it is then difficult to obtain a sufficiently high gain at the frequency of the disturbance. However with a model of the disturbance it is possible to design a controller that is tuned to the disturbance. Such a controller can have a very high gain at the disturbance frequency.

1.1 The Process

To illustrate the power of adaptive control in light of changing disturbance patterns, we will investigate a simple first order system

$$H(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})} = \frac{bq^{-1}}{1 - aq^{-1}} \tag{1}$$

with some disturbance $v(t)$, such that the finite difference equation becomes

$$y(t+h) = ay(t) + b(u(t) + v(t)) \tag{2}$$

The process dynamics is thus very simple - an integrator with known gain. The disturbance is modelled narrow band noise, where $A_d(q^{-1})v(t) = e(t)$, with

$$A_d(q^{-1}) = 1 - a_d q + q^{-2}, \tag{3}$$

such that the finite difference equation takes the form

$$v(t) - a_d v(t-h) + v(t-2h) = e(t). \tag{4}$$

with $a_d = 2\cos(\omega h)$ (see Figure 1). This generates coloured noise with a sinusoidal disturbance in the time-domain located at the frequency ω (see Figure 2). Nominal parameter values are $a = b = h = 1$ and $\omega = 10^{-1}$, corresponding to the red magnitude plot in the bode diagram.

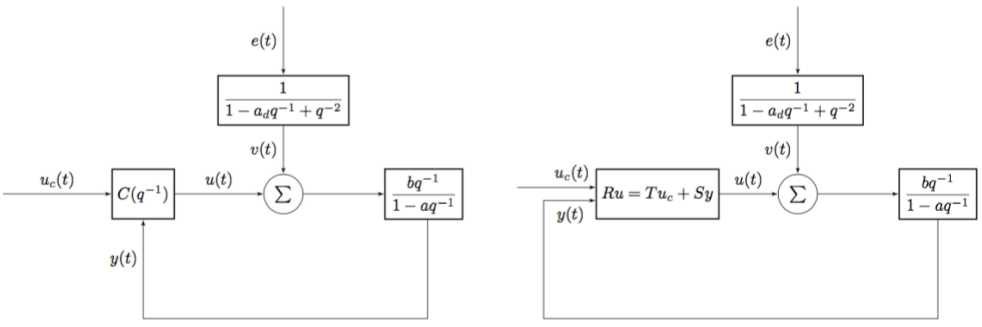

Figure 1 Flow charts for the PI and minimum variance controllers (left) and the direct adaptive minimum variance controller (right).

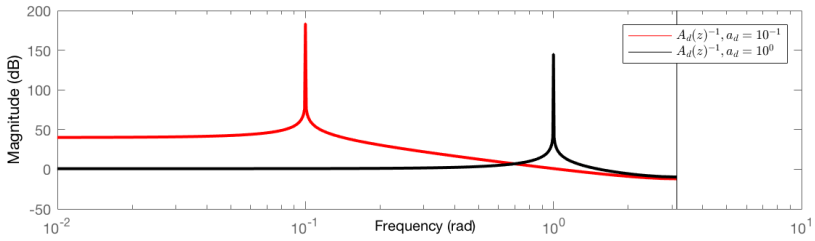

Figure 2 Noise model magnitude at $\omega = 10^{-1}$ (red) and $\omega = 1$ (black).

1.2 Controller Design

We will compare three different controllers for the process, a proportional-integral controller (PI), a minimum variance controller (MV) and finally an adaptive minimum variance controller (DAMV). The PI controller does not have enough complexity to sufficiently reduce the effects of the disturbance, while the MV-controller incorporates the disturbance model to achieve much better performance. The DAMV-controller is a direct adaptive controller based on the minimum variance controller, which will be shown to yield good performance and the capability of adapting to attenuate disturbances where $\omega(t)$ is time-varying.

In order to evaluate the controllers qualitatively, we define a metric in which to compare them. Let $t \in [0, T]$ denote time during which the system is simulated. Furthermore, and assume that the frequency of the disturbance is constant at some frequency $\omega = \omega^*$ over the course of a simulation. Let $u_c(t)$ be the controller reference, and $y(t)$ denote the system response. Then we define

$$E\Big|_{\omega=\omega^*} = \int_0^T |u_c(t) - y(t)|dt \approx \sum_{k=1}^{T} h|u_c(hk) - y(hk)|,$$

as the absolute error between reference and response, taken over the entire simulation time.

