2

Virtual Instruments and Soft Sensors

2.1 Virtual Instruments

In this chapter, the concepts of virtual instruments (VIs) and of soft sensors will be introduced in some detail. In particular, we will commence in this section with an introduction to VIs, to focus in the next one on soft sensors, that are the main topic of the book.

VIs can be considered as a wider class than soft sensors. In fact, soft sensors focus on the process of estimation of any system variable or product quality by using mathematical models, substituting some physical sensors and using data acquired from some other available ones.

For their part, VIs are based on software that performs any of the typical actions involved in a measurement and/or control problem, by exploiting available instrumentation, computers and software. This action can either involve, or not, modeling capabilities typical of soft sensors.

VIs are the result of the rapid diffusion that has taken place in the last 20 years of low-cost Windows-based personal computers, Macs, and workstations in any engineering application field, along with performing software (Foster ,1998).

They represent an alternative paradigm to traditional instruments and allow us both to customize the measuring facility capabilities to user application and to take control of the way measurement results are used and presented.

The flexibility of VIs is obtained by software that is used to transform a collection of facilities into the customized instrumentation suitable for the measurement task of interest. Also, the use of software allows the adaptation, at different times, of the available resources to new measuring problems in such a way as to adapt the measuring system to new scenarios, and hence to better exploit available resources.

Three main functional blocks can be recognized in any instrument, and all of them have been affected by changes introduced by the introduction of VIs (Combs, 1999):

- measurement;
- computation;
- user interface.

As regards measurement, this is the very first action that a measuring system performs on the observed variable, in order to extract some kind of meaningful signal. Signals extracted from real plants are analog in nature, while the majority of modern instrumentation is digital. An analog to digital (A/D) converter is, therefore, required at this stage along with conditioning circuitry, to adapt real-world signals to the A/D input span.

At this stage, VIs are used mainly to the ease the setting and control of measuring systems, especially in routine measurement surveys that require a large number of actions, and are usefully automatized using adequate software. This is the typical application when a VI is designed, and used, to drive a stand-alone system.

Data acquired from the measurement hardware do not necessarily correspond to the searched information. It is generally required either to filter data in some way, to combine data acquired from the same device in different times or different points, or to combine data acquired from different measuring facilities. At this level, computation capability plays a key role and it is possible either to have instruments with local computation capabilities or to send raw data to some kind of intelligent system that performs the required data manipulation. Of course, these two solutions impose different constraints on the designer and have conflicting performances. The first one is generally more expensive, because of the need to have a number of local intelligent systems, while the other will need a faster communication system to handle the large quantity of raw data that needs to be transferred.

The computation level is of course one of the most characterizing aspects of VIs. In fact, by using adequate software it is possible to use general measuring devices, *e.g.* acquisition cards or modular instrumentation, to acquire data and combine them in a virtually infinite number of ways. Moreover, the same devices can be re-used in different applications simply by changing the algorithms used.

Though modern electronic instrumentation is totally different from older analog measuring systems, producers try to maintain the traditional look of the instruments. This is because users experience great difficulty when the objects they are used to change their appearance. As an example, a modern digital oscilloscope has an internal structure that is totally different from traditional analog ones. Nevertheless, both of them look very similar and present a number of input channels, some knobs and buttons to control the instrument operations, and a screen. This allows the user to change from older systems to newer ones with a minimum of difficulty: only if he is interested, will he search at some convenient time for new functionalities.

This general rule has been maintained in the case of VIs. The measuring system might be realized by using traditional stand-alone instrumentation, with its own

user interface, or by using modular instrumentation; in that case the user interface is almost totally missing, and so the VI designer will produce, on a host PC monitor, a user interface that mimics the traditional instrumentation front panel and that will allow interaction with the measuring system. In this way, the user will find digital knobs or buttons to control the measurement system and will observe the information in which he is interested on some kind of graph, indicator, or in the case of surveillance systems, for example, a simulated LED will be turned on by the software together with some kind of acoustic signaling.

An example of a front panel of a VI is shown in Figure 2.1.

