System Identification through ITSIE

(Time-discrete linear SISO systems)

Marta Sofia Alves Esteves

Universidade de Trás-os-Montes e Alto Douro

Escola de Ciências e Tecnologias
Departamento de Engenharias
Vila Real, Portugal

June 2011

Contents

List of Figures

List of acronyms

Acronym	Definition
ACF	Autocorrelation function
AR	Autogressive
ARMA	Autoregressive moving average
ARMAX	Autoregressive moving average with exogenous input
ARX	Autoregressive exogenous
BB	Black-box
BJ	Box-Jenkins model
CA	Correlation analysis
CF	Crest factor
CRA	Correlation analysis in ITSIE
GUI	Graphical user interface
IO	Input and output
ITSIE	Interactive Tool for System Identification Education
LTI	Linear time-invariant
LS	Least squares
MA	Moving average
OE	Output-error model
PRBS	Pseudo-random binary sequence
RMS	Root mean square
SA	Spectral analysis
SI	System identification
SISO	Single input and single output
SNR	Signal-to-noise ratio
SS	State-space
SP	Stochastic process
FFT	Fast Fourier transform

TF	Transfer function

List of notation

Notation	Definition
a_{mag}	Magnitude of the input signal
e(t)	White noise
g_t	Impulse response
ρ	Correlation coefficient
n_k	Input delay that characterises the delay response time
n_r	Number of registers in ITSIE
n_1 and n_2	White noise sources
$S_x(n)$	Spectral density of x and length n
$R(\tau)$	Autocorrelation
T_s	Sampling time
T_{sw}	Switching time
τ	Time constant
$\bar{x}$	Mean value
$x(t)$	Input random process
σ_x	Standard deviation
σ^2	Variance
$u(t)$	Input signal at instant t
$y(t)$	Output signal at instant t
α_s	Parameter that specifies the high frequency range of interest in the signal
β_s	Parameter that specifies the low frequency ranges of interest in the signal
τ_{dom}^H	High dominant time constant
τ_{dom}^L	Low dominant time constant
λ	Scaling factor for multisine input
α_i	Fourier coefficients

N_s	Signal length
x_i	Vector of controlled variables
θ^T	Vector of unknown parameters
$p^*(q)$	Zero-order-hold equivalent transfer function
q	Forward-shift operator
A	State matrix
B	Input matrix
C	Output matrix
K	Matrix that determines the noise properties or Noise matrix
n_a	Number of poles for the model
n_b	Number of zeros plus 1
n_c	Number os poles for the disturbance model
n_s	Number of harmonics
$N^oCycles$	Number of cycles
N^oReg	Number of registers in ITSIE
w	Angular frequency
N^oSin	Numer of sine
$R(\tau)$	Normalized autocorrelation
$\rho_x(\tau)$	Autocorrelation or autocovariance
$c_{xy}(\tau)$	Cross-correlation function
$S_{xy}(n)$	Cross-spectral density
$C_{xy}(n)$	Co(-incident) spectral density -(in phase)
$Q_{xy}(n)$	Quad(-rature) spectral density -(out of phase)
$\eta(t)$	Disturbance
$\bar{y}(t)$	Output without disturbance
$\hat{S}$	Spectral density of noise
$SeqLength$	Size of the sequence in `Input signal` window

Chapter 1

What is System Identification?

System identification (SI) is a methodology for building mathematical models of dynamic systems from experimental data, i.e., using measurements of system input and output (IO) signals.

The process of SI requires the following steps:

- Measurement of the IO signals of the system in time or frequency domain.
- Selection of a candidate model structure.
- Choice and application of a method to estimate the value for the adjustable parameters in the candidate model structure.
- Validation and evaluation of the estimated model to see if the model is right for the application needs, preferably with a different set of data.

In the sequel, we will focus on the identification of linear discrete systems systems/single input and single output (SISO).

SI is about building dynamic models from data. Namelly, it uses the IO signals of a system to estimate the values of adjustable parameters in a given model structure.

1.1 What are dynamic systems?

In loose terms, a system is an object in which variables 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. Others are called disturbances and can be divided into those that are directly measured and those whose influence is only observed on the output.

In a dynamic system, the values of the output signals depend on both the instantaneous values of its input signals and also on the past behaviour of the system, that is, a dynamic system changes with time.

1.2 What is a model?

When analysing a system, some concept of how its variables relate to each other is needed. In a broad sense, an assumed relationship among observed signals is called a model of the system. Clearly, models may come in various shapes and be phrased with varying or not degrees of mathematical formalism. The intended use will determine the degree of sophistication that is required to make the model purposeful.

Also there are many systems that are dealt with using mental models and do not involve any mathematical formalisation at all.

We can formalise a dynamic system in the following manner:

where t represents an instant in a predefined time horizon $\mathcal{T}$, $t\epsilon\mathcal{T}$, $u(t)$ is the input signal and $y(t)$ is the output signal, both at instant t.

In continuous time, a simple mathematical model that describes a dynamic system is usually a differential equation

$$f_{wy}\left(\frac{d^n y(t)}{dt^n}, \frac{dy^{n-1}(t)}{dt^{n-1}}, \ldots, \frac{dy(t)}{dt}, y(t), \frac{d^m u(t)}{dt^m}, \ldots, \frac{du(t)}{dt}, u(t), t\right) = 0. \tag{1.1}$$

Since the focus of this work is on linear systems, model (1.1) can be simplified as

$$a_n(t)\frac{d^n y(t)}{dt^n} + \ldots + a_1(t)\frac{dy(t)}{dt} + a_0(t)y(t) = b_m(t)\frac{d^m u(t)}{dt^m} + \ldots + b_1(t)\frac{du(t)}{dt^m} + b_0 u(t), \tag{1.2}$$

and the discrete counterpart of model (1.2) is

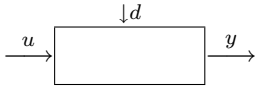
$$a_n\left[y(k-n)T_s\right] + \ldots + a_1\left[y(k-1)T_s\right] + a_0\left[y(kT_s)\right] = b_m\left[u(k-m)T_s\right] + \ldots + b_1\left[u(k-1)T_s\right] + b_0\left[u(kT_s)\right], \tag{1.3}$$

where k represents the discrete time-steps and T_s is the time between successive data samples and is called the sampling time, i.e, $(k+1) = k + T_s$.

Obtaining a good model for a system depends on how well the measured data reflects the behavior of the system.

In practice, it appears that the system response is not completely coincident with the models. The deviations may be due to modeling errors, inaccuracies in the sensors and converters and variations. In linear models, these phenomena can be represented as a disturbing signal in the output of the system.

It is convenient to distinguish between input signals and output signals. The outputs are partly determined by the inputs. In most cases, the outputs are also affected by more signals than the measured inputs. Such "unmeasured inputs" will be called disturbance signals or noise. If we denote inputs, outputs and disturbances by u, y and d, respectively, the relationship between these signals can be depicted in the diagram the following figure.



A system is said to be deterministic if the input, together with the initial conditions, uniquely determines the system. Otherwise it said to be stochastic.

In stochastic control theory, the disturbances are assumed to be stochastic processes (SP) with zero mean and covariance stationary. The spectral density factorisation Theorem (see Appendix A) allows them to be modeled as output signals of a linear minimum phase system excited by white noise. To describe these systems, one can use IO and State-space (SS) models.

This Theorem shows us that the condition for stability is absolutely essential for the stationarity of the process. The condition of minimum phase is essential in order to make your prediction.

In this context, a SP can be described by the stationary model

$$\eta(t) = \sum_{k=0}^{\infty} C(k)e(t-k), \tag{1.4}$$

where the poles and zeros are all inside the unit circle and $e(t)$ is the white noise with zero mean and variance σ^2.

The discrete model (1.3) can then be enlarged to accomodate its stochastic part, for instance adding noise as an extra input:

$$\begin{aligned} a_n y[(k-n)T_s] + \ldots + a_1 y[(k-1)T_s] + a_0[y(kT_s)] = \\ b_m u[(k-1)T_s] + \ldots + b_1 u[(k-1)T_s] + b_0[u(kT_s)] + \\ c_0 e(k) + c_1 e(k-1) + \ldots + c_{n_c} e(k-n_c). \end{aligned} \tag{1.5}$$

The data may be available either in time domain or in frequency domain, and this determines diferent choice of models.

Time domain data consists of the IO variables of the system that we record at a uniform sampling time, T_s, over a period of time.

For correct identification, in addition to having to be careful in the choice of the estimator, we should also be very careful with the choice of the excitation signal and the T_s used.

When working with uniformly sampled data, we should use the actual T_s of the experiment. Each data value is assigned to a sample time, which is calculated from the start time and T_s. When working with nonuniformly sampled data one can specify a vector of artificial instants.

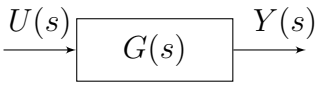
Frequency domain data represents measurements of the system IO variables that we record or store in the frequency domain. The frequency domain signals are Fourier transforms of the corresponding time domain signals.

Frequency response plots show the complex values of the a transfer function (TF) as a function of frequency.

In the case of linear continous time systems, the TF G is essentially an operator that takes the input $U(s)$ of a linear system to the output $Y(s)$.

$$Y(s) = G(s) \times U(s), \tag{1.6}$$

where $U(s) = \mathscr{L}\{u(t)\}$ and $Y(s) = \mathscr{L}\{y(t)\}$, i.e,



With $G(s) =\mid G(s) \mid e^{j\angle G(s)}$ and $U(s) =\mid U(s) \mid e^{j\angle U(s)}$.

In this case, the frequency function $G(jw)$ is the TF evaluated on the imaginary axis $s = jw$. Equation (1.6) becomes

$$Y(jw) = G(jw) \times U(jw) =\mid G(jw) \mid\mid U(jw) \mid e^{j(\angle G(jw)+\angle U(jw))}.$$

For a discrete-time system sampled with a time interval T_s, the TF relates the Z-transform of the input $U(z)$ and output $Y(z)$:

$$Y(z) = G(z) \times U(z), \quad z\epsilon\mathbb{C}.$$

In this case, the frequency function $G(e^{iwT})$ is the TF $G(z)$ evaluated within the unit circle. The argument of the frequency function $G(e^{iwT})$ is scaled by T_s to make the frequency function periodic with the sampling frequency $2\pi/T$.

It is possible to plot the frequency response of a model in order to gain insight into the characteristics of the linear model dynamics, including the frequency of the peak response and stability margins. Frequency-response plots are available for all linear parametric models and spectral analysis (SA) (nonparametric) models.

The frequency response of a linear dynamic model describes how the model reacts to sinusoidal inputs. If the input, $u(t)$, is a sinusoid of a certain frequency, then the output, $y(t)$, is also a sinusoid of the same frequency, with amplitude $\mid G(jw) \mid\mid U(jw) \mid$ and a phase shift of $\angle G(jw)$.

1.3 Model structures

A model structure is a mathematical relationship between IO variables that contain unknown parameters. Examples of model structures are with adjustable poles and zeros, SS equations with unknown system matrices and nonlinear parameterised functions.

The SI process requires the choice of a model structure and applies the estimation methods to determine the numerical values of the model parameters.

The following approaches can be used to choose the model structure:

- To choose a model that is able to reproduce your measured data and is as simple as possible. This modeling approach is called black-box (BB) modeling.
- To choose a specific structure for the model which may have been derived from physics principles, but whose parameters are unknown. The model structure may be represented, for instance, as a set of differential equations or SS systems.

1.3.1 BB modeling

In this section, the BB model type is explained as well as its use to SI. Also, we distinguish between parametric and nonparametric models and highlight the importance of each type of models.

Nonparametric identification methods are techniques to estimate model behavior without necessarily using a given parametrised model set.

Direct estimation of the impulse or the frequency response of the system are often called nonparametric estimation methods. These do not impose any assumptions on the structure of the system other than its linearity. Include correlation analysis (CA), which estimates a system's impulse response and

spectral analysis (SA), which estimates the system's frequency response are typical estimation nonparametric methods.

Direct estimation of the impulse response

A linear system can be described by its impulse response g_t with the property that:

$$y(t) = \sum_{k=1}^{\infty} g_k u(t-k). \tag{1.7}$$

The name derives from the fact that if the $u(t)$ is an impulse, i.e., $u(t) = 1$ when t=0 and $u(t) = 0$ when $t > 0$, then the output $y(t)$ will be g_t. In what follows, t represents a discrete variable in order to "uniformise" notation with ITSIE.

The impulse response coefficients are estimated directly from the IO data using CA.

In statistics, correlation expresses the relationship between two or more random variables or observed data values.

In general statistical usage, correlation can refer to any departure of two or more random variables from independence, but most commonly refers to a more specialised type of relationship between mean values. There are several correlation coefficients, ρ or r, measuring the degree of correlation. These concepts are explained in Appendix A.

Direct Estimation of the Frequency Response

The relation between IO is often written as

$$y(t) = Gu(t) + Ce(t), \tag{1.8}$$

where G is the TF and e is the additive disturbance at the input. Usually, we can estimate $G(jw)$ as a stationary stochastic process.

The system's frequency response is directly estimated using SA. This estimates the spectrum of the $e(t)$ in the system description. This description of the system gives considerable engineering insight into its properties.

A specific model structure is assumed, and the parameters in the chosen structure are estimated from data. This opens up a large variety of possibilities, corresponding to different ways of describing the system.

Parametric identification methods are techniques to estimate the parameters of given model structures. Basically, it is a matter of finding, by numerical search, those numerical values of the parameters that give the best agreement between the model's (either simulated or predicted) output and the measured output. In other words, a common and general method for estimating the parameters is the prediction error approach, where simply the parameters of the model are chosen so that the difference between the model's (predicted) output and the measured output is minimised. This method is available to all model structures.

In what follows, we distinguish three diferent model structures. Namely polynomial models, least-squares (LS) model and SS models.

1.4 Types of parametric methods

In this section, we present differents polynomial model structures such as: Moving average models (MA), Autoregressive models (AR), Autoregressive moving average models (ARMA), Autoregressive exogenous models (ARX), Exogenous ARMA models (ARMAX), Output-Error (OE) models and Box-Jenkins (BJ) models, and then, the LS model and SS models. The choice of these models follows ITSIE.

1.4.1 Polynomial model types

In order to review the most common types of polynomial models, we first define the time-shift operator q.

The general polynomial equation is written in terms of the time-shift operator q^{-1}. To understand this time-shift operator, consider the discrete-time difference equation (1.3). After changing the order of the terms and dividing every term by a_n and considering the coefficients $a_1, \ldots, a_{n_a}$ and $b_1, \ldots, b_{n_b}$ it can be written as:

$$y(t)+a_1y(t-T_s)+...+a_{n_a}y(t-n_aT_s) = b_1u(t-T_s)+...+b_{n_b}u(t-n_bT_s). \quad (1.9)$$

Defining the time-shift operator as $q^{-1}u(t) = u(t-T_s)$, (1.9) becomes:

$$y(t) + a_1q^{-1}y(t) + ... + a_{n_a}q^{-n_a}y(t) = b_1q^{-1}u(t) + ... + b_{n_b}q^{-n_b}u(t). \quad (1.10)$$

If we define $A(q) = 1 + a_1q^{-1} + a_2q^{-2} + \cdots + a_{n_a}q^{-n_a}$ and $B(q) = b_1q^{-1} + b_2q^{-2} + \cdots + b_{n_b}q^{-n_b}$, equation (1.10) can be rewritten in the compact form as:

$$A(q)y(t) = B(q)u(t). \quad (1.11)$$

Whether the stochastic part of the model is also included, model (1.11) becomes

$$A(q)y(t) = B(q)u(t) + C(q)e(t), \quad (1.12)$$

where in model (1.4) we consider $C(q) = 1 + c_1q^{-1} + \ldots + c_{n_c}q^{-n_c}$ and considering n_c terms in the sum, $\eta(t) = C(q)e(t)$.

The model types differ among themselves by how many of the polynomials of (1.12) are included. Thus, different model types provide varying levels of flexibility for modeling the dynamics and noise characteristics.

The parametric polynomials models types are subsets of the following general polynomial equation that for SISO is

$$A(q)y(t) = \left[\frac{B(q)}{F(q)}\right]u(t-n_k) + \frac{C(q)}{D(q)}e(t), \quad (1.13)$$

where $D(q) = 1+d_1q^{-1}+\ldots+d_{n_d}q^{-n_d}$ and $F(q) = 1+f_1(q)^{-1}+\ldots+f_{n_f}q^{-n_f}$.

$u(t)$ is the input and n_k is the input delay that characterises the delay response time, i.e., the number of samples, n_k, corresponding to the input time delay, given by the number of samples before the output responds to the input. The variance of the white noise $e(t)$ is assumed to be λ. $A(q)$ corresponds to the poles that are common for the dynamic model and the noise model. Using common poles for dynamics and noise is useful when the

disturbances enter the system at the input. F determines the poles unique to the system dynamics, and D determines the poles unique to the disturbances.

To estimate polynomial models of type (1.13), we must select some orders as a set of integers that represent the number of coefficients for each polynomial. Thus, a selected structure should include n_a parameters of A, n_b parameters of B and n_c parameters of C, n_d parameters of D and n_f parameters of F.

