哈爾濱工業大學深圳研究生院
Harbin Institute of Technology Shenzhen Graduate School

讲　义

Course Handouts

开课学院
School　机电工程与自动化

课程名称
Course Title　系统辨识

开课学期
Semester　秋季学期

授课教师
Lecturer　吴爱国

职称
Title　教授

联系电话
Telephone　26033899

System Identification

Dr. Ai-Guo Wu
Harbin Institute of Technology Shenzhen Graduate School

Contents

List of Notations **vii**

1 Introduction **1**
1.1 Dynamics systems and Model 1
1.2 Model Types 1
1.3 Mechanism modelling 2
1.3.1 Electrical systems 2
1.3.2 Mechanical systems 4
1.4 White and Black-box 5
1.5 Definition of system identification 6
1.6 Linear-parameter model and nonlinear-parameter model 9

2 Identification models and least squares estimation **11**
2.1 Identification models 11
2.1.1 Time series model 12
2.1.2 Equation error type models 13
2.1.3 Output error type models 15
2.2 Least squares principle 15
2.2.1 One-dimensional case 15
2.2.2 Two-dimensional case 16
2.2.3 Matrix calculus 19
2.3 Least square estimation 29
2.3.1 Single-output case 29
2.3.2 Multi-output case 32
2.4 Orthonormal approaches for least square estimation 33
2.4.1 QR factorization approach 33
2.4.2 Gram-Schmidt approach 35
2.5 Statistical properties for least squares estimation 37

3 Recursive least squares identification **41**
3.1 Recursive least squares method 41
3.2 Forgetting factor recursive least squares algorithm 46
3.2.1 Data saturation 46
3.2.2 Forgetting factor recursive least squares algorithm 47

3.3 Fixed memory identification 50
3.3.1 The first type of recursive algorithms 51
3.3.2 The second type of recursive algorithms 54
3.4 Fixed memory identification with a forgetting factor 55
3.4.1 The first type of algorithms 57
3.4.2 The second type of algorithms 59

4 Weighted recursive least squares algorithms 61
4.1 Weighted least squares estimation 61
4.1.1 Normalized gain version 65
4.2 Multivariable case 66
4.3 Weighted fixed memory recursive least squares algorithm 66
4.3.1 First type weighted fixed memory recursive least squares algorithm 68
4.3.2 The second type weighted fixed memory recursive least squares algorithm 71

5 Recursive extended and generalized least squares identification 75
5.1 Recursive extended least squares identification 75
5.2 Recursive generalized least squares identification 78
5.2.1 Recursive generalized least squares identification 78
5.2.2 Filtering based Recursive generalized least squares algorithm 80
5.3 Recursive Generalized Extended Least Squares algorithm 83
5.3.1 Recursive generalized extended least squares algorithm . . 83
5.3.2 Filtering based Recursive generalized least squares algorithm 85
5.3.3 Multi-stage Least Squares Identification 89
5.4 Instrumental variable based least squares identification 93
5.5 Bias compensation based recursive least squares identification . . 95
5.5.1 Bias compensation based recursive least squares identification 95

6 Auxiliary model based identification 101
6.1 Output error model 101
6.2 Auxiliary model based extended least squares identification . . . 103
6.3 Output error autoregressive model 107
6.4 Box-Jenkins model 111

7 Stochastic gradient based identification algorithm 115
7.1 The projection identification algorithm 115
7.2 Stochastic gradient algorithm 117

Preface

List of Notations

Chapter 1

Introduction

1.1 Dynamics systems and Model

A system is an object in which variable of different kinds interact and produce observable signals. The observable signals that are of interest to us are usually called outputs. The system is also affected by external stimuli. External signals that can be manipulated by the observer are called inputs.

A system is dynamic, which means that the current output value depends not only on the current external stimuli but also on their earlier values.

1.2 Model Types

The purpose of modelling is to understand the relationships among various physical variables in a system, in particular that between the input and the output of the system.

An assumed relationship among observed signals is called as a model of the system.

When we interact with a system, we need some concept of how its variables relate to each other. With a broad definition, we shall call such an assumed relationship among observed signals a model of the system. Clearly, models may come in various shapes and be phrased with varying degrees of mathematical formalism. The intended use will determine the degree of sophistication that is required to make the model purposeful.

No doubt, in daily life many systems are dealt with using mental models, which do not involve any mathematical formalization at all. To drive a car, for example,requires the knowledge that turning the steering wheel to the left includes a left turn,together with subtle information built up in the muscle memory. The importance and degree of sophistication of the latter should of course not be underestimated.

For certain systems it is appropriate to describe their properties using numerical tables and/or plots. We shall call such descriptions graphical models.

Linear systems, for example, can be uniquely described by their impulse or step responses or by their frequency functions. Graphical representation of these are widely used for various design purposes. The nonlinear characteristics of, say, a valve are also well suited to be described by a graphical model.

For more advanced applications, it may be necessary to use models that describe the relationships among the system variables in terms of mathematical expressions like difference or differential equations. We shall call such models mathematical (or analytical) models. Mathematical models may be further characterized by a number of adjectives (time continuous or time discrete, lumped or distributed, deterministic or stochastic, linear or nonlinear, etc.) signifying the type of difference or differential equation used. The use of mathematical models is inherent in all fields of engineering and physics. In fact, a major part of the engineering field deals with how to make good designs based on mathematical models. They are also instrumental for simulation and forecasting (prediction), which is extensively used in all fields, including nontechnical areas like economy,ecology abd biology.

The model used in a computer simulation of a system is a program. For complex systems, this program may be built up by many interconnected subroutines and lookup tables, and it may not be feasible to summarize it analytically as a mathematical model. We use the term software model for sunch computerized descriptions. They have come to play an increasingly important role in decision making for complicated systems.

1.3 Mechanism modelling

The purpose of modelling is to understand the relationships among various physical variables in a system, in particular that between the input and the output of the system. Such a relationship is called a mathematical model. There are several ways to obtain a mathematical model. The first way is to write down a complete set of equations relating the different variables in the system based on physical laws. The equations are usually differential equations. This method is called the first principle modelling. The second method is to find out the relationship between input and output, as well as other signals, from a set of experimental data. This method is called system identification. In most cases, actually, modelling is done by combining the above two methods. It is quite often that the structure of the model is determined from physical principles and the parameters are obtained from experiments. Modelling is in general time consuming, problem dependent, theoretically underdeveloped, but is a very important step towards controller design.

1.3.1 Electrical systems

First let us consider a simple RLC circuit shown in Figure 1.3.1, where the diamond symbol labeled with gv_1 means a voltage controlled current source whose current is proportional to the voltage across the capacitor $C1$.

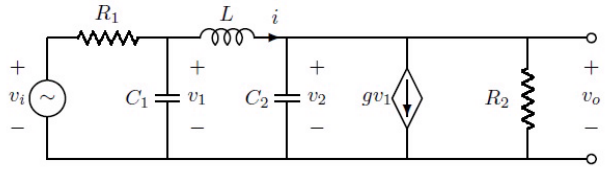

Figure 1.3.1: A simple RLC circuit

A systematic way to write down a differential equation model for an RLC circuit is as follows:

Algorithm 1.3.1 Deviring a mathematical model for a circuit

Step 1 Choose capacitor voltages and inductor currents as state variables.

Step 2 Write a Kirchhoff's Current Law (KCL) equation relating the state variables and the input for each node connecting to one and only one capacitor.

Step 3 Write a Kirchhoff's Voltage Law (KVL) equation relating the state variables and the input for each loop containing one and only one inductor.

Step 4 Relate the output with the state variables and the input.

Going back to the RLC circuit shown in 1.3.1, we choose $v_1(t)$; $v_2(t)$; $i(t)$ as state variables. Then applying KCL to the node connecting to $R1$, $C1$, and L gives

$$C_1 \frac{\mathrm{d}\, v_1(t)}{\mathrm{d}\, t} = \frac{v_i(t) - v_1(t)}{R} - i(t). \tag{1.3.1}$$

Applying KCL to the node connecting to L, C_2, R_2, and the controlled source gives

$$C_2 \frac{\mathrm{d}\, v_2(t)}{\mathrm{d}\, t} = i(t) - gv_1 - \frac{v_2(t)}{R_2}. \tag{1.3.2}$$

Applying KVL to the loop containing C_1, L, C_2 gives

$$L \frac{\mathrm{d}\, i(t)}{\mathrm{d}\, t} = v_1(t) - v_2(t). \tag{1.3.3}$$

Finally, the output $v_o(t)$ can be related to the state variables and the input as follows

$$v_o(t) = v_2(t). \tag{1.3.4}$$

The differential equations (1.3.1), (1.3.2), and (1.3.3), as well as the algebraic equation (1.3.4), give a model of the RLC circuit.

The model can also be written in the following matrix form

$$\begin{bmatrix} \dot{v}_1(t) \\ \dot{v}_2(t) \\ \dot{i}(t) \end{bmatrix} = \begin{bmatrix} -\frac{1}{R_1 C_1} & 0 & -\frac{1}{C_1} \\ -\frac{g}{C_2} & -\frac{1}{R_2 C_2} & \frac{1}{C_2} \\ \frac{1}{L} & -\frac{1}{L} & 0 \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \\ i(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{R_1 C_1} \\ 0 \\ 0 \end{bmatrix} v_i(t), \tag{1.3.5}$$

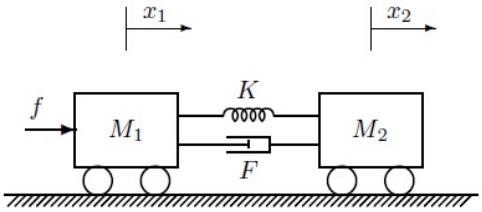

Figure 1.3.2: Two cart system

$$v_o(t) = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \\ i(t) \end{bmatrix}. \tag{1.3.6}$$

1.3.2 Mechanical systems

Consider two carts moving on a table, as shown in Figure 1.3.2. The carts are assumed to have masses M_1 and M_2 respectively, connected by a spring. A force $f(t)$ is applied to cart M_1 and we wish to control the position of cart M_2. An ideal spring generates a force which is proportional to the relative displacement of its two ends. The spring in this system is assumed to be non-ideal and its effect is equivalent to the sum of an ideal spring, depicted by the symbol in the upper part of the connection, and a friction device, depicted by the symbol in the lower part of the connection, which generates a force proportional to the relative velocity of its two ends. Assume that the reference points for the positions of the two carts are chosen so that the spring is at rest when $x_1(t) = x_2(t)$.

To model a mechanical system like this, we usually choose the positions of all independent rigid bodies as the internal variables and write down the equations according to the Newton's second law. Applying the Newton's law to the first body, we obtain

$$M_1 \frac{\mathrm{d}^2 x_1(t)}{\mathrm{d}\, t^2} = f(t) - K\left(x_1(t) - x_2\left(t\right)\right) - F\left(\frac{\mathrm{d}\, x_1(t)}{\mathrm{d}\, t} - \frac{\mathrm{d}\, x_2\left(t\right)}{\mathrm{d}\, t}\right).$$

Applying the Newton's law to the second cart, we obtain

$$M_2 \frac{\mathrm{d}^2 x_2(t)}{\mathrm{d}\, t^2} = K\left(x_1(t) - x_2\left(t\right)\right) + F\left(\frac{\mathrm{d}\, x_1(t)}{\mathrm{d}\, t} - \frac{\mathrm{d}\, x_2\left(t\right)}{\mathrm{d}\, t}\right).$$

Finally we identify the output and relate it to the internal variables and the input. In this case, the output is simply $x_2(t)$. We can also reorganize the equations into the following matrix form

$$\begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \begin{bmatrix} \ddot{x}_1(t) \\ \ddot{x}_2(t) \end{bmatrix} + \begin{bmatrix} F & -F \\ -F & F \end{bmatrix} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} + \begin{bmatrix} K & -K \\ -K & K \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} f(t). \tag{1.3.7}$$

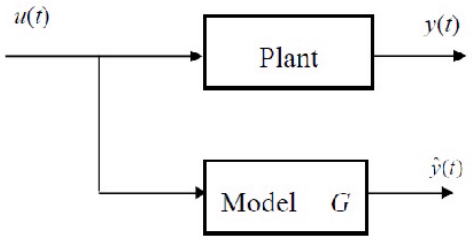

Figure 1.5.1:

Models of the form (1.3.7) are very typical for mechanical systems. Such a model is called a second-order model.

1.4 White and Black-box

Mechanism modelling is also called white box modeling. One could build a so-called white-box model based on first principles, e.g. a model for a physical process from the Newton equations, but in many cases such models will be overly complex and possibly even impossible to obtain in reasonable time due to the complex nature of many systems and processes.

A much more common approach is therefore to start from measurements of the behavior of the system and the external influences (inputs to the system) and try to determine a mathematical relation between them without going into the details of what is actually happening inside the system. This approach is called system identification. Two types of models are common in the field of system identification:

- Grey box model: although the peculiarities of what is going on inside the system are not entirely known, a certain model based on both insight into the system and experimental data is constructed. This model does however still have a number of unknown free parameters which can be estimated using system identification. Grey box modeling is also known as semi-physical modeling.

- Black box model: No prior model is available. The mathematical model can be only determined according to the Most system identification algorithms are of this type.

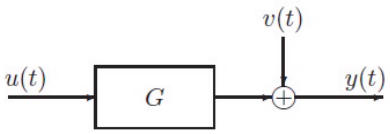

Figure 1.5.2: The system to be identified

1.5 Definition of system identification

System identification is to determine a system to which the system under test is equivalent on the basis of observation of input and output. If the measured data contains the system's input and output, system identification is to fit a mathematical model from the observered data such that the input-output property of the model is similar to the one of the real system. This can be characterized by the following **error criterion function**:

$$J_1 = \int_0^\infty [y(t) - \hat{y}(t)]^2 \,\mathrm{d}\, t = \int_0^\infty [y(t) - Gu(t)]^2 \,\mathrm{d}\, t, \text{ for continous-time case.}$$

$$J_1 = \sum_{t=0}^{\infty} [y(t) - \hat{y}(t)]^2 = \sum_{t=0}^{\infty} [y(t) - Gu(t)]^2, \text{ for discrete-time case.}$$

In some literature, a error criterion function is also called a **cost function**. If $J_1 = 0$, then the output of the model is equal to the output of the real system for any t. That is, $\hat{y}(t) = Gu(t) = y(t)$. This is only an ideal case. For a practical system, this equality does not hold since the system is affected by disturbance, and thus there is error in the measured data. In this case, we shoud find a model G making the error criterion function J_1 as small as possible. In general, the error $y(t) - Gu(t)$ is random, and thus is called noise. We use $v(t)$ to denote this error. Then,

$$v(t) = y(t) - Gu(t)$$

or

$$y(t) = Gu(t) + v(t).$$

Such models are called **output error type** models, which can be depicted in 1.5.2. **Identification can be defined as the determination of a mathematical model from the observed input and output data by minimizing some error criterion function.** In this course, we focus on the discrete-time systems, so as an example we consider a specific estimation problem that contains most of the central issues in this course.

The input and output at time t of the considered system are denoted by $u(t)$ and $y(t)$, respectively. Perhaps the most basic relationship between the input and output is the **linear difference equation**:

$$y(t)+a_1y(t-1)+\cdots+a_ny(t-n) = b_1u(t-1)+\cdots+b_mu(t-m)+v(t). \quad (1.5.1)$$

We have chosen to represent the system in discrete time, primarily since observered data are always collected by smpling. It is thus more straightforward to relate observed data to discrete-time modes. In (1.5.1), it is assumed that the sampling interval to be one time unit. This is not essentical, but makes notation easier.

A pragmatic and useful way to see (1.5.1) is to view it as a way of determining the next output value given previous observations:

$$y(t) = -a_1 y(t-1) - \cdots - a_n y(t-n) + b_1 u(t-1) + \cdots + b_m u(t-m) + v(t). \quad (1.5.2)$$

For more compact notation, we introduce the vectors:

$$\theta = \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_n & b_1 & b_2 & \cdots & b_m \end{array} \right]^{\mathrm{T}}, \quad (1.5.3)$$

$$\varphi(t) = \left[\begin{array}{cccccccc} -y(t-1) & \cdots & -y(t-n) & u(t-1) & \cdots & u(t-m) \end{array} \right]^{\mathrm{T}}. \quad (1.5.4)$$

With these, (1.5.2) can be rewritten as

$$y(t) = \varphi^{\mathrm{T}}(t)\theta + v(t).$$

To emphasize that the calculation of $y(t)$ from the past data (1.5.4) indeed depends on the parameters in θ, we shall call $\varphi^{\mathrm{T}}(t)\theta$ calculated value and write

$$\hat{y}\,(t|\theta) = \varphi^{\mathrm{T}}(t)\theta. \quad (1.5.5)$$

Now suppose for a given system that we do not know the values of the parameters in θ, but that we have recorded inputs and outputs over a time interval $1 \leq t \leq N$:

$$Z^N = \{u(1),\, y(1),\, u(2),\, y(2),\, \cdots,\, u(N),\, y(N)\}\,. \quad (1.5.6)$$

An obvious approach is then to select θ in (1.5.1) through (1.5.5) so as to fit the calculated values $\hat{y}\,(t|\theta)$ as well as possible to the measured outputs by the least squares method, that is, by minimizing the following error criterion function

$$J_1 = \sum_{t=1}^{N} \left[y(t) - \hat{y}\,(t|\theta)\right]^2 = \sum_{t=1}^{N} \left[y(t) - \varphi^{\mathrm{T}}(t)\theta\right]^2. \quad (1.5.7)$$

We denote the value of θ that minimizes (1.5.7) by $\hat{\theta}$. By setting the derivation with respect to θ, one has

$$0 = \sum_{t=1}^{N} \varphi(t) \left[y(t) - \varphi^{\mathrm{T}}(t)\theta\right]$$

which gives

$$\sum_{t=1}^{N} \varphi(t) y(t) = \sum_{t=1}^{N} \varphi(t)\varphi^{\mathrm{T}}(t)\theta$$

or

$$\hat{\theta} = \left[\sum_{t=1}^{N} \varphi(t)\varphi^{\mathrm{T}}(t)\right]^{-1} \sum_{t=1}^{N} \varphi(t)y(t).$$

Notice: A typical convention is to take values outside the measured range to zero.

By L. A. Zadeh,
System identification: Determination, on the basis of observation of input and output, of a system within a specified class of systems to which the system under test is equivalent.

By L. Ljung,
The identification procedure is based on three entities: the data, the set of models, and the criterion. Identification, then, is to select that model in the model set that describes the data best, according to the criterion.

The above definition on system identification emphasizes the three entities: dada, the set of models and criterion. The data is the basis of system identification, the criterion is the objective function to be optimized.

Three Basic Entities

As we saw in the previous section,the construction of a model from data involves three basic entities:

1. A data set
2. A set of candidate models
3. A rule by which candidate models can be assessed using the data,like the Least Squares selection rule(1.9).

Let us comment on each of these:

1. The data record. The input-output data are sometimes recorded during a specifically designed identification experiment,where the user may determine which signals to measure and when to measure them and may also choose the input signals. The objective with experiment design is thus to make these choices so that the data become maximally informative,subject to constraints that may be at hand.
2. The set of models or the model structure. A set of candidate models is obtained by specifying within which collection of models we are going to look for a suitable one. This is no doubt the most important and, at the same time, the most difficult choice of the system identification procedure. It is here that a priori knowledge and engineering intuition and insight have to be combined with formal properties of models.Sometimes the model set is obtained after careful modeling. Then a model with some unknown physical parameters is constructed from basic physical laws and other well-established relationships.In other cases standard linear models may be employed, without reference to the physical background. Such a model set,whose parameters are basically viewed as vehicles for adjusting the fit to the data and do not reflect physical considerations

in the systems, is called a black box. Model sets with adjustable parameters with physical interpretation may, accordingly, be called gray boxes.

3. Determining the "best" model in the set, guided by the data. This is the identification method. The assessment of model quality is typically based on how the models perform when they attempt to reproduce the measured data. The basic approaches to this will be dealt with independently of the model structure used.

In fact, the optimization is also very important. When different optimization approach is used, different identification approach can be obtained. Four entities: data, the set of models, criterion, and optimization approaches.

The main task of system identification is to propose new identification approaches, and to analyze the convergence properties of the proposed approach.

1.6 Linear-parameter model and nonlinear-parameter model

If the system output (or, virtual output) can be represented by a function of parameters, then the model is called linear-parameter model, and the corresponding system is called the linear-parameter system. Otherwise, the model is called nonlinear-parameter model. A linear-parameter system may be a nonlinear systems.

Chapter 2

Identification models and least squares estimation

2.1 Identification models

In this book, we introduce the **forward shift operator** z by

$$zu(t) = u(t+1)$$

and the **backward shift operator** z^{-1} by

$$z^{-1}u(t) = u(t-1).$$

A general single-input-single-output (SISO) system can be represented by

$$F(z)y(t) = \frac{B(z)}{A(z)}u(t) + \frac{D(z)}{C(z)}v(t). \tag{2.1.1}$$

In this course, we use $\{u(t)\}$ and $\{y(t)\}$ to denote the input and output series, respectively, and $\{v(t)\}$ is a white noise series with zero mean. $A(z)$, $B(z)$, $C(z)$, $D(z)$ and $F(z)$ are some polynomials with time-invariant coefficients with respect to the backword shift operator z^{-1}. They can be defined as

$$\begin{aligned}
A(z) &:= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}, \\
B(z) &:= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}, \\
C(z) &:= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}, \\
D(z) &:= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}, \\
F(z) &:= 1 + f_1 z^{-1} + f_2 z^{-2} + \cdots + f_{n_f} z^{-n_f}.
\end{aligned}$$

If there is no delay in the system, $B(z)$ can be defined as

$$B(z) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}.$$

The models in the field of system identification can be typically divided into three categories: time series models, equation error type models and output error type models.

2.1.1 Time series model

Time series model includes three basic models: autoregressive (AR) model, moving average (MA) model, and autoregressive moving average (ARMA) model. In addition, the deterministic ARMA model and the autoregressive integtated moving average model can also fall into the time series model. A time series model has two variables, the measured output $y(t)$ and the random white noise $v(t)$.

1. Autogressive (AR) model

$$A(z)\,y(t) = v(t)\,,$$

or

$$y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) = v(t)\,.$$

2 Moving Average (MA) model

$$y(t) = D(z)\,v(t)\,,$$

or

$$y(t) = v(t) + d_1 v(t-1) + d_2 v(t-2) + \cdots + d_{n_d} v(t-n_d)\,.$$

3 AutoRegressive Moving Average (ARMA) model

$$A(z)\,y(t) = D(z)\,v(t)\,,$$

or

$$\begin{aligned} & y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) \\ = \;& v(t) + d_1 v(t-1) + d_2 v(t-2) + \cdots + d_{n_d} v(t-n_d)\,. \end{aligned}$$

4 Deterministic ARMA (DARMA) model

$$A(z)\,y(t) = B(z)\,u(t)\,,$$

or

$$\begin{aligned} & y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) \\ = \; & b_1 u(t-1) + b_2 u(t-2) + \cdots + b_{n_b} u(t-n_b). \end{aligned}$$

The only difference of the DARMA model from the ARMA model is that the variable $u(t)$ is a deterministic signal instead of a random white noise.

5 AutoRegressive Integtated Moving Average (ARIMA) model

$$A(z)\left(1-z^{-1}\right)^d y(t) = D(z)\, v(t),$$

where $d \geq 0$. When $d = 0$, ARIMA=ARMA.

2.1.2 Equation error type models

Probably the simplest input-output relationship is obtained by describing it as a linear difference equation:

$$A(z)\, y(t) = B(z)\, u(t) + w(t) \tag{2.1.2}$$

where $w(t)$ is a white noise, or a colored noise, and $A(z)$ and $B(z)$ are two polynomials with respect to the backward shift operator z^{-1} as shown previously. Since the white/colored noise $w(t)$ here enters as a direct error in the difference equation, the model (2.1.2) is often called an **equation error type model** (structure). The equation error type models have the following form:The equation error type models include the following typtical models.

1. The first special case of (2.1.2) is

$$A(z)\, y(t) = B(z)\, u(t) + v(t) \tag{2.1.3}$$

where $v(t)$ is a white noise. This model is called **equation error model** since the white noise $v(t)$ enters as a direct error in the difference equation. The model (2.1.3) is also called an **ARX** model, where AR refers to the autogressive part $A(z)\, y(t)$ and X to the extra input $B(z)\, u(t)$. In the special case where $n_a = 0$, $y(t)$ is modeled as a finite impulse response (FIR). Such model sets are particularly common in signal-processing applications. The model (2.1.3) is perhaps not the most natural one from a physical point of view: the white noise is assumed to go through the denominator dynamics of the system before being added to the output. Nevertheless, the equation error model set has a very important property that makes it a prime choice in many applications: The predictor defines a linear regression.

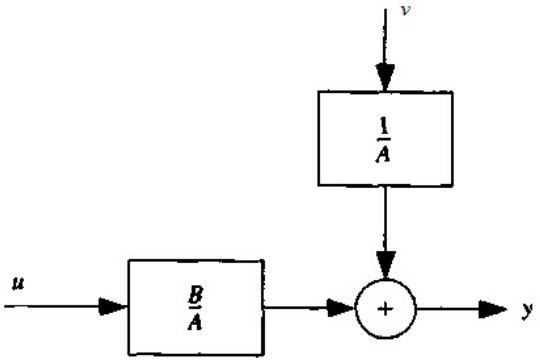

Figure 2.1.1: The ARX model structure

2 The basic disadvantage with the simple model (2.1.3) is the lack of the adequate freedom in describing the properties of the disturbance term. We could add flexibility to that by describing the equation error as a moving average of white noise. This is the following model

$$A(z)\,y(t) = B(z)\,u(t) + D(z)\,v(t) \tag{2.1.4}$$

where $A(z)$, $B(z)$ and $D(z)$ are defined as previously, and $w(t) = D(z)\,v(t)$ is in the form of a MA model, and is colored noise. In view of the moving average (MA) part $D(z)\,v(t)$, the model (2.1.4) will be called **ARMAX**. It is also called a controlled ARMA (CARMA) model. The ARMAX model has become a standard tool in control for both system description and control design.

3 Controlled ARAR (CARAR) model

$$A(z)\,y(t) = B(z)\,u(t) + \frac{1}{C(z)}v(t)$$

where $w(t) = \frac{1}{C(z)}v(t)$ is in the form of an AR model, and $w(t)$ is a colored noise.

4 Controlled ARARMA (CARARMA) model

$$A(z)\,y(t) = B(z)\,u(t) + \frac{D(z)}{C(z)}v(t)$$

where $w(t) = \frac{D(z)}{C(z)}v(t)$ is in the form of a ARMA model, and is colored noise.

2.1.3 Output error type models

An output error type model has the following form

$$y(t) = \frac{B(z)}{A(z)} u(t) + w(t).$$

In this model, the rational fractional term $\frac{B(z)}{A(z)} u(t) =: x(t)$ can be viewed as the true output of the considered system.

1. Output error (OE) model

$$y(t) = \frac{B(z)}{A(z)} u(t) + v(t)$$

2. Output error moving average (OEMA) model

$$y(t) = \frac{B(z)}{A(z)} u(t) + D(z) v(t)$$

3. Output error autoregressive (OEAR) model

$$y(t) = \frac{B(z)}{A(z)} u(t) + \frac{1}{C(z)} v(t)$$

4. Box-Jenkins (Output error autoregressive moving average) model

$$y(t) = \frac{B(z)}{A(z)} u(t) + \frac{D(z)}{C(z)} v(t)$$

2.2 Least squares principle

2.2.1 One-dimensional case

We first consider a one-demensional case. For a desk, we assume that n persons obtain n different values of the length of this desk x_1, x_2, $\cdots$, x_n. Now our question is, what is the most likely length of this desk? Intuitively, if x is the length of this desk, it should minimize the following error function

$$f(x) = (x_1 - x)^2 + (x_2 - x)^2 + \cdots + (x_n - x)^2,$$

which is the sum of the squares of the error. By setting the derivation to zero, one has

$$\frac{\mathrm{d} f(x)}{\mathrm{d} x} = -2(x_1 - x) - 2(x_2 - x) - \cdots - 2(x_n - x) = 0,$$

which gives

$$x = \frac{x_1 + x_2 + \cdots x_n}{n}.$$

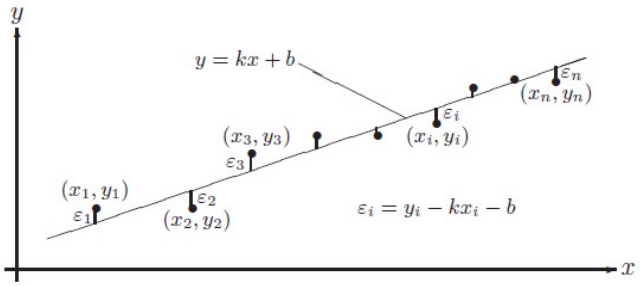

Figure 2.2.1: The linear fit of discrete dots

In addition, it is easily obtained that

$$\frac{\mathrm{d}^2 f(x)}{\mathrm{d}\, x^2} = 2n > 0.$$

Therefore, $f\left(\frac{x_1+x_2+\cdots x_n}{n}\right)$ is the minimum value of the error function $f(x)$. This is the basic principle of the method of least squares.

2.2.2 Two-dimensional case

Next we consider a two-dimensional example. Figure 2.2.1 shows that there are many dots (x_1, y_1), (x_2, y_2), $\cdots$, (x_n, y_n) in the 2-dimensional plane, which are near to some straight line. It is assumed that this line is

$$y = kx + b. \tag{2.2.1}$$

In general, it is impossible that all these dots (x_1, y_1), (x_2, y_2), $\cdots$, (x_n, y_n) satisfy the equation (2.2.1). Intuitively, we should choose parameters k and b that minimize the error

$$J(k, b) = (y_1 - kx_1 - b)^2 + (y_2 - kx_2 - b)^2 + \cdots + (y_n - kx_n - b)^2, \tag{2.2.2}$$

which is the sum of the squares of the residual

$$\varepsilon_i := y_i - kx_i - b. \tag{2.2.3}$$

This is the problem of least squares fitting. In general, least squares fitting is a mathematical procedure for finding the best-fitting curve to a given set of points by minimizing the sum of the squares of the offsets ("the residuals") of the points from the curve. The sum of the squares of the offsets is used instead of the offset absolute values because this allows the residuals to be treated as a continuous differentiable quantity.

Differentiating $J(k,b)$ yields

$$\begin{aligned}\frac{\partial J(k,b)}{\partial k} &= -2x_1(y_1-kx_1-b)-2x_2(y_2-kx_2-b)-\cdots-2x_n(y_n-kx_n-b),\\ \frac{\partial J(k,b)}{\partial b} &= -2(y_1-kx_1-b)-2(y_2-kx_2-b)-\cdots-2(y_n-kx_n-b).\end{aligned}$$

Setting the two partial derivations to zero yields

$$\begin{aligned}x_1(y_1-kx_1-b)+x_2(y_2-kx_2-b)+\cdots+x_n(y_n-kx_n-b) &= 0,\\ (y_1-kx_1-b)+(y_2-kx_2-b)+\cdots+(y_n-kx_n-b) &= 0.\end{aligned}$$

We may rewrite these equations as

$$\begin{aligned}\left(\sum_{i=1}^n x_i^2\right)k+\left(\sum_{i=1}^n x_i\right)b &= \sum_{i=1}^n x_iy_i,\\ \left(\sum_{i=1}^n x_i\right)k+nb &= \sum_{i=1}^n y_i.\end{aligned}$$

We have obtained that the values of k and b which minimize the error (defined in (2.2.2)) satisfy the following matrix equation:

$$\begin{bmatrix}\sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i\\ \sum_{i=1}^n x_i & n\end{bmatrix}\begin{bmatrix}k\\ b\end{bmatrix}=\begin{bmatrix}\sum_{i=1}^n x_iy_i\\ \sum_{i=1}^n y_i\end{bmatrix}.$$

We will show the matrix is invertible, which implies

$$\begin{bmatrix}k\\ b\end{bmatrix}=\begin{bmatrix}\sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i\\ \sum_{i=1}^n x_i & n\end{bmatrix}^{-1}\begin{bmatrix}\sum_{i=1}^n x_iy_i\\ \sum_{i=1}^n y_i\end{bmatrix}.$$

In order to deal with the high-dimensional case, we define the information vectors φ_i composed of the observed data and the parameter vector θ for the fitting model (2.2.1) as

$$\varphi_i=\begin{bmatrix}x_i\\ 1\end{bmatrix},\ \theta=\begin{bmatrix}k\\ b\end{bmatrix}.$$

It follows from (2.2.3) that

$$\begin{aligned}y_i &= kx_i+b+\varepsilon_i\\ &= \begin{bmatrix}x_i & 1\end{bmatrix}\begin{bmatrix}k\\ b\end{bmatrix}+\varepsilon_i\\ &= \varphi_i^{\mathrm{T}}\theta+\varepsilon_i.\end{aligned}$$

In the following, we consider the estimation problem in a vector form. The error criterion function is

$$J(\theta) = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(y_i - \varphi_i^{\mathrm{T}}\theta\right)^2.$$

Differentiating $J(\theta)$ yields

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \theta} &= -2\sum_{i=1}^{n} \varphi_i \left(y_i - \varphi_i^{\mathrm{T}}\theta\right) \\ &= -2\left[\sum_{i=1}^{n} \varphi_i y_i - \left(\sum_{i=1}^{n} \varphi_i \varphi_i^{\mathrm{T}}\right)\theta\right]. \end{aligned}$$

By setting the partial derivation to zero, one has

$$\left(\sum_{i=1}^{n} \varphi_i \varphi_i^{\mathrm{T}}\right)\theta = \sum_{i=1}^{n} \varphi_i y_i,$$

which implies that the least square estimation $\hat{\theta}$ of θ can be given by

$$\hat{\theta} = \left(\sum_{i=1}^{n} \varphi_i \varphi_i^{\mathrm{T}}\right)^{-1} \sum_{i=1}^{n} \varphi_i y_i. \tag{2.2.4}$$

This is a general case, can be used in some high-dimensional case. If we use this result to the prevous two-dimensional case, one has

$$\begin{aligned} \hat{\theta} &= \begin{bmatrix} k \\ b \end{bmatrix} \\ &= \left(\sum_{i=1}^{n} \begin{bmatrix} x_i \\ 1 \end{bmatrix} \begin{bmatrix} x_i & 1 \end{bmatrix}\right)^{-1} \sum_{i=1}^{n} \begin{bmatrix} x_i \\ 1 \end{bmatrix} y_i \\ &= \begin{bmatrix} \sum\limits_{i=1}^{n} x_i^2 & \sum\limits_{i=1}^{n} x_i \\ \sum\limits_{i=1}^{n} x_i & n \end{bmatrix}^{-1} \begin{bmatrix} \sum\limits_{i=1}^{n} x_i y_i \\ \sum\limits_{i=1}^{n} y_i \end{bmatrix}. \end{aligned}$$

Now we define

$$Y_n = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n,\ H_n := \begin{bmatrix} \varphi_1^{\mathrm{T}} \\ \varphi_2^{\mathrm{T}} \\ \vdots \\ \varphi_n^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^{n\times m},\ \varepsilon_n := \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \in \mathbb{R}^n.$$

By calculations, one has

$$H_n^{\mathrm{T}} H_n = \left[\begin{array}{cccc} \varphi_1 & \varphi_2 & \cdots & \varphi_n \end{array}\right] \left[\begin{array}{c} \varphi_1^{\mathrm{T}} \\ \varphi_2^{\mathrm{T}} \\ \vdots \\ \varphi_n^{\mathrm{T}} \end{array}\right] = \sum_{i=1}^{n} \varphi_i \varphi_i^T,$$

$$H_n^{\mathrm{T}} Y_n = \left[\begin{array}{cccc} \varphi_1 & \varphi_2 & \cdots & \varphi_n \end{array}\right] \left[\begin{array}{c} y_1 \\ y_2 \\ \vdots \\ y_n \end{array}\right] = \sum_{i=1}^{n} \varphi_i y_i.$$

With these results, the expression (2.2.4) can be equivalently written as

$$\hat{\theta} = \left(H_n^{\mathrm{T}} H_n\right)^{-1} H_n^{\mathrm{T}} Y_n.$$

It follows from $y_i = \varphi_i^{\mathrm{T}} \theta + \varepsilon_i$ that

$$\left\{\begin{array}{c} \varepsilon_1 = y_1 - \varphi_1^{\mathrm{T}} \theta, \\ \varepsilon_2 = y_2 - \varphi_2^{\mathrm{T}} \theta, \\ \cdots \\ \varepsilon_n = y_n - \varphi_n^{\mathrm{T}} \theta \end{array}\right. ,$$

which can be written in a compact vector form

$$\left[\begin{array}{c} \varepsilon_1 \\ \varepsilon_2 \\ \cdots \\ \varepsilon_n \end{array}\right] = \left[\begin{array}{c} y_1 \\ y_2 \\ \vdots \\ y_n \end{array}\right] - \left[\begin{array}{c} \varphi_1^{\mathrm{T}} \\ \varphi_2^{\mathrm{T}} \\ \vdots \\ \varphi_n^{\mathrm{T}} \end{array}\right] \theta = Y_n - H_n \theta.$$

So the error criterion function $J(\theta)$ can be rewritten as

$$J(\theta) = \sum_{i=1}^{n} \varepsilon_i^2 = \left(Y_n - H_n \theta\right)^{\mathrm{T}} \left(Y_n - H_n \theta\right). \tag{2.2.5}$$

2.2.3 Matrix calculus

At the end of this section, we investigate the least squares method in the matrix form. Firstly, we collect some useful formulas of matrix calculus. Matrix calculus is concerned with rules for operating on functions of matrices.