Design of PI controller A PI-controller is first designed. This controller has a simple structure.

$$(q-1)u(t) = -(s_0 q + s_1)y(t)$$

With the process model

$$(q-a)y(t) = bu(t)$$

the pole placement design equation becomes

$$(q-a)(q-1) + b(s_0 q + s_1) = q(q-a_m)$$

where the observer pole is placed in the origin. We find

$$\begin{aligned} q^1 &: \quad -1 - a + bs_0 = -a_m \\ q^0 &: \quad a + bs_1 = 0 \end{aligned}$$

This gives

$$\begin{aligned} s_0 &= (1 + a - a_m)/b \\ s_1 &= -a/b \end{aligned}$$

and the controller becomes

$$u(t) = u(t-1) - s_0 y(t) - s_1 y(t-1)$$

Exercise 1 (preparation) Implement the PI controller in the Simulink model `/exercise1/pi_sim.mdl`. How does it perform for $\omega = 0.1$? What about if if the frequency of the disturbance is increased to $\omega = 1$? Write down the resulting error metric E in both cases, and plot the bode diagram of the transfer function from $v(t)$ to $y(t)$ and comment on the diagram.

Design of Minimum Variance Controller To account for the disturbance in a better way we need a more complex controller which incorporates the disturbance model in the design procedure. We will therefore design a minimum variance controller for the problem.

The noise model is given by

$$A_d(q)v(t) = v(t+2) - a_d v(t+1) + v(t) = e(t+2)$$

where $a_d = 2\cos(\omega h)$ and the process model is

$$A(q)y(t) = y(t+1) - ay(t) = b(u(t) + v(t))$$

so that

$$AA_d y(t) = bA_d u(t) + be(t+2)$$

We write this as

$$A'y = B'u + C'\bar{e}$$

with

$$\begin{aligned} A' &= AA_d = (q-a)(q^2 - a_d q + 1) \\ B' &= BA_d = b(q^2 - a_d q + 1) \\ C' &= q^3 \\ \bar{e}(t) &= be(t-1) \end{aligned}$$

The Diophantine equation we have to solve is

$$A'R + B'S = q^3 B'$$

with the constraint that B' is a factor of R. To get a monic $R(q)$ we divide the right hand side by b. This will give us a solution, $R(q)$ and $S(q)$, that is a factor b smaller which does not affect the controller $-S(q)/R(q)$. Hence we solve

$$A'R + B'S = q^3 B'/b$$

with

$$R = R'B'/b$$

Cancellation of B'/b gives

$$(q-a)(q^2-a_dq+1)R'+bS=q^3$$

The minimum degree solution has $\deg R'=0$ and $\deg S=2$. We get $R'=1$ and the coefficients in $S(q)$ can be computed from

$$\begin{array}{ll} q^2: & -a-a_d+bs_0=0 \\ q^1: & 1+a_da+bs_1=0 \\ q^0: & -a+bs_2=0 \end{array}$$

This gives

$$\begin{array}{rcl} R(q) & = & q^2+r_1q+r_2=R'(q)B'(q)/b=q^2-a_dq+1 \\ S(q) & = & s_0q^2+s_1q+s_2=((a+a_d)q^2+(-1-a_da)q+a)/b \end{array}$$

and the control law becomes

$$u(t)=a_du(t-1)-u(t-2)+\frac{1}{b}\Big(-(a+a_d)y(t)+(1+a_da)y(t-1)-ay(t-2)\Big)$$

Exercise 2 (preparation) Implement the MV controller in the Simulink model `/exercise1/mv_sim.mdl` tuned to attenuate disturbances at $\omega=0.1$. How does it perform for $\omega=0.1$? What about if if the frequency of the disturbance is increased to $\omega=1$? Write down the resulting error metric E in both cases, and plot the bode diagram of the transfer function from $v(t)$ to $y(t)$ and comment on the diagram.

Design of Direct Adaptive Controller The minimum variance controller was designed using the disturbance model and is therefore tuned to that model. If the characteristics of the disturbance change, the controller might perform badly. To cope with changing disturbances we will therefore design an adaptive version of the controller. If we let the design equation (with A_d canceled) operate on the output $y(t)$ we get

$$\big(A(q)R(q)+B(q)S(q)\big)y(t)=B(q)q^3/by(t)$$

Using the input-output relation $A(q)y(t)=B(q)u(t)$ we get

$$B(q)(R(q)u(t)+S(q)y(t))=B(q)q^3/by(t)$$

and we can cancel $B(q)$. This results in

$$R(q)u(t)+S(q)y(t)=q^3/by(t)$$

or

$$b((q^2+r_1q+r_2)u(t)+(s_0q^2+s_1q+s_2)y(t))=q^3y(t)$$

The relation above holds for the correct controller and we can therefore use it to directly estimate the unknown controller parameters. The parameter b is assumed to be known so we can rewrite the relation as

$$y(t)/b-u(t-1)=\phi^T(t-1)\theta$$

where the left hand side is known and

$$\phi(t-1) = \begin{pmatrix} u(t-2) & u(t-3) & y(t-1) & y(t-2) & y(t-3) \end{pmatrix}^T$$

and

$$\theta = \begin{pmatrix} r_1 & r_2 & s_0 & s_1 & s_2 \end{pmatrix}^T$$

The parameter vector can now be updated using an RLS algorithm. The estimates are then used in the control law derived previously for the MV controller.