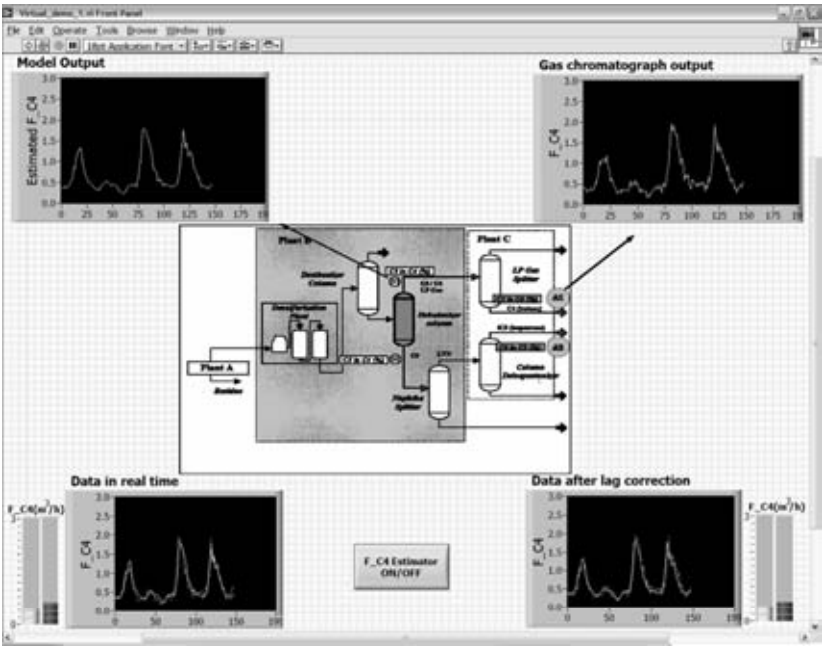

Figure 2.1. An example of a VI front panel

Figure 2.1 shows the front panel of an instrument designed for the estimation of the products of a debutanizer column that will be described in greater detail elsewhere in the book. In this case, the core of the VI was a soft sensor based on a cascade of neural network dynamic models whose objective was the prediction of products concentration without the large delay introduced by traditional measuring systems. In particular, in Figure 2.1 the presence of a number of time plots, a button, and some indicators can be recognized. The button was introduced to allow the user to turn the VI on and off, while both the time graphs and the indicators show the instrument outputs.

The availability of low-cost computers with programming capabilities has produced a far-reaching and rapid evolution also in the possible measuring hardware configurations. In fact, on the one hand, modern instrumentation is configured as a digital system whose core is a microprocessor that controls all actions required to perform measurements, while on the other one, it is more and

more common that the instruments communication capabilities allow for the realization of distributed measuring networks where a number of devices cooperate and exchange relevant information, either using a shared standard or custom communication protocol.

The simplest and most widely used class of measuring instruments are stand-alone devices. In this case, a single box contains all the resources required to perform measurements and is equipped with a front panel that is used both to set up the instrumentation and to display measurement results. VIs in this case are used to realize virtual front panels on a PC that mimics the hardware front panel. These instruments can operate by themselves and eventually, if they have digital communication capabilities, can be used in a multiple instrument system. For example, this is the typical configuration used in IEEE 488.2 compliant measurement systems.

With the evolution of computers, modular measuring instrumentation became available. In this case, the instrument front panel is totally missing and the only available option to use modular systems is to insert them into a frame and to program them by software. VIs are widely used in this context to program and use measuring stations.

Generally, modular measuring systems are realized by using some standard, such as VME, VXI or PXI.

They greatly outperform traditional stand-alone systems especially when the required system throughput is high, but they are quite expensive. Also, since they are mainly software defined, the available resources can be reconfigured virtually an infinite number of times and this is a valuable possibility both for R&D purposes, when the measured variables change frequently, and for maintenance and control applications, when measuring systems are often updated.

The latest evolution of modern electronic instrumentation is based on networked devices. In this case, each instrument acts as a computer, capable of being connected on some LAN and of sharing a common communications line or wireless link within a small geographic area, or even by using the Internet. These last scenarios are typical of monitoring and control in industrial applications due to the distributed nature of processes involved and of air quality monitoring in large urban and industrial areas, where a number of measuring stations are installed at adequately chosen points to obtain information about air quality.

Of course, also in the latter case, VIs can be very useful especially if software tools with networking programming capabilities are used, which can greatly simplify both the troubleshooting and running of the measuring system.

As mentioned before, the success of Vis is mainly due to the possibility of reconfiguring them to perform custom functions. In this sense, they look quite different from traditional instrumentation, designed for one specific measurement task.