Define the derivative of a function mapping $f : \mathbb{R}^n \to \mathbb{R}^m$ as the $n \times m$ matrix of partial derivatives. That is, if $x \in \mathbb{R}^n$, $f(x) \in \mathbb{R}^m$, the derivative of f with respect to x is defined as

$$\left[\frac{\partial f}{\partial x}\right]_{ij} = \frac{\partial f_i}{\partial x_i}.$$

If $n=1$, x reduces to a scalar. If $m=1$, y reduces to a scalar. If

$$x=\begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix},\ f(x)=\begin{bmatrix} f_1(x) \\ f_2(x) \\ \cdots \\ f_m(x) \end{bmatrix}, \tag{2.2.6}$$

then we define

$$\frac{\partial f}{\partial x}=\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_2} \\ \cdots & \cdots & & \cdots \\ \frac{\partial f_1}{\partial x_n} & \frac{\partial f_2}{\partial x_n} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}. \tag{2.2.7}$$

If f is a scalar, one has

$$\frac{\partial f}{\partial x}=\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}.$$

With this notation, for the variable $x\in\mathbb{R}^n$ and the function $f(x)$ in (2.2.6), the following form holds

$$\frac{\partial f}{\partial x}=\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_2}{\partial x} & \cdots & \frac{\partial f_2}{\partial x} \end{bmatrix}. \tag{2.2.8}$$

Remark 2.2.1 Many authors, notably in statistics and economics, define the derivatives as the transposes of those given above This has the advantage of better agreement of matrix products with composition schemes such as the chain rule. Evidently the notation is not yet stable.

Lemma 2.2.1 *Given $A\in\mathbb{R}^{n\times n}$, $f(x)=x^{\mathrm{T}}Ax$, one has*

$$\frac{\partial f}{\partial x}=Ax+A^{\mathrm{T}}x.$$

If A is symmetric, then $\frac{\partial f}{\partial x}=2Ax$.

Proof. Denote

$$x=\begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix},\ A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}.$$

Then

$$f(x)=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}x_ix_j.$$

By definition, one has

$$\begin{aligned}
&\frac{\partial f}{\partial x} \\
= &\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n} a_{1j}x_j + \sum_{i=1}^{n} a_{i1}x_i \\ \sum_{j=1}^{n} a_{2j}x_j + \sum_{i=1}^{n} a_{i2}x_i \\ \cdots \\ \sum_{j=1}^{n} a_{nj}x_j + \sum_{i=1}^{n} a_{in}x_i \end{bmatrix} \\
= &Ax + A^{\mathrm{T}}x.
\end{aligned}$$

The proof is thus completed. ■

Lemma 2.2.2 *Given $A \in \mathbb{R}^{m\times n}$, $f(x) = Ax$, one has*

$$\frac{\partial f}{\partial x} = A^{\mathrm{T}}.$$

Proof. The scalar x are defined as in the previous lemma, and $A = [a_{ij}]_{m\times n}$. Then one has

$$f(x) = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix}.$$

From this relation, it is easily obtained that

$$\begin{aligned}
\frac{\partial f_1}{\partial x_1} &= a_{11},\ \frac{\partial f_2}{\partial x_1} = a_{21},\ \cdots,\ \frac{\partial f_m}{\partial x_1} = a_{m1} \\
\frac{\partial f_1}{\partial x_2} &= a_{12},\ \frac{\partial f_2}{\partial x_2} = a_{22},\ \cdots,\ \frac{\partial f_n}{\partial x_2} = a_{m2} \\
&\cdots \\
\frac{\partial f_1}{\partial x_n} &= a_{1n},\ \frac{\partial f_2}{\partial x_n} = a_{2n},\ \cdots,\ \frac{\partial f_n}{\partial x_n} = a_{mn}
\end{aligned}$$

which implies the result. ■

A special case of this lemma is that the matrix A is an identity matrix. By using this lemm, one has

$$\frac{\partial x}{\partial x} = I.$$

In the above lemma, if A is a row vector, the function $f(x)$ is a scalar. By applying this result to this case, one can obtain the following useful corrollary.

Lemma 2.2.3 *Given $\alpha \in \mathbb{R}^n$, $f(x) = \alpha^{\mathrm{T}} x$, one has*

$$\frac{\partial f}{\partial x} = \frac{\partial (\alpha^{\mathrm{T}} x)}{\partial x} = \alpha.$$

Lemma 2.2.4 *[The chain rule for vector functions] Let*

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}, \; y = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_r \end{bmatrix}, \; z = \begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_m \end{bmatrix}.$$

where z is a function of y, which is in turn a function of x. Then,

$$\frac{\partial z}{\partial x} = \frac{\partial y}{\partial x} \frac{\partial z}{\partial y}.$$

Proof. Using the definition (2.2.7), we can write

$$\left(\frac{\partial z}{\partial x} \right)^{\mathrm{T}} = \begin{bmatrix} \frac{\partial z_1}{\partial x_1} & \frac{\partial z_1}{\partial x_2} & \cdots & \frac{\partial z_1}{\partial x_n} \\ \frac{\partial z_2}{\partial x_1} & \frac{\partial z_2}{\partial x_2} & \cdots & \frac{\partial z_2}{\partial x_n} \\ \cdots & \cdots & & \cdots \\ \frac{\partial z_m}{\partial x_1} & \frac{\partial z_m}{\partial x_2} & \cdots & \frac{\partial z_m}{\partial x_n} \end{bmatrix}.$$

Each entry of this matrix may be expanded as

$$\frac{\partial z_j}{\partial x_i} = \sum_{q=1}^{r} \frac{\partial z_j}{\partial y_q} \frac{\partial y_q}{\partial x_i}.$$

Then

$$\begin{aligned}
\left(\frac{\partial z}{\partial x} \right)^{\mathrm{T}} &= \begin{bmatrix} \frac{\partial z_1}{\partial x_1} & \frac{\partial z_1}{\partial x_2} & \cdots & \frac{\partial z_1}{\partial x_n} \\ \frac{\partial z_2}{\partial x_1} & \frac{\partial z_2}{\partial x_2} & \cdots & \frac{\partial z_2}{\partial x_n} \\ \cdots & \cdots & & \cdots \\ \frac{\partial z_m}{\partial x_1} & \frac{\partial z_m}{\partial x_2} & \cdots & \frac{\partial z_m}{\partial x_n} \end{bmatrix} \\
&= \begin{bmatrix} \sum_{q=1}^{r} \frac{\partial z_1}{\partial y_q} \frac{\partial y_q}{\partial x_1} & \sum_{q=1}^{r} \frac{\partial z_1}{\partial y_q} \frac{\partial y_q}{\partial x_2} & \cdots & \sum_{q=1}^{r} \frac{\partial z_1}{\partial y_q} \frac{\partial y_q}{\partial x_n} \\ \sum_{q=1}^{r} \frac{\partial z_2}{\partial y_q} \frac{\partial y_q}{\partial x_1} & \sum_{q=1}^{r} \frac{\partial z_2}{\partial y_q} \frac{\partial y_q}{\partial x_2} & \cdots & \sum_{q=1}^{r} \frac{\partial z_2}{\partial y_q} \frac{\partial y_q}{\partial x_n} \\ \cdots & \cdots & & \cdots \\ \sum_{q=1}^{r} \frac{\partial z_m}{\partial y_q} \frac{\partial y_q}{\partial x_1} & \sum_{q=1}^{r} \frac{\partial z_m}{\partial y_q} \frac{\partial y_q}{\partial x_2} & \cdots & \sum_{q=1}^{r} \frac{\partial z_m}{\partial y_q} \frac{\partial y_q}{\partial x_n} \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial z_1}{\partial y_1} & \frac{\partial z_1}{\partial y_2} & \cdots & \frac{\partial z_1}{\partial y_r} \\ \frac{\partial z_2}{\partial y_1} & \frac{\partial z_2}{\partial y_2} & \cdots & \frac{\partial z_2}{\partial y_r} \\ \cdots & \cdots & & \cdots \\ \frac{\partial z_m}{\partial y_1} & \frac{\partial z_m}{\partial y_2} & \cdots & \frac{\partial z_m}{\partial y_r} \end{bmatrix} \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \cdots & \cdots & & \cdots \\ \frac{\partial y_r}{\partial x_1} & \frac{\partial y_r}{\partial x_2} & \cdots & \frac{\partial y_r}{\partial x_n} \end{bmatrix} \\
&= \left(\frac{\partial z}{\partial y} \right)^{\mathrm{T}} \left(\frac{\partial y}{\partial x} \right)^{\mathrm{T}} = \left(\frac{\partial y}{\partial x} \frac{\partial z}{\partial y} \right)^{\mathrm{T}}.
\end{aligned}$$

On transposing both sides, we finally obtain $\frac{\partial z}{\partial x} = \frac{\partial y}{\partial x}\frac{\partial z}{\partial y}$. ■

Lemma 2.2.5 *[Linearity rule] If $f(x)$, $g(x) \in \mathbb{R}^m$ are two vector functions with respect to the vector variable $x \in \mathbb{R}^n$, for two real scalars c_1 and c_2 the following relation holds*

$$\frac{\partial [c_1 f(x) + c_2 g(x)]}{\partial x} = c_1 \frac{\partial f(x)}{\partial x} + c_2 \frac{\partial g(x)}{\partial x}.$$

The proof is very simple, and thus is omitted.

Lemma 2.2.6 *[Product rule] If $f(x)$, $g(x) \in \mathbb{R}^m$ are two vector functions with respect to the vector variable $x \in \mathbb{R}^n$, then*

$$\frac{\partial f^{\mathrm{T}}(x)\, g(x)}{\partial x} = \frac{\partial f(x)}{\partial x} g(x) + \frac{\partial g(x)}{\partial x} f(x).$$

Proof. Firstly, we consider the case where $f(x)$ and $g(x)$ are two scalar functions. By using the definition, one has

$$\begin{aligned}
&\frac{\partial f(x)\, g(x)}{\partial x} \\
= &\begin{bmatrix} \frac{\partial f(x) g(x)}{\partial x_1} \\ \frac{\partial f(x) g(x)}{\partial x_2} \\ \cdots \\ \frac{\partial f(x) g(x)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} f(x)\frac{\partial g(x)}{\partial x_1} + \frac{\partial f(x)}{\partial x_1} g(x) \\ f(x)\frac{\partial g(x)}{\partial x_2} + \frac{\partial f(x)}{\partial x_2} g(x) \\ \cdots \\ f(x)\frac{\partial g(x)}{\partial x_n} + \frac{\partial f(x)}{\partial x_n} g(x) \end{bmatrix} \\
= &f(x) \begin{bmatrix} \frac{\partial g(x)}{\partial x_1} \\ \frac{\partial g(x)}{\partial x_2} \\ \cdots \\ \frac{\partial g(x)}{\partial x_n} \end{bmatrix} + \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \cdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} g(x) \\
= &f(x)\frac{\partial g(x)}{\partial x} + \frac{\partial f(x)}{\partial x} g(x).
\end{aligned}$$

With this result as a tools, by letting

$$\begin{aligned}
f(x) &= \begin{bmatrix} f_1(x) & f_2(x) & \cdots & f_m(x) \end{bmatrix}^{\mathrm{T}}, \\
g(x) &= \begin{bmatrix} g_1(x) & g_2(x) & \cdots & g_m(x) \end{bmatrix}^{\mathrm{T}},
\end{aligned}$$

it can be obtained

$$\begin{aligned}
& \frac{\partial f^{\mathrm{T}}(x)\, g(x)}{\partial x} \\
= & \frac{\partial\left[\sum_{i=1}^{m} f_i(x)\, g_i(x)\right]}{\partial x} \\
= & \sum_{i=1}^{m} \frac{\partial f_i(x)\, g_i(x)}{\partial x} \\
= & \sum_{i=1}^{m}\left[f_i(x)\frac{\partial g_i(x)}{\partial x} + \frac{\partial f_i(x)}{\partial x} g_i(x)\right] \\
= & \sum_{i=1}^{m} f_i(x)\frac{\partial g_i(x)}{\partial x} + \sum_{i=1}^{m}\frac{\partial f_i(x)}{\partial x} g_i(x) \\
= & \left[\begin{array}{cccc} \frac{\partial g_1(x)}{\partial x} & \frac{\partial g_2(x)}{\partial x} & \cdots & \frac{\partial g_m(x)}{\partial x} \end{array}\right]\left[\begin{array}{c} f_1(x) \\ f_2(x) \\ \cdots \\ f_m(x) \end{array}\right] \\
& + \left[\begin{array}{cccc} \frac{\partial f_1(x)}{\partial x} & \frac{\partial f_2(x)}{\partial x} & \cdots & \frac{\partial f_m(x)}{\partial x} \end{array}\right]\left[\begin{array}{c} g_1(x) \\ g_2(x) \\ \cdots \\ g_m(x) \end{array}\right] \\
= & \frac{\partial g(x)}{\partial x} f(x) + \frac{\partial f(x)}{\partial x} g(x).
\end{aligned}$$

This is the result of the lemma. ■

Lemma 2.2.7 *[Quotient rule] If $f(x) \in \mathbb{R}^m$ and $g(x) \in \mathbb{R}$ are functions with repect to the vector variable $x \in \mathbb{R}^n$, then*

$$\frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x} = \frac{1}{g^2(x)}\left[g(x)\frac{\partial f(x)}{\partial x} - f(x)\right]$$

Proof. Firstly, we consider the case where $f(x)$ is a scalar function. By the

definition, one has

$$\begin{aligned}
&\frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x}\\
=&\begin{bmatrix}\frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x_1}\\ \frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x_1}\\ \cdots\\ \frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x_1}\end{bmatrix}=\begin{bmatrix}\frac{1}{g^2(x)}\left[g(x)\frac{\partial f(x)}{\partial x_1}-f(x)\frac{\partial g(x)}{\partial x_1}\right]\\ \frac{1}{g^2(x)}\left[g(x)\frac{\partial f(x)}{\partial x_2}-f(x)\frac{\partial g(x)}{\partial x_2}\right]\\ \cdots\\ \frac{1}{g^2(x)}\left[g(x)\frac{\partial f(x)}{\partial x_n}-f(x)\frac{\partial g(x)}{\partial x_n}\right]\end{bmatrix}\\
=&\frac{1}{g^2(x)}\begin{bmatrix}g(x)\frac{\partial f(x)}{\partial x_1}-f(x)\frac{\partial g(x)}{\partial x_1}\\ g(x)\frac{\partial f(x)}{\partial x_2}-f(x)\frac{\partial g(x)}{\partial x_2}\\ \cdots\\ g(x)\frac{\partial f(x)}{\partial x_n}-f(x)\frac{\partial g(x)}{\partial x_n}\end{bmatrix}\\
=&\frac{1}{g^2(x)}\left\{g(x)\begin{bmatrix}\frac{\partial f(x)}{\partial x_1}\\ \frac{\partial f(x)}{\partial x_2}\\ \cdots\\ \frac{\partial f(x)}{\partial x_n}\end{bmatrix}-f(x)\begin{bmatrix}\frac{\partial g(x)}{\partial x_1}\\ \frac{\partial g(x)}{\partial x_2}\\ \cdots\\ \frac{\partial g(x)}{\partial x_n}\end{bmatrix}\right\}\\
=&\frac{1}{g^2(x)}\left[g(x)\frac{\partial f(x)}{\partial x}-f(x)\frac{\partial g(x)}{\partial x}\right].
\end{aligned}$$

Let

$$f(x)=\begin{bmatrix}f_1(x) & f_2(x) & \cdots & f_m(x)\end{bmatrix}^{\mathrm{T}}.$$

Then, by applying the result of scalar case it is obtained that

$$\begin{aligned}
&\frac{\partial\left[\frac{f(x)}{g(x)}\right]}{\partial x}\\
=&\frac{\partial\begin{bmatrix}\frac{f_1(x)}{g(x)} & \frac{f_2(x)}{g(x)} & \cdots & \frac{f_m(x)}{g(x)}\end{bmatrix}^{\mathrm{T}}}{\partial x}\\
=&\begin{bmatrix}\frac{\partial\left[\frac{f_1(x)}{g(x)}\right]}{\partial x} & \frac{\partial\left[\frac{f_2(x)}{g(x)}\right]}{\partial x} & \cdots & \frac{\partial\left[\frac{f_2(x)}{g(x)}\right]}{\partial x}\end{bmatrix}\\
=&\frac{1}{g^2(x)}\begin{bmatrix}g(x)\frac{\partial f_1(x)}{\partial x}-f_1(x)\frac{\partial g(x)}{\partial x} & g(x)\frac{\partial f_2(x)}{\partial x}-f_2(x)\frac{\partial g(x)}{\partial x} & \cdots & g(x)\frac{\partial f_2(x)}{\partial x}-f_2(x)\frac{\partial g(x)}{\partial x}\end{bmatrix}\\
=&\frac{1}{g^2(x)}\begin{bmatrix}g(x)\frac{\partial f_1(x)}{\partial x} & g(x)\frac{\partial f_2(x)}{\partial x} & \cdots & g(x)\frac{\partial f_2(x)}{\partial x}\end{bmatrix}\\
&-\frac{1}{g^2(x)}\begin{bmatrix}\frac{\partial g(x)}{\partial x}f_1(x) & \frac{\partial g(x)}{\partial x}f_2(x) & \cdots & \frac{\partial g(x)}{\partial x}f_2(x)\end{bmatrix}\\
=&\frac{1}{g^2(x)}\begin{bmatrix}\frac{\partial f_1(x)}{\partial x} & \frac{\partial f_2(x)}{\partial x} & \cdots & \frac{\partial f_2(x)}{\partial x}\end{bmatrix}g(x)-\frac{1}{g^2(x)}\frac{\partial g(x)}{\partial x}\begin{bmatrix}f_1(x) & f_2(x) & \cdots & f_2(x)\end{bmatrix}\\
=&\frac{1}{g^2(x)}\left[\frac{\partial f(x)}{\partial x}g(x)-\frac{\partial g(x)}{\partial x}f^{\mathrm{T}}(x)\right].
\end{aligned}$$

The proof is thus completed. ■

Now we give some results on the derivative of scalar functions of a matrix.

Let $X = ([x_{ij}]$ be a matrix of order $m \times n$ and let $y = f(X)$ be a scalar function of X. The derivative of y with respect to X, denoted by $\frac{\partial y}{\partial X}$, is defined as the following matrix of order $m \times n$,

$$G = \frac{\partial y}{\partial X} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{12}} & \cdots & \frac{\partial y}{\partial x_{1n}} \\ \frac{\partial y}{\partial x_{21}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{2n}} \\ \cdots & \cdots & & \cdots \\ \frac{\partial y}{\partial x_{m1}} & \frac{\partial y}{\partial x_{m2}} & \cdots & \frac{\partial y}{\partial x_{mn}} \end{bmatrix} = \left[\frac{\partial y}{\partial x_{ij}} \right].$$

Next we provide some properties of derivative of a scalar function with respect to a matrix variable.

Lemma 2.2.8 *[Linearity rule] Given two real scalar functions $f(X)$, $g(X)$ with respect to the matrix variable X, for two real constants c_1 and c_2 there holds*

$$\frac{\partial [c_1 f(X) + c_2 g(X)]}{\partial X} = c_1 \frac{\partial f(X)}{\partial X} + c_2 \frac{\partial g(X)}{\partial X}.$$

Lemma 2.2.9 *[Product rule] If $f(X)$ and $g(X)$ are two scalar functions with respect to the matrix variable X, then*

$$\frac{\partial f(X)\, g(X)}{\partial X} = g(X) \frac{\partial f(X)}{\partial X} + f(X) \frac{\partial g(X)}{\partial X}.$$

Proof. Denote $X = [x_{ij}]_{n \times m}$. By using the definiton, one has

$$\begin{aligned} \frac{\partial f(X)\, g(X)}{\partial X} &= \left[\frac{\partial f(X)\, g(X)}{\partial x_{ij}} \right]_{n \times m} \\ &= \left[g(X) \frac{\partial f(X)}{\partial x_{ij}} + f(X) \frac{\partial g(X)}{\partial x_{ij}} \right]_{n \times m} \\ &= g(X) \left[\frac{\partial f(X)}{\partial x_{ij}} \right]_{n \times m} + f(X) \left[\frac{\partial g(X)}{\partial x_{ij}} \right]_{n \times m} \\ &= g(X) \frac{\partial f(X)}{\partial X} + f(X) \frac{\partial g(X)}{\partial X}, \end{aligned}$$

which implies the conclusion of the lemma. ■

Lemma 2.2.10 *[Quotient rule] If $f(X)$ and $g(X)$ are two scalar functions with respect to the matrix variable X, then*

$$\frac{\partial \left[\frac{f(X)}{g(X)} \right]}{\partial X} = g^2(X) \left[g(X) \frac{\partial f(X)}{\partial X} - f(X) \frac{\partial g(X)}{\partial X} \right].$$

Lemma 2.2.11 *Let $X \in \mathbb{R}^{m\times n}$ be a matrix variable, and $A \in \mathbb{R}^{n\times m}$ be constant matrix. Then*

$$\frac{\partial \operatorname{tr}(XA)}{\partial X} = \frac{\partial \operatorname{tr}(AX)}{\partial X} = A^{\mathrm{T}}.$$

Proof. Let $X = [x_{ij}]_{m\times n}$ and $A = [a_{jk}]_{n\times m}$. Then

$$AX = \begin{bmatrix} \sum_{j=1}^{m} a_{1j}x_{j1} & * & * & * \\ * & \sum_{j=1}^{m} a_{2j}x_{j2} & * & * \\ * & * & \ddots & * \\ * & * & * & \sum_{j=1}^{m} a_{nj}x_{jn} \end{bmatrix},$$

where * denote any entry that we do not care. With this, it is easily known that

$$\operatorname{tr}(AX) = \sum_{j=1}^{m} a_{1j}x_{j1} + \sum_{j=1}^{m} a_{2j}x_{j2} + \cdots + \sum_{j=1}^{m} a_{nj}x_{jn} = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_{ji}.$$

Then, one has

$$\begin{aligned} \frac{\partial \operatorname{tr}(AX)}{\partial x_{11}} &= a_{11},\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{12}} = a_{21},\ \cdots,\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{1n}} = a_{n1}, \\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{21}} &= a_{12},\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{22}} = a_{22},\ \cdots,\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{2n}} = a_{n2}, \\ &\cdots \\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{m1}} &= a_{1m},\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{m2}} = a_{2m},\ \cdots,\ \frac{\partial \operatorname{tr}(AX)}{\partial x_{mn}} = a_{nm}. \end{aligned}$$

Thus, it is obtained that

$$\frac{\partial \operatorname{tr}(AX)}{\partial X} = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \cdots & \cdots & & \cdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{bmatrix} = A^{\mathrm{T}}.$$

The conlusion is thus proven. ■

Lemma 2.2.12 *Let $X \in \mathbb{R}^{m\times n}$ be a matrix variable, and $A \in \mathbb{R}^{m\times n}$ be constant matrix. Then*

$$\frac{\partial \operatorname{tr}(X^{\mathrm{T}}A)}{\partial X} = \frac{\partial \operatorname{tr}(AX^{\mathrm{T}})}{\partial X} = A.$$

Lemma 2.2.13 *Let $X \in \mathbb{R}^{m\times n}$ be a matrix variable, and $A \in \mathbb{R}^{m\times m}$ be constant matrix. Then*

$$\frac{\partial \operatorname{tr}(X^{\mathrm{T}}AX)}{\partial X} = \left(A^{\mathrm{T}} + A\right)X.$$

Proof. Let $X = [x_{ij}]_{m\times n}$, $A = [a_{jk}]_{m\times m}$. Then

$$
\begin{aligned}
X^{\mathrm{T}}AX &= \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{m1} \\ x_{12} & x_{22} & \cdots & x_{m2} \\ \cdots & \cdots & & \cdots \\ x_{1n} & x_{2n} & \cdots & x_{mn} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \cdots & \cdots & & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} X \\
&= \begin{bmatrix} \sum_{j=1}^{m} x_{j1}a_{j1} & \sum_{j=1}^{m} x_{j1}a_{j2} & \cdots & \sum_{j=1}^{m} x_{j1}a_{jm} \\ \sum_{j=1}^{m} x_{j2}a_{j1} & \sum_{j=1}^{m} x_{j2}a_{j2} & \cdots & \sum_{j=1}^{m} x_{j2}a_{jm} \\ \cdots & \cdots & & \cdots \\ \sum_{j=1}^{m} x_{jn}a_{j1} & \sum_{j=1}^{m} x_{jn}a_{j2} & \cdots & \sum_{j=1}^{m} x_{jn}a_{jm} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & & \cdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \\
&= \begin{bmatrix} \sum_{k=1}^{m}\sum_{j=1}^{m} x_{j1}a_{jk}x_{k1} & * & * & * \\ * & \sum_{k=1}^{m}\sum_{j=1}^{m} x_{j2}a_{jk}x_{k2} & * & * \\ * & * & \ddots & * \\ * & * & * & \sum_{k=1}^{m}\sum_{j=1}^{m} x_{jn}a_{jk}x_{kn} \end{bmatrix}.
\end{aligned}
$$

With this, one has

$$
\begin{aligned}
&\frac{\partial \operatorname{tr}(X^{\mathrm{T}}AX)}{\partial X} \\
&= \begin{bmatrix} \sum_{k=1}^{m} a_{1k}x_{k1} + \sum_{j=1}^{m} a_{j1}x_{j1} & \sum_{k=1}^{m} a_{1k}x_{k2} + \sum_{j=1}^{m} a_{j1}x_{j2} & \cdots & \sum_{k=1}^{m} a_{1k}x_{kn} + \sum_{j=1}^{m} a_{j1}x_{jn} \\ \sum_{k=1}^{m} a_{2k}x_{k1} + \sum_{j=1}^{m} a_{j2}x_{j1} & \sum_{k=1}^{m} a_{2k}x_{k2} + \sum_{j=1}^{m} a_{j2}x_{j2} & \cdots & \sum_{k=1}^{m} a_{2k}x_{kn} + \sum_{j=1}^{m} a_{j2}x_{jn} \\ \cdots & \cdots & & \cdots \\ \sum_{k=1}^{m} a_{mk}x_{k1} + \sum_{j=1}^{m} a_{jm}x_{j1} & \sum_{k=1}^{m} a_{mk}x_{k2} + \sum_{j=1}^{m} a_{jm}x_{j2} & \cdots & \sum_{k=1}^{m} a_{mk}x_{kn} + \sum_{j=1}^{m} a_{jm}x_{jn} \end{bmatrix} \\
&= \left[\sum_{k=1}^{m} a_{ik}x_{kj}\right]_{ij} + \left[\sum_{j=1}^{m} a_{ji}x_{jk}\right]_{ik} \\
&= AX + A^{\mathrm{T}}X.
\end{aligned}
$$

The proof is thus completed. ■

Lemma 2.2.14 *Let $X \in \mathbb{R}^{m\times n}$ be a matrix variable, and $A \in \mathbb{R}^{n\times n}$ be constant*

matrix. Then

$$\frac{\partial \operatorname{tr}\left(XAX^{\mathrm{T}}\right)}{\partial X} = X\left(A^{\mathrm{T}} + A\right).$$

The proof of this lemma is left to the students.

Now we investigate the least squares fitting problem by using the matrix calculus. For the error criterion function (2.2.5), Let

$$f = Y_n - H_n\theta.$$

Then the error criterion function (2.2.5) can be rewritten as

$$J = f^{\mathrm{T}} f.$$

By using Lemma 2.2.2, one has

$$\frac{\partial f}{\partial \theta} = -H_n^{\mathrm{T}}.$$

Applying the chain rule for vector functions, gives

$$\begin{aligned}\frac{\partial J}{\partial \theta} &= \frac{\partial f}{\partial \theta}\frac{\partial J}{\partial f} \\ &= -H_n^{\mathrm{T}}(2f) \\ &= -2H_n^{\mathrm{T}}\left(Y_n - H_n\theta\right).\end{aligned}$$

By setting this derivation to zero, one can obtain the least square estimation of θ as

$$\hat{\theta} = \left(H_n^{\mathrm{T}} H_n\right)^{-1} H_n^{\mathrm{T}} Y_n.$$

2.3 Least square estimation

2.3.1 Single-output case

Consider variables y, x_1, x_2, $\cdots$, x_n. If y is a linear combination of x_1, x_2, $\cdots$, x_n, then there exist a group of parameters θ_1, θ_2, $\cdots$, θ_n such that the following relation holds

$$y = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n.$$

The above relation holds for the deterministic case. In general, there exits measurement error, the above relation does not hold. In this case, we add a noise term into the right-hand of the above equality, and obtain a linear regressive model

$$y = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n + v.$$

In time instant t, we obtain the observed values $y(t)$ and $x_i(t)$ of the variables y and x_i. The corresponding measurement error is denoted by $v(t)$. In this case,

we can obtain t equations

$$\begin{cases} y(1) = \theta_1 x_1(1) + \theta_2 x_2(1) + \cdots + \theta_n x_n(1) + v(1), \\ y(2) = \theta_1 x_1(2) + \theta_2 x_2(2) + \cdots + \theta_n x_n(2) + v(2), \\ \vdots \\ y(t) = \theta_1 x_1(t) + \theta_2 x_2(t) + \cdots + \theta_n x_n(t) + v(t). \end{cases} \tag{2.3.1}$$

These equations can be simply denoted by

$$y(t) = \theta_1 x_1(t) + \theta_2 x_2(t) + \cdots + \theta_n x_n(t) + v(t),\ t = 1, 2, 3, \cdots. \tag{2.3.2}$$

$\{v(t)\}$ can be viewed as white noise series with zero mean and variance σ^2. n is the number of the parameters. When $t \leq 0$, it is assumed that $y(t) = 0$, $x_i(t) = 0$, $v(t) = 0$.

Define parameter vector

$$\theta := \left[\begin{array}{cccc} \theta_1 & \theta_2 & \cdots & \theta_n \end{array} \right]^{\mathrm{T}} \in \mathbb{R}^n,$$

and information vector

$$\varphi(t) := \left[\begin{array}{cccc} x_1(t) & x_2(t) & \cdots & x_n(t) \end{array} \right]^{\mathrm{T}} \in \mathbb{R}^n.$$

By the definition of θ and $\varphi(t)$, relation (2.3.2) can be rewritten as

$$\begin{aligned} y(t) &= \theta_1 x_1(t) + \theta_2 x_2(t) + \cdots + \theta_n x_n(t) + v(t) \\ &= \left[\begin{array}{cccc} x_1(t) & x_2(t) & \cdots & x_n(t) \end{array} \right] \left[\begin{array}{c} \theta_1 \\ \theta_2 \\ \cdots \\ \theta_n \end{array} \right] + v(t) \\ &= \varphi^{\mathrm{T}}(t)\,\theta + v(t), \end{aligned}$$

which is a linear regression model. In the field of system identification, it is called an identification model or an identification representation. The expression (2.3.2) can be written as

$$\begin{cases} y(1) = \varphi^{\mathrm{T}}(1)\,\theta + v(1), \\ y(2) = \varphi^{\mathrm{T}}(2)\,\theta + v(2), \\ \cdots \\ y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t). \end{cases}$$

These equations can be rewritten in the following matrix form

$$\left[\begin{array}{c} y(1) \\ y(2) \\ \cdots \\ y(t) \end{array} \right] = \left[\begin{array}{c} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{array} \right] \theta + \left[\begin{array}{c} v(1) \\ v(2) \\ \cdots \\ v(t) \end{array} \right]. \tag{2.3.3}$$

Denote

$$Y_t := \begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t) \end{bmatrix} \in \mathbb{R}^t,\ H_t := \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times n},\ V_t := \begin{bmatrix} v(1) \\ v(2) \\ \cdots \\ v(t) \end{bmatrix} \in \mathbb{R}^t. \tag{2.3.4}$$

(2.3.3) becomes the following simple form

$$Y_t = H_t\theta + V_t. \tag{2.3.5}$$

Principle of least square identification: by using the observed data $\{y(t), \varphi(t)\}$, to obtain an estimate $\hat{\theta}_{LS}$ of θ such that the following quadratic criterion function

$$\begin{aligned} J(\theta) \quad &: \quad = \sum_{i=1}^{t} v^2(i) = \sum_{i=1}^{t} \left[y(i) - \varphi^{\mathrm{T}}(i)\,\theta\right]^2 \\ &= \quad V_t^{\mathrm{T}} V_t = (Y_t - H_t\theta)^{\mathrm{T}} (Y_t - H_t\theta) \end{aligned} \tag{2.3.6}$$

is minimum when $\theta = \hat{\theta}_{LS}$.

By setting the derivation to zero, one has

$$H_t^{\mathrm{T}} (Y_t - H_t\theta) = 0, \tag{2.3.7}$$

which implies that the estimate $\hat{\theta}_{LS}$ should satisfy

$$\left(H_t^{\mathrm{T}} H_t\right) \hat{\theta}_{LS} = H_t^{\mathrm{T}} Y_t, \tag{2.3.8}$$

which is called normal equation. Under the condition of persistent excitation, the matrix $\left(H_t^{\mathrm{T}} H_t\right)$ is positive definite when the data length t is large enough. In this case, from (2.3.8) the estimate can be obtained as

$$\hat{\theta}_{LS} = \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^T Y_t. \tag{2.3.9}$$

The above estimate $\hat{\theta}_{LS}$ is called the least squares estimate, and simply LS estimate. Since it is related to the data length t, it can be denoted by $\hat{\theta}_{LS}(t)$. In a recursive algorithm, it is denoted by $\hat{\theta}(t)$. Substituting the definition (2.3.2) of H_t and Y_t into (2.3.9), gives

$$\begin{aligned} \hat{\theta}(t) \quad &= \quad \hat{\theta}_{LS}(t) = \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} Y_t \\ &= \quad \left[\sum_{i=1}^{t} \varphi(i)\,\varphi^{\mathrm{T}}(i)\right]^{-1} \left[\sum_{i=1}^{t} \varphi(i)\,y(i)\right]. \end{aligned}$$

2.3.2 Multi-output case

Now we consider the case the multi-output case. It is assumed that each component

$$y = \left[\begin{array}{c} y_1 \\ y_2 \\ \cdots \\ y_m \end{array} \right]$$

is linear combination of variables x_1, x_2, $\cdots$, x_m. There θ_{ij}, $i =$ such that

$$\left\{ \begin{array}{l} y_1 = \theta_{11}x_1 + \theta_{12}x_2 + \cdots + \theta_{1n}x_n \\ y_2 = \theta_{21}x_1 + \theta_{22}x_2 + \cdots + \theta_{2n}x_n \\ \cdots \\ y_m = \theta_{m1}x_1 + \theta_{m2}x_2 + \cdots + \theta_{mn}x_n \end{array} \right. .$$

In general, there exits measurement error, the above relation does not hold. In this case, we add a noise term into the right-hand of the above equality, and obtain a linear regressive model

$$\left\{ \begin{array}{l} y_1 = \theta_{11}x_1 + \theta_{12}x_2 + \cdots + \theta_{1n}x_n + v_1 \\ y_2 = \theta_{21}x_1 + \theta_{22}x_2 + \cdots + \theta_{2n}x_n + v_2 \\ \cdots \\ y_m = \theta_{m1}x_1 + \theta_{m2}x_2 + \cdots + \theta_{mn}x_n + v_m \end{array} \right. .$$

In time instant t, we obtain the observed values $y(t)$ and $x_i(t)$ of the variables y and x_i. The corresponding measurement error is denoted by $v_i(t)$, $i = 1,\ 2,\ \cdots,\ m$. In this case, we can obtain

$$\left\{ \begin{array}{l} y_i\left(1\right) = \theta_{i1}x_1\left(1\right) + \theta_{i2}x_2\left(1\right) + \cdots + \theta_{in}x_n\left(1\right) + v_i\left(1\right), \\ y_i\left(2\right) = \theta_{i1}x_1\left(2\right) + \theta_{i2}x_2\left(2\right) + \cdots + \theta_{in}x_n\left(2\right) + v_i\left(2\right), \\ \vdots \\ y_i\left(t\right) = \theta_{i1}x_1\left(t\right) + \theta_{i2}x_2\left(t\right) + \cdots + \theta_{in}x_n\left(t\right) + v_i\left(t\right). \end{array} \right.$$

Denote

$$\begin{array}{rcl} \varphi\left(t\right) & = & \left[\begin{array}{cccc} x_1\left(t\right) & x_2\left(t\right) & \cdots & x_n\left(t\right) \end{array} \right]^{\mathrm{T}}, \\ \theta_i & = & \left[\begin{array}{cccc} \theta_{i1} & \theta_{i2} & \cdots & \theta_{in} \end{array} \right]^{\mathrm{T}},\ i = 1,\ 2,\ \cdots,\ m. \end{array}$$

Then we have

$$\left\{ \begin{array}{l} y_i\left(1\right) = \varphi^{\mathrm{T}}\left(1\right)\theta_i + v_i\left(1\right), \\ y_i\left(2\right) = \varphi^{\mathrm{T}}\left(2\right)\theta_i + v_i\left(2\right), \\ \vdots \\ y_i\left(t\right) = \varphi^{\mathrm{T}}\left(t\right)\theta_i + v_i\left(t\right). \end{array} \right. \tag{2.3.10}$$

It is easily known that

$$\begin{array}{rcl} y^{\mathrm{T}}(t) & = & \left[\begin{array}{cccc} y_1(t) & y_2(t) & \cdots & y_m(t) \end{array} \right] \\ & = & \left[\begin{array}{cccc} \varphi^{\mathrm{T}}\left(t\right)\theta_1 + v_1\left(t\right) & \varphi^{\mathrm{T}}\left(t\right)\theta_2 + v_2\left(t\right) & \cdots & \varphi^{\mathrm{T}}\left(t\right)\theta_m + v_m\left(t\right) \end{array} \right] \\ & = & \left[\begin{array}{cccc} \varphi^{\mathrm{T}}\left(t\right)\theta_1 & \varphi^{\mathrm{T}}\left(t\right)\theta_2 & \cdots & \varphi^{\mathrm{T}}\left(t\right)\theta_2 \end{array} \right] + \left[\begin{array}{cccc} v_1\left(t\right) & v_2\left(t\right) & \cdots & v_m\left(t\right) \end{array} \right] \\ & = & \varphi^{\mathrm{T}}\left(t\right)\left[\begin{array}{cccc} \theta_1 & \theta_2 & \cdots & \theta_2 \end{array} \right] + \left[\begin{array}{cccc} v_1\left(t\right) & v_2\left(t\right) & \cdots & v_m\left(t\right) \end{array} \right]. \end{array}$$

Define the parameter matrix

$$\begin{aligned}\Theta &= \left[\begin{array}{cccc}\theta_1 & \theta_2 & \cdots & \theta_m\end{array}\right] \\ &= \left[\begin{array}{cccc}\theta_{11} & \theta_{21} & \cdots & \theta_{m1} \\ \theta_{12} & \theta_{22} & \cdots & \theta_{m2} \\ \cdots & \cdots & & \\ \theta_{1n} & \theta_{2n} & \cdots & \theta_{mn}\end{array}\right],\end{aligned}$$

and noise vector

$$v = \left[\begin{array}{cccc}v_1 & v_2 & \cdots & v_m\end{array}\right]^{\mathrm{T}}.$$

With this, one has

$$y^{\mathrm{T}}(t) = \varphi^{\mathrm{T}}(t)\,\Theta + v^{\mathrm{T}}(t).$$

Define

$$Y_t = \left[\begin{array}{c}y^{\mathrm{T}}(1) \\ y^{\mathrm{T}}(2) \\ \cdots \\ y^{\mathrm{T}}(t)\end{array}\right], H_t = \left[\begin{array}{c}\varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t)\end{array}\right], V_t = \left[\begin{array}{c}v^{\mathrm{T}}(1) \\ v^{\mathrm{T}}(2) \\ \cdots \\ v^{\mathrm{T}}(t)\end{array}\right].$$

Then the group of (2.3.10) can be rewritten as

$$\begin{aligned}Y_t &= \left[\begin{array}{c}\varphi^{\mathrm{T}}(1)\,\Theta \\ \varphi^{\mathrm{T}}(2)\,\Theta \\ \cdots \\ \varphi^{\mathrm{T}}(t)\,\Theta\end{array}\right] + \left[\begin{array}{c}v^{\mathrm{T}}(1) \\ v^{\mathrm{T}}(2) \\ \cdots \\ v^{\mathrm{T}}(t)\end{array}\right] \\ &= H_t\Theta + V_t.\end{aligned}$$

The error criterion function should be

$$\begin{aligned}J(\Theta) &= \sum_{k=1}^{n}\sum_{i=1}^{t}\left(y_k(i) - \varphi^{\mathrm{T}}(i)\,\theta_k\right)^2 \\ &= \sum_{k=1}^{n}\sum_{i=1}^{t} v_k^2(i) \\ &= \mathrm{tr}\left(V_t^{\mathrm{T}}V_t\right) \\ &= \mathrm{tr}\left[\left(Y_t - H_t\Theta\right)^{\mathrm{T}}\left(Y_t - H_t\Theta\right)\right].\end{aligned}$$

Please the student complete the detailed procedure for obtaining the estimate of Θ.