Exercise 3 (preparation) The direct adaptive MV controller has already been implemented in the Simulink model `/exercise1/da_sim.mdl`, and initialized to attenuate disturbances at $\omega = 0.1$. How does it perform for $\omega = 0.1$? What about if if the frequency of the disturbance is increased to $\omega = 1$? Write down the resulting error metric E in both cases. What if you pause the simulation at $t = T/2 \approx 250$ and change the disturbance, does the controller adapt the MV-controller? How is the rate of adaption and noise sensitivity affected by the forgetting factor λ?

2. Control of the DC-servo

Next, we will design and evaluate controllers for angular position control of a simple DC-servo with a flywheel. The system is modelled using Netwon's second law as a second order process

$$J\ddot{y}(t) = -d\dot{y}(t) + ku(t) + v(t), \tag{5}$$

where

$y\ [rad]$	Angular position,
$v\ [rad]$	Angular disturbance,
$u\ [A]$	Control signal current,
$J\ [kg \cdot m^2]$	Moment of inertia,
$d\ [N \cdot m \cdot s]$	Coefficient of viscous friction
$k\ [N \cdot m/A]$	Motor torque constant.

Inserting values found in a simple system identification experiment,

$$G(s) \triangleq \frac{k/J}{s(s + d/J)} = \frac{11.2}{s(s + 0.12)}. \tag{6}$$

The corresponding pulse transfer operator is then

$$H(q) = \frac{B(q)}{A(q)} = \frac{b_1 q + b_2}{q^2 + a_1 q + a_2}. \tag{7}$$

The closed-loop characteristic polynomials $A_m(q)$ and $A_o(q)$ are conveniently defined as the discrete-time counterparts to the continuous time polynomials

$$A_{mc}(s) = s^2 + 2\zeta_m\omega_m s + \omega_m^2, \quad (8)$$

$$A_{oc}(s) = s + \omega_0. \quad (9)$$

For future reference, the sampling interval is set to $T_s = 0.1$. The nominal controller design (conforming with typical industry standards) is a fixed relative damping of $\zeta_m = 0.7$, and a natural frequency of $\omega_m = 5$. Furthermore, we will use an observer defined by $\omega_o = 7$. The system with a self-tuning regulator is shown in Figure 3.

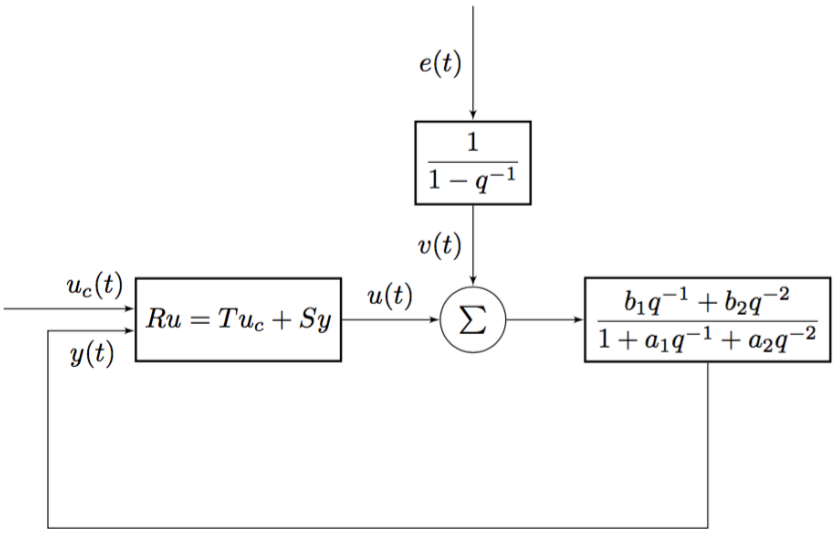

Figure 3 The closed loop system with the DC servo, step disturbance model and a self-tuning regulator.

2.1 Fixed PID control

Instead of deriving of the adaptive controller directly, we start simple by controlling the angular position using a fixed PID controller. The goal is to get to know the DC servo and test the robustness of your tuning when changing the process parameters. The well-known feedback law is defined with control error $e(t) = u_c(t) - y(t)$, whereby control signal $u(t)$ is computed by

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int e(t)dt + T_d\frac{de(t)}{dt}\right).$$

Exercise 4 (Preparation) Discretize a PID controller using finite difference approximation (use backward difference for the derivative and forward difference for the integral part) and determine the resulting pulse transfer function. What are the orders of the numerator and denominator?

Exercise 5 Tune a PID controller with an acceptable control performance for the DC process by opening the `pidDCsim.slx`. Which values of parameters K, T_i, T_d seem suitable? Test the robustness of your tuning by changing the process parameters during the simulation.