Apart from the initial system design, which for VIs must start with the identification of some minimum hardware resources, the design of a VI is a matter of software programming. The importance of having access to powerful programming languages is therefore fundamental. It is by using programming languages that general measuring tools are customized by the user to the intended application. In this sense, the VI designer has much more freedom than the

traditional instrumentation designers, even if such flexibility can pose serious problems of re-using available software, unless close attention is paid to realizing modular software in which previously designed software procedures can be included for new applications.

There are two categories of development environments for VIs available for the designer:

- textual languages;
- graphical languages.

In the case of textual languages, traditional programming languages, such as BASIC or C/C++, are used to realize a VI. Generally, programming in this case involves the use of functions specifically developed by the vendor for the hardware in question, and that govern the specific instrument functionalities and I/O functions for communication purposes.

Graphical programming allows the design of VIs using functional blocks (icons) that perform specific tasks. These blocks perform desired actions, from simple to very complex ones. Information is transferred from one icon to another by suitably wiring them. Also, programming languages come with a number of libraries (*e.g.* to perform signal spectrum analysis or digital signal processing) that greatly improve the language potentialities.

To better explain how graphical programming languages work, we will refer to the widely used LabVIEW™ by National Instruments, which can be considered a standard *de facto* for VI design, though competing languages have been proposed by other companies and still exist on the market.

A VI consists of a front panel that mimics the user interface and a block diagram that lies at the back of the front panel and is used to graphically define the VI functionalities. Moreover, during the design phase of a VI, other available VIs can be used as subroutines to perform simpler tasks in a hierarchical way.

The front panel of a VI, presented on the computer monitor, represents the user interface and intentionally looks very like a traditional instrument panel. This is the part of the VI that allows the user to act on the instrument, and eventually on the connected measuring hardware, by setting measurement parameters and loading data files. To obtain this capability, the designer can use a number of knobs, buttons, switches and so forth. Also, the front panel is used to show the user the results of VI operations (including measurement and computation results).

A number of tools are available to show VI outputs in the best way. The designer can, in fact, use time plots, XY graphs, numeric indicators, to give just a few examples, or even digital LEDs in those cases when an immediate alarm is better suited to the application.

As an example, in Figure 2.2 a very simple front panel showing a time graph, a numerical indicator, and a LED is reported.

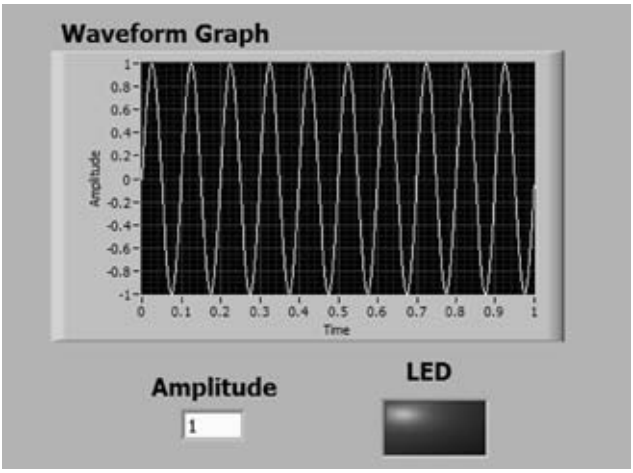

Figure 2.2. Example of a typical VI front panel

The block diagram is the core of a VI: the designer assembles here a number of available functional blocks, in the form of icons that carry out specific actions on input data, and produce corresponding output results. These blocks can either be part of the graphical programming language or have been realized by the user during the development of previous VIs. The function of each block can be simple, such as adding two input variables and giving the resulting sum, or very complex, such as performing sophisticated statistical analysis of input vectors.

Data are passed from one block to the next one using software wires. In the same way, elements in the front panel have icons in the block diagram that can be wired to functional blocks in the block diagram to allow commands from the user to be passed to the graphical software and final results to be shown on the front panel.

Figure 2.3 shows an example of a VI block diagram. The VI accepts numerical inputs from the VI front panel and returns their sum on a front panel indicator.