2.4 Orthonormal approaches for least square estimation

2.4.1 QR factorization approach

The estimate $\hat{\theta}_{LS}(t)$ can be obtained from the normal equation (2.3.8). The coefficient matrix $P(t) = H_t^{\mathrm{T}}H_t$ may be ill conditioned, in particular if its

dimension is high. There exists methods to find $\hat{\theta}_{LS}(t)$ that are much better numerically behaved, which do not have the normal equation as starting point. In this section, we shall here describe an efficient way using QR-**factorization**. The QR-**factorization** of an $n \times d$ matrix A is defined as

$$A = QR,\ QQ^{\mathrm{T}} = I,\ R \text{ upper triangular}$$

where Q is $n \times n$ and R is $n \times d$. The error criterion

$$J(\theta) = (Y_t - H_t\theta)^{\mathrm{T}} (Y_t - H_t\theta)$$

is obviously not affected by any orthonormal transformation applied to the vector $Y_t - H_t\theta$. If $Q \in \mathbb{R}^{t\times t}$ is orthonormal, that is, $QQ^{\mathrm{T}} = I$, then

$$\begin{aligned} J(\theta) &= (Y_t - H_t\theta)^{\mathrm{T}} (Y_t - H_t\theta) \\ &= \left[Q^{\mathrm{T}} (Y_t - H_t\theta)\right]^{\mathrm{T}} \left[Q^{\mathrm{T}} (Y_t - H_t\theta)\right]. \end{aligned}$$

Now, introduce the QR-factorization

$$\begin{bmatrix} H_t & Y_t \end{bmatrix} = QR,\ R = \begin{bmatrix} R_0 \\ 0 \end{bmatrix}.$$

Here R_0 is an upper triangular $(n+1) \times (n+1)$ matrix, which we decompose as

$$R_0 = \begin{bmatrix} R_1 & R_2 \\ 0 & R_3 \end{bmatrix},\ R_1 \in \mathbb{R}^{n\times n},\ R_2 \in \mathbb{R}^{n\times 1},\ R_3 \text{ is a scalar.}$$

This means that

$$\begin{aligned} J(\theta) &= \left[Q^{\mathrm{T}} (Y_t - H_t\theta)\right]^{\mathrm{T}} \left[Q^{\mathrm{T}} (Y_t - H_t\theta)\right] \\ &= \begin{bmatrix} R_2 - R_1\theta \\ R_3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} R_2 - R_1\theta \\ R_3 \end{bmatrix} \\ &= [R_2 - R_1\theta]^{\mathrm{T}} [R_2 - R_1\theta] + R_3^2, \end{aligned}$$

which clearly is minimized for

$$R_1\hat{\theta}_{LS}(t) = R_2, \tag{2.4.1}$$

giving $J\left(\hat{\theta}_{LS}(t)\right) = R_3^2$. There are three important advantages with this way of solving the least square estimation.

1. With $P(t) = H_t^{\mathrm{T}} H_t = R_1^{\mathrm{T}} R_1$, so the condition number of R_1 is the square root of $P(t)$. Therefore, it is numerically more reliable to obtain the estimate $\hat{\theta}_{LS}(t)$ from (2.4.1) than its counterpart (2.3.8).

2. R_1 is uppertriangular matrix, so the equation is easy to solve.

3. If QR factorization is performed for a regressor, the the solutions and error criterion function are easily obtained from R_0.

Note that the big matrix Q is never required to find $\hat{\theta}_{LS}(t)$ and the error criterion function. All information is contained in the "small" matrix R_0.

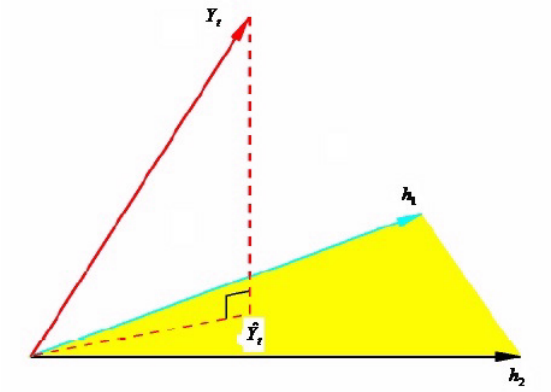

Figure 2.4.1: Geometry of least square estimation

2.4.2 Gram-Schmidt approach

Apart from the estimate $\hat{\theta}_{LS}(t)$, we further denote

$$\hat{Y}_t = H_t \hat{\theta}_{LS}(t) = H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^T Y_t.$$

In the previous section, it is shown that

$$\hat{H}_t = H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^T$$

is idempotent matrix. Such a fact shows that $\hat{Y}_t$ can be viewed as a projection of Y_t on the space spaned by the columns of matrix H_t. Figure 2.4.1 shows a different geometrical representation of the least squares estimation. This orthogonality is expressed in (2.3.7). The hat matrix Q computes the orthogonal projection, and hence it is also known as a projection matrix. In the end of this subsection, we construct the so-called Gram-Schmidt procedure to obtain the estimate $\hat{\theta}_{LS}(t)$ based on the projection explaintion.

Decompose H_t as

$$H_t = \left[\begin{array}{cccc} h_1 & h_2 & \cdots & h_n \end{array}\right].$$

The first step is to constructe a group of orthonormal basis $\{z_1, z_2, \cdots, z_n\}$ of the space spaned by the columns of H_t. First, one can choose $z_1 = h_1$. The z_2 is chosen to be orthogonal with z_1. In addition, z_2 can be linearly represented by z_1 and h_2. It is assumed that

$$z_2 = h_2 - \gamma_{12} z_1.$$

Since $z_2 \perp z_1$, one has

$$\left(h_2 - \gamma_{12} z_1\right)^{\mathrm{T}} z_1 = 0$$

which implies that

$$\gamma_{12} = \frac{h_2^{\mathrm{T}} z_1}{\|z_1\|^2}.$$

Now, it is assumed that

$$z_3 = h_3 - \gamma_{23} z_2 - \gamma_{13} z_1.$$

Since $z_3 \perp z_2$, $z_3 \perp z_1$, one has

$$\begin{aligned} \left(h_3 - \gamma_{23} z_2 - \gamma_{13} z_1\right)^{\mathrm{T}} z_2 &= 0, \\ \left(h_3 - \gamma_{23} z_2 - \gamma_{13} z_1\right)^{\mathrm{T}} z_1 &= 0. \end{aligned}$$

In view that $z_2 \perp z_1$, it follows from the previous expressions that

$$\gamma_{23} = \frac{h_3^{\mathrm{T}} z_2}{\|z_2\|^2},\ \gamma_{13} = \frac{h_3^{\mathrm{T}} z_1}{\|z_1\|^2}.$$

Similarly to this line, it is assumed that

$$z_i = h_i - \gamma_{i-1,i} z_{i-1} - \gamma_{i-2,i} z_{i-2} - \cdots - \gamma_{2i} z_2 - \gamma_{1i} z_1,\ i = 1, 2, \cdots, n.$$

In view of the orthogonality, it is easy to obtain

$$\gamma_{j,i} = \frac{h_i^{\mathrm{T}} z_j}{\|z_j\|^2},\ j = 1,\ 2,\ \cdots,\ i-1.$$

With the above relations, one has

$$\begin{cases} h_1 = z_1 \\ h_2 = \gamma_{12} z_1 + z_2 \\ h_3 = \gamma_{13} z_1 + \gamma_{23} z_2 + z_3 \\ \cdots \\ h_{n-1} = \gamma_{1,n-1} z_1 + \gamma_{2,n-1} z_2 + \gamma_{3,n-1} z_3 + \cdots + \gamma_{n-2,n-1} z_{n-2} + z_{n-1} \\ h_n = \gamma_{1n} z_1 + \gamma_{2n} z_2 + \gamma_{3n} z_3 + \cdots + \gamma_{n-1,n} z_{n-1} + z_n \end{cases},$$

which can be rewritten in the following matrix format

$$\begin{aligned} & \begin{bmatrix} h_1 & h_2 & \cdots & h_{n-1} & h_n \end{bmatrix} \\ = & \begin{bmatrix} z_1 & z_2 & \cdots & z_{n-1} & z_n \end{bmatrix} \begin{bmatrix} 1 & \gamma_{12} & & \gamma_{1,n-1} & \gamma_{1n} \\ & 1 & & \gamma_{2,n-1} & \gamma_{2n} \\ & & \ddots & & \\ & & & 1 & \gamma_{n-1,n} \\ & & & & 1 \end{bmatrix}. \end{aligned}$$

Denote

$$\begin{aligned} Z &= \begin{bmatrix} z_1 & z_2 & \cdots & z_{n-1} & z_n \end{bmatrix}, \\ \Gamma &= \begin{bmatrix} 1 & \gamma_{12} & & \gamma_{1,n-1} & \gamma_{1n} \\ & 1 & & \gamma_{2,n-1} & \gamma_{2n} \\ & & \ddots & & \\ & & & 1 & \gamma_{n-1,n} \\ & & & & 1 \end{bmatrix}. \end{aligned}$$

In addition, we introduce the following dianogal matrix

$$D = \mathrm{diag}\left(\|z_1\|,\ \|z_2\|,\ \cdots,\ \|z_n\|\right).$$

Then, we have

$$H_t = Z\Gamma = ZD^{-1}D\Gamma.$$

Let

$$Q = ZD^{-1},\ R = D\Gamma.$$

It is easily checked that Q is a orthomormal matrix which satisfies $Q^{\mathrm{T}}Q = I$, and R is an upper triangular matrix. In this case, one has

$$H_t^{\mathrm{T}}H_t = (QR)^{\mathrm{T}}QR = R^{\mathrm{T}}R.$$

It follows from the normal equation that

$$R^{\mathrm{T}}R\hat{\theta}(t) = R^{\mathrm{T}}QY_t,$$

which implies that

$$R\hat{\theta}(t) = QY_t.$$

This equation with restpect to $\hat{\theta}(t)$ is easily solved since R is an upper triangular matrix.

2.5 Statistical properties for least squares estimation

Regarding the LS estimate $\hat{\theta}_{LS}$ given by the off-line algorithm (2.3.9), we have some properties.

Theorem 2.5.1 *Consider the system (2.3.1), and suppose that V_t is of mean zero, and V_t and H_t are statistically indepent. Then the least squares estimate $\hat{\theta}_{LS}$ in (2.3.9) is an unbiased estimate, that is,*

$$\mathrm{E}\left[\hat{\theta}_{LS}\right] = \theta.$$

Proof. The proof is very simple. By using (2.3.5), one has

$$\begin{aligned}
\mathrm{E}\left[\hat{\theta}_{LS}\right] &= \mathrm{E}\left[\left(H_t^T H_t\right)^{-1} H_t^T Y_t\right] \\
&= \mathrm{E}\left[\left(H_t^T H_t\right)^{-1} H_t^T \left(H_t\theta + V_t\right)\right] \\
&= \mathrm{E}\left[\theta + \left(H_t^T H_t\right)^{-1} H_t^T V_t\right] \\
&= \theta + \mathrm{E}\left[\left(H_t^T H_t\right)^{-1} H_t^T V_t\right] = \theta.
\end{aligned}$$

■

Theorem 2.5.2 *(Covariance theorem for estimation error) Consider the system (2.3.1), and suppose that V_t is of mean zero, and has the covariance matrix $\mathrm{cov}[V_t] = R_v$. In addition, it is assumed that V_t and H_t are statistically independent. Then the covariance matrix of the estimation error $\tilde{\theta}_{LS}(t) := \hat{\theta}_{LS}(t) - \theta$ is given by*

$$\mathrm{cov}\left[\tilde{\theta}_{LS}(t)\right] = \mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} R_v H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1}\right].$$

Proof. It follows from the proof of the previous theorem that

$$\tilde{\theta}_{LS}(t) = \hat{\theta}_{LS}(t) - \theta = \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} V_t.$$

With this, one has

$$\begin{aligned}
\mathrm{cov}\left[\tilde{\theta}_{LS}(t)\right] &= \mathrm{E}\left[\tilde{\theta}_{LS}(t)\,\tilde{\theta}_{LS}^{\mathrm{T}}(t)\right] \\
&= \mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} V_t V_t^{\mathrm{T}} H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1}\right] \\
&= \mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} R_v H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1}\right].
\end{aligned}$$

■

If $\{v(t)\}$ is white series with zero mean, and its variance is σ^2. That is,

$$\mathrm{E}[v(t)] = 0,\ \mathrm{E}\left[v^2(t)\right] = \sigma^2.$$

In this case, one has

$$\mathrm{cov}[V_t] = R_v = \sigma^2 I_t.$$

With this, the covariance matrix of the estimation error can be given by

$$\begin{aligned}
\mathrm{cov}\left[\tilde{\theta}_{LS}(t)\right] &= \mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} \sigma^2 H_t \left(H_t^{\mathrm{T}} H_t\right)^{-1}\right] \\
&= \sigma^2\,\mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1}\right].
\end{aligned}$$

By taking trace in both sides of this relation, one has

$$\begin{aligned}
\mathrm{E}\left[\left\|\hat{\theta}_{LS}(t) - \theta\right\|^2\right] &= \mathrm{tr}\left\{\mathrm{cov}\left[\tilde{\theta}_{LS}(t)\right]\right\} \\
&= \sigma^2\,\mathrm{tr}\left\{\mathrm{E}\left[\left(H_t^{\mathrm{T}} H_t\right)^{-1}\right]\right\}.
\end{aligned}$$

If

$$\left(H_t^T H_t\right) \geq C I_n \ln\ln t,$$

then

$$\begin{aligned}
\lim_{t\to\infty} \mathrm{E}\left[\left\|\hat{\theta}_{LS}(t) - \theta\right\|^2\right] &= \lim_{t\to\infty} \sigma^2\,\mathrm{tr}\left\{\mathrm{E}\left[\left(H_t^T H_t\right)^{-1}\right]\right\} \\
&\leq \lim_{t\to\infty} \sigma^2\,\mathrm{tr}\left\{\mathrm{E}\left[\left(C I_n \ln\ln t\right)^{-1}\right]\right\} \\
&= \lim_{t\to\infty} \frac{n\sigma^2}{C\ln\ln t} = 0.
\end{aligned}$$

The following theorem provides an estimate of the variance of the white noise $v(t)$.

Theorem 2.5.3 *Consider the system (2.3.1), and suppose that*

$$\begin{cases} \mathrm{E}\left[v\left(t\right)\right]=0, & \mathrm{E}\left[v\left(t\right)v\left(i\right)\right]=0,\ t\neq i, \\ \mathrm{E}\left[v^{2}\left(t\right)\right]=\sigma^{2}, & \mathrm{cov}\left[V_{t}\right]=\sigma^{2}I_{t}, \end{cases}$$

and V_t and H_t is statastically indepent. The value of the error criterion function (2.3.6) at

$$\theta=\hat{\theta}_{LS}\left(t\right)=\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}Y_{t} \tag{2.5.1}$$

given in (2.3.9) is

$$J\left(\hat{\theta}_{LS}\left(t\right)\right)=\left[Y_{t}-H_{t}\hat{\theta}_{LS}\left(t\right)\right]^{\mathrm{T}}\left[Y_{t}-H_{t}\hat{\theta}_{LS}\left(t\right)\right].$$

Then, an estimate $\hat{\sigma}^2$ of the variance σ^2 of the white noise $v(t)$ can be given by

$$\hat{\sigma}^{2}=\frac{J\left(\hat{\theta}_{LS}\left(t\right)\right)}{t-n},\ t \text{ large enough},$$

where $n=\dim\theta$.

Proof. Define output residual $\varepsilon\left(t\right)$ and residual vector ε_t, respectively, as

$$\varepsilon\left(i\right):=y\left(i\right)-\varphi^{\mathrm{T}}\left(i\right)\hat{\theta}_{LS}\left(t\right),\ i=1,\ 2,\ \cdots,\ t,$$

$$\varepsilon_{t}:=\begin{bmatrix}\varepsilon\left(1\right)\\ \varepsilon\left(2\right)\\ \cdots\\ \varepsilon\left(t\right)\end{bmatrix}=Y_{t}-H_{t}\hat{\theta}_{LS}\left(t\right).$$

Substituting (2.5.1) into the expression of ε_t, gives by applying (2.3.5)

$$\begin{aligned}\varepsilon_{t} &= Y_{t}-H_{t}\hat{\theta}_{LS}\left(t\right)\\ &= Y_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}Y_{t}\\ &= \left[I_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\right]Y_{t}\\ &= \left[I_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\right]\left(H_{t}\theta+V_{t}\right)\\ &= \left[I_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\right]V_{t}\\ &= :QV_{t}.\end{aligned}$$

By simple calculation, one has

$$\begin{aligned}Q^{2} &= \left[I_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\right]^{2}\\ &= I_{t}-2H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}+H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\\ &= I_{t}-2H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}+H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}}\\ &= I_{t}-H_{t}\left(H_{t}^{\mathrm{T}}H_{t}\right)^{-1}H_{t}^{\mathrm{T}},\end{aligned}$$

and

$$Q^{\mathrm{T}} = Q.$$

These two facts imply that Q is an idempotent matrix. With this, by applying

$$\begin{aligned} \mathrm{E}\left\{\mathrm{tr}\left[A\right]\right\} &= \mathrm{tr}\left\{\mathrm{E}\left[A\right]\right\}, \\ \mathrm{tr}\left(AB\right) &= \mathrm{tr}\left(BA\right), \\ \mathrm{tr}\, A &= \mathrm{tr}\, A^{\mathrm{T}}, \end{aligned}$$

one has

$$\begin{aligned} \mathrm{E}\left[\varepsilon_t^{\mathrm{T}}\varepsilon_t\right] &= \mathrm{E}\left[V_t^{\mathrm{T}}Q^{\mathrm{T}}QV_t\right] = \mathrm{E}\left[V_t^{\mathrm{T}}QV_t\right] \\ &= \mathrm{E}\left[\mathrm{tr}\left(V_t^{\mathrm{T}}QV_t\right)\right] \\ &= \mathrm{E}\left[\mathrm{tr}\left(QV_tV_t^{\mathrm{T}}\right)\right] \\ &= \mathrm{E}\left[\mathrm{tr}\left(QV_t^{\mathrm{T}}V_t\right)\right] \\ &= \sigma^2\,\mathrm{tr}\left[\mathrm{E}\left(Q\right)\right] \\ &= \sigma^2\,\mathrm{E}\left[\mathrm{tr}\left(Q\right)\right] \\ &= \sigma^2\,\mathrm{E}\left[\mathrm{tr}\left(I_t - H_t\left(H_t^{\mathrm{T}}H_t\right)^{-1}H_t^{\mathrm{T}}\right)\right] \\ &= \sigma^2\left(t - \dim\theta\right). \end{aligned}$$

Therefore,

$$\sigma^2 = \frac{\mathrm{E}\left[\varepsilon_t^{\mathrm{T}}\varepsilon_t\right]}{t-\dim\theta} = \frac{\mathrm{E}\left[J\left(\hat{\theta}_{LS}\left(t\right)\right)\right]}{t-\dim\theta}.$$

This implies that an unbiased estimate of σ^2 can be given by

$$\hat{\sigma}^2 = \frac{J\left(\hat{\theta}_{LS}\left(t\right)\right)}{t-\dim\theta}.$$

■

Least square principle can be used to investigate the aforementioned parameter identification for linear-parameter space. Some nonlinear systems can also be identification by using this principle.

Two examples:

1. A nonlinear system:

$$y(t) = a_1 y^2\left(t-1\right) + a_2 y(t-1)u(t) + e^{b_1}u(t) + b_2 u(t-1)u(t-2) + v(t)$$

2. The following typical nonlinear system

$$\begin{aligned} y\left(t\right) &= \theta_1 f_1\left(x_1\left(t\right), x_2\left(t\right), \cdots, x_n\left(t\right)\right) + \theta_2 f_2\left(x_1\left(t\right), x_2\left(t\right), \cdots, x_n\left(t\right)\right) \\ &\quad + \cdots + \theta_n f_n\left(x_1\left(t\right), x_2\left(t\right), \cdots, x_n\left(t\right)\right) + v(t). \end{aligned}$$

Chapter 3

Recursive least squares identification

3.1 Recursive least squares method

In many cases it is necessary, or useful, to have a model of the system available on-line while the system is in operation. The model should then be based on observations up to the current time.

The on-line computation of the model must also be done in such a way that the processing of the measurements from one sample can, with certainty, be completed during one sampling interval. Otherwise the model building cannot keep up with information flow.

Identification techniques that comply with this requirement will here be called **recursive identification methods** since the measured input-output data are processed recursively (sequentially) as they become available. Other commonly used terms for such techniques are on-line or real-time identification.

Firstly, we consider the Controlled AutoRegressive (CAR, or ARX) model shown in Figure 3.1.1

$$A(z)\,y(t) = B(z)\,u(t) + v(t) \tag{3.1.1}$$

where $A(z)$ and $B(z)$ are given, respectively, by

$$\begin{aligned} A(z) &: = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}, \\ B(z) &: = 1 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}. \end{aligned}$$

Denote $n := n_a + n_b$.

Define parameter vector

$$\theta := \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^n$$

and information vector

$$\varphi(t) := \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^n.$$

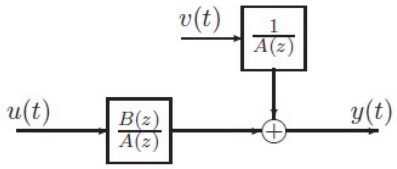

Figure 3.1.1: The systems described by CAR models

Adding some procedure here.??????????????The system (3.1.1) can be written the following identification representation

$$y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t)\,. \tag{3.1.2}$$

The structures such as $y(t) = \varphi^{\mathrm{T}}(t)\,\theta$ that are linear in θ are known in statistics as **linear regressions**. The vector $\varphi(t)$ is called the **regression vector**, and its components are the **regressor**s. "Regress" here alludes to the fact that we try to calculate (or describe) $y(t)$ by "going back" to $\varphi(t)$.

When $t = 1,\ 2,\ \cdots,\ t$, one can obtain from (3.1.2)

$$\begin{cases} y(1) = \varphi^{\mathrm{T}}(1)\,\theta + v(1)\,, \\ y(2) = \varphi^{\mathrm{T}}(2)\,\theta + v(2)\,, \\ \qquad \cdots \\ y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t)\,. \end{cases}$$

These equations can be rewritten in the following matrix form

$$\begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t) \end{bmatrix} = \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \theta + \begin{bmatrix} v(1) \\ v(2) \\ \cdots \\ v(t) \end{bmatrix}.$$

Denote

$$Y_t := \begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t) \end{bmatrix} \in \mathbb{R}^t,\ H_t := \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times n},\ V_t := \begin{bmatrix} v(1) \\ v(2) \\ \cdots \\ v(t) \end{bmatrix} \in \mathbb{R}^t.$$

(2.3.3) becomes the following simple form

$$Y_t = H_t\theta + V_t. \tag{3.1.3}$$

According to least square identification principle, one can define quadratic criterion function:

$$J(\theta) = V_t^{\mathrm{T}}V_t = (Y_t - H_t\theta)^{\mathrm{T}}(Y_t - H_t\theta)\,. \tag{3.1.4}$$

Similarly to the procedure, one can obtain the least squares estimate of the parameter vector θ as

$$\hat{\theta}_{LS}(t) = \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} Y_t. \tag{3.1.5}$$

In a recursive algorithm, $\hat{\theta}_{LS}(t)$ is denoted by $\hat{\theta}(t)$. Substituting the definition of H_t and Y_t into (3.1.5), gives

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}_{LS}(t) = \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} Y_t \\ &= \left[\sum_{i=1}^{t} \varphi(i)\,\varphi^{\mathrm{T}}(i)\right]^{-1} \left[\sum_{i=1}^{t} \varphi(i)\, y(i)\right]. \end{aligned} \tag{3.1.6}$$

For the CAR model (3.1.1), the error criterion function can be written as

$$J(\theta) = V_t^{\mathrm{T}} V_t = \sum_{i=1}^{t} v^2(i) = \sum_{i=1}^{t} \left[A(z)\, y(i) - B(z)\, u(i)\right]^2. \tag{3.1.7}$$

(Adding the detailed procedure to obtain the above relation.)

Before giving the recursive algorithm, we firstly provide a lemma on matrix inversion.

Lemma 3.1.1 *Given $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r}$ and $C \in \mathbb{R}^{r\times n}$, if the matrices A and $\left(I + CA^{-1}B\right)$ are both invertible, then the following relation holds*

$$(A + BC)^{-1} = A^{-1} - A^{-1}B\left(I + CA^{-1}B\right)^{-1} CA^{-1}.$$

Define matrix

$$\begin{aligned} P^{-1}(t) &:= H_t^{\mathrm{T}} H_t = \sum_{i=1}^{t} \varphi(i)\,\varphi^{\mathrm{T}}(i) \\ &= \sum_{i=1}^{t-1} \varphi(i)\,\varphi^{\mathrm{T}}(i) + \varphi(t)\,\varphi^{\mathrm{T}}(t), \end{aligned} \tag{3.1.8}$$

which implies the following recursive relation

$$P^{-1}(t) = P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t),\ P(0) = p_0 I > 0. \tag{3.1.9}$$

It follows from this recursive relation that

$$P^{-1}(t) = P^{-1}(0) + \sum_{i=1}^{t} \varphi(i)\,\varphi^{\mathrm{T}}(i). \tag{3.1.10}$$

Comparing this relation with (3.1.8), one knows that the intial value $P^{-1}(0)$ should be 0. According to the definitions of Y_t and H_t, one has

$$Y_t = \begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t-1) \\ y(t) \end{bmatrix} = \begin{bmatrix} Y_{t-1} \\ y(t) \end{bmatrix} \in \mathbb{R}^t,\ H_t = \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t-1) \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} = \begin{bmatrix} H_{t-1} \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times n}.$$

By applying (3.1.9), it follows from (3.1.6) that

$$\begin{aligned}
\hat{\theta}(t) &= \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} Y_t = P(t) H_t^{\mathrm{T}} Y_t \\
&= P(t) \begin{bmatrix} H_{t-1} \\ \varphi^{\mathrm{T}}(t) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Y_{t-1} \\ y(t) \end{bmatrix} \\
&= P(t) \begin{bmatrix} H_{t-1}^{\mathrm{T}} & \varphi(t) \end{bmatrix} \begin{bmatrix} Y_{t-1} \\ y(t) \end{bmatrix} \\
&= P(t) \left(H_{t-1}^{\mathrm{T}} Y_{t-1} + \varphi(t) y(t)\right) \\
&= P(t) \left(P^{-1}(t-1) P(t-1) H_{t-1}^{\mathrm{T}} Y_{t-1} + \varphi(t) y(t)\right) \\
&= P(t) \left[P^{-1}(t-1) \hat{\theta}(t-1) + \varphi(t) y(t)\right] \\
&= P(t) \left[P^{-1}(t) - \varphi(t) \varphi^{\mathrm{T}}(t)\right] \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) - P(t) \varphi(t) \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) + P(t) \varphi(t) \left[y(t) - \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1)\right].
\end{aligned}$$

Applying Lemma 3.1.1 to (3.1.9), gives

$$P(t) = P(t-1) - \frac{P(t-1) \varphi(t) \varphi^{\mathrm{T}}(t) P(t-1)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)}. \tag{3.1.11}$$

Post-multiplying both sides of the above relation by $\varphi(t)$ yields

$$\begin{aligned}
P(t) \varphi(t) &= P(t-1) \varphi(t) - \frac{P(t-1) \varphi(t) \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)} \\
&= P(t-1) \varphi(t) \left[1 - \frac{\varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)}\right] \\
&= \frac{P(t-1) \varphi(t)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)}.
\end{aligned}$$

Denote

$$L(t) := P(t) \varphi(t).$$

Then, one has

$$\begin{aligned}
P(t) &= P(t-1) - \frac{P(t-1) \varphi(t) \varphi^{\mathrm{T}}(t) P(t-1)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)} \\
&= P(t-1) - \frac{P(t-1) \varphi(t)}{1 + \varphi^{\mathrm{T}}(t) P(t-1) \varphi(t)} \varphi^{\mathrm{T}}(t) P(t-1) \\
&= P(t-1) - L(t) \varphi^{\mathrm{T}}(t) P(t-1) \\
&= \left[I - L(t) \varphi^{\mathrm{T}}(t)\right] P(t-1) \\
&= P(t-1) - P(t-1) \varphi(t) L^{\mathrm{T}}(t) \\
&= P(t-1) \left[1 - \varphi(t) L^{\mathrm{T}}(t)\right].
\end{aligned}$$

We can obtain the following **recursive least squares algorithm for CAR** model (3.1.1):

Algorithm CAR-RLS1

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
P^{-1}(t) &= P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t),\ P(0) = p_0 I, \\
\varphi(t) &= \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\mathrm{T}}.
\end{aligned}$$

In order to avoid the inversion of the **covariance matrix** $P(t)$, the **CAR-RLS** algorithm can be written as

Algorithm CAR-RLS2

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
P(t) &= P(t-1) - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{1+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)},\ P(0) = p_0 I, \\
\varphi(t) &= \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\mathrm{T}}.
\end{aligned}$$

With the help of the vector $L(t) = P(t)\varphi(t)$, one can obtain the following **CAR-RLS** algorithm:

Algorithm CAR-RLS3

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right] \\
L(t) &= \frac{P(t-1)\varphi(t)}{1+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)} \\
P(t) &= \left[I - L(t)\varphi^{\mathrm{T}}(t)\right]P(t-1),\ P(0) = p_0 I, \\
\varphi(t) &= \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\mathrm{T}}.
\end{aligned}$$

The order to carry out the Algorithm CAR-RLS3:

$$P(0) \to L(1) \to \hat{\theta}(1) \to P(1) \to L(2) \to \hat{\theta}(2) \to \cdots.$$

In the above algorithms, $L(t)$ is called gain vector, $P(t)$ is called covariance matrix, and

$$e(t) := y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)$$

is prediction error according to the current model, and is called **innovation**. In addition, we define the residual

$$\varepsilon(t) := y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t).$$

Remark 3.1.1 Regarding the residual and innovation, the following relations hold

$$\begin{aligned}
e(t) &= \left[1 + \varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)\right]\varepsilon(t), \\
\varepsilon(t) &= \left[1 - \varphi^{\mathrm{T}}(t)P(t)\varphi(t)\right]e(t).
\end{aligned}$$

In fact,

$$\begin{aligned}
\varepsilon(t) &= y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t) \\
&= y(t)-\varphi^{\mathrm{T}}(t)\left[\hat{\theta}(t-1)+L(t)\,e(t)\right] \\
&= y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)-\varphi^{\mathrm{T}}(t)\,L(t)\,e(t) \\
&= e(t)-\varphi^{\mathrm{T}}(t)\,L(t)\,e(t) \\
&= \left[1-\varphi^{\mathrm{T}}(t)\,L(t)\right]e(t) \\
&= \left[1-\varphi^{\mathrm{T}}(t)\,P(t)\,\varphi(t)\right]e(t) \\
&= \left[1-\varphi^{\mathrm{T}}(t)\,\frac{P(t-1)\,\varphi(t)}{1+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}\right]e(t) \\
&= \frac{e(t)}{1+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}.
\end{aligned}$$

Remark 3.1.2 When the initial value $P(0)$ takes $p_0 I > 0$, it follows from (3.1.10) that

$$\begin{aligned}
P(t) &= \left[\frac{1}{p_0}I+\sum_{i=1}^{t}\varphi(i)\,\varphi^{\mathrm{T}}(i)\right]^{-1} \\
&= \left[\frac{1}{p_0}I+H_t^{\mathrm{T}}H_t\right]^{-1}.
\end{aligned}$$

This implies that

$$\lim_{p_0\to\infty}P(t)=\lim_{p_0\to\infty}\left[\frac{1}{p_0}I+H_t^{\mathrm{T}}H_t\right]^{-1}=\lim_{p_0\to\infty}\left[H_t^{\mathrm{T}}H_t\right]^{-1}=P(0).$$

This fact shows that the p_0 should be chosen to be as large as possible.

3.2 Forgetting factor recursive least squares algorithm

3.2.1 Data saturation

Consider the previous equation error model

$$A(z)y(t)=B(z)u(t)+v(t). \tag{3.2.1}$$

Adding some details. ???

For the recursive least squares algorithm in the previous section, it follows from (3.1.11) one has

$$P(t)=P(t-1)-\frac{P(t-1)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{1+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}.$$

This implies that

$$P(t-1)-P(t)=P\left(t-1\right)\frac{\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right)}{1+\varphi^{\mathrm{T}}\left(t\right)P\left(t-1\right)\varphi\left(t\right)}P\left(t-1\right). \tag{3.2.2}$$

In general, the initial value $P(0)$ is chosen to be a positive definite matrix, so it follows from the relation

$$P\left(t\right)=\left[P^{-1}\left(t-1\right)+\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right)\right]^{-1}$$

that all the matrices $P(1)$, $P(2)$, $\cdots$, $P\left(t\right)$ are positive definite. Combining this fact with (3.2.2), gives

$$P(t)\leq P(t-1).$$

In addition, in order to identify the model, the observed data should satisfy persistent excitation assumption

$$\alpha I\leq\frac{1}{t}\sum_{i=1}^{t}\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\leq\beta I,\text{ a. s.}$$

With this condition and the relation (3.1.10), one has

$$\left(\alpha t+\frac{1}{p_0}\right)I\leq P^{-1}(t)=\sum_{i=1}^{t}\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)+P^{-1}\left(0\right)\leq\left(\beta t+\frac{1}{p_0}\right)I,\text{ a. s.}$$

This implies that $\lim_{t\to\infty}P\left(t\right)=0$. It follows from this fact that the modified term $P\left(t\right)\varphi\left(t\right)\left[y(t)-\varphi^{\mathrm{T}}\left(t\right)\hat{\theta}(t-1)\right]\to0$ when $t\to0$. That is to say, the new data have no contribution to improve the estimate $\hat{\theta}(t)$ of the parameter θ. Such a phenomenon is called **data saturation**.

3.2.2 Forgetting factor recursive least squares algorithm

In order to increase the contribution of new data and to reduce the influence of old data, one way is to introduce a factor for old data. With this idea, the previous matrices H_t and Y_t are modified as

$$Y_t=\left[\begin{array}{c}\rho Y_{t-1}\\ y(t)\end{array}\right]\in\mathbb{R}^t,H_t=\left[\begin{array}{c}\rho H_{t-1}\\ \varphi^{\mathrm{T}}\left(t\right)\end{array}\right].$$

In this case, the estimate $\hat{\theta}(t)$ can be given

$$\begin{aligned}
\hat{\theta}(t) &= \left(H_t^{\mathrm{T}} H_t\right)^{-1} H_t^{\mathrm{T}} Y_t = P(t) H_t^{\mathrm{T}} Y_t \\
&= P(t) \begin{bmatrix} \rho H_{t-1} \\ \varphi^{\mathrm{T}}(t) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \rho Y_{t-1} \\ y(t) \end{bmatrix} \\
&= P(t) \begin{bmatrix} \rho H_{t-1}^{\mathrm{T}} & \varphi(t) \end{bmatrix} \begin{bmatrix} \rho Y_{t-1} \\ y(t) \end{bmatrix} \\
&= P(t) \left(\rho^2 H_{t-1}^{\mathrm{T}} Y_{t-1} + \varphi(t) y(t)\right) \\
&= P(t) \left(P^{-1}(t-1) P(t-1) \rho^2 H_{t-1}^{\mathrm{T}} Y_{t-1} + \varphi(t) y(t)\right) \\
&= P(t) \left[P^{-1}(t-1) \rho^2 \hat{\theta}(t-1) + \varphi(t) y(t)\right]. \\
&= P(t) \left[P^{-1}(t) - \varphi(t) \varphi^{\mathrm{T}}(t)\right] \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) - P(t) \varphi(t) \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) + P(t) \varphi(t) \left[y(t) - \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1)\right].
\end{aligned} \tag{3.2.3}$$

In addition,

$$\begin{aligned}
P^{-1}(t) &= H_t^{\mathrm{T}} H_t \\
&= \begin{bmatrix} \rho H_{t-1} \\ \varphi^{\mathrm{T}}(t) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \rho H_{t-1} \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \\
&= \rho^2 H_{t-1}^{\mathrm{T}} H_{t-1} + \varphi(t) \varphi^{\mathrm{T}}(t) \\
&= \rho^2 P^{-1}(t-1) + \varphi(t) \varphi^{\mathrm{T}}(t).
\end{aligned} \tag{3.2.4}$$

Combining (3.2.4) with (3.2.3), gives

$$\begin{aligned}
\hat{\theta}(t) &= P(t) \left[P^{-1}(t-1) \rho^2 \hat{\theta}(t-1) + \varphi(t) y(t)\right]. \\
&= P(t) \left[P^{-1}(t) - \varphi(t) \varphi^{\mathrm{T}}(t)\right] \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) - P(t) \varphi(t) \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1) + P(t) \varphi(t) y(t) \\
&= \hat{\theta}(t-1) + P(t) \varphi(t) \left[y(t) - \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1)\right].
\end{aligned}$$

Denote $\lambda := \rho^2$. Now we have obtain the following forgetting factor recursive least squares (FF-RLS) algorithm.