Exercise 6 When you have found a controller that works well in simulations, you can try it on the real process. Open the `pidDCreal.slx` model, enter your parameters from the previous exercise and vary the process parameters (e.g., increase the moment of inertia of the plant, change the damping). How does the control performance change? Would you opt to re-tune the PID controller if the process parameters change?

2.2 Direct Adaptive Control

With a PID controller one can make an arbitrary pole placement for a system of order no greater than 2. In this section, this fact is exploited to derive a method for online adaptation of the parameters in a self-tuning PID (STUPID) controller - a direct adaptive controller with integral action.

Exercise 7 (Preparation) Similar to the MV controller, derive the linear regression model for the Direct Self-Tuning PID with integral action and zero cancellation, i.e. a controller incorporating the noise model $A_d = (q - 1)$. In addition, find the explicit equations for computing the control signal u. *Before you start with this assignment, it is strongly recommended to read Chapter 3 in Adaptive Control - Second Edition, paying special attention to pages 121-128. It may also be helpful to revise chapter 6 in the course book.*

Exercise 8 (Preparation) Since the controller output is limited by a saturation nonlinearity, the integrator could give rise to the wind-up phenomena. Introduce an anti-windup observer polynomial A_{ow}, specify how it enters the computation of the control signal u, and also give the order of the anti-windup polynomial.

Exercise 9 Simulate the Self-Tuning PID and find a suitable parameter tuning. How does the control signal behave? Can you explain? Change the process parameters during simulation and observe the behaviour of your controller.

Exercise 10 Test your best controller setup on the real process. Try to vary the process parameters (e.g. vary the moment of inertia of the plant). How does the controller behave? How much variation in the parameters can you handle? Investigate the effect of the forgetting factor in your controller.

3. Iterative Learning Control

In this part of the laboratory session we will examine the method of Iterative Learning Control (ILC) for improving the reference following when performing repeated tasks. Errors in the reference following can have many causes; unmodelled dynamics, poorly tuned control parameters, different kinds of disturbances, measurement noise, friction etc. If the system response is highly repeatable, in the sense that a certain reference signal sequence will generate roughly the same response whenever it is repeated, then ILC can be applied to decrease the control error. However, if stochastic disturbances dominate or if the dynamics vary from one run to another, other methods than ILC should be considered.

The ILC method is intuitive and simple. We apply the desired reference signal sequence $\{\mathbf{y_d}\}$ to the system and record the whole error sequence

$\{\mathbf{e_k}\} = \{\mathbf{y_d}\} - \{\mathbf{y_k}\}$. Based on the error sequence, we then modify the control signal $\{\mathbf{u_{k+1}}\}$ to improve the response for the next iteration. These two steps are repeated this until the deviation from the reference is acceptable.

The ILC-correction $\{\mathbf{u_k}\}$ is pre-determined and applied in "open-loop" during one run, but is updated from feedback information between the runs. In the lectures, we have seen different kinds of update laws for Iterative Learning Control in the lectures. Here we will try with a version of the so called *heuristic approach*.

$$\{\mathbf{u}_{k+1}\} = Q_d(\{\mathbf{u}_k\} + L_d\{\mathbf{e}_k\})$$

where $\{\mathbf{e}_k\}$ is the sequence of errors from previous run, $\{\mathbf{u}_{k+1}\}$ is the sequence of new ILC-corrections, Q_d and L_d are linear discrete time filters. As the filtering is made "off-line" between two runs, when we have access to the whole sequences of data and may choose Q_d and L_d to be non-causal filters.

Exercise 11 (Preparatory) Study the Matlab script `ILC_setup.m` and make sure you know the difference between *causal* and *acausal* filtering. Try to filter a sequence $[1:10]$ with the filter $G_1 = z^3$ using `noncausalfilter.m` and with the filter $G_2 = 1/z^3$ using `filter.m`, respectively.

Exercise 12 Open the model `/exercise3/ILC_pidDCsim.mdl` and run the ILC iterations by pressing the yellow box (which essentially calls the `run_ILC_iteration.m` script). Modify the filters $Q_d = 1/(s/p+1)$ and $L_d = \alpha z^\beta$ in the Matlab script `ILC_setup.m`. How does β affect the ILC performance, and what may be a reasonable value? How does α affect the rate of convergence, and what may be a reasonable value? How does the pole p affect the control signal sequence, any how should it be chosen?

Exercise 13 Once you have found a suitable filter, open and run the ILC on the real process. How will the system perform? Can you mention any reason for why the method starts to degenerate after a number of iterations?

Document history:
Created: September 2004 by Stefan Solyom and Anders Robertsson
Additional material by: K. J. Åström and H. Olsson
Additional material and heavy revision: Marcus Greiff