The number of coefficients in the polynomials' denominator equals the number of poles, and the number of coefficients in the polynomials' numerator is equal to the number of zeros plus 1. When the dynamics from $u(t)$ to $y(t)$ contains a delay of n_k samples, then the first n_k coefficients of B are zero.

Moving average model

If we consider model (1.4) written in terms of the time-shift operator, q^{-1}, this model may be written as

$$\eta(t) = C(q)e(t),$$

as we have already seen.

As this model is a regression input, a model of this type is known as a moving average (MA) model.

These models are very simple and can describe any SP. However, to have acceptable precision it is often necessary to have a high order (n_c very large).

Autoregressive models

Sometimes, it may be preferable to describe $e(t)$ by a model of type

$$\eta(t) + d_1\eta(t-1) + \cdots + d_{n_d}\eta(t-n_d) = e(t),$$

$$D(q)\eta(t) = e(t).$$

In these models there is a regression in $\eta(t)$ and for this reason these are generally known as auto-regressive (AR) models. Their TF is

$$C(q) = \frac{1}{D(q)}$$

and its order, n_d, is usually significantly lower than that of MA models.

Autoregressive moving average models

However, if we use regression models in the output $\eta(t)$ and moving average in the input $e(t)$, we obtain lower order models. These models consist in

$$\eta(t) + d_1\eta(t-1) + \cdots + d_{n_d}\eta(t-n_d) = e(t) + c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c)$$

and can be written in the compact form as

$$D(q)\eta(t) = C(q)e(t). \tag{1.14}$$

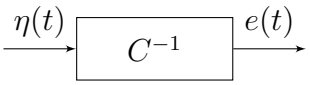
These models are known as autoregressive moving average models (ARMA). The AR models are a special case of ARMA when $c_1 = c_2 = \cdots = c_{n_c}$ and MA models are a special case when $d_1 = d_2 = \cdots = d_{n_d} = 0$.

The ARMA models is an important class of noise models, in which the process noise is considered as a solution of a difference equation, whose input is white noise.

The stochastic processes are known by the name of the models that generate them. Thus, SP generated by ARMA models are ARMA processes; in the same away, the models generated by AR processes are generated by the AR and MA models are MA processes.

The invertibility of $C(q)$ requires $\frac{1}{C(q)}$ to be a stable function. This allows it to generate a white noise feeding time with a linear time-invariant (LTI) system whose TF is $\frac{1}{C(q)}$ and the process is $\eta(t)$.

For this reason, $\frac{1}{C(q)}$ is also known as whitening filter, as can seen in following diagram:



Consider again the equation $y(t) = \frac{B(q)}{A(q)}u(t) + \eta(t)$. If $\eta(t)$ is a SP with zero mean, stationary covariance and spectral density, $C(e^{(jw)})C(e^{(-jw)})\sigma^2$, it can be considered that $\eta(t)$ is an output signal of a system and $C(q)$ is excited by a succession of random variable uncorrelated and having variance σ^2, i.e., white noise.

By incorporating the ARMA model in the linear system $\eta(t) = \frac{C(q)}{D(q)}e(t)$, one obtains the stochastic dynamic system model.

Autoregressive exogenous models

ARX models are obtained by adding a moving average of the input to the AR model. The "X" stands for the exogenous variable or external input system.

The most used model type is the simple linear difference equation:

$$\begin{aligned} y(t) + a_1 y(t-1) + ... + a_{n_a} y(t-n_a) = \\ b_1 u(t-n_k) + ... + b_{n_b} u(t-n_k-n_b+1) + e(t). \end{aligned} \tag{1.15}$$

In this type, n_a is equal to the number of poles, n_b is the number of zeros plus 1 and $e(t)$ is the white-noise disturbance.

The model orders n_a, n_b and n_k need to be specified to estimate ARX models.

A general IO linear model for a SISO system with input u and output y, for the ARX model is given by

$$A(q)y(t) = B(q)u(t-n_k) + e(t). \tag{1.16}$$

Exogenous ARMA models

As the ARX models, ARMAX models are obtained by adding the entry end of the ARMA noise.

For a SISO system, the ARMAX model is:

$$\begin{aligned} y(t) + a_1 y(t-1) + ... + a_{n_a} y(t-n_a) = \\ b_1 u(t-n_k) + ... + b_{nb} u(t-n_k-n_b+1) + \\ e(t) + c_1 e(t-1) + ... + c_{n_c} e(t-n_c) \end{aligned} . \tag{1.17}$$

The ARMAX model is more flexible than the ARX model, because the ARMAX model type contains an extra polynomial to model the additive disturbance.

A general IO linear model for a SISO system with input u and output y can be written as:

$$A(q)y(t) = B(q)u(t-n_k) + C(q)e(t). \tag{1.18}$$

In this model type, AR refers to the A-polynomial, MA to the noise C-polynomial and X to the "extra" input $B(q)u(t-n_k)$.

ARMAX extends the ARX structure by providing more flexibility for modeling noise using the C parameters of equation (1.18). Use ARMAX when the dominating disturbances enter at the input. Such disturbances are called load disturbances.

Output-Error model

When $A(q)$, $C(q)$, and $D(q)$ are equal to 1, the general-linear polynomial model reduces to the OE model. This model describes the system dynamics separately from the stochastic dynamics. The OE model does not use any parameters for simulating the disturbance characteristics

$$y(t) = \left[\frac{B(q)}{F(q)}\right] u(t-n_k) + e(t). \tag{1.19}$$

The parameters of the OE model type are estimated using a prediction error method.

OE should be used to parameterise dynamics, without estimating a noise model.

Box-Jenkins model

The parameters of BJ model

$$y(t) = \left[\frac{B(q)}{F(q)}\right] u(t - n_k) + \frac{C(q)}{D(q)} e(t) \tag{1.20}$$

are estimated using a prediction error method.

Use BJ models when the noise does not enter at the input, but is primarily a measurement disturbance. This structure provides additional flexibility for modeling noise.

1.4.2 Least squares

Consider N experiments

$$y_i = \sum_{j=1}^{p} x_{ji}\theta, \quad i = 1, \ldots, N, \tag{1.21}$$

where $x_i = (x_1, x_2, ..., x_p)$ is a vector of controlled variables and $\theta^T = (\theta_1, ..., \theta_p)$ is a vector of unknown parameters.

A more realistic mathematical model for the phenomenon is

$$y_i = x\theta^T + \epsilon_i, \quad i = 1, \ldots, N, \tag{1.22}$$

where ϵ_i is a random variable representing the error of experiment i.

Considering (1.22) for $i = 1, ..., N$, hence:

$$\begin{aligned} y_1 &= x_{11}\theta_1 + x_{12}\theta_2 + \ldots + x_{1p}\theta_p + \epsilon_1 \\ y_2 &= x_{21}\theta_1 + x_{22}\theta_2 + \ldots + x_{2p}\theta_p + \epsilon_2 \\ &\vdots \\ y_N &= x_{N1}\theta_1 + x_{N2}\theta_2 + \ldots + x_{Np}\theta_p + \epsilon_N. \end{aligned} \tag{1.23}$$

and with,

$$Y = [y_1, ..., y_N]^T, \tag{1.24}$$

$$X = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1p} \\ ... & ... & ... & ... \\ ... & ... & ... & ... \\ x_{N1} & ... & ... & x_{Np} \end{bmatrix} = \begin{bmatrix} x_1 \\ ... \\ ... \\ x_N \end{bmatrix} \tag{1.25}$$

and E is

$$E = [\epsilon_1 ... \epsilon_N]^T. \tag{1.26}$$

Then

$$Y = X\theta + E. \tag{1.27}$$

In this context, the parameters can be calculated (estimated) minimising the following function:

$$E = \sum_{i=1}^{N} \mid E_i \mid^2 \tag{1.28}$$

that is, seeking to minimize the sum of squared errors. In matrix notation it becomes:

$$E = (Y - X\theta)^T (Y - X\theta), \tag{1.29}$$

Differentiating E according to θ, we obtain

$$\frac{dE}{d\theta} = -2X^T Y + 2X^T X\theta. \tag{1.30}$$

To calculate its minimum we need to solve the normal equations

$$\frac{dE}{d\theta} = 0 \Longrightarrow X^T X\theta = X^T Y. \tag{1.31}$$

If $X^T X$ is invertible a unique estimate exists

$$\hat{\theta} = (X^T X)^{-1} X^T Y.$$

Note that $N \geq p$ is a necessary condition for $X^T X$ to be invertible. When the matrix $X^T X$ is singular the estimate is not unique.

1.4.3 State-Space model

In this section we consider the general problem of estimating parameter for linear SS models. Many of the models that we have studied so far are special cases of this problem. The extension to SS models also opens up the possibility to study estimation problems.

The basic SS model in innovation form can be written as

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + Ke(t), \\ y(t) &= Cx(t) + Du(t) + e(t), \end{aligned} \tag{1.32}$$

where $A\epsilon\mathbb{R}^{n_k \times n_k}$, $B\epsilon\mathbb{R}^{n_k \times n_u}$, $C\epsilon\mathbb{R}^{n_y \times n_k}$ and $D\epsilon\mathbb{R}^{n_y \times n_u}$ are matrices with suitable dimensions. The $x(t)$ is the state vector at instant t. The $u(t)$ is the input signal and the $y(t)$ is the output signal. On the other hand, n_y is the number of outputs and n_u is the number of inputs.

A is said the state matrix, B the input matrix, C the output matrix, D is the matrix of direct transmission between input and output and matrix K determines the noise properties.

To estimate an SS model, one needs to supply only 1 parameter. This parameter is the number os states, n_k.

1.5 BB model structure and orders' selection

BB modeling is useful when our primary interest is to fit data regardless of a particular mathematical structure of the model. These model structures vary in complexity according to the needed flexibility to account for the dynamics and noise in the system. We can choose one of these structures and compute its parameters to fit the measured response data.

Typically, one starts with a simple linear model and progresses to more complex structures.

The simplest linear BB require the fewest options to configure. For instance:

- A linear ARX model, which is the simplest IO polynomial model, or
- A SS model, can be estimated by specifying the number of model states.

To select a model structure using the model order is also possible. But one needs to be aware that the definition of the model order varies with the type of model selected. In some cases, such as for linear ARX and SS model structures, we can estimate the model order from the data.

If the simple model structures do not produce good models, then we can select more complex model structures according to the rules:

- Specify a higher model order for the same linear structure.
  Higher model orders increase the model flexibility for capturing complex phenomena. However, unnecessarily high orders can make the model less reliable.

- Adopt explicit modeling of the noise, as shown in equation (1.8), where C models the additive disturbance by treating the disturbance as the output of a linear system driven by a white noise source $e(t)$.
  Using a model structure that explicitly models the additive disturbance can help to improve the accuracy of the measured component G. Furthermore, such a model structure is useful when our main interest is to use the model for predicting future response values.

- Use a different linear structure.

- Use a nonlinear model structure. Nonlinear models have more flexibility in capturing complex phenomena than liner models of similar orders.

  However, a linear model is often sufficient to accurately describe the system dynamics and, in most cases, we should first try to fit linear models. If the linear model output does not adequately reproduce the measured output, we might need then to use a nonlinear model.

  We can assess the need for using a nonlinear model structure by plotting the response of the system to an input.

For real data, there is no such thing as a "correct model structure". However, different models can give quite different model quality. The only way to find this out is to try out a number of different models and compare the properties of the obtained model.

Ultimately, the simplest model structure that provides the best fit to measured data should be chosen.

1.6 Model quality evaluation

This section shows how to evaluate model quality. Some necessary statistical concepts are defined in Appendix A that are necessary to this section.

After a model is estimated, it can be evaluated by:

- Comparing the model response to the measured response.
- Analysing residual.
- Analysing model uncertainty.

1.6.1 Comparison between the model simulated response and the measured response

It is natural to evaluate the quality of a model by comparing the model response to the measured output for the same input signal.

The models are obtained with a particular input and the simulated responses are compared against the measured values for the same input applied to the real system. In this way, the model that has a better fit percentenge should be chosen. Consider

$$\frac{|| \, y_{means} - y_{sim} \, ||_2}{|| \, y_{means} \, ||_2} \times 100, \tag{1.33}$$

where y_{means} is the measured output, y_{sim} is the simulated output, and $y_{means} - y_{sim}$ are the residuals. Expression (1.33) is called the fit percentage .

1.6.2 Residuals

It is important to run a residual analysis to assess the quality of the model. Residuals represent the portion of the output data that is not explained by the estimated model. A good model has residuals uncorrelated with past inputs, i.e, the residuals should be independent of the input and white noise.

In equation (1.8), the noise source $e(t)$ represents that part of the output that the model could not reproduce. Otherwise, they would be more in the

output that originates from the input and that the model has not picked up. To test this independence, we have the cross-correlation function between input and residuals and the autocorrelation function. It is wise also to display the confidence region for this function. For an ideal model the correlation function should lie entirely between the confidence lines for positive lags.

1.6.3 Model uncertainty analysis

When the model parameters are estimated from the data their nominal values should be accurate within a confidence region. The size of this region is determined by the values of the parameter uncertainties computed during estimation. The magnitude of the uncertainties in parameters can result from unnecessarily high model orders, inadequate excitation levels in the input data and poor signal-to-noise ratio (SNR) [1] in the measured data.

1.7 Concluding remarks

In this chapter, we studied all aspects relating to the IS, including all concepts underlying the same the explanation of the ITSIE, so that we can give the reader a more complete view of all processes surrounding the identification of a system.

After reading this chapter, the reader/user should be able, among other things, to distinguish IO signals of a system, to select the best model type from the candidates studied, and apply the methods of estimation and validation to a model.

[1] Signal-to-noise ratio is a measure used in science and engineering to quantify how much a signal has been corrupted by noise. It is defined as the ratio of signal power to the noise power corrupting the signal.

Chapter 2

Interactive Software Tool for System Identification

This chapter contains the conceptual basis and describes the main features and functionality of an interactive software tool developed to support SI education and discovery.

The Interactive Tool for System Identification Education (ITSIE) is an interactive tool where all stages of identification of systems co-exist in the same screen in windows properly organised which is very useful from an educational point of view. I.e, the great advantage of this tool is to make an identification experiment using a single screen, with the different stages connected interactively in such a way that a modification in one stage is automatically visualised in the remaining stages.

This novel interactive software tool for SI was developed as a prototype tool for SI education and in collaboration with Professor Sebastián Dormido, from Universidad Nacional de Educacion a Distancia, UNED-Madrid, Spain, and Professors José Luis Guzmán Sanchez and Manuel Berenguel Soria from the University of Almeria, Spain, and Daniel Rivera from the Arizona State University, USA, in 2009.

The interactive software tools have been proven as particularly useful techniques with high impact on control education [5]. Interactive tools supply a real-time connection between decisions made during the design phase and the results obtained in the analysis phase of any control-related project. Also, SI is a field rich in visual contents that can be represented intuitively and

geometrically [7].

ITSIE has been developed using Sysquake, a Matlab-like language with fast execution and excellent facilities for interactive windows, and is delivered as a stand-alone executable that is made readily accessible to students and engineers.

SI is an interactive process that deals with the problem of building dynamic models from experimental data and is a key component in the control engineering practice. SI education forms an essential part of any comprehensive control engineering curriculum and, as such, requires flexible and simple software tools. Although there are many others software tools for SI, they all present several disadvantages when viewed from a primarily educational point of view [3]. Usualy, these tools do not evaluate all stages of the SI process and also have a graphical layout that can be confusing for students and can lead to lost of interest from their point of view. Thence, a new generation of software tools addressing these concerns are needed in support of advancing SI education.

The ITSIE tool is freely available through `http://aer.ual.es/ITSIE/` and does not require a Sysquake license in order to execute. One consideration that must be kept in mind is that the tool's main feature-interactivity-cannot be easily illustrated with written text. Nonetheless, some of the features and advantages of the application are shown below.

When developing a tool of this kind, it is very important that the developer keeps in mind the organisation of the main windows and menus to facilitate to the user an understanding of the identification technique.

The purpose of this chapter is to explain the functionality of this tool.

As mentioned in Chapter 1, a comprehensive SI procedure consists of the study the IO signals, data preprocessing, selection of a candidate model structure and applicattion of methods and estimation of the parameters for the candidate model as well as of the validation of the estimated models to see if the model choice is adequate. These are some of the theoretical aspects described in the following sections.

In ITSIE, the plant to be identified consists of a discrete-time system, sampled at a value specified by the user (the default value is $T_s = 1$ min) and subject to noise and disturbances according to the following model:

$$y(t) = p^*(q)(u(t) + n_1) + n_2. \tag{2.1}$$

The notation follows Chapter 1, therefore, in equation (2.1) $y(t)$ is the measured output signal and $u(t)$ is the input signal that is designed by the user, $p^*(q)$ is the zero-order-hold equivalent TF for $p(q)$, where q is the time-shift operator. The system is subject to two stationary white noise sources, $n_1(t)$ and $n_2(t)$, introduced at different locations in the plant. n_1 allows evaluation of the effects of the autocorrelated disturbances in the data, while $n_2(t)$ introduces white noise directly to the output signal.