FF-RLS1

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t) \varphi(t) \left[y(t) - \varphi^{\mathrm{T}}(t) \hat{\theta}(t-1)\right] \\
P^{-1}(t) &= \lambda P^{-1}(t-1) + \varphi(t) \varphi^{\mathrm{T}}(t).
\end{aligned}$$

It is convenient to analyze convergence properties by using FF-RLS1. Similarly to the derivation in the previous section, we use Lemma 3.1.1 to avoid the inverse

of the covariance matrix $P(t)$. By using Lemma 3.1.1, one has

$$\begin{aligned} P(t) &= \left[\lambda P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t)\right]^{-1} \\ &= \frac{1}{\lambda}P(t-1) - \frac{1}{\lambda}P(t-1)\frac{\varphi(t)}{1+\varphi^{\mathrm{T}}(t)\frac{1}{\lambda}P(t-1)\varphi(t)}\varphi^{\mathrm{T}}(t)\frac{1}{\lambda}P(t-1) \\ &= \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right]. \end{aligned} \tag{3.2.5}$$

Denote

$$L(t) = P(t)\varphi(t).$$

Then one has

$$\begin{aligned} L(t) &= P(t)\varphi(t) \\ &= \frac{1}{\lambda}\left[P(t-1)\varphi(t) - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right] \\ &= \frac{1}{\lambda}P(t-1)\varphi(t)\left[1 - \frac{\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right] \\ &= \frac{1}{\lambda}P(t-1)\varphi(t)\frac{\lambda}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)} \\ &= \frac{P(t-1)\varphi(t)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}. \end{aligned}$$

With this relation, from (3.2.5) one has

$$\begin{aligned} P(t) &= \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right] \\ &= \frac{1}{\lambda}\left[I - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right]P(t-1) \\ &= \frac{1}{\lambda}\left[I - L(t)\varphi^{\mathrm{T}}(t)\right]P(t-1). \end{aligned}$$

Organzing the above results, we have obtained the following FF-RLS algorithm which can be used to practical identification.

FF-RLS2

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right] \\ L(t) &= \frac{P(t-1)\varphi(t)}{\lambda+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)} \\ P(t) &= \frac{1}{\lambda}\left[I - L(t)\varphi^{\mathrm{T}}(t)\right]P(t-1) \end{aligned}$$

The smaller λ is, the smaller contribution of previous samples. This makes the estimator more sensitive to recent samples, which means more fluctuations in the estimator coefficients. The $\lambda = 1$ case is referred to as the **growing window RLS** algorithm.

At the end of this section, we establish a relation between the innovation and residual in the previous algorithms.

Lemma 3.2.1 *For the Algorithm FF-RLS2, define the innovation as*

$$e(t) = y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1),$$

and the residual as

$$\varepsilon(t) = y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t).$$

Then the following relations hold

$$e(t) = \left[1 + \frac{1}{\lambda}\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)\right] e(t).$$

Proof. According the algorithm, one has

$$\begin{aligned}
\varepsilon(t) &= y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t) \\
&= y(t) - \varphi^{\mathrm{T}}(t)\left[\hat{\theta}(t-1) + L(t)\,e(t)\right] \\
&= y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) - \varphi^{\mathrm{T}}(t)\,L(t)\,e(t) \\
&= \left[1 - \varphi^{\mathrm{T}}(t)\,L(t)\right] e(t) \\
&= \left[1 - \frac{\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}{\lambda + \varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}\right] e(t) \\
&= \frac{\lambda}{\lambda + \varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}\, e(t).
\end{aligned}$$

This implies the conclusion. ■

3.3 Fixed memory identification

First, we consider the following linear regression model

$$y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t), \tag{3.3.1}$$

where $v(t)$ is a white noise. In this section, we consider another recursive identification method for this linear regression model. In this method, only a finite data set $\{y(i),\, u(i)\colon\, i = t-p+1,\, t-p+2,\, \cdots,\, t\}$, which is the most recent data with data length p, is used. That is to, an old observation should be deleted when a new observation is added to the identification procedure. In this case, we choose the following error criterion function

$$J(\theta) = \sum_{i=t-p+1}^{t} \left(y(i) - \varphi^{\mathrm{T}}(i)\,\theta\right)^2. \tag{3.3.2}$$

Denote

$$Y(p,t) \quad : \quad = \begin{bmatrix} y(t-p+1) \\ y(t-p+2) \\ \cdots \\ y(t-1) \\ y(t) \end{bmatrix} \in \mathbb{R}^t, \; H(p,t) := \begin{bmatrix} \varphi^{\mathrm{T}}(t-p+1) \\ \varphi^{\mathrm{T}}(t-p+2) \\ \cdots \\ \varphi^{\mathrm{T}}(t-1) \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times n},$$

$$V(p,t) \quad : \quad = \begin{bmatrix} v(t-p+1) \\ v(t-p+2) \\ \cdots \\ v(t-1) \\ v(t) \end{bmatrix} \in \mathbb{R}^t,$$

and

$$P(t) = H^{\mathrm{T}}(p,t)\, H(p,t).$$

Minimizing the error criterion function (3.3.2), one can obtain the following estimate $\hat{\theta}(t)$ of θ

$$\begin{aligned} \hat{\theta}(t) &= [P(t)]^{-1} H^{\mathrm{T}}(p,t)\, Y(p,t) \\ &= \left[H^{\mathrm{T}}(p,t)\, H(p,t)\right]^{-1} H^{\mathrm{T}}(p,t)\, Y(p,t). \end{aligned} \tag{3.3.3}$$

3.3.1 The first type of recursive algorithms

In view of the expression of $Y(p,t)$ and $H(p,t)$, one has

$$\begin{aligned} \hat{\theta}(t) &= \left[H^{\mathrm{T}}(p,t)\, H(p,t)\right]^{-1} H^{\mathrm{T}}(p,t)\, Y(p,t) \\ &= \left[\sum_{i=t-p+1}^{t} \varphi(i)\,\varphi^{\mathrm{T}}(i)\right]^{-1} \left[\sum_{i=t-p+1}^{t} \varphi(i)\, y(i)\right] \\ &= \left[\sum_{i=t-p+1}^{t-1} \varphi(i)\,\varphi^{\mathrm{T}}(i) + \varphi(t)\,\varphi^{\mathrm{T}}(t)\right]^{-1} \left[\sum_{i=t-p+1}^{t-1} \varphi(i)\, y(i) + \varphi(t)\, y(t)\right]. \end{aligned} \tag{3.3.4}$$

This implies that

$$\begin{aligned}
& \left[\sum_{i=t-p+1}^{t-1}\varphi(i)\,\varphi^{\mathrm{T}}(i)+\varphi(t)\,\varphi^{\mathrm{T}}(t)\right]\hat{\theta}(t)\\
= & \left[\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+\sum_{i=t-p+1}^{t-1}\varphi(i)\,\varphi^{\mathrm{T}}(i)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]\hat{\theta}(t)\\
= & \left[\sum_{i=(t-1)-p+1}^{t-1}\varphi(i)\,\varphi^{\mathrm{T}}(i)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]\hat{\theta}(t)\\
= & \left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]\hat{\theta}(t) \qquad (3.3.5)\\
= & \sum_{i=t-p+1}^{t-1}\varphi(i)\,y(i)+\varphi(t)\,y(t)\\
= & \varphi(t-p)\,y(t-p)+\sum_{i=t-p+1}^{t-1}\varphi(i)\,y(i)+\varphi(t)\,y(t)-\varphi(t-p)\,y(t-p)\\
= & H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)+\varphi(t)\,y(t)-\varphi(t-p)\,y(t-p)\,.
\end{aligned}$$

In addition, it follows from (3.3.3) that

$$\begin{aligned}
\hat{\theta}(t-1) &= \left[P(t-1)\right]^{-1}H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)\\
&= \left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)\right]^{-1}H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)
\end{aligned}$$

With this relation, one has

$$H^{\mathrm{T}}(p,t-1)\,H(p,t-1)\,\hat{\theta}(t-1)=H^{\mathrm{T}}(p,t-1)\,Y(p,t-1).$$

Combining this relation with (3.3.5), gives

$$\begin{aligned}
& \left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]\left[\hat{\theta}(t)-\hat{\theta}(t-1)\right]\\
= & H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)+\varphi(t)\,y(t)-\varphi(t-p)\,y(t-p)-H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)\\
& -\varphi(t)\,\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)+\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\\
= & \varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right]-\varphi(t-p)\left[y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\right].
\end{aligned}$$

It follows from this relation that

$$\begin{aligned}
\hat{\theta}(t) = & \hat{\theta}(t-1)+\left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]^{-1}\cdot\\
& \begin{bmatrix}\varphi(t) & -\varphi(t-p)\end{bmatrix}\begin{bmatrix}y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\\ y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\end{bmatrix}.
\end{aligned}$$

With the previous notation, one can obtain the following recursive fixed memory least squares algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\left[\begin{array}{c} y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) \\ y(t-p) - \varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\end{array}\right] \\ P^{-1}(t) &= P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t) - \varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,.\end{aligned} \tag{3.3.6}$$

By applying matrix inverse lemma, one has

$$\begin{aligned}P(t) &= P(t-1) - P(t-1)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\cdot \\ &\quad \left(I + \left[\begin{array}{c}\varphi^{\mathrm{T}}(t) \\ \varphi^{\mathrm{T}}(t-p)\end{array}\right] P(t-1)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\right)^{-1}\left[\begin{array}{c}\varphi^{\mathrm{T}}(t) \\ \varphi^{\mathrm{T}}(t-p)\end{array}\right] P(t-1) \\ &= P(t-1) - P(t-1)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\cdot \\ &\quad \left[\begin{array}{cc}1+\varphi^{\mathrm{T}}(t)\,P(t)\varphi(t) & -\varphi^{\mathrm{T}}(t)\,P(t)\varphi(t-p) \\ \varphi^{\mathrm{T}}(t-p)\,P(t)\varphi(t) & 1-\varphi^{\mathrm{T}}(t-p)\,P(t)\varphi(t-p)\end{array}\right]^{-1}\left[\begin{array}{c}\varphi^{\mathrm{T}}(t) \\ \varphi^{\mathrm{T}}(t-p)\end{array}\right] P(t-1)\end{aligned}$$

In the proceding formula, the matrix

$$\left[\begin{array}{cc}1+\varphi^{\mathrm{T}}(t)\,P(t)\varphi(t) & -\varphi^{\mathrm{T}}(t)\,P(t)\varphi(t-p) \\ \varphi^{\mathrm{T}}(t-p)\,P(t)\varphi(t) & 1-\varphi^{\mathrm{T}}(t-p)\,P(t)\varphi(t-p)\end{array}\right]$$

is of dimension 2×2, and thus it is easy to obtain its inversion. Nevetheless, it still involve the inverse of a matrix. Now, we introduce the following matrix

$$Q(t) = \left[P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t)\right]^{-1}.$$

With this notation, the algorithm (3.3.6) becomes

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\left[\begin{array}{c} y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) \\ y(t-p) - \varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\end{array}\right] \\ Q^{-1}(t) &= P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t) \\ P^{-1}(t) &= Q^{-1}(t) - \varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,.\end{aligned} \tag{3.3.7}$$

By using Matrix Inverse Lemma for the second and third expressions in (3.3.7) , one has

$$\begin{aligned}Q(t) &= P(t-1) - P(t-1)\varphi(t)\left(1+\varphi^{\mathrm{T}}(t)\,P(t-1)\varphi(t)\right)^{-1}\varphi^{\mathrm{T}}(t)\,P(t-1), \\ P(t) &= Q(t) + Q(t)\varphi(t-p)\left(1-\varphi^{\mathrm{T}}(t-p)\,Q(t)\varphi(t-p)\right)^{-1}\varphi^{\mathrm{T}}(t-p)\,Q(t).\end{aligned}$$

Now we obtain the following algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc}\varphi(t) & -\varphi(t-p)\end{array}\right]\left[\begin{array}{c} y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) \\ y(t-p) - \varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\end{array}\right] \\ Q(t) &= P(t-1) - \frac{P(t-1)\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{1+\varphi^{\mathrm{T}}(t)\,P(t-1)\varphi(t)} \\ P(t) &= Q(t) + \frac{Q(t)\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,Q(t)}{1-\varphi^{\mathrm{T}}(t-p)\,Q(t)\varphi(t-p)}\end{aligned} \tag{3.3.8}$$

3.3.2 The second type of recursive algorithms

In this subsection, we give another recursive algorithm for finite-date window identification. We adopt the same notation as in the preceding subsection. Let

$$P_{\alpha}(t-1)=\left[H^{\mathrm{T}}(p-1,t-1)\,H(p-1,t-1)\right]^{-1}.$$

With this notation, one has

$$\begin{aligned}P^{-1}(t) &= \sum_{i=t-p+1}^{t-1}\varphi(i)\,\varphi^{\mathrm{T}}(i)+\varphi(t)\,\varphi^{\mathrm{T}}(t)\\ &= P_{\alpha}^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t),\end{aligned}$$

$$\begin{aligned}P^{-1}(t-1) &= H^{\mathrm{T}}(p,t-1)\,H(p,t-1)\\ &= \left[\begin{array}{cc}\varphi(t-p) & H^{\mathrm{T}}(p-1,t-1)\end{array}\right]\left[\begin{array}{c}\varphi^{\mathrm{T}}(t-p)\\ H(p-1,t-1)\end{array}\right]\\ &= \varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+H^{\mathrm{T}}(p-1,t-1)\,H(p-1,t-1)\\ &= \varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+P_{\alpha}^{-1}(t-1).\end{aligned}$$

In addition, in order to find the recursive relation between $\hat{\theta}(t)$ and $\hat{\theta}(t-1)$, we need to define an intermediate "estimate"

$$\begin{aligned}\alpha(t-1) &= \left[H^{\mathrm{T}}(p-1,t-1)\,H(p-1,t-1)\right]^{-1}H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1)\\ &= P_{\alpha}(t-1)\,H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1).\end{aligned}$$

From (3.3.4), it is easy to obtain

$$\begin{aligned}\hat{\theta}(t) &= \left[H^{\mathrm{T}}(p,t)\,H(p,t)\right]^{-1}H^{\mathrm{T}}(p,t)\,Y(p,t)\\ &= P(t)\left[\begin{array}{cc}H^{\mathrm{T}}(p-1,t-1) & \varphi(t)\end{array}\right]\left[\begin{array}{c}Y(p-1,t-1)\\ y(t)\end{array}\right]\\ &= P(t)\left[H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1)+\varphi(t)\,y(t)\right]\\ &= P(t)\left[P_{\alpha}^{-1}(t-1)\,P_{\alpha}(t-1)\,H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1)+\varphi(t)\,y(t)\right]\\ &= P(t)\left[P_{\alpha}^{-1}(t-1)\,\alpha(t-1)+\varphi(t)\,y(t)\right]\\ &= P(t)\,P_{\alpha}^{-1}(t-1)\,\alpha(t-1)+P(t)\,\varphi(t)\,y(t)\\ &= \left[I-P(t)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\right]\alpha(t-1)+P(t)\,\varphi(t)\,y(t)\\ &= \alpha(t-1)+P(t)\,\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\,\alpha(t-1)\right];\end{aligned}$$

$$\begin{aligned}\hat{\theta}(t-1) &= \left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)\right]^{-1}H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)\\ &= P(t-1)\left[\begin{array}{cc}\varphi(t-p) & H^{\mathrm{T}}(p-1,t-1)\end{array}\right]\left[\begin{array}{c}y(t-p)\\ Y(p-1,t-1)\end{array}\right]\\ &= P(t-1)\left[\varphi(t-p)\,y(t-p)+H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1)\right]\\ &= P(t-1)\left[\varphi(t-p)\,y(t-p)+P_{\alpha}^{-1}(t-1)\,P_{\alpha}(t-1)\,H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1)\right]\\ &= P(t-1)\left[\varphi(t-p)\,y(t-p)+P_{\alpha}^{-1}(t-1)\,\alpha(t-1)\right].\end{aligned}$$

It follows from the preceding expression on $\hat{\theta}(t-1)$ that

$$\begin{aligned}\alpha(t-1) &= P_\alpha(t-1)P^{-1}(t-1)\hat{\theta}(t-1) - P_\alpha(t-1)\varphi(t-p)y(t-p)\\ &= \left[I + P_\alpha(t-1)\varphi(t-p)\varphi^{\mathrm{T}}(t-p)\right]\hat{\theta}(t-1) - P_\alpha(t-1)\varphi(t-p)y(t-p)\\ &= \hat{\theta}(t-1) - P_\alpha(t-1)\varphi(t-p)\left[y(t-p) - \varphi^{\mathrm{T}}(t-p)\hat{\theta}(t-1)\right].\end{aligned}$$

From the above equation, we have obtained the following algorithm

$$\begin{aligned}\hat{\theta}(t) &= \alpha(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\alpha(t-1)\right]\\ P^{-1}(t) &= P_\alpha^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t)\\ \alpha(t-1) &= \hat{\theta}(t-1) - P_\alpha(t-1)\varphi(t-p)\left[y(t-p) - \varphi^{\mathrm{T}}(t-p)\hat{\theta}(t-1)\right]\\ P_\alpha^{-1}(t-1) &= P^{-1}(t-1) - \varphi(t-p)\varphi^{\mathrm{T}}(t-p)\end{aligned}$$

By applying the Matrix Inversion Lemma, this algorithm can be written as

$$\begin{aligned}\hat{\theta}(t) &= \alpha(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\alpha(t-1)\right]\\ P(t) &= P_\alpha(t-1) - \frac{P_\alpha(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P_\alpha(t-1)}{1+\varphi^{\mathrm{T}}(t)P_\alpha(t-1)\varphi(t)}\\ \alpha(t-1) &= \hat{\theta}(t-1) - P_\alpha(t-1)\varphi(t-p)\left[y(t-p) - \varphi^{\mathrm{T}}(t-p)\hat{\theta}(t-1)\right]\\ P_\alpha(t-1) &= P(t-1) + \frac{P(t-1)\varphi(t-p)\varphi^{\mathrm{T}}(t-p)P(t-1)}{1-\varphi^{\mathrm{T}}(t-p)P(t-1)\varphi(t-p)}\end{aligned}$$

Consider the following equation error model

$$A(z)y(t) = B(z)u(t) + v(t)$$

where

$$\begin{aligned}A(z) &= 1 + \sum_{i=1}^{n_a} a_i z^{-i},\\ B(z) &= \sum_{i=1}^{n_b} b_i z^{-i},\end{aligned}$$

and $v(t)$ is a white noise.

3.4 Fixed memory identification with a forgetting factor

Consider the following linear regression model

$$y(t) = \varphi^{\mathrm{T}}(t)\theta + v(t), \tag{3.4.1}$$

where $v(t)$ is a white noise. In the previous section, we give fixed memory least square identification method. A main feature of this method is that only a finite data set $y(i)$, $u(i)$: $\{y(i), u(i)$: $i = t-p+1, t-p+2, \cdots, t\}$ is used to estimate the parameter θ. In addition, it is easily found that the new data has the same contribution as the older data. Similarly to the idea of forgetting factor least square identification, we can introduce a weighting coefficient to reduce the infuence of old data. With this idea, the previous matrices H_t and Y_t in the previous section are modified as

$$Y(p,t) := \begin{bmatrix} \rho^{p-1}y(t-p+1) \\ \rho^{p-2}y(t-p+2) \\ \cdots \\ \rho y(t-1) \\ y(t) \end{bmatrix} \in \mathbb{R}^p,\ H(p,t) := \begin{bmatrix} \rho^{p-1}\varphi^{\mathrm{T}}(t-p+1) \\ \rho^{p-2}\varphi^{\mathrm{T}}(t-p+2) \\ \cdots \\ \rho\varphi^{\mathrm{T}}(t-1) \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{p\times n}.$$

With this definition, one has

$$Y(p,t-1) := \begin{bmatrix} \rho^{p-1}y(t-p) \\ \rho^{p-2}y(t-p+1) \\ \cdots \\ \rho y(t-2) \\ y(t-1) \end{bmatrix} \in \mathbb{R}^p,\ H(p,t-1) := \begin{bmatrix} \rho^{p-1}\varphi^{\mathrm{T}}(t-p) \\ \rho^{p-2}\varphi^{\mathrm{T}}(t-p+1) \\ \cdots \\ \rho\varphi^{\mathrm{T}}(t-2) \\ \varphi^{\mathrm{T}}(t-1) \end{bmatrix} \in \mathbb{R}^{p\times n}.$$

By comparing the expressions of $Y(p,t-1)$ and $Y(p,t)$, the following relations hold

$$Y(p,t) = \begin{bmatrix} \rho Y(p-1,t-1) \\ y(t) \end{bmatrix},\ Y(p,t-1) = \begin{bmatrix} \rho^{p-1}y(t-p) \\ Y(p-1,t-1) \end{bmatrix}$$

if we denote

$$Y(p-1,t-1) = \begin{bmatrix} \rho^{p-2}y(t-p+1) \\ \rho^{p-3}y(t-p+2) \\ \cdots \\ y(t-1) \end{bmatrix}.$$

Similarly, we have

$$H(p,t) = \begin{bmatrix} \rho H(p-1,t-1) \\ \varphi^{\mathrm{T}}(t) \end{bmatrix},\ H(p,t-1) = \begin{bmatrix} \rho^{p-1}\varphi^{\mathrm{T}}(t-p) \\ H(p-1,t-1) \end{bmatrix} \tag{3.4.2}$$

if we let

$$H(p-1,t-1) = \begin{bmatrix} \rho^{p-2}\varphi^{\mathrm{T}}(t-p+1) \\ \rho^{p-3}\varphi^{\mathrm{T}}(t-p+2) \\ \cdots \\ \varphi^{\mathrm{T}}(t-1) \end{bmatrix}.$$

The estimate $\hat{\theta}(t)$ of θ is given by

$$\hat{\theta}(t) = \left[H^{\mathrm{T}}(p,t)\,H(p,t)\right]^{-1} H^{\mathrm{T}}(p,t)\,Y(p,t). \tag{3.4.3}$$

We will provide two types of recursive algorithms for computing the estimate $\hat{\theta}(t)$ in (3.4.3). In the rest of this section, we denote $\lambda = \rho^2$.

3.4.1 The first type of algorithms

Let

$$P(t)=\left[H^{\mathrm{T}}(p,t)\,H(p,t)\right]^{-1}, \tag{3.4.4}$$

$$P_{\alpha}(t-1)=\left[H^{\mathrm{T}}(p-1,t-1)\,H(p-1,t-1)\right]^{-1}. \tag{3.4.5}$$

According to (3.4.2), it is easily obtained that

$$\begin{aligned} P^{-1}(t) &= \left[\begin{array}{cc} \rho H^{\mathrm{T}}(p-1,t-1) & \varphi(t) \end{array}\right]\left[\begin{array}{c} \rho H(p-1,t-1) \\ \varphi^{\mathrm{T}}(t) \end{array}\right] \\ &= \lambda H^{\mathrm{T}}(p-1,t-1)H(p-1,t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t) \\ &= \lambda P_{\alpha}^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t), \end{aligned} \tag{3.4.6}$$

$$\begin{aligned} P^{-1}(t-1) &= \left[\begin{array}{cc} \rho^{p-1}\varphi(t-p) & H^{\mathrm{T}}(p-1,t-1) \end{array}\right]\left[\begin{array}{c} \rho^{p-1}\varphi^{\mathrm{T}}(t-p) \\ H(p-1,t-1) \end{array}\right] \\ &= \lambda^{p-1}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+H^{\mathrm{T}}(p-1,t-1)H(p-1,t-1) \\ &= \lambda^{p-1}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+P_{\alpha}^{-1}(t-1). \end{aligned} \tag{3.4.7}$$

In addition, we introduce an intermediate estimate

$$\begin{aligned} \alpha(t-1) &= \left[H^{\mathrm{T}}(p-1,t-1)\,H(p-1,t-1)\right]^{-1}H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1) \\ &= P_{\alpha}(t-1)\,H^{\mathrm{T}}(p-1,t-1)\,Y(p-1,t-1). \end{aligned}$$

With these relations, we have

$$\begin{aligned} \hat{\theta}(t) &= \left[H^{\mathrm{T}}(p,t)\,H(p,t)\right]^{-1}H^{\mathrm{T}}(p,t)\,Y(p,t) \\ &= P(t)\left[\begin{array}{cc} \rho H^{\mathrm{T}}(p-1,t-1) & \varphi(t) \end{array}\right]\left[\begin{array}{c} \rho Y(p-1,t-1) \\ y(t) \end{array}\right] \\ &= P(t)\left[\lambda H^{\mathrm{T}}(p-1,t-1)Y(p-1,t-1)+\varphi(t)\,y(t)\right] \\ &= P(t)\left[\lambda P_{\alpha}(t-1)\,P_{\alpha}^{-1}(t-1)\,H^{\mathrm{T}}(p-1,t-1)Y(p-1,t-1)+\varphi(t)\,y(t)\right] \\ &= P(t)\left[\lambda P_{\alpha}(t-1)\,\alpha(t-1)+\varphi(t)\,y(t)\right]. \end{aligned} \tag{3.4.8}$$

It follows from (3.4.6) that

$$\lambda P(t)P_{\alpha}^{-1}(t-1)=I-P(t)\varphi(t)\,\varphi^{\mathrm{T}}(t).$$

Substituting this relation into (3.4.8), gives

$$\begin{aligned} \hat{\theta}(t) &= \left[I-P(t)\varphi(t)\,\varphi^{\mathrm{T}}(t)\right]\alpha(t-1)+P(t)\,\varphi(t)\,y(t) \\ &= \alpha(t-1)+P(t)\,\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\,\alpha(t-1)\right]. \end{aligned}$$

In addition, one has

$$\begin{aligned}
\hat{\theta}(t-1) &= \left[H^{\mathrm{T}}(p,t-1)\,H(p,t-1)\right]^{-1} H^{\mathrm{T}}(p,t-1)\,Y(p,t-1) \\
&= P(t-1)\left[\begin{array}{cc} \rho^{p-1}\varphi(t-p) & H^{\mathrm{T}}(p-1,t-1) \end{array}\right]\left[\begin{array}{c} \rho^{p-1}y(t-p) \\ Y(p-1,t-1) \end{array}\right] \\
&= P(t-1)\left[H^{\mathrm{T}}(p-1,t-1)Y(p-1,t-1)+\lambda^{p-1}\varphi(t-p)\,y(t-p)\right] \qquad (3.4.9) \\
&= P(t-1)\left[P_{\alpha}^{-1}(t-1)\,P_{\alpha}(t-1)\,H^{\mathrm{T}}(p-1,t-1)Y(p-1,t-1)+\lambda^{p-1}\varphi(t-p)\,y(t-p)\right] \\
&= P(t-1)\left[P_{\alpha}^{-1}(t-1)\,\alpha(t-1)+\lambda^{p-1}\varphi(t-p)\,y(t-p)\right].
\end{aligned}$$

It can be obtained from this relation that

$$\alpha(t-1)=P_{\alpha}(t-1)\,P^{-1}(t-1)\,\hat{\theta}(t-1)-\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\,y(t-p).$$

It follows from (3.4.7) that

$$P_{\alpha}(t-1)\,P^{-1}(t-1)=\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+I.$$

With the preceding two relations, one has

$$\begin{aligned}
\alpha(t-1) &= \left[\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+I\right]\hat{\theta}(t-1)-\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\,y(t-p) \\
&= \hat{\theta}(t-1)-\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\left[y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\right].
\end{aligned}$$

By organizing the previous equations, one can obtain the following recursive algorithm:

FDW-FF-RLS1

$$\begin{aligned}
\hat{\theta}(t) &= \alpha(t-1)+P(t)\,\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\,\alpha(t-1)\right] \\
P^{-1}(t) &= \lambda P_{\alpha}^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t) \\
\alpha(t-1) &= \hat{\theta}(t-1)-\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\left[y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\right] \\
P_{\alpha}^{-1}(t-1) &= P^{-1}(t-1)-\lambda^{p-1}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p).
\end{aligned}$$

In order to avoid the inversion of a matrix, by using the Matrix Inversion Lemma one can obtain the following recursive algorithm:

FDW-FF-RLS2

$$\begin{aligned}
\hat{\theta}(t) &= \alpha(t-1)+P(t)\,\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\,\alpha(t-1)\right] \\
P(t) &= \frac{1}{\lambda}\left[P_{\alpha}(t-1)-\frac{P_{\alpha}(t-1)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P_{\alpha}(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)\,P_{\alpha}(t-1)\,\varphi(t)}\right] \\
\alpha(t-1) &= \hat{\theta}(t-1)-\lambda^{p-1}P_{\alpha}(t-1)\,\varphi(t-p)\left[y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\right] \\
P_{\alpha}(t-1) &= P(t-1)+\frac{P(t-1)\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,P(t-1)}{\lambda^{-p}-\varphi^{\mathrm{T}}(t-p)\,P(t-1)\,\varphi(t-p)}.
\end{aligned}$$

3.4.2 The second type of algorithms

With the definite of $P(t)$ in (3.4.4), one has

$$\begin{aligned} & H^{\mathrm{T}}(p,t)\,Y(p,t) \\ = & \sum_{i=0}^{p-1}\lambda^{i}\varphi(t-i)\,\varphi^{\mathrm{T}}(t-i) \\ = & \sum_{i=1}^{p}\lambda^{i}\varphi(t-i)\,y(t-i)+\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p) \qquad (3.4.10) \\ = & \lambda H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)+\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p)\,; \end{aligned}$$

and

$$\begin{aligned} P^{-1}(t) = & \lambda^{p-1}\varphi(t-p+1)\,\varphi^{\mathrm{T}}(t-p+1)+\lambda^{p-2}\varphi(t-p+2)\,\varphi^{\mathrm{T}}(t-p+2) \\ & +\cdots+\lambda\varphi(t-1)\,\varphi^{\mathrm{T}}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t) \\ = & \lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)+\lambda^{p-1}\varphi(t-p+1)\,\varphi^{\mathrm{T}}(t-p+1) \qquad (3.4.11) \\ & +\lambda^{p-2}\varphi(t-p+2)\,\varphi^{\mathrm{T}}(t-p+2) \\ & +\cdots+\lambda\varphi(t-1)\,\varphi^{\mathrm{T}}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p) \\ = & \lambda P^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,, \end{aligned}$$

which implies that

$$\lambda P(t)P^{-1}(t-1)=I-P(t)\left[\varphi(t)\,\varphi^{\mathrm{T}}(t)-\lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right].$$

By combining this relation with (3.4.10), it can be obtained that

$$\begin{aligned} \hat{\theta}(t) = & P(t)H^{\mathrm{T}}(p,t)\,Y(p,t) \\ = & P(t)\left[\lambda H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)+\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p)\right] \\ = & P(t)\left[\lambda P^{-1}(t-1)P(t-1)H^{\mathrm{T}}(p,t-1)\,Y(p,t-1)+\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p)\right] \\ = & \lambda P(t)P^{-1}(t-1)\hat{\theta}(t-1)+P(t)\left[\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p)\right] \\ = & \hat{\theta}(t-1)-P(t)\left[\varphi(t)\,\varphi^{\mathrm{T}}(t)-\lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\right]\hat{\theta}(t-1) \\ & +P(t)\left[\varphi(t)\,y(t)-\lambda^{p}\varphi(t-p)\,y(t-p)\right] \\ = & \hat{\theta}(t-1)+ \\ & P(t)\left[\varphi(t)\left(y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right)-\lambda^{p}\varphi(t-p)\left(y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\right)\right] \\ = & \hat{\theta}(t-1)+P(t)\left[\begin{array}{cc}\varphi(t) & -\lambda^{p}\varphi(t-p)\end{array}\right]\left[\begin{array}{c} y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) \\ y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1)\end{array}\right]. \end{aligned}$$

In addition, let

$$Q^{-1}(t)=\lambda P^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)\,.$$

Then it follows from (3.4.11) that

$$P^{-1}(t)=Q^{-1}(t)-\lambda^{p}\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,.$$

By using the Matrix Inverse Lemma in Lemma 3.1.1, from the preceding two relations one has

$$\begin{aligned} Q(t) &= \frac{1}{\lambda}P(t-1) - \frac{1}{\lambda}P(t-1)\frac{\varphi(t)\,\varphi^{\mathrm{T}}(t)}{1+\varphi^{\mathrm{T}}(t)\,\frac{1}{\lambda}P(t)\varphi(t)}\frac{1}{\lambda}P(t-1) \\ &= \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)\,P(t-1)\varphi(t)}\right]; \end{aligned}$$

$$P(t) = Q(t) + \frac{Q(t)\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,Q(t)}{\lambda^{-p}-\varphi^{\mathrm{T}}(t-p)\,Q(t)\varphi(t-p)}.$$

So we have obtained the following recursive algorithm.

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc} \varphi(t) & -\lambda^{p}\varphi(t-p) \end{array}\right]\left[\begin{array}{c} y(t)-\varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1) \\ y(t-p)-\varphi^{\mathrm{T}}(t-p)\,\hat{\theta}(t-1) \end{array}\right], \\ Q(t) &= \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)\,P(t-1)\varphi(t)}\right], \\ P(t) &= Q(t) + \frac{Q(t)\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\,Q(t)}{\lambda^{-p}-\varphi^{\mathrm{T}}(t-p)\,Q(t)\varphi(t-p)}. \end{aligned}$$

Chapter 4

Weighted recursive least squares algorithms

4.1 Weighted least squares estimation

Consider the following linear regression model

$$y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t) \tag{4.1.1}$$

where $y(t) \in \mathbb{R}$ is the output, $\varphi(t) \in \mathbb{R}^n$ is the and $\theta \in \mathbb{R}^n$ is the parameter vector to be estimated. In order to estimate the parameter θ, we consider the following weighted least squares criterion

$$J(\theta) = \sum_{i=1}^{t} \beta(t,i)\left[y(i) - \varphi^{\mathrm{T}}(i)\,\theta\right]^2. \tag{4.1.2}$$

Similarly to the treatment in the previous section, we denote

$$Y_t := \begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t) \end{bmatrix} \in \mathbb{R}^t, H_t := \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times n}. \tag{4.1.3}$$

In addition, we introduce the following matrix

$$\Theta(t) = \begin{bmatrix} \beta(t,1) & & & \\ & \beta(t,2) & & \\ & & \ddots & \\ & & & \beta(t,2) \end{bmatrix}. \tag{4.1.4}$$

In this case, one has

$$\begin{aligned} J(\theta) &= \left[\begin{array}{cccc} y(1)-\varphi^{\mathrm{T}}(1)\theta & y(2)-\varphi^{\mathrm{T}}(2)\theta & \cdots & y(t)-\varphi^{\mathrm{T}}(t)\theta \end{array}\right] \cdot \\ &\quad \left[\begin{array}{cccc} \beta(t,1) & & & \\ & \beta(t,2) & & \\ & & \ddots & \\ & & & \beta(t,2) \end{array}\right] \left[\begin{array}{c} y(1)-\varphi^{\mathrm{T}}(1)\theta \\ y(2)-\varphi^{\mathrm{T}}(2)\theta \\ \cdots \\ y(t)-\varphi^{\mathrm{T}}(t)\theta \end{array}\right] \\ &= [Y_t - H_t\theta]^{\mathrm{T}} \Theta(t) [Y_t - H_t\theta]. \end{aligned}$$

In order to the estimate $\hat{\theta}_{WLS}(t)$ that minimizes the criterion function $J(\theta)$, we need to taking derivative of $J(\theta)$ with respect to θ. By using the matrix calculs introduced in the previous section, one can easily obtain that

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \theta} &= 2\frac{\partial [Y_t - H_t\theta]}{\partial \theta}\Theta(t)[Y_t - H_t\theta] \\ &= -2H_t^{\mathrm{T}}\Theta(t)[Y_t - H_t\theta]. \end{aligned}$$

By setting the above derivative to zero, one has

$$H_t^{\mathrm{T}}\Theta(t)Y_t = H_t^{\mathrm{T}}\Theta(t)H_t\theta,$$

which gives the estimate $\hat{\theta}_{WLS}(t)$ of θ as

$$\hat{\theta}_{WLS}(t) = \left[H_t^{\mathrm{T}}\Theta(t)H_t\right]^{-1} H_t^{\mathrm{T}}\Theta(t)Y_t. \tag{4.1.5}$$

Substituting (4.1.3) and (4.1.4) into (4.1.5), gives

$$\hat{\theta}_{WLS}(t) = \left[\sum_{i=1}^{t} \beta(t,i)\varphi(i)\varphi^{\mathrm{T}}(i)\right]^{-1} \left[\sum_{i=1}^{t} \beta(t,i)\varphi(i)y(i)\right]. \tag{4.1.6}$$

To compute (4.1.5) or (4.1.6) as given, we would at time t form the indicated matrix and vector vector from the data $\{\varphi(t)\}$. If we had computed $\hat{\theta}(t-1)$ previously, this would not be of any immediate help. However, it is clear from the expressions that $\hat{\theta}(t)$ and $\hat{\theta}(t-1)$ are closely related.