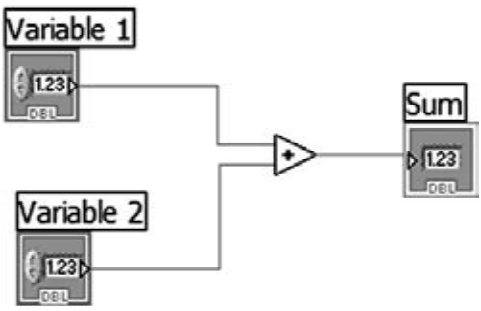

Figure 2.3. Examples of a typical VI block panel

Finally, Figure 2.4 shows a less didactic example of a VI block diagram. The reported example is part of the block diagram of the VI whose front panel is given in Figure 2.1, which was developed for the realization of a soft sensor for a debutanizer column.

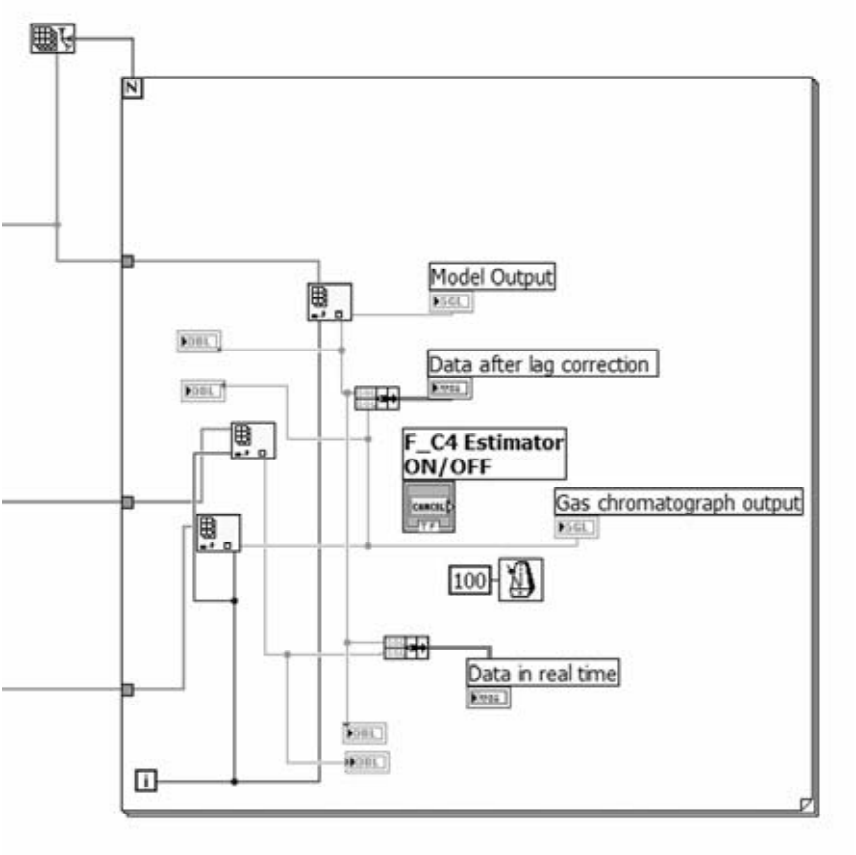

Figure 2.4. Part of a block diagram of the VI reported in Figure 2.1, for estimating debutanizer products

It is worth noticing that in the block diagram powerful tools for instrument driving, supplied by vendors, networking facilities, and data storage are available, all of which make VIs very flexible devices.

Some functional blocks can take inputs from an A/D converter and/or hardware measuring device and elaborate them on the basis of a user designed code. In other words, a block in the block diagram can be a soft sensor. Also, the soft sensor can either be designed using the graphical programming language that implements the VI or can be software, developed using a textual programming language, depending on the most suitable approach.

As a final remark on the difference between textual and graphical programming languages for VI design, it should be noted that, though graphical languages are more suitable for their eye appeal in presenting measurement results to the end user, they generally are very resource demanding and especially in real-time applications can introduce an unacceptable delay. In contrast, textual based VIs are much more conservative as regards computing resources and can be very efficient tools for real-time control applications. As usual, the final choice between the two alternatives will depend on the designer, who will need to take into account velocity constraints imposed by the application.

Of course, hybrid solutions where complex data elaboration is performed using textual programming languages, and where the user interface is realized using a graphical language, can be developed.

2.2 Applications of Soft Sensors

There are a number of reasons why soft sensors can be profitably used in industrial applications; currently they are becoming routine tools with the trend moving from open-loop information tools for the operator towards sensors in closed-loop inferential and/or adaptive control schemes.