In ITSIE graphical tool, it is very important to understand the layout, including the toolbar, available at the top of ITSIE to be able to start the study of identification, as can we see in following figure.

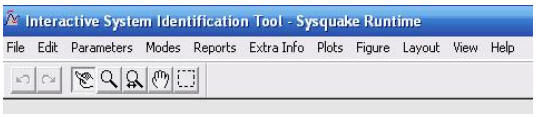


Figure 2.1: The toolbar at the top of ITSIE.

The "`Parameters`, `Modes` and `Reports`" menus are the most relevant. It is through the `Modes` menu selection that we choose the mode we want to work in, then in the `Parameters` menu, we set the values for the parameters under way, and finally, when we finish the identification procedure, we go to the `Reports` menu, and select `Generate report` option and save the report with the required parameters for the study developed. The other menus refer to Sysquake and are not relevant here, because they are not needed for ITSIE use.

2.1 Different working modes

The tool has two different modes, the simulation mode and real data mode, that are very useful from an educational and industrial point of view.

The simulation mode enables the student to evaluate the main stages of system identification, from input signal design through model validation, simultaneously and interactively in one screen on a user-specified dynamic sys-

tem. The real data mode allows the user to load experimental data obtained externally and identify suitable models in an interactive fashion.

In the Chapter 3 of this manual, we detail study these modes using practical examples. We can find two tutorials, respecting to the different working modes of ITSIE.

2.2 Simulation mode

In this section, we explain the funcionality and features of the simulation mode step by step.

2.2.1 Selection of input signals

The sucess of any identification methodology hinges on the availabilities of an informative input/output data set obtained from a sensibly designed identification experiment. An input of a test signal should be able to excite all modes in the range frequency response.

Informative input signals that are friendly to process operations are highly desirable in the identification practice, with the goal of finding a control-relevant model estimate within an acceptable time-period.

ITSIE makes available two types of input periodic signals for the identification of systems, the pseudo-random binary sequence (PRBS) and the multisine input. Both are considered deterministic and having periodic features which has some advantages in SI. We study these two excitation signals, in Section 2.2.1, respectively.

PRBS

The PRBS is a binary signal generated using a shift register module 2. A cycle of a PRBS sequence is determined by the number of bits of shift register, n_r, and a switching time, T_{sw}. The signal is repeated after $N_s T_{sw}$ time units, where $N_s = 2^{n_r} - 1$. The power spectral density for a PRBS signal is given by the following expression:

$$\Phi_u(w) = \frac{a_{mag}^2(N_s+1)T_{sw}}{N_s}\left[\frac{\sin(\frac{wT_{sw}}{2})}{\frac{wT_{sw}}{2}}\right]^2. \tag{2.2}$$

In the expression given above, a_{mag} is the magnitude of the PRBS signal, w is the angular frequency and N_s is the signal length.

The specification of parameters and applying direct time constants based guidelines for the design of entry are evaluated in ITSIE. However, in practice, little is known about the process dynamics at the start of the identification testing, and plant operating restrictions will discourage excessively long, or very intrusive, identification experiments. A guideline that provides a suitable estimate of the frequency band, over which excitation is required, can be given by the following equation.

$$\frac{1}{\beta_s \tau_{dom}^H} \leq w \leq \frac{\alpha_s}{\tau_{dom}^L}. \tag{2.3}$$

In this equation, α_s is a factor representing the closed-loop speed of response, written as a multiple of the open loop response time and β_s is an integer factor representing the settling time of the process. This parameters specify the high and low frequency ranges of interest in the signal, respectively, for a given range of high and low dominant time constants, defined by τ_{dom}^H and τ_{dom}^L, i.e., β_s and τ_{dom}^H define the lower bound of the frequency band, α_s and τ_{dom}^L define the higher bound of the frequency band. These two factors, τ_{dom}^H and τ_{dom}^L, increase the frequency bandwidth of the input signal.

Equation (2.3) is used, to specify variables in PRBS inputs. Expressions to specify T_{sw} and n_r, based on equation (2.3), give rise to the expression (2.4):

$$T_{sw} \leq \frac{2.8\tau_{dom}^L}{\alpha_s}, \quad N_s = 2^{n_r} - 1 \geq \frac{2\Pi\beta_s\tau_{dom}^H}{T_{sw}}, \tag{2.4}$$

where n_r and N_s are integer values and T_{sw} is an integer multiple of the sampling time T_s.

The PRBS has been widely used for SI (AH Tan and KR Godfrey, 2002). This signal, in addition to the deterministic capacity and frequency, may

include properties of white noise and an optimal crest factor (CF) [2], which is an advantage and a point of interest in choosing this type of signal.

The CF is a measurement of a waveform calculated from the peak amplitude of the waveform divided by the RMS [3] value of the waveform.

$$CF = \frac{\mid x \mid_{peak}}{x_{RMS}}, \tag{2.5}$$

It is therefore a dimensionless quantity. While this quotient is most simply expressed by a positive rational number in commercial products it is also commonly stated as the ratio of two whole numbers, e.g., 2:1. In signal processing applications it is often expressed in decibels (dB).

In equation (2.5), x is the signal and $x_{RMS} = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}}$.

It provides a measure of how well distributed the signal values are over the input span. A low CF indicates that most of the elements in the input sequence are distributed near maximum values of the sequence. Reducing the crest factor of an input signal can significantly improve the SNR of the resulting plant output, contributing to plant-friendliness during experimental testing, i.e, this factor should be minimised in order to better handle the input spectrum and to match what we want.

Multisine

Another type of input signal very commonly used in SI is the multisine.

These signals, as it was already mentioned, are also deterministic and periodic. They are represented in the case of a single input, as the sum of

[2] The crest factor (CF) or peak-to-average ratio (PAR) or peak-to-average power ratio (PAPR) is a measurement of a waveform, calculated from the peak amplitude of the waveform divided by the RMS value of the waveform.

[3] In mathematics, the root mean square (abbreviated RMS or rms), also known as the quadratic mean, is a statistical measure of the magnitude of a varying quantity.
It can be calculated for a series of discrete values or for a continuously varying function. The name comes from the fact that it is the square root of the mean of the squares of the values. It is a special case of the generalized mean with the exponent $p = 2$.

sinusoids, as shown in the equation below:

$$u(k) = \lambda \sum_{i=1}^{n_s} \sqrt{2\alpha_i} \cos[w_i kT + \phi_i], \tag{2.6}$$

where the frequency is $w_i = \dfrac{2\pi i}{N_s T}, \quad n_s \leq \dfrac{N_s}{2}$.

The power spectrum of the multisine input is

$$\Phi_u(w_i) = \Big(\frac{\lambda^2 \alpha_i}{2}\Big) N_s, \quad i = 1, \cdots, n_s \tag{2.7}$$

and is directly specified through the selection of a scaling factor, λ, the Fourier coefficients, α_i, the number of harmonics, n_s, and the signal length, N_s [5]. As in the PRBS case, we can use equation (2.3) to specify variables for multisine type inputs such as N_s, as shown below.

$$N_s \leq \frac{2\pi \beta_s \tau_{dom}^H}{T}, \quad n_s \geq \frac{N_s T \alpha_s}{2\pi \tau_{dom}^L}. \tag{2.8}$$

In both cases in order to reduce the variance of the model it is benefic to apply a wider range of the input signal amplitude or a_{mag} (standard deviation) that will implement the largest possible number of input cycles m. In practice, decisions concerning the magnitude of the input signal, spectral content, duration of experimental testing are dictated by physical limitations, economics and security considerations Ljung [2].

Multisine inputs are easy to implement in a real-time setting. As deterministic signals, one cycle can be designed to include all the frequency ranges needed for consistent estimation of the plant dynamics. Under noisy testing conditions, multiple cycles can be implemented until the variance in the model estimate is reduced to acceptable levels [2].

In this case, a guideline that also provides a suitable estimate of the frequency band over which excitation by the equation (2.2). As the PRBS, this equation is also used in ITSIE to design variables in multisine inputs. Expressions for specifying T_{sw} and n_r are based on equation (2.2) from which (2.3) results.

2.2.2 Input design

A parameter definition section and three interactive windows characterise the input design stage.

The parameter definition section is called `Input signal Parameters`, being located at the top of the middle section of the tool as is shown in the following figures for PRBS and Multisine, respectively.

PRBS

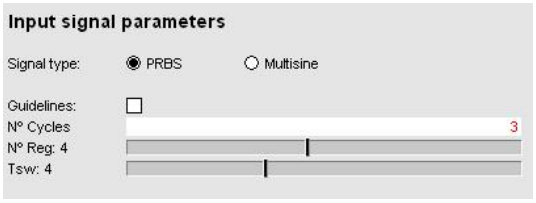


Figure 2.2: Menu input signal parameters for PRBS.

This figure, shows how to select the type of input signals i.e., between a PRBS or Multisine signals. Then if we choose guidelines [4] or not. In case the guidelines box is not active, choose the $N°cycles$, the $N°Reg$ and the T_{sw}. At the same time, from the `Input signal parameters` window, it is possible to modify the T_{sw} dragging on the magenta vertical line, the signal amplitude using the green horizontal line and the $N°cycles$ dragging on the small black triangle located at the x-axis. Furthermore, the T_{sw} can be changed from the `Power Spectrum` window using the vertical lines, we find that the number of stripes of the power spectrum relates to the value of T_{sw}.

As we change the value of the $N°cycles$, this is reflected in the `Power Spectrum`, `Step response`, `Output signal`, `Full input signal`. The $N°Reg$ and T_{sw} changes all windows on the screen.

Consider, for example, a fifth-order system whose TF is $p(q) = \dfrac{1}{(1+q)^5}$ and $T = 1$ min. First, we choose as the input of the systems a PRBS of 3 cycles.

[4] When the user does not select the guidelines, that is, the guidelines checkbox is not active, the `Input signal Parameters` can be interactively modified using specific sliders or dragging on the windows.

We select the guidelines and after $\alpha_s = 2$ and $\beta_s = 3$. These α_s and β_s will wide the frequency band of emphasis in the input signal and increase the resolution of the input signal spectrum.

Figure 2.3 describes the use of a PRBS input signal for this system and aims to illustrate the PRBS signal.

Figure 2.3: ITSIE interactive tool user interface for two cycles of a PRBS input applied to a simulated fifth-order system, with selection of different parameters from the `Input signal parameters` menu.

The Figure 2.4 clarifies the use of guidelines for $\alpha_s = 2$, $\beta_s = 3$ and a dominant time constant range ($3 = \tau_{dom}^{L} \leq \tau_{dom} \leq \tau_{dom}^{H} = 5$), this is noticeable in their effect on the input signal.

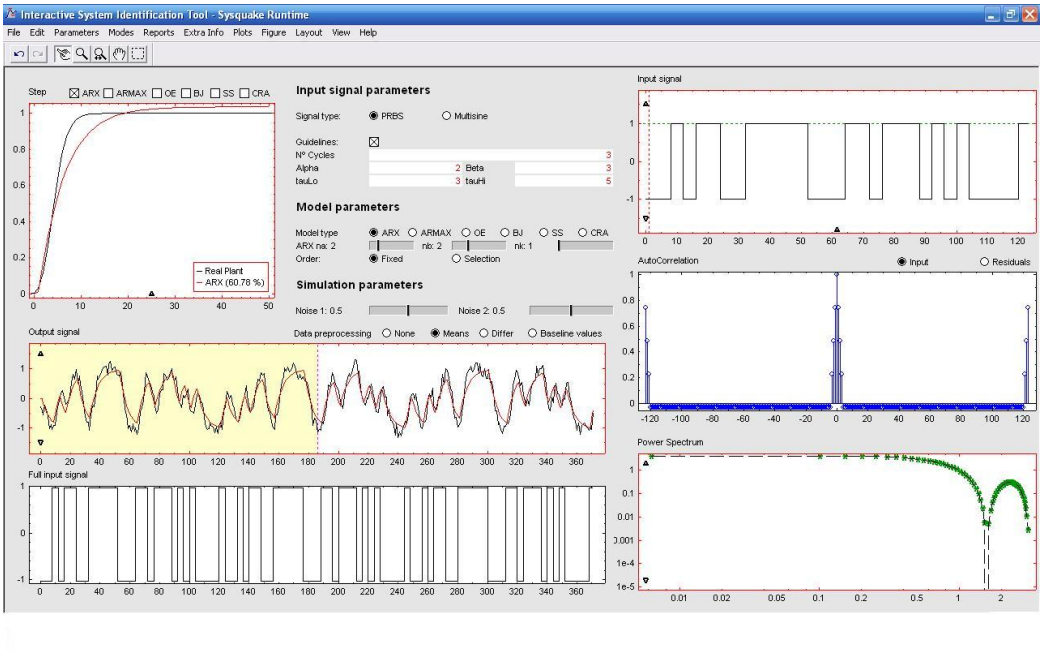


Figure 2.4: ITSIE interactive demonstration of three cycles of a PRBS input applied to simulate a system of fifth-order with selection of guidelines.

Multisine

If we select `Multisine`, we should select, in `Input signal parameters` menu the `Min Crest Factor` or not, choose the $N^{o}cycles$, the $SeqLength$, the $N^{o}Sin$, the $Maxp$, α_s and β_s, and the τ_{dom}^{L} and τ_{dom}^{H}, according the requirements.

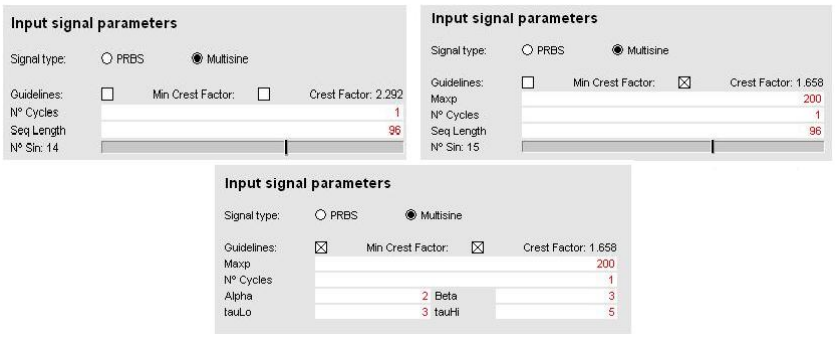


Figure 2.5: Differents Input signal parameters menus for Multisine.

At the same time, from the `Input signal parameters` window, it is possible to modify the $N^{o}cycles$ dragging on the small black triangle located at the x-axis.

As we change the value of the $N^{o}cycles$, this is reflected in multiple windows, such as: `Power Spectrum`, `Step response`, `Output signal`, `Full input signal` and `Autocorrelation`. That is, the $N^{o}cycles$ reflects how many cycles are repeated in the `Full input signal` window, as in the `Input signal` window is displayed only one signal cycle.

The $SeqLength$ is the size of the sequence of the cycle that appears in the window `Input signal`. The choice of the $N^{o}Sin$ can also be perceived in the window `Power Spectrum`, and thus is in accordance with equation (2.7).

2.2.3 Further remarks about inputs

If we analyse in detail, for example Figures 2.3 and 2.6, we can see that for a study of a practical example in ITSIE, there are some differences for both types of input.

For example, to validate the data, the model chosen for the study was the ARX. The determination of impulse and step responses are determined through certain specially chosen test signals, as they excite a system that provides an $y(t)$.

Therefore, and as an example, in the step response is noticeable that, for a PRBS input, of the fit percentage of the model is higher when compared with

Figure 2.6: Three cycles of a multisine input applied to a simulated fifth-order system.

an multisine input. The determination of the impulse and step responses are done through certain specially chosen test signals.

If the choice is the same for both signals, we also found that the PRBS has a higher compliance compared to the multisine. Another difference found is in the power spectrum. The signs show the power shifted spectra and how these are correlated in the frequency domain.

The PRBS and multisine signals are also different in terms of the CA, particularly in cross correlation between input and prediction error, namely the behavior they display, as shown in the two images presented below:

2.2.4 Plant definition and simulation parameters

The central part of the tool, is a section called `Simulation parameters`, which allows to modify interactively the noise sources of the simulated process.

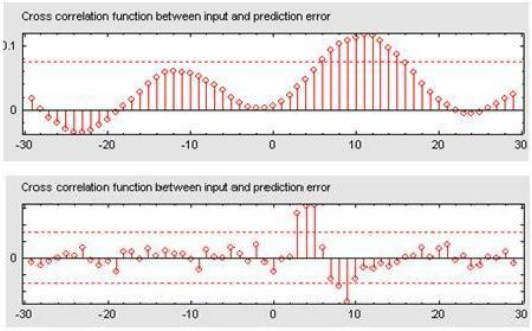


Figure 2.7: Signal multisine and signal PRBS, respectively, for ARX model.

Two sliders are available. The first one, `Noise 1`, allows for modifying the noise source $n_1(t)$ in equation (2.1) and the second one, `Noise 2`, is also used to change the noise source $n_2(t)$ in the same equation.

On the other hand, other simulation parameters, such as T_s, order selection limits, confidence intervals and baseline values are available from an entry in the parameters menu.