Suppose that the weighting sequence has the following property:

$$\begin{aligned} \beta(t,i) &= \lambda(t)\beta(t-1,i),\ i = 1,\ 2,\ \cdots,\ t-1, \\ \beta(t,t) &= 1. \end{aligned} \tag{4.1.7}$$

It follows from this relation that

$$\begin{aligned} \beta(1,1) &= 1 \\ \beta(2,1) &= \lambda(2),\ \beta(2,2) = 1 \\ \beta(3,1) &= \lambda(3)\lambda(2),\ \beta(3,2) = \lambda(3),\ \beta(3,3) = 1 \\ \beta(4,1) &= \lambda(4)\lambda(3)\lambda(2),\ \beta(4,2) = \lambda(4)\lambda(3),\ \beta(4,3) = \lambda(4),\ \beta(4,4) = 1 \\ &\cdots \end{aligned}$$

This means that we may write

$$\beta\left(t,i\right)=\prod_{j=i+1}^{t}\lambda(j).$$

We shall later discuss the significance of this assumption. Denote

$$\begin{aligned}P(t) &= \left[H_t^{\mathrm{T}}\Theta(t)H_t\right]^{-1}=\left[\sum_{i=1}^{t}\beta\left(t,i\right)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\right]^{-1},\\ f(t) &= H_t^{\mathrm{T}}\Theta(t)Y_t=\sum_{i=1}^{t}\beta\left(t,i\right)\varphi\left(i\right)y\left(i\right).\end{aligned}$$

Then, with the assumption (4.1.7) one has

$$\begin{aligned}P^{-1}\left(t\right) &= \sum_{i=1}^{t}\beta\left(t,i\right)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\\ &= \sum_{i=1}^{t}\left[\prod_{j=i+1}^{t}\lambda(j)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\right]\\ &= \sum_{i=1}^{t-1}\left[\prod_{j=i+1}^{t}\lambda(j)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\right]+\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right)\\ &= \sum_{i=1}^{t-1}\left[\lambda(t)\prod_{j=i+1}^{t-1}\lambda(j)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\right]+\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right)\\ &= \lambda(t)\sum_{i=1}^{t-1}\left[\prod_{j=i+1}^{t-1}\lambda(j)\varphi\left(i\right)\varphi^{\mathrm{T}}\left(i\right)\right]+\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right)\\ &= \lambda(t)P^{-1}\left(t-1\right)+\varphi\left(t\right)\varphi^{\mathrm{T}}\left(t\right),\end{aligned}\tag{4.1.8}$$

and

$$\begin{aligned}f(t) &= \sum_{i=1}^{t}\beta\left(t,i\right)\varphi\left(i\right)y\left(i\right)\\ &= \lambda(t)f\left(t-1\right)+\varphi\left(t\right)y\left(t\right).\end{aligned}$$

Now

$$\begin{aligned}
\hat{\theta}(t) &= P(t)f(t) \\
&= P(t)\left[\lambda(t)f(t-1)+\varphi(t)y(t)\right] \\
&= P(t)\left[\lambda(t)P^{-1}(t-1)P(t-1)f(t-1)+\varphi(t)y(t)\right] \\
&= P(t)\left[\lambda(t)P^{-1}(t-1)\hat{\theta}(t-1)+\varphi(t)y(t)\right] \\
&= P(t)\left[\left(P^{-1}(t)-\varphi(t)\varphi^{\mathrm{T}}(t)\right)\hat{\theta}(t-1)+\varphi(t)y(t)\right] \\
&= \hat{\theta}(t-1)+P(t)\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right].
\end{aligned}$$

We thus have the following weighted least squares algorithm

WRLS1

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1)+P(t)\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
P^{-1}(t) &= \lambda(t)P^{-1}(t-1)+\varphi(t)\varphi^{\mathrm{T}}(t).
\end{aligned}$$

This is a recursive algorithm. At time $t-1$, we store only the finite-dimensional information $\hat{\theta}(t-1)$ and $P(t-1)$.

To avoid inverting $P^{-1}(t)$ at each step, we apply the matrix inversion lemma in Lemma ??? to the second expression in the algorithm WRLS1. One thus has

$$\begin{aligned}
P(t) &= \left[\lambda(t)P^{-1}(t-1)+\varphi(t)\varphi^{\mathrm{T}}(t)\right]^{-1} \\
&= \frac{1}{\lambda(t)}\left[P(t-1)-\frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right].
\end{aligned}$$

Moreover, we have

$$\begin{aligned}
L(t) &:= P(t)\varphi(t) \\
&= \frac{P(t-1)\varphi(t)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}.
\end{aligned}$$

We can thus summarize this version of the algorithm as

WRLS2

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1)+L(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
L(t) &= \frac{P(t-1)\varphi(t)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}, \\
P(t) &= \frac{1}{\lambda(t)}\left[P(t-1)-\frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right].
\end{aligned}$$

In addition, by using the introduced gain vector $L(t)$ one has

$$\begin{aligned} P(t) &= \frac{1}{\lambda(t)}\left[P(t-1)-\frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}\right] \\ &= \frac{1}{\lambda(t)}\left[P(t-1)-L(t)\varphi^{\mathrm{T}}(t)P(t-1)\right] \\ &= \frac{1}{\lambda(t)}\left[I-L(t)\varphi^{\mathrm{T}}(t)\right]P(t-1). \end{aligned}$$

With this, we can obtain another version of the weighted recursive least squares algorim.

WRLS3

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1)+L(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ L(t) &= \frac{P(t-1)\varphi(t)}{\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}, \\ P(t) &= \frac{1}{\lambda(t)}\left[I-L(t)\varphi^{\mathrm{T}}(t)\right]P(t-1). \end{aligned}$$

4.1.1 Normalized gain version

The "size" of the matrix $P^{-1}(t)$ in the previous section will depend on the $\lambda(t)$. To clearly bring out the amount of modification inflicted on $\hat{\theta}(t-1)$, it is instructive to normalize $P^{-1}(t)$ such that

$$\overline{P}^{-1}(t)=\gamma(t)P^{-1}(t),\ \gamma(t)=\left[\sum_{i=1}^{t}\beta(t,i)\right]^{-1}. \tag{4.1.9}$$

It follows from the assumption (4.1.7) that $\gamma(t)$ defined in (4.1.9) satisfies

$$\frac{1}{\gamma(t)}=1+\frac{\lambda(t)}{\gamma(t-1)}. \tag{4.1.10}$$

In addition, it is obvious that $\overline{P}^{-1}(t)$ now is a weighted arithmetic mean of $\varphi(t)\varphi^{\mathrm{T}}(t)$. From (4.1.10) and (4.1.8), one has

$$\begin{aligned} \overline{P}^{-1}(t) &= \gamma(t)\left[\lambda(t)P^{-1}(t-1)+\varphi(t)\varphi^{\mathrm{T}}(t)\right] \\ &= \gamma(t)\left[\frac{\lambda(t)\gamma(t-1)P^{-1}(t-1)}{\gamma(t-1)}+\varphi(t)\varphi^{\mathrm{T}}(t)\right] \\ &= \gamma(t)\left[\frac{\lambda(t)\overline{P}^{-1}(t-1)}{\gamma(t-1)}+\varphi(t)\varphi^{\mathrm{T}}(t)\right] \\ &= \gamma(t)\left[\left(\frac{1}{\gamma(t)}-1\right)\overline{P}^{-1}(t-1)+\varphi(t)\varphi^{\mathrm{T}}(t)\right] \\ &= \overline{P}^{-1}(t-1)+\gamma(t)\left[\varphi(t)\varphi^{\mathrm{T}}(t)-\overline{P}^{-1}(t-1)\right]. \end{aligned}$$

In this case, we can have the following relation:

$$\begin{array}{rcl} & & \text{WRLSNormalized} \\ \hat{\theta}(t) & = & \hat{\theta}(t-1)+\gamma(t)\overline{P}(t)\varphi(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ \overline{P}^{-1}(t) & = & \overline{P}^{-1}(t-1)+\gamma(t)\left[\varphi(t)\varphi^{\mathrm{T}}(t)-\overline{P}^{-1}(t-1)\right]. \end{array}$$

Notice that $e(t)=y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)$ is the prediction error according to the current model. Since $\overline{P}^{-1}(t)$ is a normalized matrix, the variable $\gamma(t)$ can be viewed as an updated step size or gain in Algorithm WRLSNormalzied.

4.2 Multivariable case

When the considered plant is multiply input multiply out, we consider the following error criterion function

$$J(\theta)=\sum_{i=1}^{t}\beta(t,i)\left[y(i)-\varphi^{\mathrm{T}}(t)\theta\right]^{\mathrm{T}}\Lambda_i^{-1}\left[y(i)-\varphi^{\mathrm{T}}(t)\theta\right].$$

Entirely analogous calculations as before give the following multivariable courter part of Algorithm WRLS2

$$\begin{array}{rcl} & & \text{WRLS-MultiCase} \\ \hat{\theta}(t) & = & \hat{\theta}(t-1)+L(t)\left[y(t)-\varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ L(t) & = & P(t-1)\varphi(t)\left[\lambda(t)\Lambda_t+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)\right]^{-1}, \\ P(t) & = & \dfrac{1}{\lambda(t)}\left[P(t-1)-P(t-1)\varphi(t)\left[\lambda(t)+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)\right]^{-1}\varphi^{\mathrm{T}}(t)P(t-1)\right]. \end{array}$$

4.3 Weighted fixed memory recursive least squares algorithm

Firstly, we consider the following linear regression model

$$y(t)=\varphi^{T}(t)\theta+v(t), \tag{4.3.1}$$

and choose the following error criterion function

$$J(\theta)=\sum_{i=t-p+1}^{t}\beta(t,i)\left[y(i)-\varphi^{T}(t)\theta\right]^{2}.$$

If we introduce the following notations

$$Y(p,t) = \begin{bmatrix} y(t-p+1) \\ y(t-p+2) \\ \cdots \\ y(t-1) \\ y(t) \end{bmatrix} \in R^p \ , \quad H(p,t) = \begin{bmatrix} \varphi^T(t-p+1) \\ \varphi^T(t-p+2) \\ \cdots \\ \varphi^T(t-1) \\ \varphi^T(t) \end{bmatrix} \in R^{p\times n},$$

and

$$P(t) = \left[H^T(p,t)\Theta(p,t)H(p,t)\right]^{-1},$$

where the weighted matrix has the following form and property

$$\Theta(p,t) = \begin{bmatrix} \beta(t,t-p+1) & & & & \\ & \beta(t,t-p+2) & & & \\ & & \cdots & & \\ & & & \beta(t,t-1) & \\ & & & & \beta(t,t) \end{bmatrix} \in R^{p\times p},$$

$$\begin{aligned} \beta(t,i) &= \lambda(t)\lambda(t-1)\cdots\lambda(i+1) \\ &= \prod_{j=i+1}^{t} \lambda(j), i = t-p+1, t-p+2, \cdots, t-1, \\ \beta(t,t) &= 1, \\ \beta(t,i) &= \lambda(t)\beta(t-1,i). \end{aligned}$$

Then we can rewrite the error criterion function as follows

$$\begin{aligned} J(\theta) &= \sum_{i=t-p+1}^{t} \beta(t,i)\left[y(i) - \varphi^T(t)\theta\right]^2 \\ &= \left[Y(p,t) - H(p,t)\theta\right]^T \Theta(p,t)\left[Y(p,t) - H(p,t)\theta\right]. \end{aligned} \tag{4.3.2}$$

By using the matrix calculus, one can easily obtain that

$$\begin{aligned} \frac{\partial J(\theta)}{\partial\theta} &= \frac{\partial\left[Y(p,t) - H(p,t)\theta\right]}{\partial\theta} \cdot \frac{\partial J(\theta)}{\partial\left[Y(p,t) - H(p,t)\theta\right]} \\ &= -H^T(p,t)\cdot 2\cdot\Theta(p,t)[Y(p,t) - H(p,t)\theta] \\ &= -2H^T(p,t)\Theta(p,t)[Y(p,t) - H(p,t)\theta]. \end{aligned}$$

By setting the above derivative to zero, one has the following estimate $\hat{\theta}(t)$ of θ

$$\begin{aligned} \hat{\theta}(t) &= [H^T(p,t)\Theta(p,t)H(p,t)]^{-1}H^T(p,t)\Theta(p,t)Y(p,t) \\ &= P(t)H^T(p,t)\Theta(p,t)Y(p,t). \end{aligned} \tag{4.3.3}$$

4.3.1 First type weighted fixed memory recursive least squares algorithm

Now we introduce the first type weighted fixed memory recursive least squares algorithm. In view of the expression of $Y(p,t)$ and $H(p,t)$, one has

$$\begin{aligned}
\hat{\theta}(t) &= [H^T(p,t)\Theta(p,t)H(p,t)]^{-1}H^T(p,t)\Theta(p,t)Y(p,t) \\
&= [\sum_{i=t-p+1}^{t}\beta(t,i)\varphi(i)\varphi^T(i)]^{-1}\sum_{i=t-p+1}^{t}\beta(t,i)\varphi(i)y(i) \\
&= [\sum_{i=t-p+1}^{t-1}\beta(t,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)]^{-1} \\
&\quad [\sum_{i=t-p+1}^{t-1}\beta(t,i)\varphi(i)y(i)+\varphi(t)y(t)] \\
&= [\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)]^{-1} \\
&\quad [\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)y(i)+\varphi(t)y(t)]. \qquad (4.3.4)
\end{aligned}$$

We can rewrite equation (4.3.4) as follows

$$\begin{aligned}
&\left[\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)\right]\hat{\theta}(t) \\
= \;& \lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)y(i)+\varphi(t)y(t). \qquad (4.3.5)
\end{aligned}$$

For the left-hand side of equation (4.3.5), one has

$$
\begin{aligned}
& \left[\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)\right]\hat{\theta}(t) \\
= & \left[\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)+\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)\right. \\
& \left.-\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)\right]\hat{\theta}(t) \\
= & \left[\lambda(t)\sum_{i=t-p}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)+\varphi(t)\varphi^T(t)-\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)\right]\hat{\theta}(t) \\
= & [\lambda(t)H^T(p,t-1)\Theta(p,t-1)H(p,t-1)+ \\
& \varphi(t)\varphi^T(t)-\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)]\hat{\theta}(t). \qquad (4.3.6)
\end{aligned}
$$

From the right-hand side of equation (4.3.5), one can obtain

$$
\begin{aligned}
& \lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)y(i)+\varphi(t)y(t) \\
= & \lambda(t)\beta(t-1,t-p)\varphi(t-p)y(t-p)+\lambda(t)\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)y(i) \\
& +\varphi(t)y(t)-\lambda(t)\beta(t-1,t-p)\varphi(t-p)y(t-p) \\
= & \lambda(t)\sum_{i=t-p}^{t-1}\beta(t-1,i)\varphi(i)y(i)+\varphi(t)y(t)-\lambda(t)\beta(t-1,t-p)\varphi(t-p)y(t-p) \\
= & \lambda(t)H^T(p,t-1)\Theta(p,t-1)Y(p,t-1)+\varphi(t)y(t)-\lambda(t)\beta(t-1,t-p)\varphi(t-p)y(t-p).
\end{aligned}
$$

It follows from equation (4.3.3) that

$$
\begin{aligned}
\hat{\theta}(t-1) &= P(t-1)H^T(p,t-1)\Theta(p,t-1)Y(p,t-1) \\
&= \left[H^T(p,t-1)\Theta(p,t-1)Y(p,t-1)\right]^{-1}H^T(p,t-1)\Theta(p,t-1)Y(p,t-1)
\end{aligned}
$$

and

$$
\begin{aligned}
& H^T(p,t-1)\Theta(p,t-1)Y(p,t-1)\hat{\theta}(t-1) \\
= & H^T(p,t-1)\Theta(p,t-1)Y(p,t-1).
\end{aligned}
$$

By combining the above equation with equation (4.3.6), one can derive that

$$
\begin{aligned}
& [\lambda(t)H^T(p,t-1)\Theta(p,t-1)H(p,t-1) \\
& +\varphi(t)\varphi^T(t) - \lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)] \left[\hat{\theta}(t) - \hat{\theta}(t-1)\right] \\
= \; & \lambda(t)H^T(p,t-1)\Theta(p,t-1)Y(p,t-1) \\
& +\varphi(t)y(t) - \lambda(t)\beta(t-1,t-p)\varphi(t-p)y(t-p) \\
& -\lambda(t)H^T(p,t-1)\Theta(p,t-1)Y(p,t-1) - \varphi(t)\varphi^T(t)\hat{\theta}(t-1) \\
& +\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)\hat{\theta}(t-1) \\
= \; & \varphi(t)\left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right] \\
& -\lambda(t)\beta(t-1,t-p)\varphi(t-p)\left[y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1)\right]
\end{aligned}
$$

It follows from this relation that

$$
\begin{aligned}
\hat{\theta}(t) = \; & \hat{\theta}(t-1) + [\lambda(t)H^T(p,t-1)\Theta(p,t-1)H(p,t-1) \\
& +\varphi(t)\varphi^T(t) - \lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)]^{-1} \\
& \left[\begin{array}{cc} \varphi(t) & -\lambda(t)\beta(t-1,t-p)\varphi(t-p) \end{array}\right] \left[\begin{array}{c} y(t) - \varphi^T(t)\hat{\theta}(t-1) \\ y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1) \end{array}\right].
\end{aligned}
$$

With the previous notations, one can obtain the following algorithm

$$
\begin{aligned}
\hat{\theta}(t) = \; & \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc} \varphi(t) & -\lambda(t)\beta(t-1,t-p)\varphi(t-p) \end{array}\right] \\
& \left[\begin{array}{c} y(t) - \varphi^T(t)\hat{\theta}(t-1) \\ y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1) \end{array}\right] \\
P^{-1}(t) = \; & \lambda(t)P^{-1}(t-1) + \varphi(t)\varphi^T(t) \\
& -\lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p).
\end{aligned} \tag{4.3.7}
$$

Now we introduce the following matrix

$$
Q(t) = \left[\lambda(t)P^{-1}(t-1) + \varphi(t)\varphi^T(t)\right]^{-1}.
$$

With this notation, the algorithm (4.3.7) becomes

$$
\begin{aligned}
\hat{\theta}(t) = \; & \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc} \varphi(t) & -\lambda(t)\beta(t-1,t-p)\varphi(t-p) \end{array}\right] \\
& \left[\begin{array}{c} y(t) - \varphi^T(t)\hat{\theta}(t-1) \\ y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1) \end{array}\right] \\
P^{-1}(t) = \; & Q^{-1}(t) - \lambda(t)\beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p) \\
= \; & Q^{-1}(t) - \beta(t,t-p)\varphi(t-p)\varphi^T(t-p) \\
Q^{-1}(t) = \; & \lambda(t)P^{-1}(t-1) + \varphi(t)\varphi^T(t).
\end{aligned} \tag{4.3.8}
$$

By using Matrix Inverse Lemma for the second and third expression in (4.3.8), one has

$$\begin{aligned}
Q(t) &= \frac{P(t-1)}{\lambda(t)} - \frac{P(t-1)}{\lambda(t)}\varphi(t)\left[1+\varphi^T(t)\frac{P(t-1)}{\lambda(t)}\varphi(t)\right]^{-1}\varphi^T(t)\frac{P(t-1)}{\lambda(t)} \\
&= \frac{1}{\lambda(t)}\left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{\lambda(t)+\varphi^T(t)P(t-1)\varphi(t)}\right], \\
P(t) &= Q(t) + Q(t)\beta(t,t-p)\varphi(t-p) \\
&\quad \left[1-\varphi^T(t-p)Q(t)\beta(t,t-p)\varphi(t-p)\right]^{-1}\varphi^T(t-p)Q(t) \\
&= Q(t) + \beta(t,t-p)\frac{Q(t)\varphi(t-p)\varphi^T(t-p)Q(t)}{1-\beta(t,t-p)\varphi^T(t-p)Q(t)\varphi(t-p)}.
\end{aligned}$$

Now we obtain the following weighted fixed memory recursive least squares algorithm

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\left[\begin{array}{cc} \varphi(t) & -\beta(t,t-p)\varphi(t-p) \end{array}\right] \\
&\quad \left[\begin{array}{c} y(t)-\varphi^T(t)\hat{\theta}(t-1) \\ y(t-p)-\varphi^T(t-p)\hat{\theta}(t-1) \end{array}\right] \\
P(t) &= Q(t) + \beta(t,t-p)\frac{Q(t)\varphi(t-p)\varphi^T(t-p)Q(t)}{1-\beta(t,t-p)\varphi^T(t-p)Q(t)\varphi(t-p)} \\
Q(t) &= \frac{1}{\lambda(t)}\left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{\lambda(t)+\varphi^T(t)P(t-1)\varphi(t)}\right]
\end{aligned}$$

4.3.2 The second type weighted fixed memory recursive least squares algorithm

Now we introduce the second type weighted fixed memory recursive least squares algorithm. If we let

$$\begin{aligned}
P_\alpha(t-1) &= \left[H^T(p-1,t-1)\Theta(p-1,t-1)H(p-1,t-1)\right]^{-1} \\
&= \left[\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)\varphi^T(i)\right]^{-1},
\end{aligned}$$

then

$$\begin{aligned} P^{-1}(t) &= \sum_{i=t-p+1}^{t-1} \beta(t,i)\varphi(i)\varphi^T(i) + \varphi(t)\varphi^T(t) \\ &= \lambda(t) \sum_{i=t-p+1}^{t-1} \beta(t-1,i)\varphi(i)\varphi^T(i) + \varphi(t)\varphi^T(t) \\ &= \lambda(t)P_\alpha^{-1}(t-1) + \varphi(t)\varphi^T(t), \end{aligned} \tag{4.3.9}$$

$$\begin{aligned} P^{-1}(t-1) &= H^T(p,t-1)\Theta(p,t-1)H(p,t-1) \\ &= \sum_{i=t-p}^{t-1} \beta(t-1,i)\varphi(i)\varphi^{\mathrm{T}}(i) \\ &= \sum_{i=t-p+1}^{t-1} \beta(t-1,i)\varphi(i)\varphi^{\mathrm{T}}(i) + \beta(t-1,t-p)\varphi(t-p)\varphi^{\mathrm{T}}(t-p) \\ &= H^{\mathrm{T}}(p-1,t-1)\Theta(p-1,t-1)H(p-1,t-1) \\ &\quad +\beta(t-1,t-p)\varphi(t-p)\varphi^{\mathrm{T}}(t-p) \\ &= P_\alpha^{-1}(t-1) + \beta(t-1,t-p)\varphi(t-p)\varphi^{\mathrm{T}}(t-p). \end{aligned} \tag{4.3.10}$$

In order to find the recursive ralation between $\hat{\theta}(t)$ and $\hat{\theta}(t-1)$, we need to define an intermediate "estimate"

$$\begin{aligned} \alpha(t-1) &= [H^T(p-1,t-1)\Theta(p-1,t-1)H(p-1,t-1)]^{-1} \\ &\quad H^T(p-1,t-1)\Theta(p-1,t-1)Y(p-1,t-1) \\ &= P_\alpha(t-1)H^T(p-1,t-1)\Theta(p-1,t-1)Y(p-1,t-1) \end{aligned} \tag{4.3.11}$$

From equation (4.3.4), it's easy to obtain

$$\begin{aligned} \hat{\theta}(t) &= [H^{\mathrm{T}}(p,t)\Theta(p,t)H(p,t)]^{-1}H^{\mathrm{T}}(p,t)\Theta(p,t)Y(p,t) \\ &= P(t) \sum_{i=t-p+1}^{t} \beta(t,i)\varphi(i)y(i) \\ &= P(t) \left[\lambda(t) \sum_{i=t-p+1}^{t-1} \beta(t-1,i)\varphi(i)y(i) + \varphi(t)y(t) \right] \\ &= P(t) \left[\lambda(t)H^{\mathrm{T}}(p-1,t-1)\Theta(p-1,t-1)Y(p-1,t-1) + \varphi(t)y(t) \right], \end{aligned}$$

It follows from equation (4.3.11) that

$$P_\alpha^{-1}(t-1)\alpha(t-1) = H^{\mathrm{T}}(p-1,t-1)\Theta(p-1,t-1)Y(p-1,t-1). \tag{4.3.12}$$

Then, $\hat{\theta}(t)$ can be rewritten as

$$\begin{aligned}\hat{\theta}(t) &= P(t)\left[\lambda(t)P_{\alpha}^{-1}(t-1)\alpha(t-1)+\varphi(t)y(t)\right]\\ &= \lambda(t)P(t)P_{\alpha}^{-1}(t-1)\alpha(t-1)+P(t)\varphi(t)y(t).\end{aligned}$$

From equation (4.3.9), one can obtain

$$P(t)P^{-1}(t)=I=\lambda(t)P(t)P_{\alpha}^{-1}(t-1)+P(t)\varphi(t)\varphi^{T}(t),$$

$$\lambda(t)P(t)P_{\alpha}^{-1}(t-1)=I-P(t)\varphi(t)\varphi^{T}(t).$$

So, $\hat{\theta}(t)$ can be obtained as

$$\begin{aligned}\hat{\theta}(t) &= \left[I-P(t)\varphi(t)\varphi^{T}(t)\right]\alpha(t-1)+P(t)\varphi(t)y(t)\\ &= \alpha(t-1)+P(t)\varphi(t)\left[y(t)-\varphi^{T}(t)\alpha(t-1)\right].\end{aligned}$$

From equation (4.3.4), one can obtain

$$\begin{aligned}\hat{\theta}(t-1) &= [H^{\mathrm{T}}(p,t)\Theta(p,t)H(p,t)]^{-1}H^{\mathrm{T}}(p,t)\Theta(p,t)Y(p,t)\\ &= P(t-1)\sum_{i=t-p}^{t-1}\beta(t-1,i)\varphi(i)y(i)\\ &= P(t-1)[\sum_{i=t-p+1}^{t-1}\beta(t-1,i)\varphi(i)y(i)\\ &\quad +\beta(t-1,t-p)\varphi(t-p)y(t-p)]\\ &= P(t-1)[H^{T}(p-1,t-1)\Theta(p-1,t-1)Y(p-1,t-1)\\ &\quad +\beta(t-1,t-p)\varphi(t-p)y(t-p)].\end{aligned}$$

We substitute (4.3.12) into above equation, one gets

$$\begin{aligned}\hat{\theta}(t-1) &= P(t-1)[P_{\alpha}^{-1}(t-1)\alpha(t-1)\\ &\quad +\beta(t-1,t-p)\varphi(t-p)y(t-p)]\\ P^{-1}(t-1)\hat{\theta}(t-1) &= P_{\alpha}^{-1}(t-1)\alpha(t-1)\\ &\quad +\beta(t-1,t-p)\varphi(t-p)y(t-p). \qquad (4.3.13)\end{aligned}$$

From equation (4.3.13), one gets

$$\begin{aligned}\alpha(t-1) &= P_{\alpha}(t-1)P^{-1}(t-1)\hat{\theta}(t-1)\\ &\quad -\beta(t-1,t-p)P_{\alpha}(t-1)\varphi(t-p)y(t-p). \qquad (4.3.14)\end{aligned}$$

Substituting equation (4.3.10) into equation (4.3.14), gives

$$\begin{aligned}
\alpha(t-1) &= P_\alpha(t-1)\left[P_\alpha^{-1}(t-1) + \beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)\right] \\
&\quad \hat{\theta}(t-1) - \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p)y(t-p) \\
&= \left[I + \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p)\varphi^T(t-p)\right] \\
&\quad \hat{\theta}(t-1) - \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p)y(t-p) \\
&= \hat{\theta}(t-1) - \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p) \\
&\quad \left[y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1)\right]
\end{aligned}$$

From the above equations, we obtain the following algorithm

$$\begin{aligned}
\hat{\theta}(t) &= \alpha(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^T(t)\alpha(t-1)\right] \\
P^{-1}(t) &= \lambda(t)P_\alpha^{-1}(t-1) + \varphi(t)\varphi^T(t) \\
\alpha(t-1) &= \hat{\theta}(t-1) - \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p) \\
&\quad \left[y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1)\right] \\
P_\alpha^{-1}(t-1) &= P^{-1}(t-1) - \beta(t-1,t-p)\varphi(t-p)\varphi^T(t-p)
\end{aligned}$$

By applying the Matrix Inversion Lemma, the weighted fixed memory recursive least squares algorithm can be written as

$$\begin{aligned}
\hat{\theta}(t) &= \alpha(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^T(t)\alpha(t-1)\right] \\
P(t) &= \frac{1}{\lambda(t)}\left[P_\alpha(t-1) + \frac{P_\alpha(t-1)\varphi(t)\varphi^T(t)P_\alpha(t-1)}{\lambda(t) + \varphi^T(t)P_\alpha(t-1)\varphi(t)}\right] \\
\alpha(t-1) &= \hat{\theta}(t-1) - \beta(t-1,t-p)P_\alpha(t-1)\varphi(t-p)\left[y(t-p) - \varphi^T(t-p)\hat{\theta}(t-1)\right] \\
P_\alpha(t-1) &= P(t-1) + \frac{\beta(t-1,t-p)P(t-1)\varphi(t-p)\varphi^T(t-p)P(t-1)}{1 - \beta(t-1,t-p)\varphi^T(t-p)P(t-1)\varphi(t-p)}
\end{aligned}$$

Chapter 5

Recursive extended and generalized least squares identification

5.1 Recursive extended least squares identification

In this section, we first focus on the following ARMAX model

$$A(z)y(t) = B(z)u(t) + D(z)v(t) \tag{5.1.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$ and $D(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned} A\left(z\right) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\ B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\ D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}. \end{aligned}$$

It is assumed that the orders n_a, n_b and n_d are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

Define the extended parameter vector θ and the information vector $\varphi(t)$ containing the noise terms as follows

$$\theta = \left[\begin{array}{cccccccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} & d_1 & d_2 & \cdots & d_{n_d} \end{array} \right]^{\mathrm{T}},$$

$$\begin{aligned} \varphi(t) &= \left[\begin{array}{cccccccc} -y\left(t-1\right) & -y\left(t-2\right) & \cdots & -y\left(t-n_a\right) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \\ v\left(t-1\right) & v\left(t-2\right) & \cdots & v\left(t-n_d\right) \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

It follows from (5.1.1) that

$$y(t) = [1 - A(z)]\, y(t) + B(z)u(t) + D(z)v(t).$$

Substituting the expressions of $A(z)$, $B(z)$ and $D(z)$, gives

$$\begin{aligned} y(t) &= \left(-a_1 z^{-1} - a_2 z^{-2} - \cdots - a_{n_a} z^{-n_a}\right) y(t) \\ &\quad + \left(b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}\right) u(t) \\ &\quad + \left(1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}\right) v(t) \\ &= -a_1 y(t-1) - a_2 y(t-2) - \cdots - a_{n_a} y(t-n_a) \\ &\quad + b_1 u(t-1) + b_2 y(t-2) + \cdots + b_{n_b} y(t-n_b) \\ &\quad + v(t) + d_1 v(t-1) + d_2 v(t-2) + \cdots + d_{n_d} v(t-n_d) \\ &= \varphi^{\mathrm{T}}(t)\theta + v(t), \end{aligned}$$

which is the identification model of the system described by ARMAX model. Due to the nonlinear effect of the θ, this model is called **pseudolinear regression**.

In this first, the identification goal is to estimate the parameters of the system (5.1.1) using the input and output data $\{u(t),\, y(t)\}$. The error criterion function is chosen as

$$\begin{aligned} J(\theta) &= \sum_{i=1}^{t} v^2(i) = \sum_{i=1}^{t} \{A(z)\, y(i) - B(z)u(i) - [D(z) - 1]\, v(i)\}^2 \\ &= \sum_{i=1}^{t} \left(y(i) - \varphi^{\mathrm{T}}(t)\theta\right)^2. \end{aligned}$$

By minimizing this error criterion function, it seems that the following recursive algorithm can be obtained to estimate the parameter vector θ

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right] \\ L(t) &= \frac{P(t-1)\,\varphi(t)}{1 + \varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)} \\ P(t) &= \left[I - L(t)\varphi^{\mathrm{T}}(t)\right] P(t-1),\; P(0) = p_0 I. \end{aligned}$$

However, it is a pity that this algorithm can not be realized since the vector $\varphi(t)$ in the above algorithm contains the unmeasurable noise terms $v(t-i)$. A way to avoid such a case is to replace the noise term $v(t-i)$ by the corresponding estimate.

R-RELS: Residual based Recursive Extended Least Squares algorithm

$$
\begin{array}{rcl}
 & & \text{R-RELS} \\
\hat{\theta}(t) & = & \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right] \\
L(t) & = & \dfrac{P(t-1)\,\hat{\varphi}(t)}{1+\hat{\varphi}^{\mathrm{T}}(t)\,P(t-1)\,\hat{\varphi}(t)} \\
P(t) & = & \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1),\ P(0) = p_0 I, \\
\hat{\varphi}(t) & = & \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \\ \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) & & & & \end{array}\right] \\
\hat{v}(t) & = & y(t) - \hat{\varphi}^{\mathrm{T}}(t)\,\hat{\theta}(t)
\end{array}
$$

I-RELS: Innovation based Recursive Extended Least Squares algorithm

$$
\begin{array}{rcl}
 & & \text{I-RELS} \\
\hat{\theta}(t) & = & \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right] \\
L(t) & = & \dfrac{P(t-1)\,\hat{\varphi}(t)}{1+\hat{\varphi}^{\mathrm{T}}(t)\,P(t-1)\,\hat{\varphi}(t)} \\
P(t) & = & \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1),\ P(0) = p_0 I, \\
\hat{\varphi}(t) & = & \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \\ \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) & & & & \end{array}\right] \\
\hat{v}(t) & = & y(t) - \hat{\varphi}^{\mathrm{T}}(t)\,\hat{\theta}(t-1)
\end{array}
$$

Now we consider the recursive least square algorithm of the MA model. When $u(t) = 0$, and $A(z) = 1$, the ARMAX model (5.1.1) becomes the MA model

$$y(t) = D(z)v(t)$$

or

$$y(t) = v(t) + d_1 v(t-1) + d_2 v(t-2) + \cdots + d_{n_d} v(t-n_d).$$

The corresponding identification model is

$$
\begin{array}{rcl}
y(t) & = & \left[\begin{array}{cccc} v(t-1) & v(t-2) & \cdots & v(t-n_d) \end{array}\right]\left[\begin{array}{c} d_1 \\ d_2 \\ \cdots \\ d_{n_d} \end{array}\right] + v(t) \\
 & = & :\varphi^{\mathrm{T}}(t)\theta + v(t), \\
\varphi(t) & : & = \left[\begin{array}{cccc} v(t-1) & v(t-2) & \cdots & v(t-n_d) \end{array}\right]^{\mathrm{T}}, \\
\theta & : & = \left[\begin{array}{cccc} d_1 & d_2 & \cdots & d_{n_d} \end{array}\right]^{\mathrm{T}}.
\end{array}
$$

R-RELS: Residual based Recursive Extened Least Squares algorithm

$$\begin{array}{rcl} & & \text{R-RELS} \\ \hat{\theta}(t) & = & \hat{\theta}(t-1) + L\left(t\right)\left[y(t) - \hat{\varphi}^{\mathrm{T}}\left(t\right)\hat{\theta}(t-1)\right] \\ L(t) & = & \dfrac{P\left(t-1\right)\hat{\varphi}\left(t\right)}{1+\hat{\varphi}^{\mathrm{T}}\left(t\right)P\left(t-1\right)\hat{\varphi}\left(t\right)} \\ P\left(t\right) & = & \left[I - L(t)\hat{\varphi}^{\mathrm{T}}\left(t\right)\right]P\left(t-1\right),\, P\left(0\right) = p_0 I, \\ \hat{\varphi}(t) & = & \left[\begin{array}{cccc} \hat{v}\left(t-1\right) & \hat{v}\left(t-2\right) & \cdots & \hat{v}\left(t-n_d\right) \end{array}\right] \\ \hat{v}(t) & = & y(t) - \hat{\varphi}^{\mathrm{T}}\left(t\right)\hat{\theta}(t) \end{array}$$

I-RELS: Innovation based Recursive Extened Least Squares algorithm

$$\begin{array}{rcl} & & \text{I-RELS} \\ \hat{\theta}(t) & = & \hat{\theta}(t-1) + L\left(t\right)\left[y(t) - \hat{\varphi}^{\mathrm{T}}\left(t\right)\hat{\theta}(t-1)\right] \\ L(t) & = & \dfrac{P\left(t-1\right)\hat{\varphi}\left(t\right)}{1+\hat{\varphi}^{\mathrm{T}}\left(t\right)P\left(t-1\right)\hat{\varphi}\left(t\right)} \\ P\left(t\right) & = & \left[I - L(t)\hat{\varphi}^{\mathrm{T}}\left(t\right)\right]P\left(t-1\right),\, P\left(0\right) = p_0 I, \\ \hat{\varphi}(t) & = & \left[\begin{array}{cccc} \hat{v}\left(t-1\right) & \hat{v}\left(t-2\right) & \cdots & \hat{v}\left(t-n_d\right) \end{array}\right] \\ \hat{v}(t) & = & y(t) - \hat{\varphi}^{\mathrm{T}}\left(t\right)\hat{\theta}(t-1). \end{array}$$

5.2 Recursive generalized least squares identification

5.2.1 Recursive generalized least squares identification

Consider the ARARX model

$$A(z)y(t) = B(z)u(t) + \frac{1}{C(z)}v(t). \tag{5.2.1}$$

Define the intermediate variable

$$w(t) := \frac{1}{C(z)}v(t). \tag{5.2.2}$$

Then, one has

$$A(z)y(t) = B(z)u(t) + w(t). \tag{5.2.3}$$

Define system parameter vector θ_{s} and noise vector θ_{n}, respectively, as

$$\begin{array}{rcl} \theta_{\mathrm{s}} & = & \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array}\right]^{\mathrm{T}}, \\ \theta_{\mathrm{n}} & = & \left[\begin{array}{cccc} c_1 & c_2 & \cdots & c_{n_c} \end{array}\right]^{\mathrm{T}}; \end{array}$$

and then define the parameter vector θ as

$$\theta = \begin{bmatrix} \theta_{\mathrm{s}} \\ \theta_{\mathrm{n}} \end{bmatrix}.$$

Define system information vector $\varphi_{\mathrm{s}}(t)$ and noise information vector $\varphi_{\mathrm{s}}(t)$, respective, as

$$\begin{aligned} \varphi_{\mathrm{s}}(t) &= \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\ \varphi_{\mathrm{n}}(t) &= \begin{bmatrix} -w(t-1) & -w(t-2) & \cdots & -w(t-n_c) \end{bmatrix}^{\mathrm{T}}, \end{aligned}$$

and then define the information vector $\varphi(t)$ as

$$\varphi(t) = \begin{bmatrix} \varphi_{\mathrm{s}}(t) \\ \varphi_{\mathrm{n}}(t) \end{bmatrix}.$$

By the preceding definition, it follows from (5.2.2) that

$$w(t) = \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\theta_{\mathrm{n}} + v(t),$$

and it can be obtained from (5.2.3) that

$$\begin{aligned} y(t) &= [1 - A(z)]\, y(t) + B(z)u(t) + w(t) \\ &= \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + w(t) \\ &= \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\theta_{\mathrm{n}} + v(t) \\ &= \begin{bmatrix} \varphi_{\mathrm{s}}^{\mathrm{T}}(t) & \varphi_{\mathrm{n}}^{\mathrm{T}}(t) \end{bmatrix} \begin{bmatrix} \theta_{\mathrm{s}} \\ \theta_{\mathrm{n}} \end{bmatrix} + v(t) \\ &= \varphi^{\mathrm{T}}(t)\theta + v(t). \end{aligned} \tag{5.2.4}$$

This is a pseudolinear regressive model since the information vector $\varphi(t)$ contains the unknown intermediate variable $w(t-i)$. In this case, the following standard recursive least square algorithm can not be applied to estimate the parameter vector θ :

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ L(t) &= \frac{P(t-1)\,\varphi(t)}{1 + \varphi^{\mathrm{T}}(t)P(t-1)\,\varphi(t)}, \\ P(t) &= \left[I - L(t)\varphi^{\mathrm{T}}(t)\right] P(t-1),\ P(0) = p_0 I. \end{aligned}$$

By using the idea in the Recursive Extended Least Square identification, the unknown term $w(t-i)$ is replaced by its estimate $\hat{w}(t-i)$, and then the corresponding noise infromation vector is denoted by

$$\hat{\varphi}_{\mathrm{n}}(t) = \begin{bmatrix} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \end{bmatrix}^{\mathrm{T}}.$$

Further, define information vector

$$\hat{\varphi}(t) = \begin{bmatrix} \varphi_{\mathrm{s}}(t) \\ \hat{\varphi}_{\mathrm{n}}(t) \end{bmatrix}.$$

Let

$$\hat{\theta}(t) = \left[\begin{array}{c} \hat{\theta}_{\rm s}(t) \\ \hat{\theta}_{\rm n}(t) \end{array} \right]$$

be the estimate of θ. It follows from (5.2.4) that

$$w(t) = y(t) - \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s}.$$

By replacing $\theta_{\rm s}$ by $\hat{\varphi}_{\rm s}(t)$, we give an estimate $\hat{w}(t)$ of $w(t)$ as

$$\hat{w}(t) = y(t) - \varphi_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}\left(t\right).$$

Please note that $\hat{\varphi}(t)$ is unknown. With the preceding preliminaries, we can obtain the following Recursive Generalized Least Square algorithm to identify the ARARX model (5.2.1).

$$\begin{aligned}
\hat{\theta}\left(t\right) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}(t)\hat{\theta}(t-1)\right], \\
L(t) &= \frac{P\left(t-1\right)\hat{\varphi}(t)}{1+\hat{\varphi}^{\rm T}(t)P\left(t-1\right)\hat{\varphi}(t)}, \\
P(t) &= \left[I - L(t)\hat{\varphi}^{\rm T}(t)\right]P(t-1),\ P(0) = p_0 I, \\
\hat{\varphi}(t) &= \left[\begin{array}{c} \varphi_{\rm s}(t) \\ \hat{\varphi}_{\rm n}(t) \end{array} \right] \\
\varphi_{\rm s}(t) &= \left[\begin{array}{cccccccc} -y\left(t-1\right) & -y\left(t-2\right) & \cdots & -y\left(t-n_a\right) & u(t-1) & u(t-2) & \cdots & u\left(t-n_b\right) \end{array} \right]^{\rm T} \\
\hat{\varphi}_{\rm n}(t) &= \left[\begin{array}{cccc} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \end{array} \right]^{\rm T} \\
\hat{w}(t) &= y(t) - -\varphi_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}\left(t\right), \\
\hat{\theta}\left(t\right) &= \left[\begin{array}{c} \hat{\theta}_{\rm s}(t) \\ \hat{\theta}_{\rm n}(t) \end{array} \right].
\end{aligned}$$

5.2.2 Filtering based Recursive generalized least squares algorithm

Consider the system described by ARARX model investigated in Subsection 5.2.1

$$A(z)y(t) = B(z)u(t) + \frac{1}{C(z)}v(t) \tag{5.2.5}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$ and $C(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned}
A\left(z\right) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\
C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}.
\end{aligned}$$

It is assumed that the orders n_a, n_b and n_c are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

For this system, define the filtered input $u_{\mathrm{f}}(t)$ and filtered output $y_{\mathrm{f}}(t)$, respectively, as

$$\begin{aligned} u_{\mathrm{f}}(t) &: = C(z)u(t), \\ y_{\mathrm{f}}(t) &: = C(z)y(t). \end{aligned}$$

Define the filtered information vector $\varphi_{\mathrm{f}}(t)$, system information vector $\varphi_{\mathrm{s}}(t)$ and noise information vector $\varphi_{\mathrm{n}}(t)$, respectively, as

$$\begin{aligned} \varphi_{\mathrm{f}}(t) &= \left[\begin{array}{cccccccc} -y_{\mathrm{f}}(t-1) & -y_{\mathrm{f}}(t-2) & \cdots & -y_{\mathrm{f}}(t-n_a) & u_{\mathrm{f}}(t-1) & u_{\mathrm{f}}(t-2) & \cdots & u_{\mathrm{f}}(t-n_b) \end{array} \right]^{\mathrm{T}}, \\ \varphi_{\mathrm{s}}(t) &= \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\mathrm{T}}, \\ \varphi_{\mathrm{n}}(t) &= \left[\begin{array}{ccccc} -w(t-1) & -w(t-2) & \cdots & -w(t-n_c) \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

In addition, define system parameter vector θ_{s} and noise vector θ_{n}, respectively, as

$$\begin{aligned} \theta_{\mathrm{s}} &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array} \right]^{\mathrm{T}}, \\ \theta_{\mathrm{n}} &= \left[\begin{array}{ccccc} c_1 & c_2 & \cdots & c_{n_c} \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

Multiplying the both sides of (5.2.5) by $C(z)$, yields

$$A(z)C(z)y(t) = B(z)C(z)u(t) + v(t),$$

or

$$A(z)y_{\mathrm{f}}(t) = B(z)u_{\mathrm{f}}(t) + v(t).$$

This is a equation error type model (ARX model). For this system, one has

$$y_{\mathrm{f}}(t) = \varphi_{\mathrm{f}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + v(t). \tag{5.2.6}$$

Define an intermediate variable

$$w(t) = \frac{1}{C(z)}v(t), \tag{5.2.7}$$

which gives

$$C(z)w(t) = v(t).$$

By this relation, it can be obtained

$$w(t) = \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\theta_{\mathrm{n}} + v(t). \tag{5.2.8}$$

For the two identification models (5.2.6) and (5.2.8), one can apply the recursive least square algorithm to obtain the estimates $\hat{\theta}_{\mathrm{s}}(t)$ and $\hat{\theta}_{\mathrm{n}}(t)$ of θ_{s} and θ_{n}, respectively,

$$\begin{aligned} \hat{\theta}_{\mathrm{s}}(t) &= \hat{\theta}_{\mathrm{s}}(t-1) + L_{\mathrm{f}}(t)\left[y_{\mathrm{f}}(t) - \varphi_{\mathrm{f}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1)\right], \\ L_{\mathrm{f}}(t) &= \frac{P_{\mathrm{f}}(t-1)\varphi_{\mathrm{f}}(t)}{1 + \varphi_{\mathrm{f}}^{\mathrm{T}}(t)P_{\mathrm{f}}(t-1)\varphi_{\mathrm{f}}(t)}, \\ P_{\mathrm{f}}(t) &= \left[I - L_{\mathrm{f}}(t)\varphi_{\mathrm{f}}^{\mathrm{T}}(t)\right]P_{\mathrm{f}}(t-1), \end{aligned}$$

$$\begin{aligned}
\hat{\theta}_{\mathrm{n}}(t) &= \hat{\theta}_{\mathrm{n}}(t-1) + L_{\mathrm{n}}(t)\left[w(t) - \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{n}}(t-1)\right], \\
L_{\mathrm{n}}(t) &= \frac{P_{\mathrm{n}}(t-1)\,\varphi_{\mathrm{n}}(t)}{1+\varphi_{\mathrm{n}}^{\mathrm{T}}(t)P_{\mathrm{n}}(t-1)\,\varphi_{\mathrm{n}}(t)}, \\
P_{\mathrm{n}}(t) &= \left[I - L_{\mathrm{n}}(t)\varphi_{\mathrm{n}}^{\mathrm{T}}(t)\right]P_{\mathrm{n}}(t-1).
\end{aligned}$$

Since the polynomial $C(z)$ is unknown, then $u_{\mathrm{f}}(t)$ and $y_{\mathrm{f}}(t)$ are unknown, and thus the information vector $\varphi_{\mathrm{f}}(t)$ is also unknown. In addition, the noise information vector $\varphi_{\mathrm{n}}(t)$ is also unknown. The above algorithm is not realized since the expressions in this algorithm contains known $\varphi_{\mathrm{f}}(t)$ and $\varphi_{\mathrm{n}}(t)$.

It follows from (5.2.5) and (5.2.7) that

$$\begin{aligned}
w(t) &= A(z)y(t) - B(z)u(t) \\
&= y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) \\
&\quad -b_1 u(t-1) - b_2 u(t-2) - \cdots - b_{n_b} u(t-n_b) \\
&= y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}}.
\end{aligned}$$

Replacing θ_{s} by its estimate $\hat{\theta}_{\mathrm{s}}(t-1)$, one can obtain an estimate $\hat{w}(t)$ of $w(t)$ as

$$\hat{w}(t) = y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}\,(t-1)\,.$$

Then one can construct the estimate $\hat{\varphi}_{\mathrm{n}}(t)$ of $\varphi_{\mathrm{n}}(t)$

$$\hat{\varphi}_{\mathrm{n}}(t) = \left[\begin{array}{cccc} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}\,(t-n_c) \end{array}\right]^{\mathrm{T}}.$$

With these preliminaries, one can obtain the estimate $\hat{\theta}_{\mathrm{n}}(t) = \left[\begin{array}{cccc} \hat{c}_1(t) & \hat{c}_2(t) & \cdots & \hat{c}_{n_c}(t) \end{array}\right]^{\mathrm{T}}$. Thus, the estimate $\hat{C}(t,z)$ of $C(z)$ can be given by

$$\hat{C}(t,z) = 1 + \hat{c}_1\,(t)\,z^{-1} + \hat{c}_2\,(t)\,z^{-2} + \cdots + \hat{c}_{n_c}\,(t)\,z^{-n_c}.$$

By using this estimated polynomial $\hat{C}(t,z)$, one can obtain the estimate of the filtered input and ouput as

$$\begin{aligned}
\hat{u}_{\mathrm{f}}(t) &= \hat{C}(t,z)u(t) \\
&= u(t) + \hat{c}_1\,(t)\,u(t-1) + \hat{c}_2\,(t)\,u(t-2) + \cdots + \hat{c}_{n_c}\,(t)\,u(t-n_c); \\
\hat{y}_{\mathrm{f}}(t) &= \hat{C}(t,z)y(t) \\
&= y(t) + \hat{c}_1\,(t)\,y(t-1) + \hat{c}_2\,(t)\,y(t-2) + \cdots + \hat{c}_{n_c}\,(t)\,y(t-n_c).
\end{aligned}$$

Now, by using the estimated fitlered input and output one can construct the estimate $\hat{\varphi}_{\mathrm{f}}(t)$ of the fitered system information vector $\varphi_{\mathrm{f}}(t)$ as

$$\hat{\varphi}_{\mathrm{f}}(t) = \left[\begin{array}{cccccccc} -\hat{y}_{\mathrm{f}}\,(t-1) & -\hat{y}_{\mathrm{f}}\,(t-2) & \cdots & -\hat{y}_{\mathrm{f}}\,(t-n_a) & \hat{u}_{\mathrm{f}}(t-1) & \hat{u}_{\mathrm{f}}(t-2) & \cdots & \hat{u}_{\mathrm{f}}\,(t-n_b) \end{array}\right]^{\mathrm{T}}.$$

Now we can obtain the Filtering based Recursive Generalized Least Squares algorithm by replacing $\varphi_{\mathrm{f}}(t)$, $\varphi_{\mathrm{n}}(t)$ and $w(t)$ with their estimates $\hat{\varphi}_{\mathrm{f}}(t)$, $\hat{\varphi}_{\mathrm{n}}(t)$ and $\hat{w}(t)$, respectively, as

$$
\begin{aligned}
\hat{\theta}_{\mathrm{s}}(t) &= \hat{\theta}_{\mathrm{s}}(t-1) + L_{\mathrm{f}}(t)\left[\hat{y}_{\mathrm{f}}(t) - \hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1)\right], \\
L_{\mathrm{f}}(t) &= \frac{P_{\mathrm{f}}(t-1)\,\hat{\varphi}_{\mathrm{f}}(t)}{1+\hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)P_{\mathrm{f}}(t-1)\,\hat{\varphi}_{\mathrm{f}}(t)}, \\
P_{\mathrm{f}}(t) &= \left[I - L_{\mathrm{f}}(t)\hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)\right] P_{\mathrm{f}}(t-1),\ P_{\mathrm{f}}(0) = p_0 I, \\
\hat{\varphi}_{\mathrm{f}}(t) &= \begin{bmatrix} -\hat{y}_{\mathrm{f}}(t-1) & -\hat{y}_{\mathrm{f}}(t-2) & \cdots & -\hat{y}_{\mathrm{f}}(t-n_a) \\ \hat{u}_{\mathrm{f}}(t-1) & \hat{u}_{\mathrm{f}}(t-2) & \cdots & \hat{u}_{\mathrm{f}}(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\
\hat{y}_{\mathrm{f}}(t) &= y(t) + \hat{c}_1(t)\,y(t-1) + \hat{c}_2(t)\,y(t-2) + \cdots + \hat{c}_{n_c}(t)\,y(t-n_c) \\
\hat{u}_{\mathrm{f}}(t) &= u(t) + \hat{c}_1(t)\,u(t-1) + \hat{c}_2(t)\,u(t-2) + \cdots + \hat{c}_{n_c}(t)\,u(t-n_c)
\end{aligned}
$$

$$
\begin{aligned}
\hat{\theta}_{\mathrm{n}}(t) &= \hat{\theta}_{\mathrm{n}}(t-1) + L_{\mathrm{n}}(t)\left[\hat{w}(t) - \hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{n}}(t-1)\right], \\
L_{\mathrm{n}}(t) &= \frac{P_{\mathrm{n}}(t-1)\,\hat{\varphi}_{\mathrm{n}}(t)}{1+\hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)P_{\mathrm{n}}(t-1)\,\hat{\varphi}_{\mathrm{n}}(t)}, \\
P_{\mathrm{n}}(t) &= \left[I - L_{\mathrm{n}}(t)\hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\right] P_{\mathrm{n}}(t-1),\ P_{\mathrm{n}}(0) = p_0 I, \\
\hat{\varphi}_{\mathrm{n}}(t) &= \begin{bmatrix} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \end{bmatrix}^{\mathrm{T}}, \\
\hat{w}(t) &= y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1), \\
\hat{\theta}_{\mathrm{s}}(t) &= \begin{bmatrix} \hat{a}_1(t) & \hat{a}_2(t) & \cdots & \hat{a}_{n_a}(t) & \hat{b}_1(t) & \hat{b}_2(t) & \cdots & \hat{b}_{n_b}(t) \end{bmatrix}^{\mathrm{T}}, \\
\hat{\theta}_{\mathrm{n}}(t) &= \begin{bmatrix} \hat{c}_1(t) & \hat{c}_2(t) & \cdots & \hat{c}_{n_c}(t) \end{bmatrix}^{\mathrm{T}}.
\end{aligned}
$$

5.3 Recursive Generalized Extended Least Squares algorithm

5.3.1 Recursive generalized extended least squares algorithm

Consider the following system described by ARARMAX model

$$A(z)y(t) = B(z)u(t) + \frac{D(z)}{C(z)}v(t) \tag{5.3.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$, $C(z)$ and $D(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$
\begin{aligned}
A(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\
C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}; \\
D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}.
\end{aligned}
$$

It is assumed that the orders n_a, n_b, n_c and n_d are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$. We denote $n = n_a + n_b + n_c + n_d$. In this model,

$$w(t) := \frac{D(z)}{C(z)} v(t) \tag{5.3.2}$$

is the ARMA noise. For system (5.3.1), we definite the following parameter vectors and information vectors

$$\begin{aligned}
\theta &= \begin{bmatrix} \theta_{\mathrm{s}} \\ \theta_{\mathrm{n}} \end{bmatrix} \in \mathbb{R}^n, \\
\theta_{\mathrm{s}} &= \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{bmatrix}^{\mathrm{T}}, \\
\theta_{\mathrm{n}} &= \begin{bmatrix} c_1 & c_2 & \cdots & c_{n_c} & d_1 & d_2 & \cdots & d_{n_d} \end{bmatrix}^{\mathrm{T}}, \\
\varphi(t) &= \begin{bmatrix} \varphi_{\mathrm{s}}(t) \\ \varphi_{\mathrm{n}}(t) \end{bmatrix}, \\
\varphi_{\mathrm{s}}(t) &= \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\
\varphi_{\mathrm{n}}(t) &= \begin{bmatrix} -w(t-1) & -w(t-2) & \cdots & -w(t-n_c) & v(t-1) & v(t-2) & \cdots & v(t-n_d) \end{bmatrix}^{\mathrm{T}}.
\end{aligned}$$

It follows from (5.3.2) and the defined notations that

$$\begin{aligned}
w(t) &= [1 - C(z)]\, w(t) + D(z)v(t) \\
&= \left(-c_1 z^{-1} - c_2 z^{-2} - \cdots - c_{n_c} z^{-n_c}\right) w(t) + \\
&\quad v(t) + d_1 z^{-1} v(t) + d_2 z^{-2} v(t) + \cdots + d_{n_d} z^{-n_d} v(t) \\
&= \theta_{\mathrm{n}}^{\mathrm{T}}(t)\, \theta_{\mathrm{n}} + v(t).
\end{aligned}$$

In addition, it follows from (5.3.1) that

$$\begin{aligned}
y(t) &= \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + w(t) \\
&= \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\, \theta_{\mathrm{n}} + v(t).
\end{aligned} \tag{5.3.3}$$

This is an pseudolinear regression model, and thus the following standard recursive least square can not be used:

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
L(t) &= \frac{P(t-1)\,\varphi(t)}{1 + \varphi^{\mathrm{T}}(t) P(t-1)\,\varphi(t)}, \\
P(t) &= \left[I - L(t)\varphi^{\mathrm{T}}(t)\right] P(t-1),\ P(0) = p_0 I.
\end{aligned}$$

With the aid of the idea of recursive extended least squares identification, the unmeasured terms $w(t-i)$ and $v(t-i)$ in the information vector $\varphi_{\mathrm{n}}(t)$ will be replaced with their estimate $\hat{w}(t-i)$ and $\hat{v}(t-i)$, respectively.

It follows from (5.3.3) that

$$w(t) = y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}}.$$

Thus, one can obtain an estimate $\hat{w}(t)$ of $w(t)$ by replacing θ_{s} with its estimate $\hat{\theta}_{\mathrm{s}}(t)$

$$\hat{w}(t) = y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t).$$

In addition, it follows from (5.3.3) that

$$v(t) = y(t) - \varphi^{\mathrm{T}}(t)\theta.$$

Thus, an estimate $\hat{v}(t)$ of $v(t)$ can be otained by replacing θ and $\varphi(t)$ with its estimate $\hat{\theta}(t)$ and $\hat{\varphi}(t)$, respectively

$$\hat{v}(t) = y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t).$$

An estimate $\hat{v}(t)$ of $v(t)$ can be obtained from

$$\hat{v}(t) = \hat{w}(t) - \hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\,\hat{\theta}_{\mathrm{n}}(t).$$

With the preceding preliminaries, we can obtain the following **Recursive Generalized Extened Least Squares** identification algorithm:

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
L(t) &= \frac{P(t-1)\,\hat{\varphi}(t)}{1 + \hat{\varphi}^{\mathrm{T}}(t)P(t-1)\,\hat{\varphi}(t)}, \\
P(t) &= \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1),\ P(0) = p_0 I, \\
\hat{\theta}(t) &= \begin{bmatrix} \hat{\theta}_{\mathrm{s}}(t) \\ \hat{\theta}_{\mathrm{n}}(t) \end{bmatrix},\ \hat{\varphi}(t) = \begin{bmatrix} \varphi_{\mathrm{s}}(t) \\ \hat{\varphi}_{\mathrm{n}}(t) \end{bmatrix}, \\
\varphi_{\mathrm{s}}(t) &= \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\
\hat{\varphi}_{\mathrm{n}}(t) &= \begin{bmatrix} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) & \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) \end{bmatrix}^{\mathrm{T}}, \\
\hat{w}(t) &= y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t), \\
\hat{v}(t) &= \hat{w}(t) - \hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\,\hat{\theta}_{\mathrm{n}}(t).
\end{aligned}$$

In fact, this algorithm is based on the residual. The innovation based algorithm can be easily obtained by replacing $\hat{\theta}_{\mathrm{s}}(t)$ and $\hat{\theta}_{\mathrm{n}}(t)$ with $\hat{\theta}_{\mathrm{s}}(t-1)$ and $\hat{\theta}_{\mathrm{n}}(t-1)$, respectively.

5.3.2 Filtering based Recursive generalized least squares algorithm

Consider the system described by ARARX model investigated in Subsection 5.2.1

$$A(z)y(t) = B(z)u(t) + \frac{D(z)}{C(z)}v(t) \tag{5.3.4}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$ and $C(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned} A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\ B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\ C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}; \\ D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}. \end{aligned}$$

It is assumed that the orders n_a, n_b and n_c are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

For this system, define the filtered input $u_{\mathrm{f}}(t)$ and filtered output $y_{\mathrm{f}}(t)$, respectively, as

$$\begin{aligned} u_{\mathrm{f}}(t) &:= \frac{C(z)}{D(z)} u(t), \\ y_{\mathrm{f}}(t) &:= \frac{C(z)}{D(z)} y(t). \end{aligned}$$

Define the filtered information vector $\varphi_{\mathrm{f}}(t)$, system information vector $\varphi_{\mathrm{s}}(t)$ and noise information vector $\varphi_{\mathrm{n}}(t)$, respectively, as

$$\begin{aligned} \varphi_{\mathrm{f}}(t) &= \left[\begin{array}{cccccccc} -y_{\mathrm{f}}\,(t-1) & -y_{\mathrm{f}}\,(t-2) & \cdots & -y_{\mathrm{f}}\,(t-n_a) & u_{\mathrm{f}}(t-1) & u_{\mathrm{f}}(t-2) & \cdots & u_{\mathrm{f}}\,(t-n_b) \end{array} \right]^{\mathrm{T}}, \\ \varphi_{\mathrm{s}}(t) &= \left[\begin{array}{cccccccc} -y\,(t-1) & -y\,(t-2) & \cdots & -y\,(t-n_a) & u(t-1) & u(t-2) & \cdots & u\,(t-n_b) \end{array} \right]^{\mathrm{T}}, \\ \varphi_{\mathrm{n}}(t) &= \left[\begin{array}{cccc} -w(t-1) & -w(t-2) & \cdots & -w\,(t-n_c) \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

In addition, define system parameter vector θ_{s} and noise vector θ_{n}, respectively, as

$$\begin{aligned} \theta_{\mathrm{s}} &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array} \right]^{\mathrm{T}}, \\ \theta_{\mathrm{n}} &= \left[\begin{array}{cccccccc} c_1 & c_2 & \cdots & c_{n_c} & d_1 & d_2 & \cdots & d_{n_d} \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

Multiplying the both sides of (5.3.4) by $C(z)$, yields

$$A(z)\frac{C(z)}{D(z)} y(t) = B(z)\frac{C(z)}{D(z)} u(t) + v(t),$$

or

$$A(z) y_{\mathrm{f}}(t) = B(z) u_{\mathrm{f}}(t) + v(t).$$

This is a equation error type model (ARX model). For this system, one has

$$y_{\mathrm{f}}(t) = \varphi_{\mathrm{f}}^{\mathrm{T}}(t)\theta_{\mathrm{s}} + v(t). \tag{5.3.5}$$

Define an intermediate variable

$$w(t) = \frac{D(z)}{C(z)} v(t), \tag{5.3.6}$$

which gives

$$C(z)w(t) = D(z)v(t).$$

By this relation, it can be obtained

$$w(t) = \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\theta_{\mathrm{n}} + v(t). \tag{5.3.7}$$

For the two identification models (5.3.5) and (5.3.7), one can apply the recursive least square algorithm to obtain the estimates $\hat{\theta}_{\mathrm{s}}(t)$ and $\hat{\theta}_{\mathrm{n}}(t)$ of θ_{s} and θ_{n}, respectively,

$$\begin{aligned}
\hat{\theta}_{\mathrm{s}}(t) &= \hat{\theta}_{\mathrm{s}}(t-1) + L_{\mathrm{f}}(t)\left[y_{\mathrm{f}}(t) - \varphi_{\mathrm{f}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1)\right], \\
L_{\mathrm{f}}(t) &= \frac{P_{\mathrm{f}}(t-1)\varphi_{\mathrm{f}}(t)}{1+\varphi_{\mathrm{f}}^{\mathrm{T}}(t)P_{\mathrm{f}}(t-1)\varphi_{\mathrm{f}}(t)}, \\
P_{\mathrm{f}}(t) &= \left[I - L_{\mathrm{f}}(t)\varphi_{\mathrm{f}}^{\mathrm{T}}(t)\right]P_{\mathrm{f}}(t-1),
\end{aligned}$$

$$\begin{aligned}
\hat{\theta}_{\mathrm{n}}(t) &= \hat{\theta}_{\mathrm{n}}(t-1) + L_{\mathrm{n}}(t)\left[w(t) - \varphi_{\mathrm{n}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{n}}(t-1)\right], \\
L_{\mathrm{n}}(t) &= \frac{P_{\mathrm{n}}(t-1)\varphi_{\mathrm{n}}(t)}{1+\varphi_{\mathrm{n}}^{\mathrm{T}}(t)P_{\mathrm{n}}(t-1)\varphi_{\mathrm{n}}(t)}, \\
P_{\mathrm{n}}(t) &= \left[I - L_{\mathrm{n}}(t)\varphi_{\mathrm{n}}^{\mathrm{T}}(t)\right]P_{\mathrm{n}}(t-1).
\end{aligned}$$

Since the polynomials $C(z)$ and $D(z)$ are unknown, then $u_{\mathrm{f}}(t)$ and $y_{\mathrm{f}}(t)$ are unknown, and thus the information vector $\varphi_{\mathrm{f}}(t)$ is also unknown. In addition, the noise information vector $\varphi_{\mathrm{n}}(t)$ is also known. The above algorithm is not realized since the expressions in this algorithm contains known $\varphi_{\mathrm{f}}(t)$ and $\varphi_{\mathrm{n}}(t)$.

It follows from (5.3.4) and (5.3.6) that

$$\begin{aligned}
w(t) &= A(z)y(t) - B(z)u(t) \\
&= y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) \\
&\quad -b_1 u(t-1) - b_2 u(t-2) - \cdots - b_{n_b} u(t-n_b) \\
&= y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\theta_{\mathrm{s}}.
\end{aligned}$$

Replacing θ_{s} by its estimate $\hat{\theta}_{\mathrm{s}}(t-1)$, one can obtain an estimate $\hat{w}(t)$ of $w(t)$ as

$$\hat{w}(t) = y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1).$$

Then one can construct the estimate $\hat{\varphi}_{\mathrm{n}}(t)$ of $\varphi_{\mathrm{n}}(t)$

$$\hat{\varphi}_{\mathrm{n}}(t) = \left[\begin{array}{cccccccc} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) & \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) \end{array}\right]^{\mathrm{T}}.$$

With these preliminaries, one can obtain the estimate

$$\hat{\theta}_{\mathrm{n}}(t) = \left[\begin{array}{cccccccc} \hat{c}_1(t) & \hat{c}_2(t) & \cdots & \hat{c}_{n_c}(t) & \hat{d}_1(t) & \hat{d}_2(t) & \cdots & \hat{d}_{n_d}(t) \end{array}\right]^{\mathrm{T}}.$$

. Thus, the estimates $\hat{C}(t,z)$ and $\hat{D}(t,z)$ of $C(z)$ and $D(z)$ can be given, respectively, by

$$\begin{aligned}\hat{C}(t,z) &= 1+\hat{c}_1(t)\,z^{-1}+\hat{c}_2(t)\,z^{-2}+\cdots+\hat{c}_{n_c}(t)\,z^{-n_c},\\ \hat{D}(t,z) &= 1+\hat{d}_1(t)\,z^{-1}+\hat{d}_2(t)\,z^{-2}+\cdots+\hat{d}_{n_d}(t)\,z^{-n_d}.\end{aligned}$$

By using this estimated polynomials $\hat{C}(t,z)$ and $\hat{D}(t)$, one can obtain the estimate of the filtered input and ouput as

$$\begin{aligned}\hat{u}_{\mathrm{f}}(t) &= \frac{\hat{C}(t,z)}{\hat{D}(t,z)}u(t);\\ \hat{y}_{\mathrm{f}}(t) &= \frac{\hat{C}(t,z)}{\hat{D}(t,z)}y(t)\\ &= y(t)+\hat{c}_1(t)\,y(t-1)+\hat{c}_2(t)\,y(t-2)+\cdots+\hat{c}_{n_c}(t)\,y(t-n_c).\end{aligned}$$

This implies

$$\hat{D}(t,z)\,\hat{u}_{\mathrm{f}}(t)=\hat{C}(t,z)u(t),$$

and

$$\hat{D}(t,z)\,\hat{y}_{\mathrm{f}}(t)=\hat{C}(t,z)y(t).$$

It follows from the preceding two relations that

$$\begin{aligned}\hat{u}_{\mathrm{f}}(t) &= -\hat{d}_1(t)\hat{u}_{\mathrm{f}}(t-1)-\hat{d}_2(t)\hat{u}_{\mathrm{f}}(t-2)-\cdots-\hat{d}_{n_d}(t)\hat{u}_{\mathrm{f}}(t-n_d)\\ &\quad +u(t)+\hat{c}_1(t)\,u(t-1)+\hat{c}_2(t)\,u(t-2)+\cdots+\hat{c}_{n_c}(t)\,u(t-n_c);\\ \hat{y}_{\mathrm{f}}(t) &= -\hat{d}_1(t)\hat{y}_{\mathrm{f}}(t-1)-\hat{d}_2(t)\hat{y}_{\mathrm{f}}(t-2)-\cdots-\hat{d}_{n_d}(t)\hat{y}_{\mathrm{f}}(t-n_d)\\ &\quad +y(t)+\hat{c}_1(t)\,y(t-1)+\hat{c}_2(t)\,y(t-2)+\cdots+\hat{c}_{n_c}(t)\,y(t-n_c).\end{aligned}$$

Now, by using the estimated fitlered input and output one can construct the estimate $\hat{\varphi}_{\mathrm{f}}(t)$ of the fitered system information vector $\varphi_{\mathrm{f}}(t)$ as

$$\hat{\varphi}_{\mathrm{f}}(t)=\left[\begin{array}{ccccccccc}-\hat{y}_{\mathrm{f}}(t-1) & -\hat{y}_{\mathrm{f}}(t-2) & \cdots & -\hat{y}_{\mathrm{f}}(t-n_a) & \hat{u}_{\mathrm{f}}(t-1) & \hat{u}_{\mathrm{f}}(t-2) & \cdots & \hat{u}_{\mathrm{f}}(t-n_b)\end{array}\right]^{\mathrm{T}}.$$

Now we can obtain the Filtering based Recursive Generalized Least Squares algorithm by replacing $\varphi_{\mathrm{f}}(t)$, $\varphi_{\mathrm{n}}(t)$ and $w(t)$ with their estimates $\hat{\varphi}_{\mathrm{f}}(t)$, $\hat{\varphi}_{\mathrm{n}}(t)$ and $\hat{w}(t)$, respectively, as

$$
\begin{aligned}
\hat{\theta}_{\mathrm{s}}(t) &= \hat{\theta}_{\mathrm{s}}(t-1) + L_{\mathrm{f}}(t)\left[\hat{y}_{\mathrm{f}}(t) - \hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1)\right], \\
L_{\mathrm{f}}(t) &= \frac{P_{\mathrm{f}}(t-1)\,\hat{\varphi}_{\mathrm{f}}(t)}{1+\hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)P_{\mathrm{f}}(t-1)\,\hat{\varphi}_{\mathrm{f}}(t)}, \\
P_{\mathrm{f}}(t) &= \left[I - L_{\mathrm{f}}(t)\hat{\varphi}_{\mathrm{f}}^{\mathrm{T}}(t)\right] P_{\mathrm{f}}(t-1),\ P_{\mathrm{f}}(0) = p_0 I, \\
\hat{\varphi}_{\mathrm{f}}(t) &= \left[\begin{array}{cccc} -\hat{y}_{\mathrm{f}}(t-1) & -\hat{y}_{\mathrm{f}}(t-2) & \cdots & -\hat{y}_{\mathrm{f}}(t-n_a) \\ \hat{u}_{\mathrm{f}}(t-1) & \hat{u}_{\mathrm{f}}(t-2) & \cdots & \hat{u}_{\mathrm{f}}(t-n_b) \end{array}\right]^{\mathrm{T}}, \\
\hat{y}_{\mathrm{f}}(t) &= -\hat{d}_1(t)\hat{y}_{\mathrm{f}}(t-1) - \hat{d}_2(t)\hat{y}_{\mathrm{f}}(t-2) - \cdots - \hat{d}_{n_d}(t)\hat{y}_{\mathrm{f}}(t-n_d) + \\
&\quad y(t) + \hat{c}_1(t)\,y(t-1) + \hat{c}_2(t)\,y(t-2) + \cdots + \hat{c}_{n_c}(t)\,y(t-n_c) \\
\hat{u}_{\mathrm{f}}(t) &= -\hat{d}_1(t)\hat{u}_{\mathrm{f}}(t-1) - \hat{d}_2(t)\hat{u}_{\mathrm{f}}(t-2) - \cdots - \hat{d}_{n_d}(t)\hat{u}_{\mathrm{f}}(t-n_d) + \\
&\quad u(t) + \hat{c}_1(t)\,u(t-1) + \hat{c}_2(t)\,u(t-2) + \cdots + \hat{c}_{n_c}(t)\,u(t-n_c)
\end{aligned}
$$

$$
\begin{aligned}
\hat{\theta}_{\mathrm{n}}(t) &= \hat{\theta}_{\mathrm{n}}(t-1) + L_{\mathrm{n}}(t)\left[\hat{w}(t) - \hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{n}}(t-1)\right], \\
L_{\mathrm{n}}(t) &= \frac{P_{\mathrm{n}}(t-1)\,\hat{\varphi}_{\mathrm{n}}(t)}{1+\hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)P_{\mathrm{n}}(t-1)\,\hat{\varphi}_{\mathrm{n}}(t)}, \\
P_{\mathrm{n}}(t) &= \left[I - L_{\mathrm{n}}(t)\hat{\varphi}_{\mathrm{n}}^{\mathrm{T}}(t)\right] P_{\mathrm{n}}(t-1),\ P_{\mathrm{n}}(0) = p_0 I, \\
\hat{\varphi}_{\mathrm{n}}(t) &= \left[\begin{array}{ccccccccc} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) & \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) \end{array}\right]^{\mathrm{T}}, \\
\hat{w}(t) &= y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t-1), \\
\hat{\theta}_{\mathrm{s}}(t) &= \left[\begin{array}{ccccccccc} \hat{a}_1(t) & \hat{a}_2(t) & \cdots & \hat{a}_{n_a}(t) & \hat{b}_1(t) & \hat{b}_2(t) & \cdots & \hat{b}_{n_b}(t) \end{array}\right]^{\mathrm{T}}, \\
\hat{\theta}_{\mathrm{n}}(t) &= \left[\begin{array}{ccccccccc} \hat{c}_1(t) & \hat{c}_2(t) & \cdots & \hat{c}_{n_c}(t) & \hat{d}_1(t) & \hat{d}_2(t) & \cdots & \hat{d}_{n_d}(t) \end{array}\right]^{\mathrm{T}}.
\end{aligned}
$$

5.3.3 Multi-stage Least Squares Identification

It is assumed that the considered system is described by the following ARARMAX model

$$A(z)\,y(t) = B(z)u(t) + \frac{D(z)}{C(z)}v(t), \tag{5.3.8}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$ and $C(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$
\begin{aligned}
A(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\
C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}; \\
D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}.
\end{aligned}
$$

It is assumed that the orders n_a, n_b and n_c are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

The first stage is to estimate the parameters of the transfered the model with higher orders.