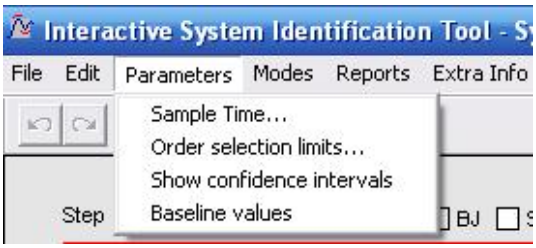


Figure 2.8: `Parameters` menu.

Furthermore, the simulated process can be configured from the `Modes` → `Simulation` menu (in addition different models can be used for the simulated process and for the n_1 filter, although by default both are the same, as shown in equation (2.1)), that also includes a couple of examples: a fifth-order system and a fluidized-bad calciner plant. The process model configuration can also be loaded and stored from files.

2.2.5 Data preprocessing

The purpose of examining the data of IO, is to find out if there are portions of the data that are not suitable for identification. That is, if the information

contents of the data is suitable in the interesting frequency regions and if the data have to be preprocessed in some away before being used for estimation.

Some aspects that we consider in data preprocessing is the detrending and prefiltering.

- Detrending the data involves removing the mean values or linear trends from the signals (the means and the linear trends are then computed and removed from each signal individually).
  It is recommended to remove at least the mean values of the data before the estimation phase, unless physical insight involving actual signal levels is built into the models.

- By filtering the IO signals through a linear filter (the same filter for all signals) we can focus on the model's fit to the system to specific frequency ranges.
  Prefiltering is a good way of removing high frequency noise in the data, and also a good alternative to detrending (by cutting out low frequencies from the pass band). Depending on the intended model used, we can also make sure that the model concentrates on the important frequency ranges. For a model used for control design, for example, the frequency band around the intended closed-loop bandwidth is of special importance.

The ITSIE data preprocessing supports mean subtraction, differenciating and subtraction of baseline values; mean detrending is applied by default.

In the tool a menu exists with several options for processing data. These options are `None`, `Means`, `Differ` and `Baseline values`, as shown in Figure 2.9.

Figure 2.9: `Data preprocessing`.

The Means option This option is used to subtract the mean values from the input and the output signals to remove ofsets, i.e.

$$m = mean(u)$$
$$u = u(t) - m$$

The None option This option is used, when the tool takes the raw real/ simulated data, without any filtering activity.

The Baseline values option The baseline values line remove from the real data the same values specified from the menu `Parameters` $\longrightarrow$ `Baseline values`. These values are called Baseline values.

The Differ option

$$u(t) = u(t) - u(t-1)$$

2.2.6 System simulation

The first thing to do is to perform a detailed simulation of a model. To do this, we choose a model from among those available on the `Model parameters` menu, in the central part of the screen bellow the choice of the `Input signal parameters`.

ITSIE has two different types of models which are the parametric and non-parametric. Within the parametric models we have, ARX, ARMAX, OE, BJ and SS, and the nonparametric include the correlation analysis (CRA), as shown bellow in Figure 2.10.

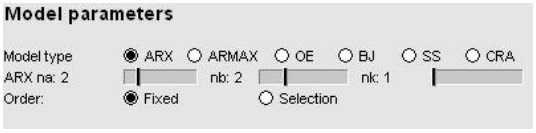


Figure 2.10: `Model parameters`.

2.2.7 System identification

As studied in Chapter 1, every model has a different structure. According to the model type different orders need to be selected. It can be `Fixed` or `Selection`, as can be seen in Figure 2.10.

In `Fixed`, we can change the orders of the model in contrast to what happens in the `Selection`. For example, if we can modify the associated orders interactively of the different models, one should first click on `Fixed`, in `Model parameters` menu and then put the cursor of the mouse over each one of the parameters to change the value according to requirements.

In ITSIE, we call model structure the choice of model type together with the choice of orders.

- To estimate an ARX model: $\left[\begin{array}{ccc} n_a & n_b & n_k \end{array}\right]$, where n_a, n_b and n_k are orders for this model, according to equation (1.16).
- To estimate an ARMAX model: $\left[\begin{array}{cccc} n_a & n_b & n_c & n_k \end{array}\right]$, where n_a ,n_b, n_c and n_k are orders for this model, according to equation (1.18).
- To estimate an OE model: $\left[\begin{array}{ccc} n_b & n_f & n_k \end{array}\right]$, where these parameters are estimated and n_b, n_f and n_k are orders for this model, according to equation (1.19).
- To estimate an BJ model: $\left[\begin{array}{ccccc} n_b & n_c & n_d & n_f & n_k \end{array}\right]$, where n_b, n_c, n_d, n_f and n_k are orders for this model, according to equation (1.20).
- To estimate an SS model: $\left[\begin{array}{c} n_k \end{array}\right]$, where n_k is order for this model.
- To estimate an CRA model: $\left[\begin{array}{c} n \end{array}\right]$, where n is order for this model ($n = 1 \ldots 100$).

The parameters' ranges are set to $n_a = n_b = n_c = n_d = n_f = n_k = 1 \ldots 10$. And can be selected or estimated.

Once a model structure is selected, the estimation and validation results for that model are shown in corresponding parts of tool.

As mentioned in Section 2.2.6, below the section `Input signal parameters` there is an area denominated `Model parameters` showing parameters to modify the orders of the different model structures. Several radio buttons are available to choose between different model structures. Also, different sliders appear making it possible to modify the associated orders interactively.

Once a input signal has been configured, the final input with all the desired cycles is shown in a window called `Full input signal`, which is located at the lower-left corner of the tool. This full input signal is applied to the simulated plant with noise in order to obtain the simulated "real data" (shown in black in the `Output signal` window), which is used as real process data

in the estimation and validation process.

Once a model structure is selected, the estimation data is used to estimate the model parameters and the validation data is used to test the resulting model. Then, for each selected model structure, the `Full input signal` is applied to the model and the results are shown in the `Output signal` window.

Finally, when we click on the `Output signal` window, it will generate a fresh realisation of the noise sequences, n_1 and n_2, enabling the user to interactively experience variability in the estimates resulting from the stochastic nature of the disturbance.

2.2.8 Model validation

On top of the `Step response` window, located on the upper left-hand corner of the tool, there is a set of checkboxes allowing to activate the different model types, as, namely, ARX, ARMAX, OE, BJ, SS and CRA, as it is shown in window `Step response`.

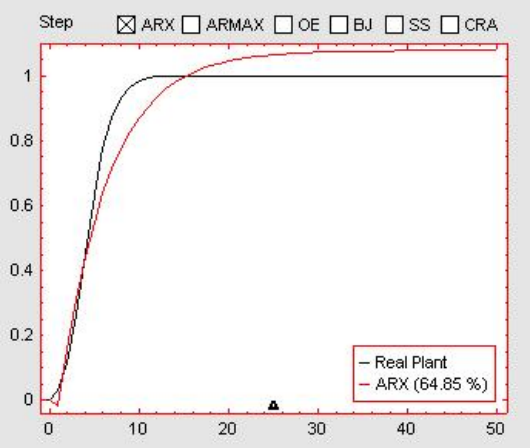


Figure 2.11: `Step response` window for ARX model, for example.

The `Step responses` window shows the step responses for each resulting model and includes a legend representing its goodness of fit percentage. The confidence intervals can also be shown in this window activating this option from the `Parameters` menu.

Different colors are used to distinguish the responses for the different models. Black for the original data, red for ARX, green for ARX with a different order selection, magenta for ARMAX, blue for OE, cyan for BJ, orange for SS and brown for CRA. These colors are consistently used in the different parts of the tool to refer to the model results.

In the `Output signal` window, an interactive magenta vertical line defines the estimation and validation data sets. The area shown in yellow (at the left of the vertical line) specifies the estimation data, whereas the white area represents the validation data (at the right side of the vertical line).

Model validation consists mainly of classical methods of simulation, cross-validation, residual analysis on the prediction errors and step responses. To enhance its educational value, the step response of the true plant is presented alongside with the ones generated by the estimated models. The percentage of the output variance explained by each model on the crossvalidation data set is reported.

The validation data is used for crossvalidation purposes. Model validation results are displayed in other three different windows: `Step responses`, `Correlation function of residuals`, and `Cross correlation function between input and output`. For all these windows, the same color distribution described before is used to represent the results of each model.

The highest percentage, present in the `Step response` window, is considered the best. As seen in this window the models in the best fits list are ordered from best at the top to worst at the bottom.

On the other hand, the `Correlation function of residuals` and the `Cross correlation function between input and output` windows, located between the `Input signal` and `Power Spectrum` windows, describe the auto and cross correlation between the input and the prediction error for each model. By default, the input `Autocorrelation` window is shown instead of these two windows. In order to switch between the input autocorrelation and residual analysis, two radio buttons are shown below the `Input signal` window that enable this commutation.

The top axes show the Autocorrelation of residuals for the output (whiteness test). The horizontal scale is the number of lags, which is the time difference (in samples) between the signals at which the correlation is estimated.

2.3 Real data mode

This mode allows to identify models from real data. To load real data, go to `Mode` → `Real data` menu. The real data can be loaded in ASCII and Matlab formats. For ASCII format, the data must be organised in columns with the following order: time, output and input signals. If the Matlab format is used, the file must contain three variables called "t", "y", and "u" for the time, the output, and the input, respectively. When real data is loaded, the tool screen is changed such as shown in detail in Chapter 3 of this manual, particularly in the choice of the example hairdryer.

Those areas in the simulation mode dedicated to input design and plant definition and simulation parameters are changed, as for example, menus to select and define the inputs and simulation parameters do not appear in this screen.

We identify a real data problem step by step.

2.3.1 Input design

Two interactive windows characterise the input design stage, the windows as `Full input signal` and `Power Spectrum`.

2.3.2 Model structure selection and parameter estimation

Once a model structure is selected, the estimation and validation results for that model are shown in corresponding parts of tool. The model structure selection and parameter estimation are exactly the same as in the **simulation mode**, described in Subsection 2.2.7, with the only difference that we are working with real data loaded from a file.

In this mode, all the model parameters are always shown simultaneously on the right side of the `Step response` window. The order selection, for the different models are specified by slidder-bars as shown in Figure 2.12.

Once an input signal has been imported, the final input with all the desired cycles is shown in a window called `Full input signal`, which is located at the lower-left corner of the tool. As in simulation mode, in the `Output`

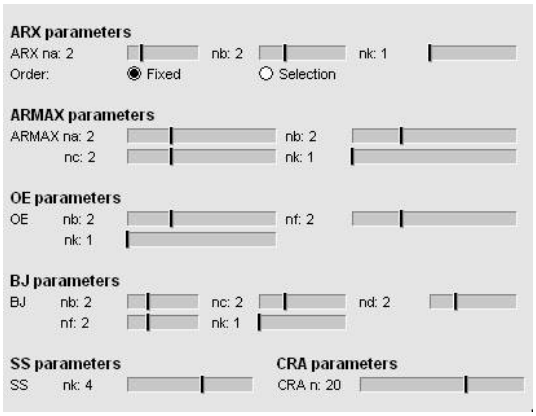


Figure 2.12: Choice of parameters of models types.

`signal` window, an interactive magenta vertical line defines the estimation and validation data sets. The area shown in yellow (at the left of the vertical line) specifies the estimation data, whereas the white area represents the validation data (at the right side of the vertical line), just like what happens in simulation mode.

2.3.3 Model validation

In this mode, also model validation consists principally of classical methods of simulation, that have been already referred in Chapter 1, such as cross-validation, residual analysis of the prediction errors and step responses.

To enhance its educational value, in the real mode, the step response of the true plant is presented alongside those that are generated by the estimated models. The percentage of the output variance explained data set is also reported.

Model validation areas, namelly windows, are exactly the same as in the simulation mode, and it follows that the entire procedure was explained in the section for the simulation mode with the only difference is that we are working with real data loaded from file.

The validation data is used for crossvalidation purposes. The model validation results are displayed in other three different windows: `Step response`, `Correlation function of residuals` and `Cross correlation function between input and output`.

On the other hand, the `Correlation function of residuals` and `Cross correlation function between input and output` windows are located at the `Power Spectrum` window.

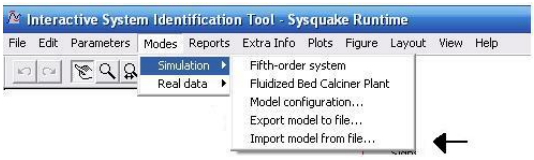
2.4 Additional options

The tool has been complemented with some additional options to be used for educational and training purposes. The options are "How to import/export models" and "Reports generation" are explained below.

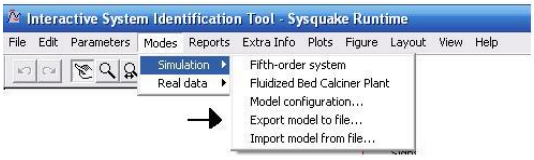
2.4.1 How to import/export models?

The user can define his own process model and the export model option saves the model TFs defined from the `Mode` $\longrightarrow$ `Simulation` $\longrightarrow$ `Model configuration` menu option, and so, we should follow the following steps.

1. Save the model through `Model` $\longrightarrow$ `Simulation` $\longrightarrow$ `Import model to file` menu option.



2. The model is saved in a binary file with `.bin` extension.
3. The model can be exported to the tool through `Model` $\longrightarrow$ `Simulation` $\longrightarrow$ `Export model from file` menu option.



Notice that this model is the "simulated model" that we are using as real plant.

2.4.2 Reports generation

After we processing the data, as described in Section 3.2.3, we can delete any data sets in `File` → `Reset Data` menu, but before we need to save our work as report.

The tool has been complemented with some additional options to be used for educational and training purposes one of these features is to print a report. Once the process model is defined, we can export the model into a file with extension txt. Load the report from the `Reports` → `Generate Report`.

The reports include information about the resulting identified models, e.g., goodness of fit, sampling time, model structure, model parameters and TF in Matlab format.

2.5 Concluding remarks

The interactive tool provides the user with multiple degrees of freedom for understanding the theoretical concepts and gives sensitivity for the impact of choices made in the different steps of the SI process. The main advantage with respect to other existing software tools is that the most important stages of SI are shown simultaneously in one screen (input design, model structure selection, parameter estimation and validation), and that the interactive features of the tool allow the user to understand and experience the relationships between these different stages, the meaning and effects of the associated parameters, the bidirectional interpretation between parameter modifications from numerical and graphical points of view.

In this chapter, we present two different examples for the simulation mode so that anyone reading this manual, may realise the functionality of this mode and the difference there is between them, through two simple case studies chosen.

In the next chapter the functionality of the tool is illustrated through an example and compared with the toolbox of Matlab, `ident`. The usefulness of this interactive is highlighted by conveying an intercative picture of SI..

Chapter 3

Using ITSIE

Our goal is to estimate and validate models from SISO data using ITSIE and find the one that best represents its system dynamics.

Large number of possible scenarios with educational value exist that can be illustrated by the ITSIE tool [5].

After reading this manual, one should be able to accomplish the following tasks using ITSIE software.

- To import data objects into ITSIE.
- To estimate and validate models from data.
- To generate a report.

We study some examples relevant for understanding the software. We work out the examples from beginning to the end, explaining throughly each step. The first example is a well known example: the hairdryer; analysed in the real data mode. The second example is a fifth-order system in the simulation mode.

This is the same example that was used in Chapter 2 to discuss the input signals.

Before trying to identify these examples, there are some theoretical concepts that should be understood, such as to distinguish input and output data, different types of processing the data, what type of models exist, etc. Concepts that were explained in previous chapters and now need to be used in the examples.

3.1 Installing the tool

To install ITSIE, one should go to the page `http://aer.ual.es/ITSIE/` in order to download the tool.

Next, we have to download the folder with extension `.zip` in accordance with the operating system of each user and then unzip this folder, to extract the software.

After installing the tool it is necessary to download the folder **documentation.zip** and then we save it in a place of our choice, that is easily accessible.

Figure 3.1: Download of ITSIE and documentation.zip.

Finally, unzip the folder documentation.zip, in order to extract the files of documentation.

Inside this folder, there is a file called dryer_data.

3.2 Case Study—Hairdryer

This system heats the air at the inlet using a mesh of resistor wire, similar to a hairdryer. The input is the power supplied to the resistor wires, and the output is the air temperature at the outlet. This can be considered as a BB, as shown in Figure 3.2.

We are going to identify this system in real data mode. This is a well-known example has been worked out in the manual of `ident`, the toolbox of Matlab.

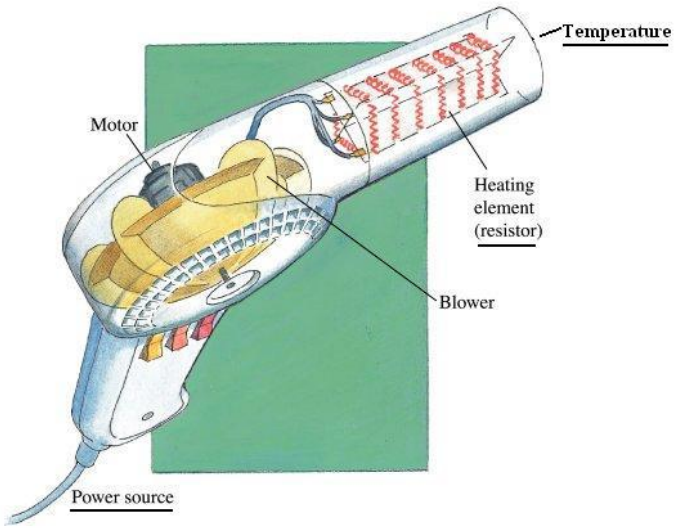


Figure 3.2: Hairdryer example [16].