It is assumed that the order (n_a, n_b, n_c, n_d) is known, and $A(z)$ and $B(z)$ have no common factor. Denote

$$\begin{cases} P(z) = A(z)C(z) = 1 + p_1 z^{-1} + \cdots + p_{n_a+n_c} z^{-(n_a+n_c)}, \\ Q(z) = B(z)C(z) = q_1 z^{-1} + q_2 z^{-2} + \cdots + q_{n_b+n_c} z^{-(n_b+n_c)}. \end{cases} \tag{5.3.9}$$

With this, the model (5.3.8) can be equivalently written as

$$P(z)y(t) = Q(z)u(t) + D(z)v(t). \tag{5.3.10}$$

Denote

$$\begin{aligned} \theta_1 &= \text{coeff}\,[P(z), Q(z), D(z)] \\ &= \left[\begin{array}{cccccccccccc} p_1 & p_2 & \cdots & p_{n_a+n_c} & q_1 & q_2 & \cdots & q_{n_b+n_c} & d_1 & d_2 & \cdots & d_{n_d} \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^{n_a+n_b+2n_c+n_d}, \end{aligned}$$

$$\begin{aligned} \varphi_1(t) = \Big[& -y(t-1) \quad -y(t-2) \quad \cdots \quad -y(t-n_a-n_c) \\ & u(t-1) \quad u(t-2) \quad \cdots \quad u(t-n_b-n_c) \\ & v(t-1) \quad v(t-2) \quad \cdots \quad v(t-n_d) \Big]^{\mathrm{T}}. \end{aligned}$$

The ARMAX model (5.3.10) can be transfered into the following psudolinear regression model

$$y(t) = \varphi_1^{\mathrm{T}}(t)\theta_1 + v(t).$$

For this model, the following algorithm can be used to give an unbiased estimate of parameter θ_1.

$$\hat{\theta}_1(t) = \hat{\theta}_1(t-1) + L(t)\left[y(t) - \hat{\varphi}_1^{\mathrm{T}}(t)\hat{\theta}_1(t-1)\right], \tag{5.3.11}$$

$$L(t) = \frac{P(t-1)\hat{\varphi}_1(t)}{1 + \hat{\varphi}_1^{\mathrm{T}}(t)P(t-1)\hat{\varphi}_1(t)}, \tag{5.3.12}$$

$$P(t) = [I - L(t)\hat{\varphi}_1^{\mathrm{T}}(t)]P(t-1), \tag{5.3.13}$$

$$\begin{aligned} \hat{\varphi}_1(t) = \Big[& -y(t-1) \quad -y(t-2) \quad \cdots \quad -y(t-n_a-n_c) \\ & u(t-1) \quad u(t-2) \quad \cdots \quad u(t-n_b-n_c) \\ & \hat{v}(t-1) \quad \hat{v}(t-2) \quad \cdots \quad \hat{v}(t-n_d) \Big]^{\mathrm{T}}, \end{aligned} \tag{5.3.14}$$

$$\hat{v}(t) = y(t) - \hat{\varphi}_1^{T}(t)\hat{\theta}_1(t). \tag{5.3.15}$$

The second stage is to give estimate of the parameters of the system model. It follows from (5.3.9) that

$$A(z)Q(z) = B(z)P(z).$$

By substituting the expressions of the polynomials $A(z)$, $B(z)$, $P(z)$, $Q(z)$ into the above equation, one has

$$\begin{aligned} & \left(1 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a}\right)\left(q_1 z^{-1} + \cdots + q_{n_b+n_c} z^{-(n_b+n_c)}\right) \\ = \; & \left(b_1 z^{-1} + b_2 z^{-2} \cdots + b_{n_b} z^{-n_b}\right)\left(1 + p_1 z^{-1} + \cdots + p_{n_a+n_c} z^{-(n_a+n_c)}\right). \end{aligned} \tag{5.3.16}$$

Denote

$$\begin{aligned}
\theta_{\mathrm{s}} &= \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{bmatrix}^{\mathrm{T}}, \\
Y_2 &= \begin{bmatrix} q_1 & q_2 & \cdots & q_{n_b+n_c} & 0 & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{(n_a+n_b+n_c)}, \\
H_2 &= \begin{bmatrix} -S_q & S_p \end{bmatrix} \in \mathbb{R}^{(n_a+n_b+n_c)\times(n_a+n_b)},
\end{aligned}$$

$$S_q = \begin{bmatrix}
0 & 0 & 0 & \cdots & 0 \\
q_1 & 0 & 0 & & \vdots \\
q_2 & q_1 & 0 & \ddots & 0 \\
q_3 & q_2 & q_1 & \ddots & 0 \\
\vdots & & & \ddots & 0 \\
q_{n_q} & & & & q_1 \\
0 & q_{n_q} & & & q_2 \\
\vdots & 0 & q_{n_q} & & q_3 \\
\vdots & & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & q_{n_q}
\end{bmatrix} \in \mathbb{R}^{(n_a+n_b+n_c)\times n_a},\ n_q = n_b + n_c,$$

$$S_p = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
p_1 & 1 & 0 & & \vdots \\
p_2 & p_1 & 1 & \ddots & 0 \\
p_3 & p_2 & p_1 & \ddots & 0 \\
\vdots & & & \ddots & 1 \\
p_{n_p} & & & & p_1 \\
0 & p_{n_p} & & & p_2 \\
\vdots & 0 & p_{n_p} & & p_3 \\
\vdots & & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & p_{n_p}
\end{bmatrix} \in \mathbb{R}^{(n_a+n_b+n_c)\times n_b}, n_p = n_{a+n_c}.$$

Then, by comparing the coefficients of powers of z^{-1} in (5.3.16) one has

$$Y_2 = H_2\theta_{\mathrm{s}}.$$

From this, the least squares estimation of parameter vector θ_{s} can be given by

$$\hat{\theta}_{\mathrm{s}} = (H_2^{\mathrm{T}}H_2)^{-1}H_2^{\mathrm{T}}Y_2.$$

where (q_i, p_i) are replaced with the estimate obtained in the algorithm (5.3.11) – (5.3.15). Since $A(s)$ and $B(s)$ have no common factor, the estimate $\hat{\theta}_{\mathrm{s}}$ is unique.

The third stage is to give the estimate of the noise model.

Denote

$$\theta_3 = \left[\begin{array}{cccc} c_1 & c_2 & \cdots & c_{n_c} \end{array} \right]^{\mathrm{T}}.$$

It follows from $P(z) = A(z)C(z), Q(z) = B(z)C(z)$, that by comparing the coefficients of powers of z^{-1}

$$Y_3 = H_3\theta_3, \tag{5.3.17}$$

$$H_3 = \left[\begin{array}{ccccc} 1 & 0 & 0 & \cdots & 0 \\ a_1 & 1 & 0 & & \vdots \\ a_2 & a_1 & 1 & \ddots & 0 \\ a_3 & a_2 & a_1 & \ddots & 0 \\ \vdots & & & \ddots & 1 \\ a_{n_a} & & & & a_1 \\ 0 & a_{n_a} & & & a_2 \\ \vdots & 0 & a_{n_a} & & a_3 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{n_a} \\ 0 & 0 & 0 & \cdots & 0 \\ b_1 & 0 & 0 & & \vdots \\ b_2 & b_1 & 0 & \ddots & 0 \\ b_3 & b_2 & b_1 & \ddots & 0 \\ \vdots & & & \ddots & 0 \\ b_{n_b} & & & & b_1 \\ 0 & b_{n_b} & & & b_2 \\ \vdots & 0 & b_{n_b} & & b_3 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & b_{n_n} \end{array} \right] \in \mathbb{R}^{(n_a+n_b+2n_c)\times n_c},$$

$$\begin{aligned} Y_3 \quad = \quad & \left[\begin{array}{ccccccc} p_1 - a_1 & \cdots & p_{n_a} - a_{n_a} & p_{n_a+1} & \cdots & p_{n_a+n_c} \\ q_2 - b_2 & \cdots & q_{n_b} - b_{n_b} & q_{n_b+1} & \cdots & q_{n_b+n_c} \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

The parameters $(a_i,\ b_i,\ q_i,\ p_i)$ are replaced with the corresponding estimate. The least squares estimation of θ_3 is given by

$$\hat{\theta}_3 = (H_3^{\mathrm{T}}H_3)^{-1}H_3^{\mathrm{T}}Y_3.$$

Thus, we obtain the parameter estiamte of the noise model

$$\hat{\theta}_n = \left[\begin{array}{cccc} \hat{\theta}_3 & \hat{d}_1 & \hat{d}_2 & \cdots & \hat{d}_{n_d} \end{array} \right].$$

5.4 Instrumental variable based least squares identification

It is assumed that the considered system is described by the following model

$$A(z)y(t) = B(z)u(t) + w(t), \tag{5.4.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are the input series and output series, respectively; $\{w(t)\}$ is a stationary noise with mean zero. $A(z)$ and $B(z)$ are two polynomials on operator z^{-1}:

$$\begin{aligned} A(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}, \\ B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}. \end{aligned}$$

It is assumed that the order n_a and n_b are known, $y(t) = 0$, $u(t) = 0$, $w(t) = 0$ for $t \leq 0$. In addition, we denote $n = n_a + n_b$.

Define the parameter vector and information vector as

$$\begin{aligned} \theta &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array} \right]^{\mathrm{T}} \in \mathbb{R}^n, \\ \varphi(t) &= \left[\begin{array}{cccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\mathrm{T}}. \end{aligned}$$

In addition, we define some related matrices as

$$Y_t = \left[\begin{array}{c} y(1) \\ y(2) \\ \vdots \\ y(t) \end{array} \right] \in \mathbb{R}^t, H_t = \left[\begin{array}{c} \varphi^T(1) \\ \varphi^T(2) \\ \vdots \\ \varphi^T(t) \end{array} \right] \in \mathbb{R}^{t \times (n_a + n_b)}, W_t = \left[\begin{array}{c} w(1) \\ w(2) \\ \vdots \\ w(t) \end{array} \right] \in \mathbb{R}^t.$$

By using the above defined notations, the relation (5.3.9) can be written as the following vector form and matrix form, respectively, as

$$y(t) = \varphi^T(t)\theta + w(t)$$

and

$$Y_t = H_t\theta + W_t. \tag{5.4.2}$$

It is assumed that there is a matrix F_t with the same dimension as H_t satisfying

$$\left\{ \begin{array}{l} \mathrm{E}[\frac{1}{t} F_t^T W_t] = 0, \\ \mathrm{E}[\frac{1}{t} F_t^T H_t] = Q, \end{array} \right. \tag{5.4.3}$$

where Q is a nonsingular matrix. Pre-multiplying both sides of (5.4.2), gives

$$F_t^{\mathrm{T}} Y_t = F_t^{\mathrm{T}} H_t \theta + F_t^{\mathrm{T}} W_t.$$

Thus, the instrumental variable based least squares estimate of θ is given by

$$\hat{\theta}_{\rm IV} = (F_t^{\rm T} H_t)^{-1} F_t^{\rm T} Y_t. \tag{5.4.4}$$

This estimate is unbiased. In fact,

$$\begin{aligned} \hat{\theta}_{\rm IV} &= (F_t^{\rm T} H_t)^{-1} F_t^{\rm T} (H_t\theta + W_t) \\ &= \theta + (F_t^{\rm T} H_t)^{-1} (F_t^{\rm T} W_t), \end{aligned}$$

$$\mathrm{E}[\hat{\theta}_{IV}] = \theta + \mathrm{E}[(F_t^{\rm T} H_t)^{-1} (F_t^{\rm T} W_t)] = \theta.$$

The aforementioned matrix F_t is referred to as instrumental matrix. The key of instrumental variable based least squares identification is to construst a matrix F_t to meet with (5.4.3). It follows from (5.4.3) that F_t and W_t are uncorrelated, and F_t and H_t are strongly correlated. Denote

$$F_t = \begin{bmatrix} \varphi_a^{\rm T}(1) \\ \varphi_a^{\rm T}(2) \\ \vdots \\ \varphi_a^{\rm T}(t) \end{bmatrix} \in \mathbb{R}^{t\times n}.$$

In the expression of F_t, $\varphi_a(t)$ is refferred to as instrumental vector, and the elements of F_t are called as instrumental variables. In general, there are the following two choices for $\varphi_a(t)$:

$$\varphi_a(t) = \begin{bmatrix} u(t-1) & u(t-2) & \cdots & u(t-n_a) & u(t-n_a-1) & \cdots & u(t-n_a-n_b) \end{bmatrix}^{\rm T} \in \mathbb{R}^n,$$

$$\begin{aligned} \varphi_a(t) &= \begin{bmatrix} -u(t-n_b-1) & -u(t-n_b-2) & \cdots & -u(t-n_b-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\rm T}. \end{aligned}$$

Define covariance matrix $P(t)$ as

$$\begin{aligned} P^{-1}(t) &= F_t^{\rm T} H_t = \sum_{i=1}^{t} \varphi_a(i)\varphi^T(i) \\ &= P^{-1}(t-1) + \varphi_a(t)\varphi^T(t), P(0) = p_0 I \quad (p_0 \gg 0). \end{aligned}$$

Then, (5.4.4) can be written as

$$\hat{\theta}_{\rm IV} = P(t) F_t^T Y_t.$$

In addition, denote $L(t) = P(t)\varphi_a(t)$. Similarly to the case of the standard least squares identification, one can obtain the following Instrumental Variable based Recursive Least Squares algorithm:

$$
\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)[y(t) - \varphi^T(t)\hat{\theta}(t-1)], \\
L(t) &= P(t)\varphi_a(t) = \frac{P(t-1)\varphi_a(t)}{1+\varphi^T(t)P(t-1)\varphi_a(t)}, \\
P(t) &= [I - L(t)\varphi^T(t)]P(t-1), P(0) = p_0 I, \\
\varphi(t) &= \left[\begin{array}{cccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\mathrm{T}}, \\
\varphi_a(t) &= \left[\begin{array}{cccccccc} -x(t-1) & -x(t-2) & \cdots & -x(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\mathrm{T}}.
\end{aligned}
$$

5.5 Bias compensation based recursive least squares identification

5.5.1 Bias compensation based recursive least squares identification

Consider the following output error (OE) model

$$y(t) = \frac{B(z)}{A(z)}u(t) + v(t) \tag{5.5.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$ and $B(z)$ are two polynomials with respect to the backward operator z^{-1} :

$$
\begin{aligned}
A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}.
\end{aligned}
$$

It is assumed that the orders n_a and n_b are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$. Denote

$$w(t) = A(z)v(t). \tag{5.5.2}$$

Then, the model (6.1.1) can be rewritten as

$$A(z)y(t) = B(z)u(t) + w(t). \tag{5.5.3}$$

Since $w(t)$ is colored noise, the parameter estimation given by the standard recursive least squares algorithm is biased. In this section, we use the bias compensation principle to investigate the identification problme for the output error modeol (6.1.1).

Define parameter vector and information vector as follows

$$\theta = \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array} \right]^{\mathrm{T}},$$

$$
\begin{aligned}
\varphi(t) &= \left[\begin{array}{cccc} -y\,(t-1) & -y\,(t-2) & \cdots & -y\,(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\mathrm{T}}, \\
\psi(t) &= \left[\begin{array}{cccccccc} v\,(t-1) & v\,(t-2) & \cdots & v\,(t-n_a) & 0 & 0 & \cdots & 0 \end{array} \right]^{\mathrm{T}} \in \mathbb{R}^{n_a+n_b}.
\end{aligned}
$$

In addition, we define the following matrices

$$Y_t = \begin{bmatrix} y(1) \\ y(2) \\ \cdots \\ y(t) \end{bmatrix} \in \mathbb{R}^t,\ H_t = \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \cdots \\ \varphi^{\mathrm{T}}(t) \end{bmatrix} \in \mathbb{R}^{t\times(n_a+n_b)},\ W_t = \begin{bmatrix} w(1) \\ w(2) \\ \cdots \\ w(t) \end{bmatrix} \in \mathbb{R}^t.$$

It follows from (6.1.1) to (5.5.3) that

$$\begin{aligned} w(t) &= \psi^{\mathrm{T}}(t)\theta + v(t), && (5.5.4)\\ y(t) &= \varphi^{\mathrm{T}}(t)\theta + w(t), && (5.5.5)\\ y(t) &= \varphi^{\mathrm{T}}(t)\theta + \psi^{\mathrm{T}}(t)\theta + v(t), && (5.5.6)\\ Y_t &= H_t\theta + W_t. && (5.5.7) \end{aligned}$$

According to the least squares identification principle, the least squares estimate of θ is given by

$$\begin{aligned} \hat{\theta}_{\mathrm{LS}}(t) &= \left[H_t^{\mathrm{T}}H_t\right]^{-1}H_t^{\mathrm{T}}Y_t \\ &= \left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]^{-1}\left[\sum_{i=1}^{t}\varphi(i)y(i)\right], \end{aligned}$$

which is equivalent to

$$\left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]\hat{\theta}_{\mathrm{LS}}(t) = \sum_{i=1}^{t}\varphi(i)y(i). \tag{5.5.8}$$

It follows from (5.5.8) that

$$\sum_{i=1}^{t}\varphi(i)\left[y(i) - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t)\right] = 0. \tag{5.5.9}$$

With (5.5.8), one can obtain from (5.5.6) that

$$\begin{aligned} &\left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]\left(\hat{\theta}_{\mathrm{LS}}(t) - \theta\right) \\ =\ &\sum_{i=1}^{t}\varphi(i)y(i) - \sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\theta \\ =\ &\sum_{i=1}^{t}\varphi(i)\left[y(i) - \varphi^{\mathrm{T}}(i)\theta\right] \\ =\ &\sum_{i=1}^{t}\varphi(i)\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right]. \end{aligned}$$

This implies that

$$\lim_{t\longrightarrow\infty}\frac{1}{t}\left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]\left(\hat{\theta}_{\mathrm{LS}}(t)-\theta\right)=\lim_{t\longrightarrow\infty}\frac{1}{t}\sum_{i=1}^{t}\varphi(i)\psi^{\mathrm{T}}(i)\theta+\lim_{t\longrightarrow\infty}\frac{1}{t}\sum_{i=1}^{t}\varphi(i)v(i). \tag{5.5.10}$$

Since $v(t)$ is white noise, one has

$$\lim_{t\longrightarrow\infty}\frac{1}{t}\sum_{i=1}^{t}\varphi(i)v(i)=0,$$

and

$$\lim_{t\longrightarrow\infty}\frac{1}{t}\sum_{i=1}^{t}\varphi(i)\psi^{\mathrm{T}}(i)\theta=-\sigma^2\Lambda\theta,$$

where

$$\Lambda=\left[\begin{array}{cc} I_{n_a} & 0 \\ 0 & 0 \end{array}\right]\in\mathbb{R}^{(n_a+n_b)\times(n_a+n_b)}.$$

It is assumed that the following weak persistent excitation condition is satisfied

$$\lim_{t\longrightarrow\infty}\frac{1}{t}\left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]=R>0.$$

Then, it follows from (5.5.10) that

$$\lim_{t\longrightarrow\infty}\hat{\theta}_{\mathrm{LS}}(t)=\theta-\sigma^2R^{-1}\Lambda\theta.$$

This implies that the least squares estimation for the output error model (6.1.1) is biased. If the compensation term $\sigma^2R^{-1}\Lambda\theta$ is introduced to the least squares estimation $\hat{\theta}_{\mathrm{LS}}(t)$, then

$$\hat{\theta}_{\mathrm{c}}(t)=\hat{\theta}_{\mathrm{LS}}(t)+\sigma^2R^{-1}\Lambda\theta$$

is an unbiased estimation of the parameter θ. This is the basic idea of bias compensation based least squares identification. The recursive form is

$$\hat{\theta}_{\mathrm{c}}(t)=\hat{\theta}_{\mathrm{LS}}(t)+t\hat{\sigma}^2(t)P(t)\Lambda\hat{\theta}_{\mathrm{c}}(t-1) \tag{5.5.11}$$

where $\hat{\theta}_{\mathrm{c}}(t)$, $\hat{\theta}_{\mathrm{LS}}(t)$, $\hat{\sigma}^2(t)$, are the bias compensation based least squares estimation of θ, the least squares estimation of θ, and the variance estimation of the noise $v(t)$, respectively; the covariance matrix is defined as

$$P(t)=\left[\sum_{i=1}^{t}\varphi(i)\varphi^{\mathrm{T}}(i)\right]^{-1}.$$

Now, in order to realize the recursive algorithm, we need provide an estimate of $\hat{\sigma}^2(t)$. Let

$$\varepsilon_{\mathrm{LS}}(i) = y(i) - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t),\ i = 1, 2, \cdots, t.$$

Then, the relation (5.5.9) becomes

$$\sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\,\varphi^{\mathrm{T}}(i) = 0.$$

With this relation, one has

$$\begin{aligned}
\sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}^2(i) &= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\left[y(i) - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t)\right] \\
&= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\,y(i) - \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\,\varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t) \\
&= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\,y(i) \\
&= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\left[\varphi^{\mathrm{T}}(i)\theta + \psi^{\mathrm{T}}(i)\theta + v(i)\right] \\
&= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}(i)\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] \\
&= \sum_{i=1}^{t} \left[y(i) - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t)\right]\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] \\
&= \sum_{i=1}^{t} \left[\varphi^{\mathrm{T}}(i)\theta + \psi^{\mathrm{T}}(i)\theta + v(i) - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t)\right]\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] \\
&= \sum_{i=1}^{t} \left[\varphi^{\mathrm{T}}(i)\theta - \varphi^{\mathrm{T}}(i)\hat{\theta}_{\mathrm{LS}}(t)\right]\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] + \sum_{i=1}^{t} \left[\psi^{\mathrm{T}}(i)\theta + v(i)\right]^2 \\
&= \sum_{i=1}^{t} \varphi^{\mathrm{T}}(i)\left[\theta - \hat{\theta}_{\mathrm{LS}}(t)\right]\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] + \sum_{i=1}^{t} \left[\psi^{\mathrm{T}}(i)\theta + v(i)\right]^2.
\end{aligned}$$

With the above relation, by using some properties of white noise one has

$$\begin{aligned}
\lim_{t\longrightarrow\infty} \frac{1}{t}\sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}^2(i) &= \lim_{t\longrightarrow\infty} \frac{1}{t}\sum_{i=1}^{t} \varphi^{\mathrm{T}}(i)\left[\theta - \hat{\theta}_{\mathrm{LS}}(t)\right]\left[\psi^{\mathrm{T}}(i)\theta + v(i)\right] \\
&\quad + \lim_{t\longrightarrow\infty} \frac{1}{t}\sum_{i=1}^{t} \left[\psi^{\mathrm{T}}(i)\theta + v(i)\right]^2 \\
&= -\sigma^2\theta^{\mathrm{T}}\Lambda \lim_{t\longrightarrow\infty}\left[\theta - \hat{\theta}_{\mathrm{LS}}(t)\right] + \sigma^2 + \sigma^2\theta^{\mathrm{T}}\Lambda\theta \\
&= \sigma^2\left[1 + \theta^{\mathrm{T}}\Lambda \lim_{t\longrightarrow\infty} \hat{\theta}_{\mathrm{LS}}(t)\right],
\end{aligned}$$

which gives

$$\sigma^2 = \frac{\lim\limits_{t\longrightarrow\infty} \frac{1}{t} \sum\limits_{i=1}^{t} \varepsilon_{\mathrm{LS}}^2 (i)}{1 + \theta^{\mathrm{T}} \Lambda \lim_{t\longrightarrow\infty} \hat{\theta}_{\mathrm{LS}}(t)}.$$

This relation implies that the estimate $\hat{\sigma}^2(t)$ of σ^2 at the time instant t can be given by

$$\hat{\sigma}^2(t) = \frac{\frac{1}{t} J(t)}{1 + \hat{\theta}_{\mathrm{c}}^{\mathrm{T}}(t) \Lambda \hat{\theta}_{\mathrm{LS}}(t)},$$

where

$$\begin{aligned} J(t) \quad : \quad &= \sum_{i=1}^{t} \varepsilon_{\mathrm{LS}}^2 (i) = \sum_{i=1}^{t} \left[y(i) - \varphi^{\mathrm{T}} (i)\, \hat{\theta}_{\mathrm{LS}}(t) \right]^2 \\ &= J(t-1) + \left[\frac{y(t) - \varphi^{\mathrm{T}} (t)\, \hat{\theta}_{\mathrm{LS}}(t-1)}{1 + \varphi^{\mathrm{T}} (t)\, P(t-1) \varphi (t)} \right]^2. \end{aligned}$$

With the above deduction, we now give the following Bias Compensation based Recursive Least Squares algorithm.

$$\begin{aligned} \hat{\theta}_{\mathrm{c}} (t) &= \hat{\theta}_{\mathrm{LS}}(t) + t\hat{\sigma}^2 (t)\, P (t)\, \Lambda \hat{\theta}_{\mathrm{c}} (t-1), \\ \hat{\theta}_{\mathrm{LS}}(t) &= \hat{\theta}_{\mathrm{LS}}(t-1) + L(t) \left[y(t) - \varphi^{\mathrm{T}} (t)\, \hat{\theta}_{\mathrm{LS}}(t-1) \right], \\ L(t) &= \frac{P(t-1)\varphi (t)}{1 + \varphi^{\mathrm{T}} (t)\, P(t-1)\varphi (t)}, \\ P(t) &= \left[I - L(t)\varphi^{\mathrm{T}} (t) \right] P (t-1), \\ J(t) &= J(t-1) + \left[\frac{y(t) - \varphi^{\mathrm{T}} (t)\, \hat{\theta}_{\mathrm{LS}}(t-1)}{1 + \varphi^{\mathrm{T}} (t)\, P(t-1)\varphi (t)} \right]^2, \\ \hat{\sigma}^2(t) &= \frac{J(t)}{t \left[1 + \hat{\theta}_{\mathrm{c}}^{\mathrm{T}}(t-1) \Lambda \hat{\theta}_{\mathrm{LS}}(t) \right]} = \frac{J(t)}{t \left[1 + \sum\limits_{i=1}^{n_a} \hat{\theta}_{\mathrm{c}i}^{\mathrm{T}}(t) \hat{\theta}_{\mathrm{LS}i}(t) \right]}, \\ \varphi (t) &= \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_a) \end{bmatrix}^{\mathrm{T}}, \\ \Lambda &= \begin{bmatrix} I_{n_a} & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(n_a+n_b)\times(n_a+n_b)}. \end{aligned}$$

Chapter 6

Auxiliary model based identification

6.1 Output error model

Consider the systems described by the following output error model

$$y(t) = \frac{B(z)}{A(z)}u(t) + v(t) \tag{6.1.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$ and $B(z)$ are two polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned} A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\ B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}. \end{aligned}$$

It is assumed that the orders n_a and n_b are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$. The model (6.1.1) can be rewritten as

$$A(z)y(t) = B(z)u(t) + A(z)v(t). \tag{6.1.2}$$

This is a special ARMAX model, which only has $(n_a + n_b)$ parameters. Let

$$\begin{aligned} \vartheta &= \left[\begin{array}{cccccccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} & a_1 & a_2 & \cdots & a_{n_a} \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^{2n_a+n_b} \\ \phi(t) &= \left[\begin{array}{cccccccccccc} -y(t-1) & -y(t-2) & \cdots & -y(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \\ v(t-1) & v(t-2) & \cdots & v(t-n_b) \end{array}\right]^{\mathrm{T}}. \end{aligned}$$

Then, one can obtain the following psudolinear regression model for (6.1.2)

$$y(t) = \phi^{\mathrm{T}}(t)\vartheta + v(t).$$

If the extended least squares algorithm is used to estimate the parameters in the output error model (6.1.1), the number of parameters to be identified is $2n_a+n_b$. However, only (n_a+n_b) parameters need to be identified in the model (6.1.1). In this section, we use the auxiliary model identification principle to identify the parameters in the model (6.1.1).

First, we define inner variable

$$x(t) = \frac{B(z)}{A(z)}u(t). \tag{6.1.3}$$

It follows from (6.1.3) that

$$\begin{aligned} & x(t) + a_1x(t-1) + a_2x(t-2) + \cdots + a_{n_a}x(t-n_a) \\ = \;& b_1u(t-1) + b_2u(t-2) + \cdots + b_{n_b}u(t-n_b), \end{aligned}$$

In addition, define parameter vector θ and information vector φ as follows

$$\begin{aligned} \theta &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^{n_a+n_b}, \\ \varphi(t) &= \left[\begin{array}{cccc} -x(t-1) & -x(t-2) & \cdots & -x(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\mathrm{T}}. \end{aligned}$$

Then, (6.1.3) can be equivalently rewritten as

$$x(t) = \varphi^{\mathrm{T}}(t)\theta.$$

Thus, we can obtain the identification model of the ouput error model (6.1.1)

$$\begin{aligned} y(t) &= x(t) + v(t) \\ &= \varphi^{\mathrm{T}}(t)\theta + v(t). \end{aligned}$$

By minimizing the error criterion function

$$J_0(\theta) = \sum_{i=1}^{t} \left[y(i) - \varphi^{\mathrm{T}}(i)\theta\right]^2,$$

the following recursive least squares algorithm can be obtained

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ P(t) &= P(t-1) - \frac{P(t-1)\varphi(t)\varphi^{\mathrm{T}}(t)P(t-1)}{1+\varphi^{\mathrm{T}}(t)P(t-1)\varphi(t)}. \end{aligned}$$

This algorithm can not be used to identify the output error model (6.1.1) since the information vector $\varphi(t)$ contains the unknown variable $x(t-i)$, $i = 1, 2, \cdots, n_a$. If $x(t)$ is replaced with the output of the following auxiliary model

$$x_{\mathrm{a}}(t) = P_{\mathrm{a}}(z)u(t) = \frac{A_{\mathrm{a}}(z)}{B_{\mathrm{a}}(z)}u(t), \tag{6.1.4}$$

or

$$x_{\mathrm{a}}(t) = \varphi_{\mathrm{a}}^{\mathrm{T}}(t)\theta_{\mathrm{a}}(t), \tag{6.1.5}$$

then the parameter identification for the output error model (6.1.1) can be solved. In the regression model (6.1.5), $\varphi_{\mathrm{a}}(t)$ and $\theta_{\mathrm{a}}(t)$ are respectively the information vector and parameter vector of the auxiliary model (6.1.4) at the time instant t. This is the basic idea of auxiliary model identification. The methods based on this idea are called auxiliary model based identification methods, or reference model identification method. If $x_{\mathrm{a}}(t)$ approximates $x(t)$, the output $x_{\mathrm{a}}(t)$ of the auxiliary model can be viewed as an estimate $\hat{x}(t)$ of $x(t)$.

Let $\hat{\theta}(t)$ be the estimate of the parameter θ. We can choose $\hat{\theta}(t)$ as the parameter vector $\theta_{\mathrm{a}}(t)$ of the auxiliary model. In this case, an auxiliary model can be chosen as

$$\begin{aligned} x_{\mathrm{a}}(t) &= \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t), \\ \hat{\varphi}(t) &= \begin{bmatrix} -x_{\mathrm{a}}(t-1) & -x_{\mathrm{a}}(t-2) & \cdots & -x_{\mathrm{a}}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}. \end{aligned}$$

By minimizing the following error criterion function

$$J(\theta) = \sum_{i=1}^{t} \left[y(i) - \hat{\varphi}^{\mathrm{T}}(t)\theta\right]^2,$$

we can obtain the following Auxiliary Model based Recursive Least Squares algorithm.

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\ L(t) &= P(t)\hat{\varphi}(t) = \frac{P(t-1)\hat{\varphi}(t)}{1+\hat{\varphi}^{\mathrm{T}}(t)P(t-1)\hat{\varphi}(t)} \\ P(t) &= P(t-1) - \frac{P(t-1)\hat{\varphi}(t)\hat{\varphi}^{\mathrm{T}}(t)P(t-1)}{1+\hat{\varphi}^{\mathrm{T}}(t)P(t-1)\hat{\varphi}(t)} \\ &= \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1),\ P(0) = p_0 I, \\ \hat{\varphi}(t) &= \begin{bmatrix} -x_{\mathrm{a}}(t-1) & -x_{\mathrm{a}}(t-2) & \cdots & -x_{\mathrm{a}}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\ x_{\mathrm{a}}(t) &= \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t). \end{aligned}$$

6.2 Auxiliary model based extended least squares identification

In this section, we consider the following output error MA model

$$y(t) = \frac{B(z)}{A(z)}u(t) + D(z)v(t) \tag{6.2.1}$$

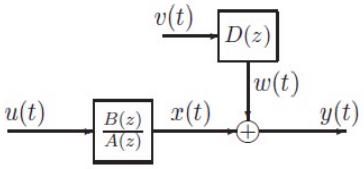

Figure 6.2.1: The output error moving average systems

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$ and $D(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned} A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\ B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\ D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}. \end{aligned}$$

It is assumed that the orders n_a, n_b and n_d are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

By multiplying both sides of (6.2.1) by $A(z)$, the output error moving average (OEMA) model (6.2.1) becomes the following ARMAX model

$$A(z)y(t) = B(z)u(t) + A(z)D(z)v(t).$$

One way to identify this mode is to use the recursive extended least square algorithm. However, in this case there are $(2n_a + n_b + n_d)$ parameters to be identified although only $(n_a + n_b + n_d)$ parameters need to be identified in the original model. This will lead to more computational load. In this section, we will introduce auxiliary model based approach to identify the model (6.2.1).