3.2.1 Objectives of the case study

1. To estimate and validate linear models for a SISO system, the hairdryer, in order to find the one that best describes the system dynamics using ITSIE.
2. To tutor the fresh user how to find the best model for a given SISO system.
3. To generate a report.
4. To compare the use of ITSIE with the Matlab Toolbox, `ident`, from the learning point of view, since the former is an interactive tool.

3.2.2 Acessing and preparing data for SI

After this section we should be able to load the data.

ITSIE software has available the data file dryer_data with extension `.txt`, i.e., the data is stored in a file with extension `.txt` which contains SISO time-domain data from Feedback Process Trainer PT326. The input and output signals each contain 1000 data samples.

Loading data into the chosen file data

To study the hairdryer example, we must follow the following steps:

1. Load the data from the `Mode` → `Real data` menu, i.e., select the options `Modes` → `Real data` → `Load real data (ASCII)`.

Figure 3.3: Load the real data from the `Modes` menu.

2. Open the folder denominated by documentation and then the documentation folder data;
3. Finally, select dryer_data.txt.

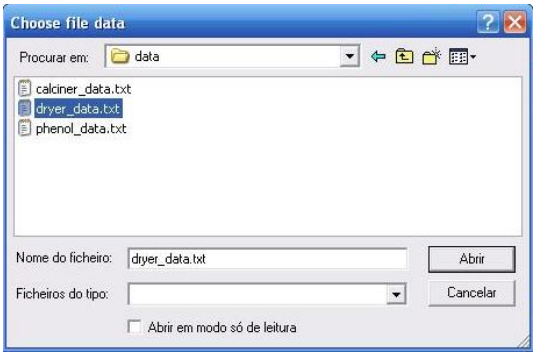


Figure 3.4: Selection of the file dryer_data.

As stated in the previous chapter, the real data can be loaded in ASCII and Matlab formats. We decided to use the ASCII format, because it is the only one that supports this file in ITSIE.

When real data is loaded, the tool screen is changed as shown in Figure 3.5.

The next step is to evaluate the data and process it for SI.

Three importants channels exist:

- Input: Power in Watts (for power units);
- Output: Temperature in °C (for temperature units);
- $T_s = 0.08$, this value is the actual sampling time in the experiment.

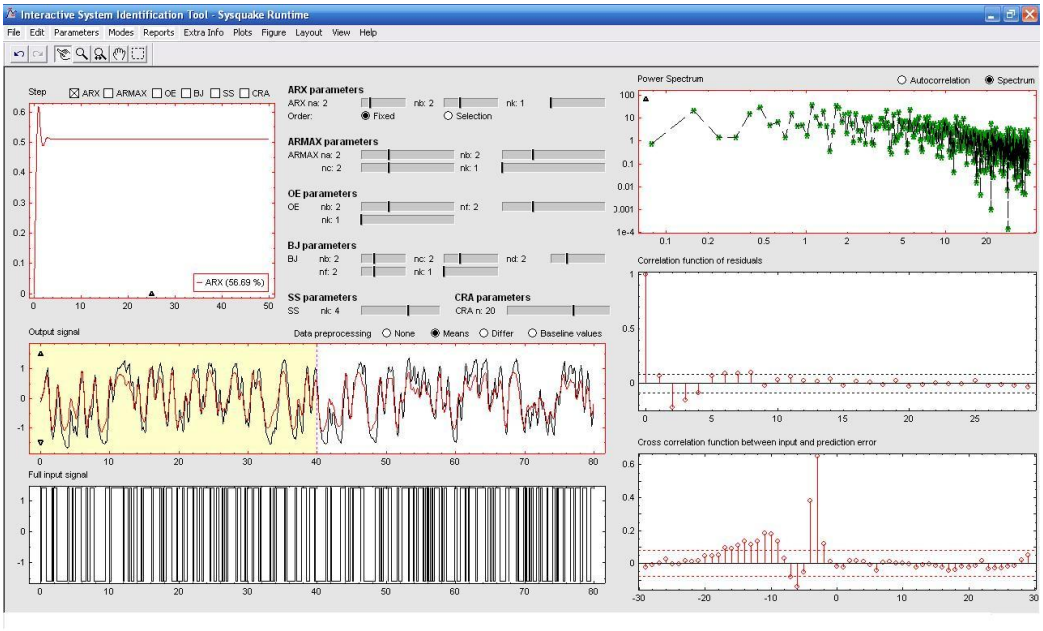


Figure 3.5: Screen layout of the ITSIE software in real mode for the hairdryer example.

3.2.3 System simulation

In this section, the system is simulated using the different models described in Chapter 1.

We will explore the following steps:

1. In the absence of any previous knowledge, it is advisable to try the various options available and use various models. But, for polynomial models, a similar advantage is realised by using the ARX model. However, models like the OE and ARMAX may also be good options for a polynomial model, due to its simplicity. So, we start to study this models.
   From the top of the `Step response` window select, for instance, a model of type ARX, ARMAX and OE.

2. After choosing the models, we should specify the orders to estimate these different models.

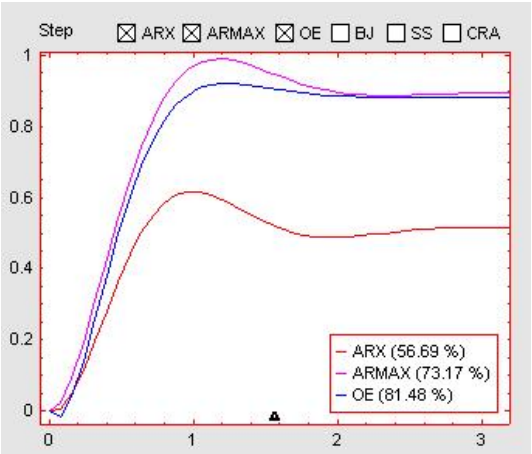


Figure 3.6: Selection of the ARX, ARMAX and OE models and fit percentage of these models for the `Step response`.

3.2.4 System identification

Now, we select the orders for different models types selected, such as [2 2 1] for ARX mode, [2 2 2 1] for the ARMAX model and [2 2 1] for the OE model, as shown in Figure 3.7.

3.2.5 Data preprocessing

In this section, we explain why to choose:

- Option `Means` in `Data preprocessing`.

  We chooce the option `Means` because it is necessary to subtract the mean values of the input (power) and the output (temperature) to remove the offsets, and so we can get better results, able to better visualise the behavior of the temperature.

- If we choose another option other than `Means`, we may notice changes in the windows `Output signal`, the `Power Spectrum`, the `Correlation function of residuals` and `Cross correlation function between input and prediction error`.

  For example, if we choose the option `Baseline values`, the change in the windows referred to in the previous point, are quite significant.

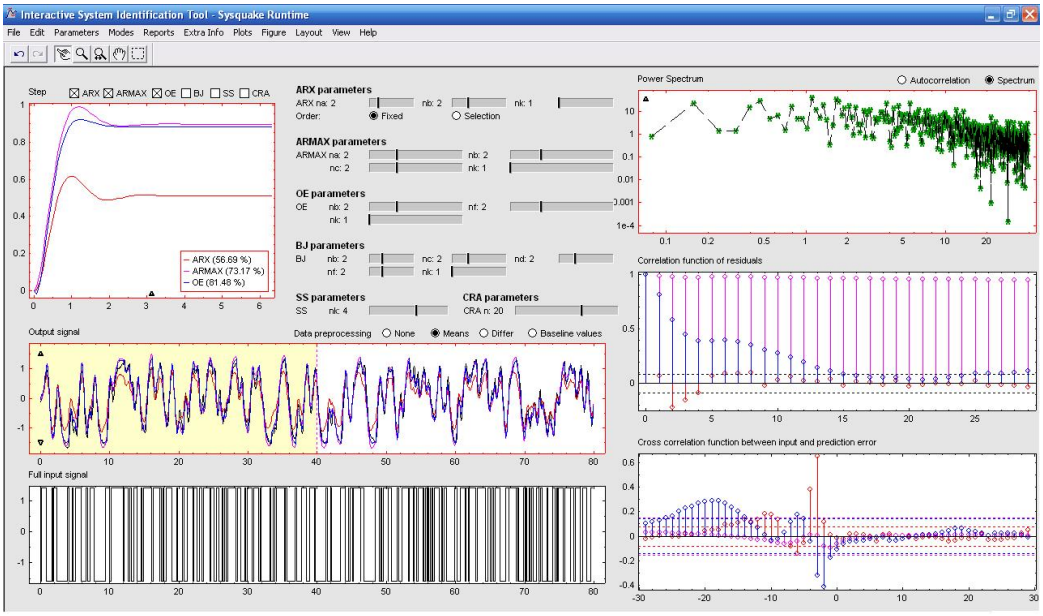


Figure 3.7: Representation of the ARX-[2 2 1], ARMAX-[2 2 2 1] and OE-[2 2 1].

To be able to view the image, we must place the cursor over the arrow on the `Output signal` window, and as you can see in the image below is the representation of the output will only be found on a larger scale.

This option remove from the real data the same values specified from the menu `Parameters` $\longrightarrow$ `Baseline values`.

- If we choose the option `None`, we see changes in the windows of tool. This option is used, when the tool takes the raw real/ simulated data, without any filtering activity, i.e, this option is not relevant for this case study, since it does nothing.
- Next, if we choose the option `Differ`, that makes changes to all windows.

  What is happening is basically $u(t) = u(t) - u(t-1)$.

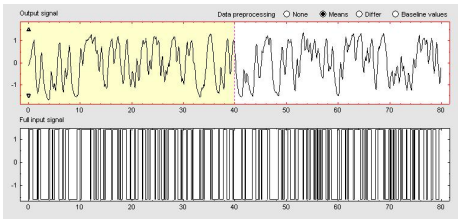


Figure 3.8: The top of this figure show the output data, temperature, and the bottom figure show the input data, power, for this example.

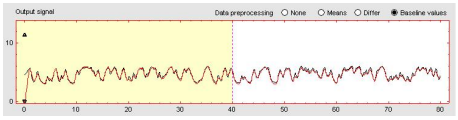


Figure 3.9: Representation of the Output signal graphic for `Baseline values` option.

3.2.6 Model validation

In order to choose the model with the best performance, we can compare among themselves.

In Figure 3.6 it appears that the OE model is the best (81.48 %), compared to the ARX (56.69 %) and ARMAX (73.17 %).

That is, we studied how efficient is the estimation model by simulating this model for a step reference and comparing the simulated output with the measured output. So, we can state that the model output window shows agreement among the different model types and the measured output in the validation data, as shown on the top left of Figure 3.7.

Bellow the `Step response` window, we can see that the output of the matches the measured ouput also for the validation data, which indicates that the models seem to capture the main system dynamics and that linear modeling is sufficient.

We can analyse the windows `Power Spectrum`, `Correlation function of residuals` and `Cross correlation between the input signal and the prediction error`, with the three chosen models, and investigate the differences, as shown in the Figure 3.7.

The spectrum has to do with the impulse response of the model. So SA also

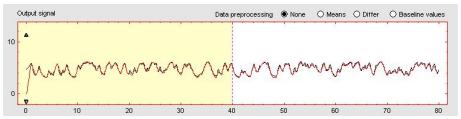


Figure 3.10: Representation of the Output signal graphic for `None` option.

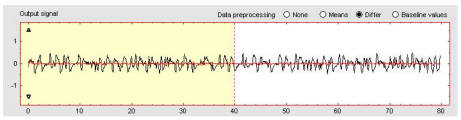


Figure 3.11: Representation of the Output signal graphic for `Differ` option.

estimates the spectrum of the additive disturbance $e(t)$ in the description of the system.

As explained in Chapter 1, in the `Cross correlation function between input and prediction error` window it is also wise to display the confidence region for this same function, the dashed dotted lines mark a 99% confidence interval. For an ideal model the correlation function should lie entirely between the confidence lines for positive lags. If the model has many positive lags, its quality is better. For example, in Figures 3.7 it appears that the model OE is the best.

A good model should have a residual autocorrelation function within this confidence interval to indicate that the residuals are uncorrelated. However, in this example, the residuals for the models appear to be correlated, which is espected since the noise model is used to make the residuals white.

A good model should have residuals uncorrelated with inputs. Evidence of correlation indicates that the model does not describe how a portion of the output relates to the corresponding input. For example, when there is a peak outside the confidence interval for lag k, this means that the contribution to the output $y(t)$ that originates from the input $u(t-k)$ is not properly described by the model.

In this example, there is no correlation between the residuals and the inputs. Thus, residual analysis indicates that this model is good. All these reasons, it can be seen in Figure 3.7, the OE model is the best and we choose this model for the case study.

But, we are still not satisfied with the overall result, because we can find a model that has more waste uncorrelated, so now let's repeat steps 1 and 2

of Section 3.2.3, and change the orders of the ARX model to investigate the differences. We select [4 3 2] for ARX model, for example.

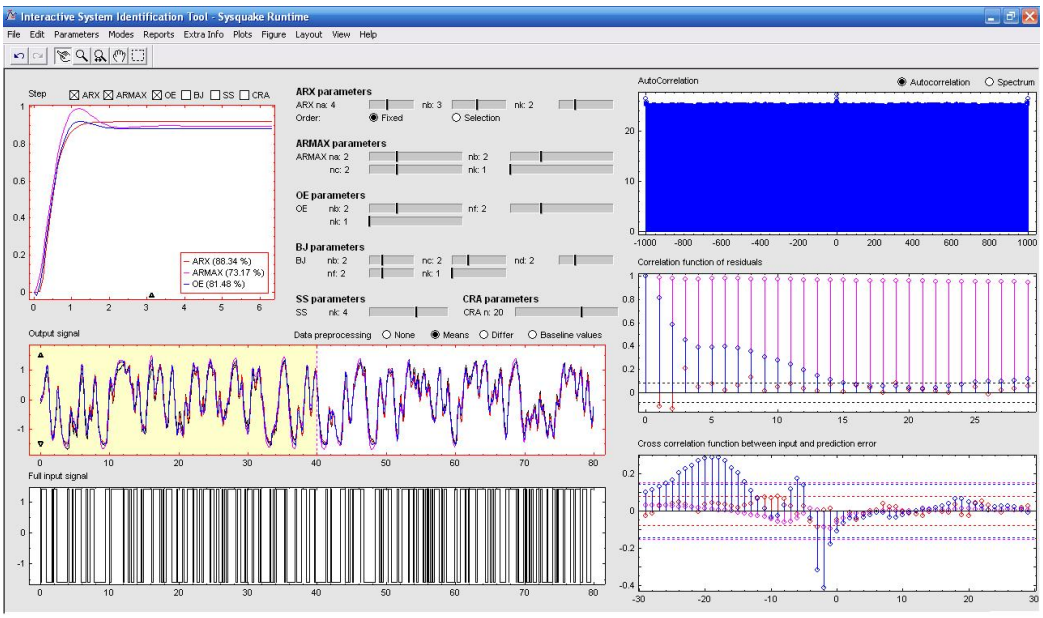


Figure 3.12: Representation of the ARX-[4 3 2], ARMAX-[2 2 2 1] and OE-[2 2 1].

In Figure 3.12, we can observe that the model that has better fit percentage is the ARX (88.34%) and also presents the residuals within the limited region and uncorrelated with past inputs, for the previous example (ARX-[2 2 1]), as these results are satisfactory, then we can consider the best ARX model, before all the trials made.

3.2.7 Report generation

We had already described the report generation, in Section 2.4.2.

We can export the model into a file with extension `.txt` and, for this example, the generated report can be seen in the following figure.

In the report of Figure 3.13 we have the sampling time, the numerators and denominators of each chosen model and also its fit percentage.

Figure 3.13: Report of a case study: hairdryer for ARX-[2 2 1], ARMAX-[2 2 2 1] and OE-[2 2 1].

Analysing first the ARX model agreement, which was learned in Chapter 1 and 2, we have $n_a = 2$, $n_b = 2$ and $n_k = 1$, so we can define $A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} = 1 - 1.698 q^{-1} + 0.772 q^{-2}$ and $B(q) = b_1 q^{-1} + b_2 q^{-2} = 0.007 q^{-1} + 0.031 q^{-2}$, therefore:

$$\begin{aligned} (1 - 1.698q^{-1} + 0.772q^{-2})y(t) = \\ (0.007q^{-1} + 0.031q^{-2})u(t-1) + e(t) \Leftrightarrow \\ \Leftrightarrow y(t) - 1.698y(t-1) + 0.772y(t-2) = \\ 0.007u(t-2) + 0.031u(t-3) + e(t) \end{aligned}$$

Then for ARMAX model we have $n_a = 2$, $n_b = 2$, $n_c = 2$ and $n_k = 1$,

so we can define $A(q) = 1 + a_1q^{-1} + a_2q^{-2} = 1 - 1.675q^{-1} + 0.734q^{-2}$, $B(q) = b_1q^{-1} + b_2q^{-2} = 0.024q^{-1} + 0.0291q^{-2}$ and $C(q) = 1 + c_1q^{-1} + c_2q^{-2}$, therefore:

$$\begin{aligned}(1 - 1.675q^{-1} + 0.734q^{-2})y(t) = \\ (0.024q^{-1} + 0.029q^{-2})u(t-1) + C(q)e(t) \Leftrightarrow \\ \Leftrightarrow y(t) - 1.675y(t-1) + 0.772y(t-2) = \\ = 0.024u(t-2) + 0.029u(t-3) + C(q)e(t)\end{aligned}$$

Finally, for OE model we have $n_b = 2$, $n_f = 2$ and $n_k = 1$, so we can define $B(q) = b_1q^{-1} + b_2q^{-2} = -0.015q^{-1} + 0.080q^{-2}$ and $F(q) = 1 + f_1q^{-1} + f_2q^{-2} = 1 - 1.5795q^{-1} + 0.652q^{-2}$, therefore:

$$\begin{aligned}(1 - 1.579q^{-1} + 0.652q^{-2})y(t) = \\ (-0.015q^{-1} + 0.080q^{-2})u(t-1) + e(t) \Leftrightarrow \\ \Leftrightarrow y(y) - 1.579y(t-1) + 0.652y(t-2) = \\ -0.015u(t-2) + 0.0809u(t-3) + C(q)e(t)\end{aligned}$$

Now, in the report of Figure 3.14, we study the report for different ARX model structure, because the other models are equal to those present in the previous report.

Analysing the ARX model, we have $n_a = 4$, $n_b = 3$ and $n_k = 2$, so we can define $A(q) = 1 + a_1q^{-1} + a_2q^{-2} + a_3q^{-3} + a_4q^{-4} = 1 - 1.098q^{-1} + 0.010q^{-2} + 0.308q^{-3} - 0.085q^{-4}$, $B(q) = b_1q^{-1} + b_2q^{-2} + b_3q^{-3} = 0.004q^{-1} + 0.066q^{-2} + 0.054q^{-3}$, therefore:

$$\begin{aligned}(1 - 1.098q^{-1} + 0.010q^{-2} + 0.308q^{-3} - 0.085q^{-4})y(t) = \\ (0.004q^{-1} + 0.066q^{-2} + 0.054q^{-3})u(t-2) + e(t) \Leftrightarrow \\ \Leftrightarrow y(t) - 1.698y(t-1) + 0.772y(t-2) + \\ 0.308y(t-3) - 0.085y(t-4) = \\ 0.004u(t-3) + 0.066u(t-4) + 0.053u(t-5) + e(t)\end{aligned}$$

Figure 3.14: Report of the case study: hairdryer for ARX-[4 3 2], ARMAX-[2 2 2 1] and OE-[2 2 1].

The fit percentage appears in both reports under the designation `Fitting` and is the same that appears in the window `Step Response`.

3.2.8 Comparison between ITSIE and `ident` through a case study

In this section, we explain the last objective of this case study.

So far we have been explaining the hairdryer in ITSIE, but this example also have been studied in other tools, such as Identification Toolbox of Matlab, ident, because this is the identification tool of Matlab.

This toolbox provides a graphical user interface (GUI). The GUI covers most of the toolbox's functions and gives easy access to all variables that are created during a session. It is started by typing `ident` in the Matlab command window and the following interface window appears.

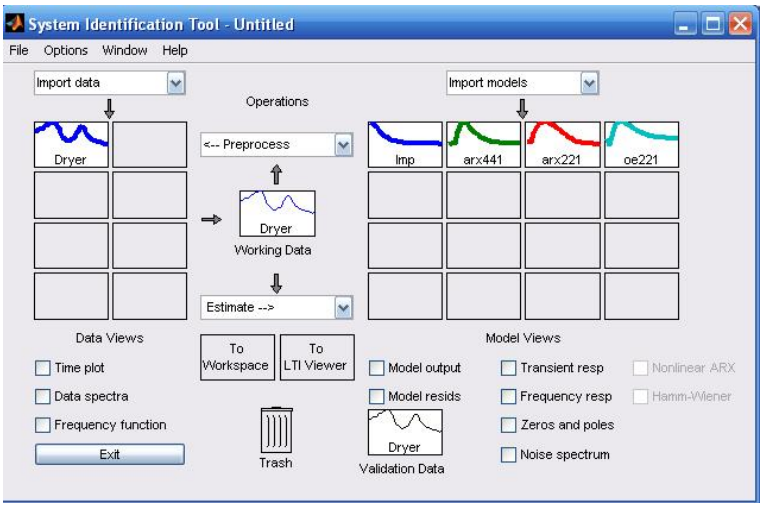


Figure 3.15: An example of importing hairdyrer in GUI.

The graphical layout of this tool is quite different from the ITSIE. This is without a shadow of doubt, one of the differences between the two tools under discussion.

In the GUI we have to specify everything we want to address in a detailed manner. For example, we must specify which channel of data input, output, and sampling time, how is the data imported, among other aspects relevant to the SI.

There are also several operation that need to be worked out in detail. For instance, the data processing, the options to choose from various types of models and to specify the orders of the same, among others.

But this tool, in my opinion, has disadvantages when viewed from a primarily educational point of view, because as explained, the GUI, do not evaluate all the stages of the SI process in an integrated fashion and provide substantial amounts of information in many different screens, wich can be quite confusing for students, unlike what happens in ITSIE.

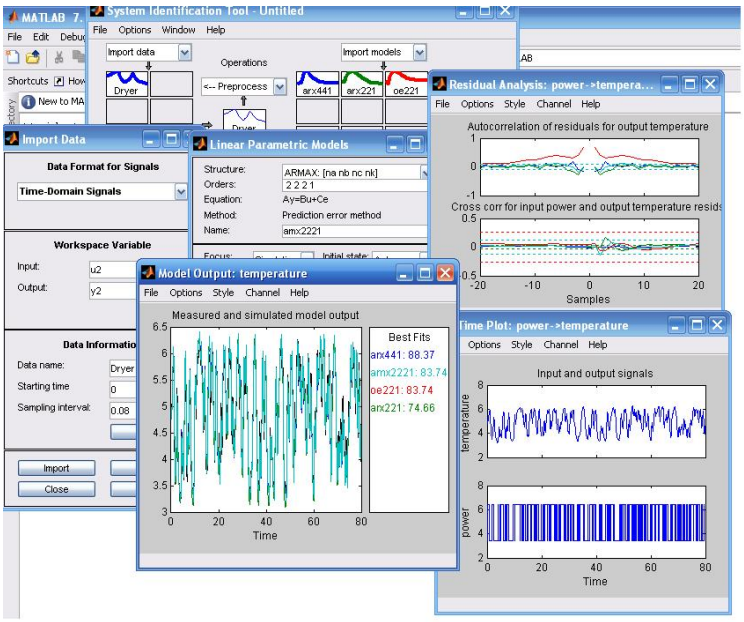


Figure 3.16: Differents screens in GUI.

3.3 Case Study—A fifth-order system

In this section, we describe the identification process, in the simulation mode, using an example.

This example was chosen because it is presented in ITSIE software and is a simple case study, where all concepts of identification can be visualised and are easily understood by the user.

3.3.1 System description

The system considered is the simulated fifth-order system that is default in the tool, represented according to the following TF which was presented in Chapter 2, particularly, in Section 2.2.2.

$$p(q) = \frac{1}{(1+q)^5} \tag{3.1}$$

3.3.2 Objectives for the case study

1. To study the choice of the input design.
2. Data preprocessing.
3. To estimate and validate linear models for a SISO system, using ITSIE.
4. To compare the application of two different models to this system under varying experimental conditions.
5. To tutor the fresh user how to find the best model for a given SISO system, using differents models types and analyse the differences.
6. To generate a report.

3.3.3 Preparing data for SI

After this section, we should be able to load an example into ITSIE and prepare it for identification.

Loading data into the simulation mode

To study the example, we should follow these steps:

1. After open the ITSIE software, we select the icon `Modes` in menu and following select `Simulation` and `Fifth-order system`;

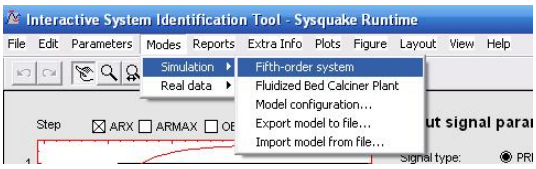


Figure 3.17: Selection of `Fifth-order system` in simulation mode.

When this example is loaded, the tool screen is changed such as shown in following Figure 3.18.

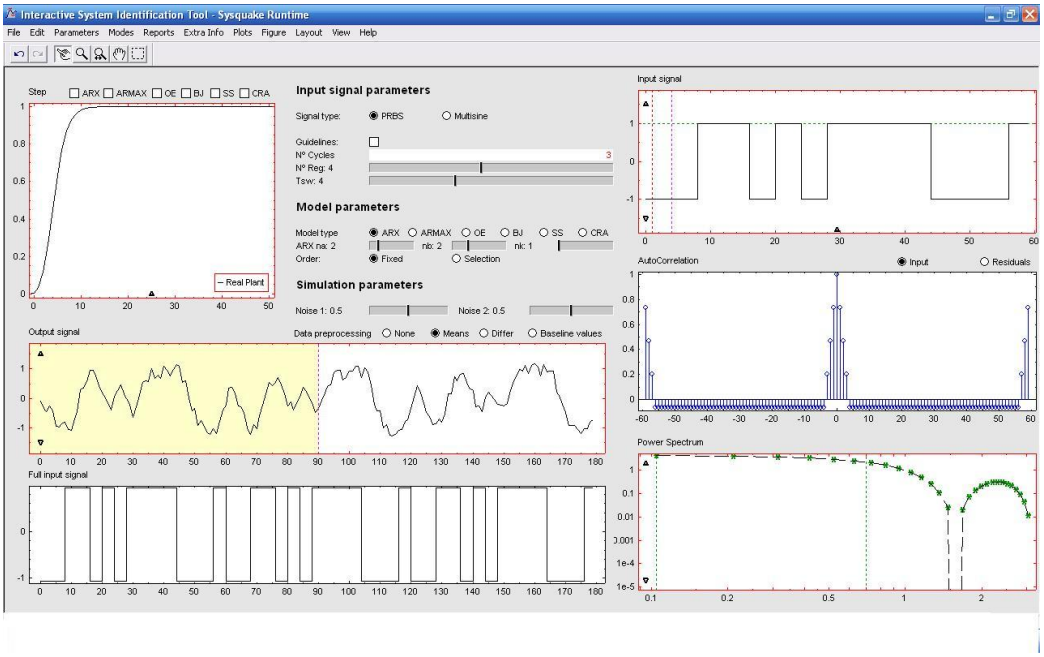


Figure 3.18: ITSIE in the simulation mode for the fifth-order system.

Next, we evaluate the data and process it for SI.

We have simulation parameters, such as:

- Input: $u(t)$;

- Output: $y(t)$;
- $T_s = 1.00$. This value is the actual sampling time in the experiment. We can modify this sampling time, to do this, we select in `Parameters` $\longrightarrow$ `Sample time`.
- Noise sources: $n_1(t)$ and $n_2(t)$. The source $n_1(t)$ allows evaluating the effects of the autocorrelated disturbances in the data, while $n_2(t)$ will introduce white noise directly to the output signal.

3.3.4 Input design

In chapter 2, we explain how to use the input signals through an example. Here, we will concretise the choice of the input signals.

Now, for example, we will study the following case and so, in the menu `Input signal parameters`, being located at the top of the middle section of the tool:

1. Select `PRBS` input;
2. Choice $N^oCycles = 2$
3. Select the use of guidelines for $\alpha_s = 2$, $\beta_s = 3$;
4. Select $\tau_{dom}^L = 3$;
5. Select $\tau_{dom}^H = 5$.

Figure 3.19: `Input signal parameters` menu with guidelines.

For each category, the step responses, autocorrelation function, cross correlation function and power spectral density are observed for three different conditions, as soon as, system forced by PRBS signal in absence of noise, noisy system forced by PRBS signal and noisy system without PRBS signal as forcing function. The autocorrelation function of the input signal and cross

correlation function between input and output signal were used to estimate the TF model of the system.

We select the PRBS signal, because, multisine input signal design which provides a shorter signal length when frequency independence is not required for model estimation, and for this reason, PRBS signals based on maximum length sequences are easy to generate, as can we see in following figure, in `Input signal` window, and the PRBS signal has a correlation function that resembles a white noise correlation function.

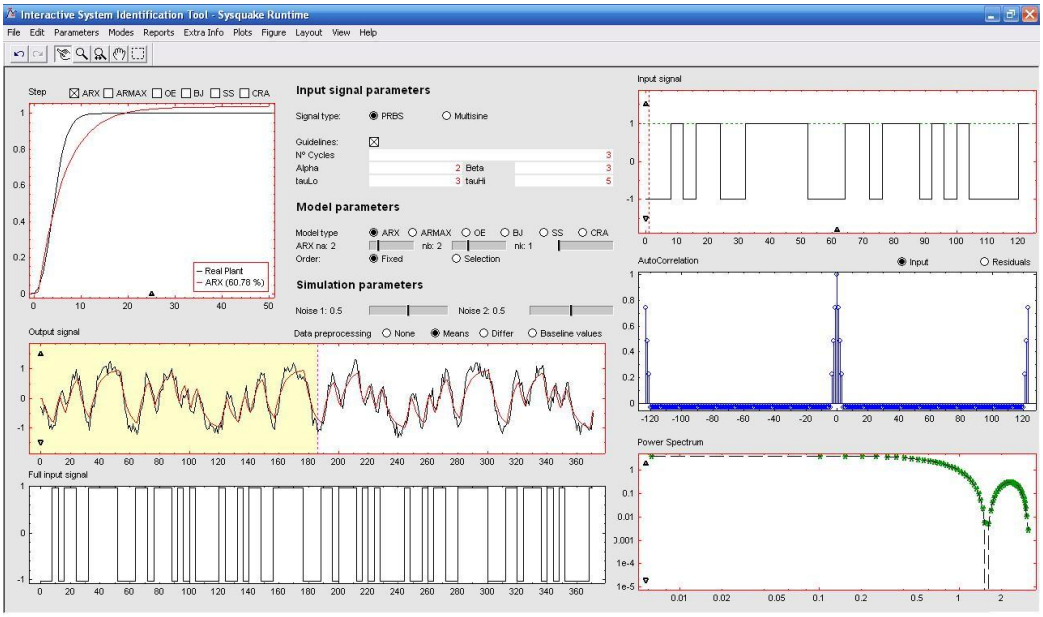


Figure 3.20: ITSIE in the simulation mode for this example of system.

Then, the three interactive windows that characterise the input design stage are modified according to the parameters of the input signal we have chosen, as we are shown in previous figure .

If we do not select the guidelines, the `Input signal parameters` of PRBS signal is different. And for this reason, we show the following example, represented in following figure.

1. We select `PRBS`;
2. Choice $N°Cycles = 3$

3. Choice $n_r = 4$;
4. Choice $T_{sw} = 4$;

Figure 3.21: ITSIE input signal parameters example of PRBS signal.

Once a input signal has been configured the final input with all the desired cycles is shown in a window called `Full input signal`. This full input signal is applied to the simulated plant with noise in order to obtain the simulated "real data" (shown in black in the `Output signal` window).

3.3.5 System simulation

Now we study, in detail, the first exeample for this fifth-order system, already shown in this chapter, in Section 3.3.4.

As we saw in the hairdryer example, a advantage is to start by using the ARX model. However, models like the OE may also be a good option for a polynomial model, due to its simplicity, for this reason, we select ARX and OE models. So, we select, for example, the ARX and OE models.

3.3.6 System identification

In this example, we select, for example, the orders [2 2 1] for ARX model and [2 2 1] for the OE model.

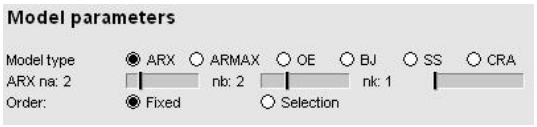


Figure 3.22: `Model parameters` menu for ARX model.

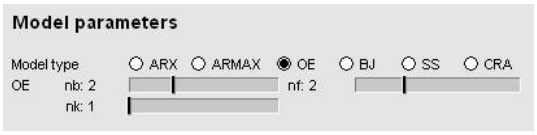


Figure 3.23: `Model parameters` menu for OE model.

3.3.7 Data preprocessing

In this part, we learn how to:

1. Subtract the mean values of the input and the output to remove offsets. So, in this example, select the option, in `Data preprocessing`, `Means`.