For OEMA model (6.2.1), we define the output of the system model and the output of noise model, respectively, as

$$x(t) = \frac{B(z)}{A(z)}u(t), \tag{6.2.2}$$

$$w(t) = D(z)v(t). \tag{6.2.3}$$

Then (6.2.1) can be rewritten as

$$y(t) = x(t) + w(t).$$

It can be seen from Fig. 6.2.1 that, $x(t)$ is true output of the system, $w(t)$ is the colored output, and $y(t)$ is the measurement of the $x(t)$ with noise. Both $x(t)$ and $w(t)$ are both the unmeasured inner variables. Denote the paramater vector $\theta_{\rm s}$ of the system model and the parameter vector $\theta_{\rm n}$ of the noise model, respectively as

$$\begin{aligned} \theta_{\rm s} &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array}\right]^{\rm T}, \\ \theta_{\rm n} &= \left[\begin{array}{cccc} d_1 & d_2 & \cdots & d_{n_d} \end{array}\right]^{\rm T}. \end{aligned}$$

With this, we define

$$\theta = \left[\begin{array}{c} \theta_{\rm s} \\ \theta_{\rm n} \end{array} \right].$$

Now we define the information vector $\varphi_{\rm s}(t)$ of the system model and the information vector $\varphi_{\rm n}(t)$ of the noise model, respectively, as

$$\begin{aligned} \varphi_{\rm s}(t) &= \left[\begin{array}{cccccccc} -x(t-1) & -x(t-2) & \cdots & -x(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array} \right]^{\rm T} \in \mathbb{R}^{n_a+n_b}, \\ \varphi_{\rm n}(t) &= \left[\begin{array}{cccc} v(t-1) & v(t-2) & \cdots & v(t-n_b) \end{array} \right]^{\rm T} \in \mathbb{R}^{n_d}. \end{aligned}$$

In addition, we denote

$$\varphi(t) = \left[\begin{array}{c} \varphi_{\rm s}(t) \\ \varphi_{\rm n}(t) \end{array} \right].$$

With the above definitions, it follows from (6.2.2) and (6.2.3) that

$$\begin{aligned} x(t) &= [1 - A(z)]\, x(t) + B(z)u(t) \\ &= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s}, \\ w(t) &= \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t). \end{aligned}$$

By substituting the above two relations to (6.2.1), one has

$$\begin{aligned} y(t) &= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s} + \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t) \\ &= \left[\begin{array}{cc} \varphi_{\rm s}^{\rm T}(t) & \varphi_{\rm n}^{\rm T}(t) \end{array} \right] \left[\begin{array}{c} \theta_{\rm s} \\ \theta_{\rm n} \end{array} \right] + v(t) \\ &= \varphi^{\rm T}(t)\theta + v(t). \end{aligned} \tag{6.2.4}$$

This is a psudolinear regresson model. Let

$$\hat{\theta}(t) = \left[\begin{array}{c} \hat{\theta}_{\rm s}(t) \\ \hat{\theta}_{\rm n}(t) \end{array} \right]$$

be the estimate of θ at the time t. A natural idea is to identify the parameter θ by using the previous recursive least squares algorithm:

$$\begin{aligned} \hat{\theta}\,(t) &= \hat{\theta}\,(t-1) + L(t)\left[y(t) - \varphi^{\rm T}(t)\hat{\theta}\,(t-1)\right], \\ L(t) &= \frac{P(t-1)\varphi(t)}{1 + \varphi^{\rm T}(t)P\,(t-1)\,\varphi(t)}, \\ P\,(t) &= \left[I - L(t)\varphi^{\rm T}(t)\right] P\,(t-1)\,,\; P(0) = p_0 I. \end{aligned}$$

However, this algorithm can not be realized since the information $\varphi(t)$ in this algorithm contains the unknown intermediate variable $x\,(t-i)$ and the unmeasurable noise term $v(t-i)$. In order to overcome this difficulty, we construst an auxiliary model

$$x_{\rm a}\,(t) = \frac{B_{\rm a}\,(z)}{A_{\rm a}\,(z)}u(t) \tag{6.2.5}$$

to approximate the true output $x(t)$. In this case, the unknown terms $x\,(t-i)$ in $\varphi_{\rm s}(t)$ of $\varphi(t)$ is replaced with the output $x_{\rm a}\,(t-i)$ of the anxiliary model, and we denote

$$\begin{aligned}\hat{\varphi}_{\rm s}(t) \;=\; & \left[\; -x_{\rm a}(t-1) \quad -x_{\rm a}(t-2) \quad \cdots \quad -x_{\rm a}(t-n_a)\right.\\ & \left. \quad u(t-1) \quad u(t-2) \quad \cdots \quad u(t-n_b)\;\right]^{\rm T}.\end{aligned}$$

In addition, the unknown noise terms $v(t-i)$ in $\varphi_{\rm n}(t)$ of $\varphi(t)$ is replaced with the estimation residual $\hat{v}(t-i)$, and we denote

$$\hat{\varphi}_{\rm n}(t) = \left[\;\hat{v}(t-1) \quad \hat{v}(t-2) \quad \cdots \quad \hat{v}(t-n_d)\;\right]^{\rm T} \in \mathbb{R}^{n_d}.$$

Similarly, the transfer function model (6.2.5) can be rewritten as the following linear regression model

$$x_{\rm a}(t) = \varphi_{\rm a}^{\rm T}(t)\theta_{\rm a},$$

where $\varphi_{\rm a}(t)$ and $\theta_{\rm a}$ are the information vector and the parameter vector of the auxiliary model, respectively. There are many ways to choose the information vector and the parameter vector of the auxiliary model. One way is to choose $\hat{\varphi}_{\rm s}(t)$ as the information vector $\varphi_{\rm a}^{\rm T}(t)$ of the auxiliary model, and choose $\hat{\theta}_s(t)$ as the parameter vector $\theta_{\rm a}$ of the auxiliary model. In this case, one has

$$x_{\rm a}(t) = \hat{\varphi}_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t).$$

Replace $\varphi_{\rm s}(t)$ and $\varphi_{\rm n}(t)$ in $\varphi(t)$ with $\hat{\varphi}_s(t)$ and $\hat{\varphi}_n(t)$, respectively, and then we denote

$$\hat{\varphi}(t) = \left[\begin{array}{c}\hat{\varphi}_s(t)\\ \hat{\varphi}_n(t)\end{array}\right] \in \mathbb{R}^{n_a+n_b+n_d}.$$

It follows from (6.2.4) that

$$v(t) = y(t) - \varphi^{\rm T}(t)\theta.$$

By replacing the unknown $\varphi(t)$ with $\hat{\varphi}(t)$, and replacing θ with the estimate $\hat{\theta}(t)$, the residual $\hat{v}(t)$ can be obtained by the following relation

$$\hat{v}(t) = y(t) - \hat{\varphi}^{\rm T}(t)\hat{\theta}(t).$$

With the above preliminaries, by minimizing the following

$$J(\theta) = \sum_{i=1}^{t}\left[y(i) - \hat{\varphi}^{\rm T}(i)\theta\right]^2,$$

and derivation similarly to the standard least squares identification, one can obtain the following recursive algorithm to estimate θ

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
L(t) &= \frac{P(t-1)\hat{\varphi}(t)}{1+\hat{\varphi}^{T}(t)P(t-1)\hat{\varphi}(t)}, \\
P(t) &= \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1), P(0) = p_0 I.
\end{aligned}$$

By summarizing the previous results, one can obtain the following Auxiliary Model based Recursive Extended Least Squares (AM-RELS) algorithm.

$$\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t-1)\right], \\
L(t) &= P(t-1)\hat{\varphi}(t)\left[1+\hat{\varphi}^{\mathrm{T}}(t)P(t-1)\hat{\varphi}(t)\right]^{-1}, \\
P(t) &= \left[I - L(t)\hat{\varphi}^{\mathrm{T}}(t)\right]P(t-1), P(0) = p_0 I, \\
\hat{\varphi}(t) &= \begin{bmatrix} \hat{\varphi}_{\mathrm{s}}(t) \\ \hat{\varphi}_{\mathrm{n}}(t) \end{bmatrix}, \\
\hat{\varphi}_{\mathrm{s}}(t) &= \begin{bmatrix} -x_{\mathrm{a}}(t-1) & -x_{\mathrm{a}}(t-2) & \cdots & -x_{\mathrm{a}}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\
\hat{\varphi}_{\mathrm{n}}(t) &= \begin{bmatrix} \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_d) \end{bmatrix}^{\mathrm{T}}, \\
x_{\mathrm{a}}(t) &= \hat{\varphi}_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t), \\
\hat{v}(t) &= y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t), \\
\hat{\theta}(t) &= \begin{bmatrix} \hat{\theta}_s(t) \\ \hat{\theta}_n(t) \end{bmatrix}, \\
\hat{\theta}_{\mathrm{s}}(t) &= \begin{bmatrix} \hat{a}_1(t) & \hat{a}_2(t) & \cdots & \hat{a}_{n_a}(t) & \hat{b}_1(t) & \hat{b}_2(t) & \cdots & \hat{b}_{n_b}(t) \end{bmatrix}^{\mathrm{T}}, \\
\hat{\theta}_{\mathrm{n}}(t) &= \begin{bmatrix} \hat{d}_1(t) & \hat{d}_2(t) & \cdots & \hat{d}_{n_d}(t) \end{bmatrix}^{\mathrm{T}}.
\end{aligned}$$

6.3 Output error autoregressive model

Consider the following system with colored noise described by the following output error autoregressive model

$$y(t) = \frac{B(z)}{A(z)}u(t) + \frac{1}{C(z)}v(t) \tag{6.3.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$, $C(z)$ and $D(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$\begin{aligned}
A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\
C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}.
\end{aligned}$$

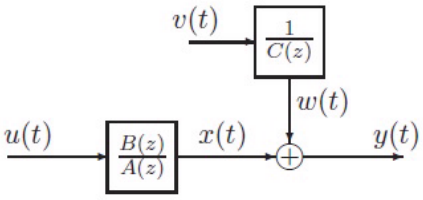

Figure 6.3.1: The output error autogressive systems

It is assumed that the orders n_a, n_b and n_c are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$.

The system (6.3.1) can be described by Fig. 6.3.1. Define the output of the system model and the output of the noise model as, respectively

$$x(t) = \frac{B(z)}{A(z)}u(t), \tag{6.3.2}$$

$$w(t) = \frac{1}{C(z)}v(t). \tag{6.3.3}$$

With this, the system (6.3.1) can be rewritten as

$$y(t) = x(t) + w(t). \tag{6.3.4}$$

It can be seen from Fig. 6.3.1 that $x(t)$ is the output of the system model, which is also referred to as true output; $w(t)$ is the output of the noise model, which is colored noise or correlated noise. $x(t)$ and $w(t)$ are both the inner variabel which can not be measurable. $y(t)$ is the measurement of $x(t)$.

Denote the system parameter vector θ, the parameter vector $\theta_{\rm s}$ of the system model, and the parameter vector $\theta_{\rm n}$ of the noise model, respectively, as

$$\begin{aligned}
\theta &= \begin{bmatrix} \theta_{\rm s} \\ \theta_{\rm n} \end{bmatrix} \in \mathbb{R}^{n_a+n_b+n_c}, \\
\theta_{\rm s} &= \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{bmatrix}^{\rm T} \in \mathbb{R}^{n_a+n_b}, \\
\theta_{\rm n} &= \begin{bmatrix} c_1 & c_2 & \cdots & c_{n_c} \end{bmatrix}^{\rm T}.
\end{aligned}$$

In addition, we define the information vector $\varphi(t)$, the information vector $\varphi_s(t)$ of the system model, the information vector $\varphi_{\rm n}(t)$ of the noise model, respectively, as

$$
\begin{aligned}
\varphi(t) &= \left[\begin{array}{c} \varphi_{\rm s}(t) \\ \varphi_{\rm n}(t) \end{array}\right] \in \mathbb{R}^{n_a+n_b+n_c}, \\
\varphi_{\rm s}(t) &= \left[\begin{array}{cccccccc} -x(t-1) & -x(t-2) & \cdots & -x(t-n_a) & u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\rm T} \in \mathbb{R}^{n_a+n_b}, \\
\varphi_{\rm n}(t) &= \left[\begin{array}{cccc} -w(t-1) & -w(t-2) & \cdots & -w(t-n_b) \end{array}\right]^{\rm T} \in \mathbb{R}^{n_c}.
\end{aligned}
$$

With the above notations, it follows from (6.3.2) and (6.3.3) that

$$
\begin{aligned}
x(t) &= [1-A(z)]\,x(t) + B(z)u(t) \qquad (6.3.5) \\
&= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s}, \\
w(t) &= [1-C(z)]\,w(t) + v(t) \qquad (6.3.6) \\
&= \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t).
\end{aligned}
$$

Substituting the above two relations into (6.3.4), gives the following identification model

$$
\begin{aligned}
y(t) &= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s} + \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t) \\
&= \left[\begin{array}{cc} \varphi_{\rm s}^{\rm T}(t) & \varphi_{\rm n}^{\rm T}(t) \end{array}\right] \left[\begin{array}{c} \theta_s \\ \theta_n \end{array}\right] + v(t) \qquad (6.3.7) \\
&= \varphi^{\rm T}(t)\theta + v(t).
\end{aligned}
$$

This is a psudolinear regression model.

Let

$$
\hat{\theta}(t) = \left[\begin{array}{c} \hat{\theta}_{\rm s}(t) \\ \hat{\theta}_{\rm n}(t) \end{array}\right]
$$

be the estimate of

$$
\theta = \left[\begin{array}{c} \theta_{\rm s} \\ \theta_{\rm n} \end{array}\right].
$$

A natural idea is to estimate the parameter vector θ by using the following standard recursive least square algorithm

$$
\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)[y(t) - \varphi^{\rm T}(t)\hat{\theta}(t-1)], \qquad (6.3.8) \\
L(t) &= \frac{P(t-1)\varphi(t)}{1+\varphi^{\rm T}(t)P(t-1)\varphi(t)}, \qquad (6.3.9) \\
P(t) &= [I - L(t)\varphi^{\rm T}(t)]P(t-1), P(0) = p_0 I. \qquad (6.3.10)
\end{aligned}
$$

However, this algorithm does not work since the information vector $\varphi(t)$ contains the unknown terms $x(t-i)$ and $w(t-i)$. In order to overcome such difficulties, we construct an auxiliary model

$$
x_{\rm a}(t) = \frac{B_{\rm a}(z)}{A_{\rm a}(z)}u(t)
$$

to approximate the true output $x(t)$. The unknown terms $x(t-i)$ in $\varphi_{\rm s}(t)$ is replaced with the output $x_{\rm a}(t-i)$ of the auxiliary model, and the modified $\varphi_{\rm s}(t)$ is denoted as

$$\begin{aligned}\hat{\varphi}_{\rm s}(t) &= \left[\begin{array}{cccc} -x_{\rm a}(t-1) & -x_{\rm a}(t-2) & \cdots & -x_{\rm a}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\rm T}.\end{aligned}$$

In addition, the noise term $w(t-i)$ in $\varphi_{\rm n}(t)$ of $\varphi(t)$ is replaced with the estimate $\hat{w}(t-i)$. The modeifed $\varphi_{\rm n}(t)$ is denoted by

$$\hat{\varphi}_{\rm n}(t) = \left[\begin{array}{cccc} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \end{array}\right]^{\rm T} \in \mathbb{R}^{n_c}.$$

Now we choose the auxiliary model as

$$x_{\rm a}(t) = \hat{\varphi}_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t),$$

and we denote

$$\hat{\varphi}(t) = \left[\begin{array}{c} \hat{\varphi}_s(t) \\ \hat{\varphi}_n(t) \end{array}\right] \in \mathbb{R}^{n_a+n_b+n_c}.$$

It follows from (6.3.7) and (6.3.6) that

$$w(t) = y(t) - \varphi_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t).$$

By replacing the $\varphi_s(t)$ with $\hat{\varphi}_{\rm s}(t)$, and replacing $\theta_{\rm s}$ with $\hat{\theta}_s(t)$, it follows from the preceding relation that the estimate $\hat{w}(t)$ of $w(t)$ can be given by

$$\hat{w}(t) = y(t) - \hat{\varphi}_s^{\rm T}(t)\hat{\theta}_s(t).$$

By replacing $\varphi(t)$ in algorithm (6.3.8) – (6.3.10) with $\hat{\varphi}(t)$, one can obtain the following recursive algorithm to compute $\hat{\theta}(t)$

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)[y(t) - \hat{\varphi}^{\rm T}(t)\hat{\theta}(t-1)], \\ L(t) &= \frac{P(t-1)\hat{\varphi}(t)}{1+\hat{\varphi}^{\rm T}(t)P(t-1)\hat{\varphi}(t)}, \\ P(t) &= [I - L(t)\hat{\varphi}^{\rm T}(t)]P(t-1), P(0) = p_0 I.\end{aligned}$$

By summarizing the previous content, we can obtain the following Auxiliary Model based Recursive Least Squares algorithm.

$$
\begin{aligned}
\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)[y(t) - \hat{\varphi}^{\mathrm{T}}(t)\hat{\theta}(t-1)], \\
L(t) &= P(t-1)\hat{\varphi}(t)[1 + \hat{\varphi}^{\mathrm{T}}(t)P(t-1)\hat{\varphi}(t)]^{-1}, \\
P(t) &= [I - L(t)\hat{\varphi}^{\mathrm{T}}(t)]P(t-1), P(0) = p_0 I, \\
\hat{\varphi}(t) &= \begin{bmatrix} \hat{\varphi}_{\mathrm{s}}(t) \\ \hat{\varphi}_{\mathrm{n}}(t) \end{bmatrix}, \\
\hat{\varphi}_{\mathrm{s}}(t) &= \begin{bmatrix} -x_{\mathrm{a}}(t-1) & -x_{\mathrm{a}}(t-2) & \cdots & -x_{\mathrm{a}}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\mathrm{T}}, \\
\hat{\varphi}_{\mathrm{n}}(t) &= \begin{bmatrix} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \end{bmatrix}^{\mathrm{T}}, \\
x_{\mathrm{a}}(t) &= \hat{\varphi}_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t), \\
\hat{w}(t) &= y(t) - x_{\mathrm{a}}(t) = y(t) - \hat{\varphi}_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t), \\
\hat{\theta}(t) &= \begin{bmatrix} \hat{\theta}_{\mathrm{s}}(t) \\ \hat{\theta}_{\mathrm{n}}(t) \end{bmatrix}.
\end{aligned}
$$

6.4 Box-Jenkins model

In this section, we consider the most general output error model

$$y(t) = \frac{B(z)}{A(z)}u(t) + \frac{D(z)}{C(z)}v(t) \tag{6.4.1}$$

where $\{u(t)\}$ and $\{y(t)\}$ are respectively input and output series, $\{v(t)\}$ is ??/ Adding some details. $A(z)$, $B(z)$, $C(z)$ and $D(z)$ are some polynomials with respect to the backward operator z^{-1} :

$$
\begin{aligned}
A\,(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}; \\
B(z) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}; \\
C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}. \\
D(z) &= 1 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_{n_d} z^{-n_d}.
\end{aligned}
$$

It is assumed that the orders n_a, n_b, n_c and n_d are unknown, and $y(t) = 0$, $u(t) = 0$, and $v(t) = 0$ for $t \leq 0$. Obviously, the model (6.4.1) is the output error autogressive moving average model. It is also called as Box-Jenkins model.

Define the output of the system model and the output of the noise model as, respectively

$$
\begin{aligned}
x(t) &= \frac{B(z)}{A(z)}u(t), \\
w(t) &= \frac{D(z)}{C(z)}v(t).
\end{aligned}
$$

With this, the system (6.4.1) can be rewritten as

$$y(t) = x(t) + w(t). \tag{6.4.2}$$

$x(t)$ is the output of the system model, which is also referred to as true output; $w(t)$ is the output of the noise model, which is colored noise or correlated noise. $x(t)$ and $w(t)$ are both the inner variabel which can not be measurable. $y(t)$ is the measurement of $x(t)$.

Denote the system parameter vector θ, the parameter vector $\theta_{\rm s}$ of the system model, and the parameter vector $\theta_{\rm n}$ of the noise model, respectively, as

$$\begin{aligned}
\theta &= \left[\begin{array}{c} \theta_{\rm s} \\ \theta_{\rm n} \end{array}\right] \in \mathbb{R}^{n_a+n_b+n_c+n_d}, \\
\theta_{\rm s} &= \left[\begin{array}{cccccccc} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{array}\right]^{\rm T} \in \mathbb{R}^{n_a+n_b}, \\
\theta_{\rm n} &= \left[\begin{array}{cccccccc} c_1 & c_2 & \cdots & c_{n_c} & d_1 & d_2 & \cdots & d_{n_d} \end{array}\right]^{\rm T} \in \mathbb{R}^{n_c+n_d}.
\end{aligned}$$

In addition, we define the information vector $\varphi(t)$, the information vector $\varphi_s(t)$ of the system model, the information vector $\varphi_{\rm n}(t)$ of the noise model, respectively, as

$$\begin{aligned}
\varphi(t) &= \left[\begin{array}{c} \varphi_{\rm s}(t) \\ \varphi_{\rm n}(t) \end{array}\right] \in \mathbb{R}^{n_a+n_b+n_c+n_d}, \\
\varphi_{\rm s}(t) &= \left[\begin{array}{cccc} -x(t-1) & -x(t-2) & \cdots & -x(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{array}\right]^{\rm T}, \\
\varphi_{\rm n}(t) &= \left[\begin{array}{cccc} -w(t-1) & -w(t-2) & \cdots & -w(t-n_b) \\ v(t-1) & v(t-2) & \cdots & v(t-n_d) \end{array}\right]^{\rm T}.
\end{aligned}$$

With the above notations, it follows from (6.4.1) and (6.4.2) that

$$\begin{aligned}
x(t) &= [1 - A(z)]\, x(t) + B(z)u(t) \qquad (6.4.3) \\
&= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s}, \\
w(t) &= [1 - C(z)]\, w(t) + D(z)v(t) \qquad (6.4.4) \\
&= \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t).
\end{aligned}$$

Substituting the above two relations into (6.4.2), gives the following identification model

$$\begin{aligned}
y(t) &= \varphi_{\rm s}^{\rm T}(t)\theta_{\rm s} + \varphi_{\rm n}^{\rm T}(t)\theta_{\rm n} + v(t) \\
&= \left[\begin{array}{cc} \varphi_{\rm s}^{\rm T}(t) & \varphi_{\rm n}^{\rm T}(t) \end{array}\right] \left[\begin{array}{c} \theta_s \\ \theta_n \end{array}\right] + v(t) \qquad (6.4.5) \\
&= \varphi^{\rm T}(t)\theta + v(t).
\end{aligned}$$

This is a psudolinear regression model.

Let

$$\hat{\theta}(t) = \begin{bmatrix} \hat{\theta}_{\mathrm{s}}(t) \\ \hat{\theta}_{\mathrm{n}}(t) \end{bmatrix}$$

be the estimate of

$$\theta = \begin{bmatrix} \theta_{\mathrm{s}} \\ \theta_{\mathrm{n}} \end{bmatrix}.$$

The standard recursive least squares algorithm can not be used to estimate the parameter vector θ since the information vector $\varphi(t)$ contains the unknown terms $x(t-i)$ and $w(t-i)$. In order to overcome such difficulties, we construct an auxiliary model

$$x_{\mathrm{a}}(t) = \frac{B_{\mathrm{a}}(z)}{A_{\mathrm{a}}(z)} u(t)$$

to approximate the true output $x(t)$. The unknown terms $x(t-i)$ in $\varphi_{\mathrm{s}}(t)$ is replaced with the output $x_{\mathrm{a}}(t-i)$ of the auxiliary model, and the modified $\varphi_{\mathrm{s}}(t)$ is denoted as

$$\begin{aligned} \hat{\varphi}_{\mathrm{s}}(t) \quad = \quad & \big[\; -x_{\mathrm{a}}(t-1) \quad -x_{\mathrm{a}}(t-2) \quad \cdots \quad -x_{\mathrm{a}}(t-n_a) \\ & \quad u(t-1) \quad u(t-2) \quad \cdots \quad u(t-n_b) \; \big]^{\mathrm{T}}. \end{aligned}$$

In addition, the noise term $w(t-i)$ in $\varphi_{\mathrm{n}}(t)$ of $\varphi(t)$ is replaced with the estimate $\hat{w}(t-i)$. The modeifed $\varphi_{\mathrm{n}}(t)$ is denoted by

$$\begin{aligned} \hat{\varphi}_{\mathrm{n}}(t) \quad = \quad & \big[\; -\hat{w}(t-1) \quad -\hat{w}(t-2) \quad \cdots \quad -\hat{w}(t-n_c) \\ & \quad \hat{v}(t-1) \quad \hat{v}(t-2) \quad \cdots \quad \hat{v}(t-n_c) \; \big]^{\mathrm{T}}. \end{aligned}$$

Now we choose the auxiliary model as

$$x_{\mathrm{a}}(t) = \hat{\varphi}_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t),$$

and we denote

$$\hat{\varphi}(t) = \begin{bmatrix} \hat{\varphi}_s(t) \\ \hat{\varphi}_n(t) \end{bmatrix} \in \mathbb{R}^{n_a+n_b+n_c}.$$

It follows from (6.4.2), (6.4.3) and (6.4.4) that

$$w(t) = y(t) - \varphi_{\mathrm{s}}^{\mathrm{T}}(t)\hat{\theta}_{\mathrm{s}}(t).$$

By replacing the $\varphi_{\mathrm{s}}(t)$ with $\hat{\varphi}_{\mathrm{s}}(t)$, and replacing θ_{s} with $\hat{\theta}_s(t)$, it follows from the preceding relation that the estimate $\hat{w}(t)$ of $w(t)$ can be given by

$$\hat{w}(t) = y(t) - \hat{\varphi}_s^{\mathrm{T}}(t)\hat{\theta}_s(t).$$

In addition, it can be obtained that

$$\begin{aligned}\hat{v}(t) &= \hat{w}(t) - \hat{\varphi}_{\rm n}^{\rm T}(t)\hat{\theta}_{\rm n}(t) \\ &= y(t) - \hat{\varphi}_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t) - \hat{\varphi}_{\rm n}^{\rm T}(t)\hat{\theta}_{\rm n}(t) \\ &= y(t) - \hat{\varphi}^{\rm T}(t)\hat{\theta}(t).\end{aligned}$$

By minimizing the following error criterion function

$$J(\theta) = \sum_{i=1}^{t}[y(i) - \hat{\varphi}^{\rm T}(i)\theta]^2,$$

one can obtain the following recursive algorithm to estimate the parameter vector θ.

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)[y(t) - \hat{\varphi}^{\rm T}(t)\hat{\theta}(t-1)], \\ L(t) &= \frac{P(t-1)\hat{\varphi}(t)}{1 + \hat{\varphi}^{\rm T}(t)P(t-1)\hat{\varphi}(t)}, \\ P(t) &= [I - L(t)\hat{\varphi}^{\rm T}(t)]P(t-1), P(0) = p_0 I, \\ \hat{\varphi}(t) &= \begin{bmatrix} \hat{\varphi}_{\rm s}(t) \\ \hat{\varphi}_{\rm n}(t) \end{bmatrix}, \\ \hat{\varphi}_{\rm s}(t) &= \begin{bmatrix} -x_{\rm a}(t-1) & -x_{\rm a}(t-2) & \cdots & -x_{\rm a}(t-n_a) \\ u(t-1) & u(t-2) & \cdots & u(t-n_b) \end{bmatrix}^{\rm T}, \\ \hat{\varphi}_{\rm n}(t) &= \begin{bmatrix} -\hat{w}(t-1) & -\hat{w}(t-2) & \cdots & -\hat{w}(t-n_c) \\ \hat{v}(t-1) & \hat{v}(t-2) & \cdots & \hat{v}(t-n_c) \end{bmatrix}^{\rm T}. \\ x_{\rm a}(t) &= \hat{\varphi}_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t), \\ \hat{w}(t) &= y(t) - x_{\rm a}(t) = y(t) - \hat{\varphi}_{\rm s}^{\rm T}(t)\hat{\theta}_{\rm s}(t), \\ \hat{v}(t) &= \hat{w}(t) - \hat{\varphi}_{\rm n}^{\rm T}(t)\hat{\theta}_n(t) = y(t) - \hat{\varphi}^{\rm T}(t)\hat{\theta}(t), \\ \hat{\theta}(t) &= \begin{bmatrix} \hat{\theta}_{\rm s}(t) \\ \hat{\theta}_{\rm n}(t) \end{bmatrix}.\end{aligned}$$

Chapter 7

Stochastic gradient based identification algorithm

We have known that an important entity of system identification is the error criterion function. In the previous sections, the involved error criterion functions are the quadratic function of the error. In this chapter, we will consider another type of error criterion function.

We consider the following linear regression model

$$y(t) = \varphi^{\mathrm{T}}(t)\,\theta + v(t) \tag{7.0.1}$$

where $y(t) \in \mathbb{R}$ is the system output, $\varphi(t)$ is the information vector in this model.

7.1 The projection identification algorithm

For the regression model (7.0.1), we consider the following quadratic cost function

$$J(\theta) = \left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta\right]^2. \tag{7.1.1}$$

The gradient $\frac{\partial J(\theta)}{\partial \theta}$ is given by

$$\frac{\partial J(\theta)}{\partial \theta} = -2\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta\right].$$

According to the stochastic approximation principle, the estimate $\hat{\theta}(t)$ of θ can be obtained by the following recursive algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) - \frac{\mu(t)}{2}\left\{-2\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right]\right\} \\ &= \hat{\theta}(t-1) + \mu(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right],\end{aligned} \tag{7.1.2}$$

where $\mu(t)$ is the step-size. Substituting $\theta = \hat{\theta}(t)$ into (7.1.1), gives

$$\begin{aligned} g\left(\mu(t)\right) &: = J(\hat{\theta}\left(t\right)) \\ &= \left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\hat{\theta}\left(t\right)\right]^2 \\ &= \left\{y(t) - \varphi^{\mathrm{T}}\left(t\right)\left[\hat{\theta}(t-1) + \mu(t)\varphi\left(t\right)\left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\theta\left(t-1\right)\right]\right]\right\}^2 \\ &= \left\{y(t) - \varphi^{\mathrm{T}}\left(t\right)\hat{\theta}(t-1) - \mu(t)\left\|\varphi\left(t\right)\right\|^2\left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\theta\left(t-1\right)\right]\right\}^2 \\ &= \left[1 - \mu(t)\left\|\varphi\left(t\right)\right\|^2\right]^2\left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\theta\left(t-1\right)\right]^2. \end{aligned}$$

The rest is to choose some $\mu(t)$ such that $g\left(\mu(t)\right)$ is minimized. It is obvious that such a $\mu(t)$ is given by

$$\mu(t) = \frac{1}{\left\|\varphi\left(t\right)\right\|^2}.$$

Substituting this relation into (7.1.2), gives the recursive algorithm

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{\varphi\left(t\right)}{\left\|\varphi\left(t\right)\right\|^2}\left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\theta\left(t-1\right)\right] \tag{7.1.3}$$

The second approach to obtain the projection algorithm.
Minimize the following error criterion function

$$J\left(\hat{\theta}\left(t\right)\right) = \frac{1}{2}\left\|\hat{\theta}\left(t\right) - \hat{\theta}(t-1)\right\|^2$$

under the constraint

$$y(t) = \varphi^{\mathrm{T}}\left(t\right)\hat{\theta}\left(t\right).$$

Introduce the Lagrange multiplier ω, and extend the criterion function as

$$J_c\left(\hat{\theta}\left(t\right), \omega\right) = \frac{1}{2}\left\|\hat{\theta}\left(t\right) - \hat{\theta}(t-1)\right\|^2 + \omega\left[y(t) - \varphi^{\mathrm{T}}\left(t\right)\hat{\theta}\left(t\right)\right].$$

By setting the first-order partial derivative to be zero, one has

$$\begin{aligned} \frac{\partial J_c}{\partial \theta} &= \hat{\theta}\left(t\right) - \hat{\theta}(t-1) - \omega\varphi\left(t\right) = 0, \\ \frac{\partial J_c}{\partial \omega} &= y(t) - \varphi^{\mathrm{T}}\left(t\right)\hat{\theta}\left(t\right) = 0. \end{aligned}$$

With these two relations, one has

$$y(t) = \varphi^{\mathrm{T}}\left(t\right)\left[\hat{\theta}(t-1) + \omega\varphi\left(t\right)\right],$$

which implies that

$$\omega = \frac{y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)}{\|\varphi(t)\|^2}.$$

So we have

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + \omega\varphi(t) \\ &= \hat{\theta}(t-1) + \frac{\varphi(t)}{\|\varphi(t)\|^2}\left[y(t) - \varphi^{\mathrm{T}}(t)\,\hat{\theta}(t-1)\right].\end{aligned}$$

This basic principle can be applied to some typical models such as equation error type models, output error type models.

7.2 Stochastic gradient algorithm

The parameter estimates $\hat{\theta}(t)$ given by the projection algorithm in (7.1.3) is sensitive to noise $v(t)$ and cannot converge to their true values. In order to overcome this drawback, one can obtain the following stochastic gradient (SG) algorithm of estimating θ by taking the step-size

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + \frac{\varphi(t)}{r(t)}\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right], \\ r(t) &= r(t-1) + \|\varphi(t)\|^2,\ r(0) = 1.\end{aligned}$$

Now we give a unified framework of previous algorithm.

For the system (7.0.1), we consider the following error criterion function

$$J(\theta) = \mathrm{E}\left\{\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta\right]^2\right\}.$$

The gradient of $J(\theta)$ at $\theta = \hat{\theta}(t-1)$ is

$$\left.\frac{\partial J(\theta)}{\partial\theta}\right|_{\theta=\hat{\theta}(t-1)} = -2\,\mathrm{E}\left\{\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta\right]\right\},$$

and the Hessian matrix is

$$H\left(\hat{\theta}(t-1)\right) = \left.\frac{\partial^2 J(\theta)}{\partial\theta\partial\theta^{\mathrm{T}}}\right|_{\theta=\hat{\theta}(t-1)} = 2\,\mathrm{E}\left[\varphi(t)\,\varphi^{\mathrm{T}}(t)\right].$$

With this, one can obtain the following Newton's algorithm for estimating the parameter θ

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) - H^{-1}\left(\hat{\theta}(t-1)\right)\left.\frac{\partial J(\theta)}{\partial\theta}\right|_{\theta=\hat{\theta}(t-1)} \\ &= \hat{\theta}(t-1) + \left\{\mathrm{E}\left[\varphi(t)\,\varphi^{\mathrm{T}}(t)\right]\right\}^{-1}\mathrm{E}\left\{\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right]\right\}.\end{aligned}$$

In practical, let

$$R(t) = \mathrm{E}\left[\varphi(t)\,\varphi^{\mathrm{T}}(t)\right],$$

and the latter expectation is removed, one can obtain the following recursive format

$$\hat{\theta}(t) = \hat{\theta}(t-1) + R^{-1}(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right].$$

- Recursive least squares algorithm

  If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P^{-1}(t) = P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t) \end{cases},$$

By using the matrix inversion lemma, we can obtain the recursive least squares algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right], \\ P(t) &= P(t-1) - \frac{P(t-1)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{1+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)},\ P(0) = p_0 I. \end{aligned}$$

- Recursive least squares algorithm with forgetting factor

  If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P^{-1}(t) = \lambda P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t),\ 0 < \lambda < 1 \end{cases},$$

we can obtain the following algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right], \\ P(t) &= \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)}\right],\ P(0) = p_0 I. \end{aligned}$$

- Kalman filtering algorithm

  If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P(t) = \left[P^{-1}(t-1) + \varphi(t)\,\varphi^{\mathrm{T}}(t)\right]^{-1} + Q \end{cases},$$

we can obtain the Kalman filtering algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\,\theta(t-1)\right], \\ P(t) &= P(t-1) - \frac{P(t-1)\,\varphi(t)\,\varphi^{\mathrm{T}}(t)\,P(t-1)}{\lambda+\varphi^{\mathrm{T}}(t)\,P(t-1)\,\varphi(t)} + Q. \end{aligned}$$

- Fixed memory least squares algorithm

Let p is the length of the data window. If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P^{-1}(t) = \sum\limits_{i=0}^{L-1} \varphi(t-i)\varphi^{\mathrm{T}}(t-i) \\ = P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t) - \varphi(t-p)\varphi^{\mathrm{T}}(t-p) \end{cases},$$

we can obtain the following reduced fixed memory least squres algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\theta(t-1)\right], \\ P^{-1}(t) &= P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t) - \varphi(t-p)\varphi^{\mathrm{T}}(t-p). \end{aligned}$$

- Fixed memory least squares algorithm with forgetting factor

Let p is the length of the data window. If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P^{-1}(t) = \sum\limits_{i=0}^{L-1} \lambda^i \varphi(t-i)\varphi^{\mathrm{T}}(t-i) \\ = \lambda P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t) - \lambda^p \varphi(t-p)\varphi^{\mathrm{T}}(t-p) \end{cases},$$

we can obtain the following reduced fixed memory least squres algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^{\mathrm{T}}(t)\theta(t-1)\right], \\ P^{-1}(t) &= \lambda P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t) - \lambda^p \varphi(t-p)\varphi^{\mathrm{T}}(t-p). \end{aligned}$$

- Projection identification

If we choose

$$\begin{cases} R(t) = P^{-1}(t) \\ P^{-1}(t) = I + \varphi(t)\varphi^{\mathrm{T}}(t) \end{cases},$$

by using matrix inversion lemma we have

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{\varphi(t)}{1 + \|\varphi(t)\|^2}\left[y(t) - \varphi^{\mathrm{T}}(t)\theta(t-1)\right].$$

- Stochastic gradient algorithm

If we choose

$$\begin{cases} R(t) = r(t)I \\ r(t) = \mathrm{tr}\left[P^{-1}(t)\right] \\ P^{-1}(t) = P^{-1}(t-1) + \varphi(t)\varphi^{\mathrm{T}}(t) \end{cases}$$

we can obtain the following stochastic gradient algorithm

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + \frac{\varphi(t)}{r(t)}\left[y(t) - \varphi^{\mathrm{T}}(t)\theta(t-1)\right] \\ r(t) &= r(t-1) + \|\varphi(t)\|^2. \end{aligned}$$

- Modefied stochastic gradient algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1)+\frac{\varphi(t)}{r^{\varepsilon}(t)}\left[y(t)-\varphi^{\mathrm{T}}(t)\,\theta(t-1)\right],\ \frac{1}{2}<\varepsilon\leq 1\\ r(t) &= r(t-1)+\|\varphi(t)\|^{2}.\end{aligned}$$

- Stochastic Gradient identification algorithm with a Forgetting Factor

If we choose

$$\begin{cases} R(t)=r(t)I\\ r(t)=\mathrm{tr}\left[P^{-1}(t)\right]\\ P^{-1}(t)=\lambda P^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)\end{cases}$$

we can obtain the following stochastic gradient algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1)+\frac{\varphi(t)}{r(t)}\left[y(t)-\varphi^{\mathrm{T}}(t)\,\theta(t-1)\right]\\ r(t) &= \lambda r(t-1)+\|\varphi(t)\|^{2}.\end{aligned}$$

- Generalized projection algorithm

If we choose

$$\begin{cases} R(t)=r(t)I\\ r(t)=\mathrm{tr}\left[P^{-1}(t)\right]\\ P^{-1}(t)=\sum\limits_{i=0}^{L-1}\varphi(t-i)\,\varphi^{\mathrm{T}}(t-i)\\ =P^{-1}(t-1)+\varphi(t)\,\varphi^{\mathrm{T}}(t)-\varphi(t-p)\,\varphi^{\mathrm{T}}(t-p)\end{cases}$$

we can obtain

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1)+\frac{\varphi(t)}{r(t)}\left[y(t)-\varphi^{\mathrm{T}}(t)\,\theta(t-1)\right]\\ r(t) &= \sum_{i=0}^{L-1}\|\varphi(t-i)\|^{2}=r(t-1)+\|\varphi(t)\|^{2}-\|\varphi(t-p)\|^{2}.\end{aligned}$$

A forgetting factor can also be introduced into the above projection algorithm

$$\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1)+\frac{\varphi(t)}{r(t)}\left[y(t)-\varphi^{\mathrm{T}}(t)\,\theta(t-1)\right]\\ r(t) &= \sum_{i=0}^{L-1}\|\varphi(t-i)\|^{2}=r(t-1)+\|\varphi(t)\|^{2}-\|\varphi(t-p)\|^{2}.\end{aligned}$$

The algorithms that is not involved with the variance matrix $P(t)$ is called Least Mean Square algorithm.

Bibliography

[1]

[2]

[3] P. Benner, E. S. Quintana-Qrti, and G. Quintana-Qrti. Solving stable stein equations on distributed memory computers. *Lecture Notes in Computer Science*, 1685:1120–1123, 1999.

[4] R. R. Bitmead. Explicit solutions of the discrete-time lyapunov matrix equation and kalman-yabovich equations. *IEEE Transactions on Automatic Control*, AC-26(6):1291–1294, 1981.

[5] R. R. Bitmead and H. Weiss. One the solution of the discrete-time lyapunov matrix in controllable canonical form. *IEEE Transactions on Automatic Control*, AC-24(3):481–482, 1979.

[6] B. Hanzon. A faddeev sequence method for solving lyapunov and sylvester equations. *Linear Algebra and Its Application*, 241-243:401–430, 1996.

[7] L. Huang. The extension of roth's theorem for matrix equation over a ring. *Linear Algebra and Its Application*, 259:229–235, 1997.

[8] T. Jiang and M. Wei. On solutions of the matrix equations x-axb=c and x-axb=c. *Linear Algebra and Its Application*, 367:225–233, 2003.

[9] T. Mom, N. Fukuma, and M. Kuwahara. Eigenvalue bounds for the discrete lyapunov matrix equation. *IEEE Transactions on Automatic Control*, AC-30(9):925–926, 1985.

[10] T. Penzl. A cyclic low-rank smith method for large sparse lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000.

[11] P. G. Smith. Numerical solution of the matrix equation ax+xat+b=0. *IEEE Transactions on Automatic Control*, AC-16(3):278–279, 1971.

[12] R. A. Smith. Matrix equation xa+bx=c. *SIAM J. Appl. Math.*, 16:198–201, 1968.

[13] A. G. Wu, H. Q. Wang, and G. R. Duan. On matrix equations x-axf=c and x-axf=c. *Journal of Computational and Applied Mathematics*, 230:690–698, 2009.

[14] B. Zhou and G. R. Duan. Parametric solutions to the generalized discrete sylvester matrix equation mxn-x=ty and their applications. *IMA Journal of Mathematical Control and Information*, 26:59–78, 2009.

[15] B. Zhou, J. Lam, and G. R. Duan. On smith-type iterative algorithms for the stein matrix equation. *Applied Mathematics Letters*, 22:1038–1044, 2009.