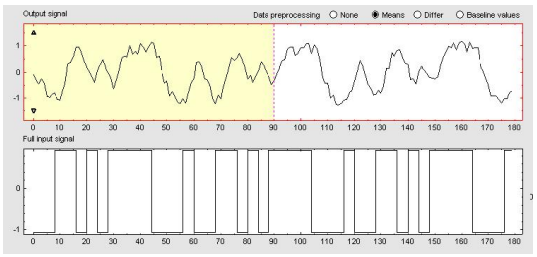


Figure 3.24: The top of figure represents the output data and the bottom figure show the input data and selection the option Means.

   The reason for selecting this option, and consequently we subtract the mean values from each signal is because, we build linear models that describe the response for deviations from a physical equilibrium. With steady-state data, it is reasonable to assume that the mean levels of the signals correspond to such an equilibrium. So, we can seek models around zero without modeling the absolute equilibrium levels in physical units.

2. If we choose another option other than `Means`, we may notice changes in the windows: `Output signal`, the `Power Spectrum`, the `Correlation function of residuals`, the `Cross correlation function between input and prediction error` and `Step response` graphic.

3. For example, if we choose the option `None` or other, and we would not get as good results, as can we see in changes in the windows referred to in the previous point, including the `Output signal` window, i.e., we see changes in the windows of tool. This option is used, when the tool

takes the raw real/simulated data, without any filtering activity, i.e, this option is not relevant for this case study, since it does nothing.

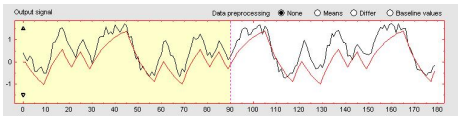


Figure 3.25: Representation of the Output signal graphic for `None` option.

4. For example, if we choose the option `Baseline values`, the change in the windows referred to in the previous point, are quite significant. This option remove from the real data the same values specfied from the menu `Parameters` $\longrightarrow$ `Baseline values`.

   To be able to view the image, we must place the cursor over the arrow on the `Output signal` window, and as we can see in the image below is the representation of the output will only be found on a larger scale.

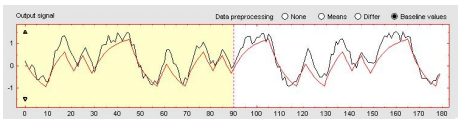


Figure 3.26: Representation of the Output signal graphic for `Baseline values` option.

5. Next, if we choose the option `Differ`, that makes changes to all windows.

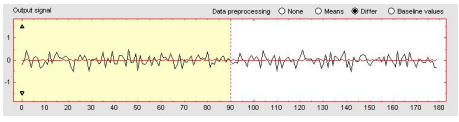


Figure 3.27: Representation of the Output signal graphic for `Differ` option.

   What is happening is basically $u(t) = u(t) - u(t-1)$.

3.3.8 Model validation

In this section we perform the following steps:

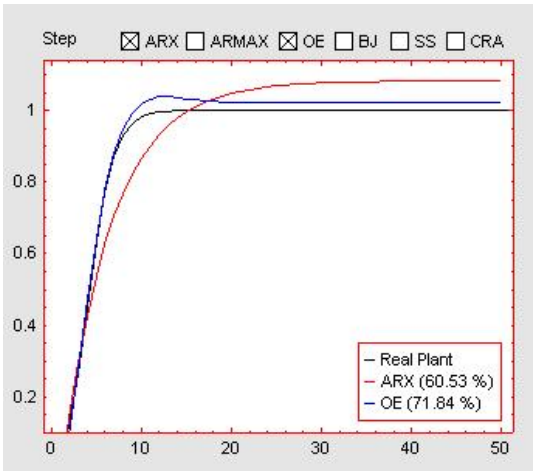


Figure 3.28: `Step response` window for this fifth-order system.

1. As has been referenced in Chapter 2, on top of the `Step response` window, we will see the representation of the models chosen.

   In this case, the ARX is what has the highest fit percentage.

2. See the `Output signal` window.

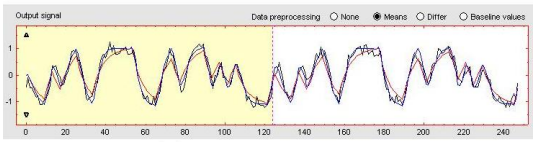


Figure 3.29: Representation of Output signal.

   We can verify in Figure 3.29, which the model output window shows agreement among the different model structures and the measured output in validation data.

3. If we see the `Step response` window in Figure 3.28, notes that the OE model will be the best (71.84 %) than the ARX (60.53%), for the model structures chosen.
   In this example, the step response of the models indicates that this models seem to capture the main system dynamics and that linear modeling is sufficient.

4. In the `Autocorrelation` window, we can now select `Residuals` instead of being input in order to see `Input`, as we can see in Figure 3.30.

As was explained in Chapter 1 and we can see in these Figure 3.30, in `Correlation function of residuals` window, shows that adding a noise model produces uncorrelated residuals: the set of axes or dotted lines show that the autocorrelation values should be present inside the confidence bounds. This indicates a more accurate model. As can we see in Figure 3.30, the chosen models, show any fluctuations within this confidence interval, which are considered to be insignificant.

Validation criteria that indicate the inadequacy of these models (in the absence of knowledge of the true system as provided in the step response) include the poor fits to both estimation and validation data and the wide discrepancy in model step responses.

Curiously, for this data set correlation function of residuals for both model estimates falls within the standard error bounds, incorrectly implying model adequacy. Because of the short duration of the tests, the standard error bounds (determined by $\pm\frac{1}{\sqrt{N}}$, where N is the length of the data set) are high, indicating to users that CA may be unreliable for short data sets under these experimental conditions.

As we can see, in the Figures 3.28 and 3.30, the model OE is the best, and so, we must choose this model.

Now, we will analyse the other situation, to see the differents behaviors and if we can get better results for this example, using, for example, a different order for ARX model.

To accomplish this, it is important to know whether a higher order model can be estimated and thus the variance of parameters estimated by the ARX model shoud be more pronounced than in OE.

So, we can select a different orders for ARX model, for example, [3 8 1] and [2 2 1] for OE model, as can we see in Figure 3.31.

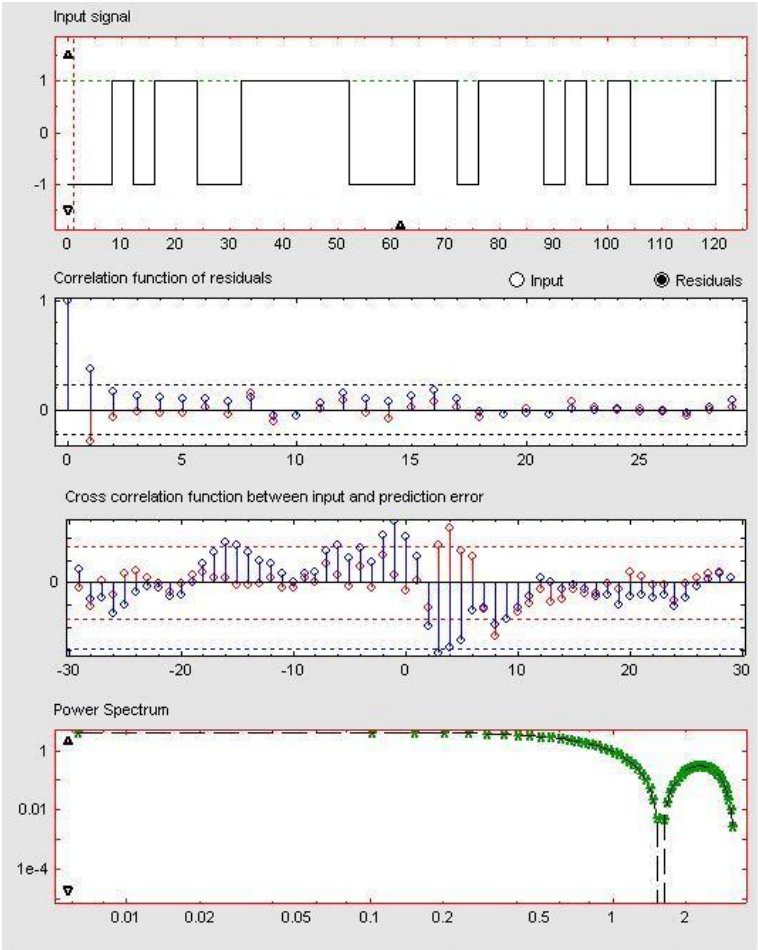


Figure 3.30: Model validation.

In this situation, OE model is also compared to the orders [3 8 1] for ARX model, obtained for systematic order selection over a range of model has a higher fit over the crossvalidation dataset. I.e., we will see in `Step response` that the AR model is the best (76.06%) than the OE (74.44%), as can we see in previous figure.

So, ARX model represents a good precursor model and can be further refined through model reduction or other means.

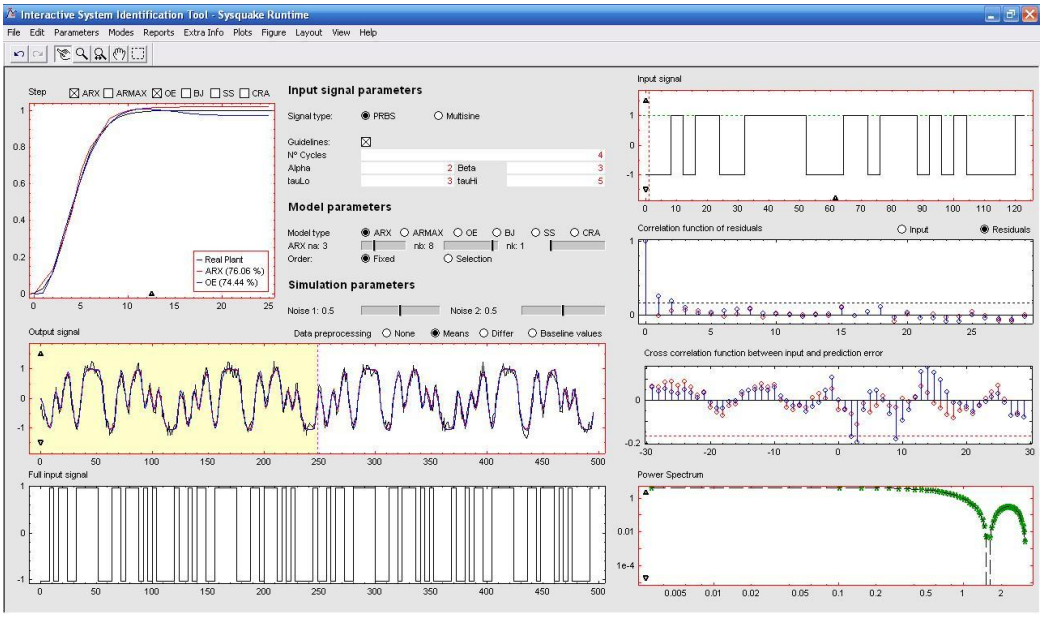


Figure 3.31: ITSIE tool user interface demonstrating four cycles of a PRBS input applied to this system, where ARX model is compared with an OE.

3.3.9 Report generating

Once the process model is defined, we can export the model into a file with extension `.txt`. For this case study the reports generated are as follows:

In this report of Figure 3.32 we have the sampling time, the numerators and denominators of each model is chosen and also the fit percentage of each.

Analysing first the ARX model agreement, which was learned in Chapter 1 and 2, we have $n_a = 2$, $n_b = 2$ and $n_k = 1$, so we can define $A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} = 1 - 0.955q^{-1} + 0.102q^{-2}$ and $B(q) = b_1 q^{-1} + b_2 q^{-2} = -0.0024q^{-1} + 0.184q^{-2}$, therefore:

$$\begin{aligned}(1 - 0.955q^{-1} + 0.102q^{-2})y(t) = \\ (-0.024q^{-1} + 0.184q^{-2})u(t-1) + e(t) \Leftrightarrow \\ \Leftrightarrow y(t) - 0.955y(t-1) + 0.102y(t-2) = \\ -0.024u(t-2) + 0.184u(t-3) + e(t)\end{aligned}$$

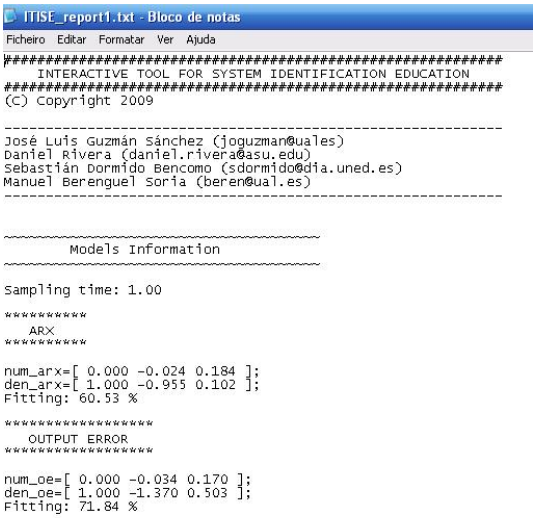


Figure 3.32: Report of a fifth-order system for ARX-[2 2 1] and OE-[2 2 1].

Finally, for OE model we have $n_b = 2$, $n_f = 2$ and $n_k = 1$, so we can define $B(q) = b_1 q^{-1} + b_2 q^{-2} = -0.034q^{-1} + 0.170q^{-2}$ and $F(q) = 1 + f_1 q^{-1} + f_2 q^{-2} = 1 - 1.370q^{-1} + 0.503q^{-2}$, therefore:

$$\begin{aligned}(1 - 1.370q^{-1} + 0.503q^{-2})y(t) = \\ (-0.034q^{-1} + 0.170q^{-2})u(t-1) + e(t) \Leftrightarrow \\ \Leftrightarrow y(t) - 0.034y(t-1) + 0.170y(t-2) = \\ -1.370u(t-2) + 0.503u(t-3) + C(q)e(t)\end{aligned}$$

For the second example:

Now in report of Figure 3.33, we study the report for different ARX model structures, because the other models structures are equal to those presented in the previous report.

Analysing the ARX model, we have $n_a = 3$, $n_b = 8$ and $n_k = 1$, so we can define $A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + a_3 q^{-3} + a_4 q^{-4} = 1 - 0.135q^{-1} - 0.111q^{-2} - 0.058q^{-3}$ and $B(q) = b_1 q^{-1} + b_2 q^{-2} + b_3 q^{-3} + b_4 q^{-4} + b_5 q^{-5} + b_6 q^{-6} + b_7 q^{-7} + b_8 q^{-8} = 0.063q^{-1} + 0.061q^{-2} + 0.124q^{-3} + 0.134q^{-4} + 0.196q^{-5} + 0.063q^{-6} + 0.019q^{-7} + 0.052q^{-8}$, therefore:

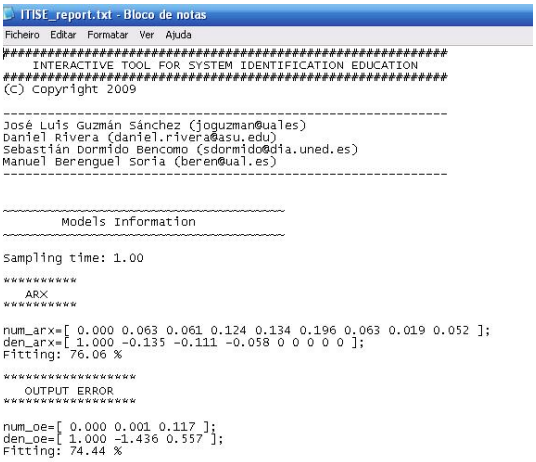


Figure 3.33: Report of a fifth-order system for ARX-[3 8 1] and OE-[2 2 1].

$$
\begin{aligned}
(1 - 0.135q^{-1} - 0.111q^{-2} - 0.058q^{-3})y(t) = \\
(0.063q^{-1} + 0.061q^{-2} + 0.124q^{-3} + 0.134q^{-4} + \\
0.196q^{-5} + 0.063q^{-6} + 0.019q^{-7} + 0.052q^{-8})u(t-1) + e(t) \Leftrightarrow \\
\Leftrightarrow y(t) - 0.135y(t-1) - 0.111y(t-2) - 0.058y(t-3) = \\
0.063u(t-2) + 0.061u(t-3) + 0.124u(t-4) + \\
0.134u(t-5) + 0.196u(t-6) + 0.063u(t-7) + \\
0.019u(t-8) + 0.052u(t-9) + e(t)
\end{aligned}
$$

The fit percentage, which appears in both reports, `Fitting` is the same that appears in the window `Step Response`.

Through theses reports, we can better conclude whether we are happy with the results.

3.4 Concluding remarks

In this chapter we analysed two case studies, referring to real and simulated data.

These two examples were studied and explained in this manual, step by step, and in a systematic way, since all the concepts underlying the concepts of SI.

After these tutorials, one person who initially knew nothing of identification and want to learn using the graphical tool ITSIE, or anyone wishing to depen their knowledge, should be able to make the study of identification of a system without no problems, and also test these examples to see if everyone, working or studying with identification, understood the theoretical concepts related to SI, presented in the previous chapters are illustrated here.

Appendix A

Random processes - basic concepts

In this appendix, we will understand what a random process is and discuss in greater detail some concepts in order to understand how this is relevant to the identification problems.

First we will review the properties of a single random process such the concepts of mean, standard deviation, autocorrelation and spectral density, and also stationarity and ergodicity. Next, for two or more random processes we will explain the notions of correlation, covariance, cross spectral density. Finally, we will study the concept of white noise.

A.1 Deterministic and random processes

A random process $\{x_k(t)\}_{k\in N}, -\infty < t < \infty$, also called a time series or SP, is an ensemble of real-valued (or complexed-valued) functions that can be characterised through its probability structure. For convenience, the variable t will be interpreted as time.

Each particular function $x_k(t)$, where t is variable and k is fixed, is called a sample function. In practice, a sample function may be thought of as the observed result of a single experiment. For any number N and any fixed times $t_1, t_2, \cdots, t_N$, the quantities $x_k(t_1), x_k(t_2), \cdots, x_k(t_N)$, represent N random

variables over index k. It is required that a well-defined N-dimensional probability distribution function exists for every N.

A particular sample function $x_k(t)$, in general, would not be suitable for representing the entire random process $\{x_k(t)\}$ to which it belongs.

A determinist process is a physical process, which is represented by an explicit mathematical relation.

A determinist process is a physical process, which is represented by an explicit mathematical relation. For example, if in determinist processes, we have a physical process, this is represented by an explicit mathematical relation. A case study is, for example, the response of a single mass-spring-damper in free vibration in laboratory. In turn, a random processes results from a large number of separate causes and is, therefore, described in probabilistic terms and by properties which are averages, as we will see in what follows.

In the Figure A.1 we have the representation of a random process over time, t.

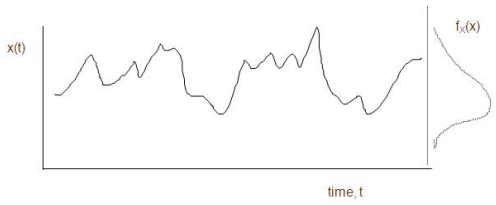


Figure A.1: Representation of random process.

The probability density function describes the general distribution of the magnitude of the random process, but it gives no information on the time or frequency content of the process.

A.2 Basic characterisation of a random process

In this section, we will introduce the basic properties for a single random process, such as mean, standard deviation, autocorrelation and spectral density.

The mean value, $\bar{x}$, is the height of the rectangular area having the same

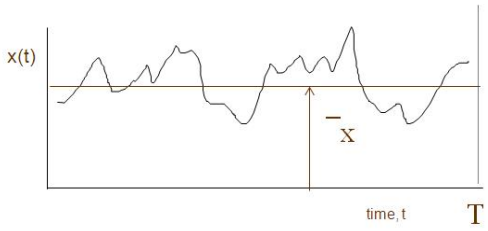


Figure A.2: Mean value.

area as that under the function $x(t)$, as shown in the Figure (A.2).

$$\bar{x} = \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)\mathrm{d}t. \tag{A.1}$$

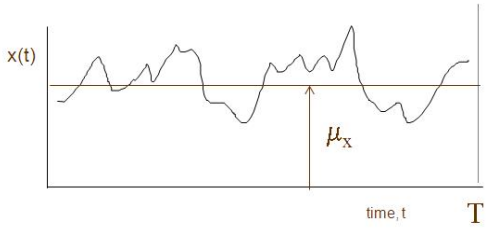


Figure A.3: Mean value.

The mean square value, can be given by

$$\bar{x^2} = \lim_{T\to\infty} \frac{1}{T} \int_0^T x^2(t)\mathrm{d}t. \tag{A.2}$$

The variance is given by

$$\sigma_x^2 = \overline{[x(t) - \bar{x}]^2} = \lim_{T\to\infty} \frac{1}{T} \int_0^T [x(t) - \bar{x}]^2\mathrm{d}t. \tag{A.3}$$

The standard deviation, σ_x, is the square root of the variance.

The autocorrelation or autocovariance, describes the general dependency of $x(t)$ on its value at later instants, $x(t+\tau)$, where τ is the displacement in time and is represented by the following expression:

$$\rho_x(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_0^T [x(t) - \bar{x}]^2[x(t+\tau) - \bar{x}]^2\mathrm{d}t. \tag{A.4}$$

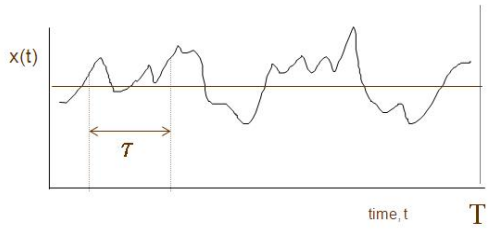


Figure A.4: Autocorrelation.

The value of $\rho_x(\tau)$ at τ equal to 0 is the variance σ_x^2.

The normalised autocorrelation is given by

$$R(\tau) = \frac{\rho_x(\tau)}{\sigma_x^2}. \tag{A.5}$$

For $R(0)$ this expression (A.5) equals 1.

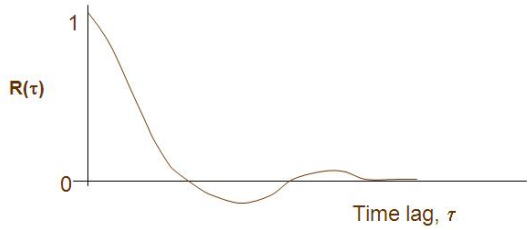


Figure A.5: Auto-Correlation.

The autocorrelation for a random process eventually decays to zero at large τ.

The autocorrelation for a sinusoidal process (deterministic) is a cosine function which does not decay to zero. Example: With $\tau = 2\pi$, the correlation is 1.

The spectral density (auto-spectral density, power spectral density, spectrum) describes the average frequency content of a random process $x(t)$.

The first basic relationship is

$$\sigma_x^2 = \int_0^\infty S_x(n)\mathrm{d}n, \tag{A.6}$$

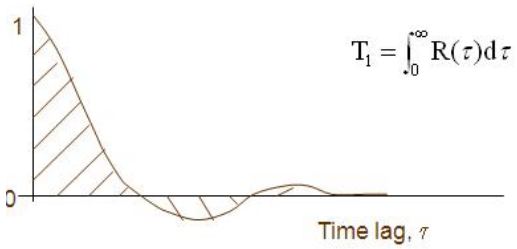


Figure A.6: Autocorrelation.

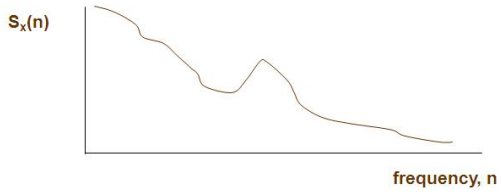


Figure A.7: Spectral density.

where $S_x(n)$ is the spectral density. The quantity $S_x(n) \times \sigma_n$ represents the contribution to σ_x^2 from the frequency increment σ_n. The units of $S_x(n)$ are [units of x]^2 $\times sec$.

The second basic relationship is

$$S_x(n) = \lim_{T\to\infty} \left(\frac{2}{T} \mid X_T(n) \mid^2 \right), \tag{A.7}$$

where $X_T(n)$ is the Fourier transform of the process $x(t)$ taken over the time interval $-T/2 < t < T/2$. The above relationship is the basis for the usual method of obtaining the spectral density of experimental data, usually a fast Fourier transform (FFT) algorithm is used.

The third basic relationship is

$$S_x(n) = 2 \int_{-\infty}^{\infty} \rho_x(\tau) e^{(-i2\pi n\tau)} \mathrm{d}\tau. \tag{A.8}$$

The spectral density is twice the Fourier transform of the autocorrelation function. Otherwise, the inverse relationship is

$$\rho_x(\tau) = Real\Big\{ \int_0^{\infty} S_x(n) e^{(-i2\pi n\tau)} \mathrm{d}n \Big\} = \int_0^{\infty} S_x(n) \cos(2\pi n\tau). \tag{A.9}$$

Thus the spectral density and autocorrelation are closely linked. They basically provide the same information about the process $x(t)$.

A.3 Averaging, stationarity and ergodicity

To study the average of a process, we must have a record set of samples with values at corresponding times, so that through them we can withdraw the necessary conclusions.

If all marginal and joint density function of the process do not depend on the choice of the time origin, the process is said to be stationary.

A stationary process is a process in which averages from a single record are the same as those obtained from averaging over the ensemble. If almost every number of the ensemble shows the same statistical behavior as the whole ensemble, then it is possible to determine the statistical behavior by examining only one typical sample function. These process is defined as an ergodic process.

A most stationary random processes can be treated as ergodic.

A.4 Statistical relationships between two or more random processes

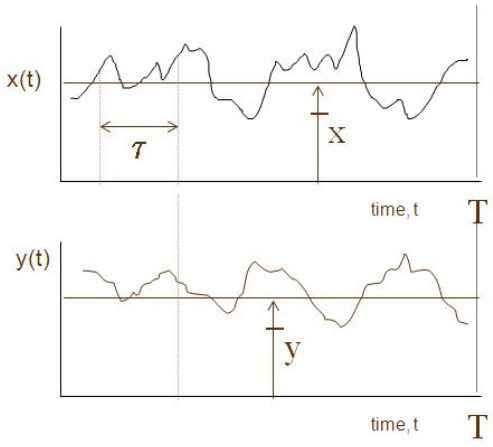


Figure A.8: Cross-correlation.

The cross-correlation function describes the general dependency of $x(t)$ on another random process $y(t+\tau)$, also delayed by a time delay, τ, given by the expression

$$c_{xy}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_0^T [x(t) - \bar{x}][y(t+\tau) - \bar{y}]\mathrm{d}t. \tag{A.10}$$

The covariance is the cross-correlation function with the time delay, τ, set to zero.

$$c_{xy}(0) = \lim_{T\to\infty} \frac{1}{T} \int_0^T [x(t) - \bar{x}][y(t) - \bar{y}]\mathrm{d}t. \tag{A.11}$$

The correlation coefficient, ρ, is the covariance normalized by the standard deviations of x and y.

$$\rho = \frac{c_{xy}(0)}{\sigma_x \times \sigma_y}. \tag{A.12}$$

When x and y are identical to each other, the value of ρ is 1 (full correlation). When $y(t) = -x(t)$, the value of ρ is -1. In general, the variation of ρ is $-1 < \rho < 1$.

By analogy with the spectral density, we define

$$S_{xy}(n) = 2 \int_{-\infty}^{\infty} c_{xy}(\tau) e^{(-j2\pi n\tau)} \mathrm{d}\tau. \tag{A.13}$$

The cross-spectral density is twice the Fourier transform of the cross-correlation function for the processes $x(t)$ and $y(t)$.

The cross-spectral density (cross-spectrum) is a complex number, $S_{xy}(n) = C_{xy}(n) + jQ_{xy}$

$C_{xy}(n)$ is the co(-incident) spectral density -(in phase). And $Q_{xy}(n)$ is the quad(-rature) spectral density -(out of phase).

Define normalised co-spectral density as

$$\rho_{xy}(n) = \frac{C_{xy}(n)}{\sqrt{S_x(n) * S_y(n)}}, \tag{A.14}$$

that is effectively a correlation coefficient for fluctuations at frequency n.

Theorem 1 (The spectral density factorisation theorem) *When*

$$H(z) = \sum_{k=0}^{\infty} h(k) z^{-k} \tag{A.15}$$

is a stable TF with $h(0) = 1$ *with all zeros inside the unit circle. Then the spectral density function* $\phi_x(w)$ *can be factorised as*

$$\phi_y(w) = H(e^{(jw)})H(e^{(-jw)})\sigma^2. \tag{A.16}$$

This theorem allows any SP model as the output of a system excited by white noise e.

It is noted that the term spectral density is also used to designate the Z-transform of the autocorrelation sequence, ie,

$$\phi_x(z) = \sum_{\tau=0}^{\infty} r_x(\tau) z^{-\tau}$$

A.4.1 White noise process

White noise is a random signal (or process) with a flat power spectral density. White noise draws its name from white light in which the power spectral density of the light is distributed over the visible band in such a way that the eye's three color receptors (cones) are approximately equally stimulated.

In practice, it appears that the system response is not completely coincident with the models. The deviations may be due to errors modeling, inaccuracies in the sensors and converters, variations in and charge interactions with the environment. In linear models, these phenomena can be represented as a disturbing signal output system.

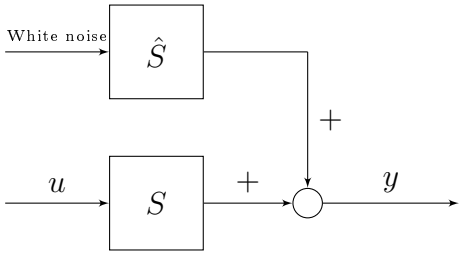
$$y(t) = \bar{y}(t) + \eta(t) \tag{A.17}$$

where $\eta(t)$ is the disturbance and $\bar{y}(t)$ is the output without disturbance.

In stochastic control theory, it is often considered that the disturbances are SP with zero mean and covariance stationary. The spectral density factorisation theorem, allows them to be modeled as output signals of linear minimum phase excited by white noise. To describe these systems, we can use models for IO and state models.

A usual strategy is to consider a simple white noise or other processes and then get through filters, i.e., from the white noise through a linear system.

The coefficients of the linear system specify the covariance of the noise, according to the figure, where $\hat{S}$ is spectral density of noise, as you can see in following diagram.



Usually, the mean of white noise is assumed to be 0 and the standard deviation 1. White noise is used appropriately to generate other processes with different SP.

Figure A.9 represents the white noise.

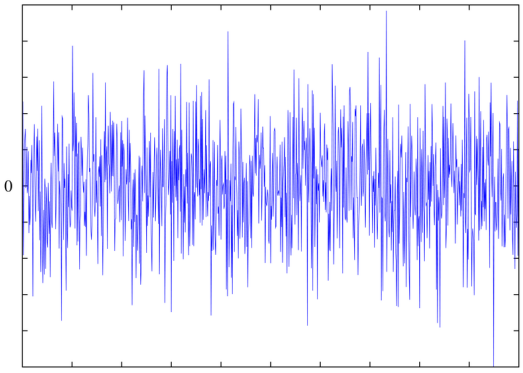


Figure A.9: Representation of white noise.

Bibliography

[1] B. De Moor, P. De Gersem, B. De Schutter, and W Favoreel, DAISY: A database for identification systems in *Journal A*, Vol. 38, No. 3, pg. 4-5, Sep. 1997.

[2] L. Ljung, *System identification: Theory for the User, 2nd edition.* Englewood Cliffs, NJ: Prentice-Hall, 1999.

[3] L. Ljung, MATLAB *System identification toolbox users guide, Version 6.* Natick, MA: The Mathworks, 2004.

[4] L. Ljung, and T. Glad, Modeling of Dynamic Systems, PTR Prentice Hall, Upper Saddle River, NJ, 1994.

[5] J.L. Guzmán, D. E. Rivera, S. Dormido M. Berenguel, ITSIE: an Interactive Software Tool for System Identification Education em *Proc. 15th IFAC Symposium on System identification (SYSID 2009)*, July 2009, St. Malo, France, http://aer.ual.es/ITSIE/

[6] J.L. Guzmán, D. E. Rivera, S. Dormido M. Berenguel, Teaching System Identification through interactivity em *Proc. 8th IFAC Symposium on Advances in Control education*, October 2009, Kumamoto, Japan.

[7] L. Ljung, Educational aspects of identification software user interfaces, In S. Weiland P. van der Hof, B. Wahlberg, editor, Proc. 13th IFAC Symp. on System Identification, pages 1590-1594, Rotterdam, The Netherlands, August 2003a.

[8] T. Soderström, Discrete-time Stochastic Systems, London, Springer, 2002.

[9] T. Katayama, *Subspace Methods for System Identification.* London, Springer, 2005.

[10] H. kwakernaak, G. Meinnsma, Time Series Analysis and System Identification, University of Twente, Departament of Applied Mathemathics, 1999.

[11] D. Rivera, H. Lee, M. Braun, H.Mittelmann, Plant-friendly system identification: a challenge for the process industries, in: 13th IFAC Symposium on System Identification (SYSID 2003), Rotterdam, Netherlands, pp.917-922.

[12] Harrison H. Barrett, Theory of random processes, Part I: Basic concepts, University of Arizona, March 2011, http://optics.nuigalway.ie/people/barrett/Lecture_13.pdf.

[13] A. Paulo G.M. Moreira, Paulo J. G. Costa, Paulo J. Lopes dos Santos, Introdução à identificação de modelos discretos para sistemas dinâmicos, March 2011, http://paginas.fe.up.pt/~amoreira/documentospdf/identif_jan2003.pdf.

[14] Jack Herrick, Ben Rubenstein, Imperatrix, Brett, Sondra C, Dvortygirl, Teresa, Peter, Charles Carter and Max, How to Write a Manual, June 2011, http://www.wikihow.com/Write-a-Manual.

[15] Wikipedia, the free encyclopedia, Crest factor, April 2011, http://en.wikipedia.org/wiki/Crest_factor.

[16] Petervaldivia, Technology Department, Electricity: Moving charges, May 2011, http://www.petervaldivia.com/technology/electricity/electrical-power.php.