Moreover, the wide availability of on-line analyzers and digital systems that are used both for monitoring and control give designers and operators the tools required for the design and implementation of soft sensors with a minimum, or even null, increase in the initial costs. In what follows, a description of typical soft sensors applications is given.

2.2.1 Back-up of Measuring Devices

A huge number of measuring devices connected to realize distributed monitoring networks, are used from industrial plants for monitoring and control pourposes. They routinely acquire a very large quantity of data. In fact, monitoring the state of a plant, even at a fixed time instant, could require sampling hundreds or even thousands of different variables.

Such measuring devices, and the corresponding data transmission systems, are required to face very harsh working environments. It is not surprising that working conditions impose both the use of very robust measuring hardware and periodic maintenance procedures. Notwithstanding such precautions, faults in measuring devices occur. Faults can come either in the form of abrupt changes in the working mode of measuring devices or in the form of slow changes of metrological characteristics. The latter can be even more dangerous than the former, because it is more difficult to detect and can hence cause malfunctioning of control systems.

Irrespective of whether a maintenance intervention be programmed or accidental, the measuring hardware needs to be turned off and suitably substituted. The back-up of measuring instrumentation is a typical application of soft sensors: an inferential model is in this case specifically designed to momentarily substitute unavailable measuring equipment and to avoid degradation of plant performance and rises in cost.

This particular scenario imposes restrictions on the soft sensor designer's possible choices. In particular, care must be taken in those cases when the variable inferred by the soft sensor is the output of a dynamic system. In fact, two possible choices are common:

- to restrict the model structure to moving average (MA) or nonlinear moving average (NMA) models that do not require past samples of the output variable. This corresponds to limiting the class of possible models to the class of finite impulse response (FIR) structure (including their nonlinear generalization), eventually decreasing the model performance with respect to more general model structures;
- to use autoregressive models as infinite-step-ahead predictors. In this case, the model has among its inputs past samples of its own estimations with corresponding feedback of model errors. Such structures are generally more efficient than the corresponding MA or NMA structures in the very first predicting steps but, generally, their performance quickly degrades due to error propagation. This is very true when the envisaged maintenance interval is very large compared to the system dynamics, so that a large number of successive samples are required to be estimated.

2.2.2 Reducing the Measuring Hardware Requirements

Using a software tool instead of a measuring hardware device represents, of course, a source of possible budget saving. Experts can therefore be encouraged to design inferential models that are intended to definitively substitute hardware devices, which become available for further reallocation.

Also in this case, a NMA model should be preferred to autoregressive structures. In any case much care should be taken to critically analyze model performance, due to the lack of any redundancy, and periodic model validation should be performed by temporarily inserting measuring devices and eventually proceeding to soft sensor retuning.

The problem of periodic soft sensors validation and eventual retuning is actually a common issue for any application of soft sensors. The need for such retuning can be due to a change in a new process working point (not considered during the soft sensor design phase), which can be detected by critically analyzing system inputs. This analysis should be constantly performed by checking for violation of suitable thresholds imposed for each input variable.

Soft sensor retuning is also needed when a change in system parameters occurs, in the case of slowly time-varying systems (*e.g.* due to seasonal variations).

The application scenario considered in this subsection is particularly sensitive to these problems. In fact, in the other applications described in this section, measuring hardware is always available (at least after a finite maintenance period) and this allows the required soft sensor validation operations to be accomplished. When the designer intends to eliminate measuring hardware, the availability of measuring facilities for sensor validation must be suitably planned, and extra hardware must be used temporarily.

2.2.3 Real-time Estimation for Monitoring and Control

The real-time estimation of system variables obtained using soft sensors, as opposed to its delayed measurement by means of hardware measuring devices, represents the most valuable feature of soft sensors; this is due both to the possibility to design a very efficient soft sensor and to the importance of the corresponding benefits in terms of process performance.

Any measuring instrument requires a finite time to perform the actions needed to give the final variable measurement. Though such a time can be very small in a number of applications, in some cases it can be significant. To give an example, this is the case with some gas chromatographs that require measuring times of the order of minutes or even greater. Moreover, due to the high cost of some measuring devices used in industrial applications, variables can sometimes be inferred on the basis of data acquired using measuring hardware that can be located on different processes, with a corresponding further delay (see, for example, the application reported in Chapter 8). Should this time be comparable with system dynamics, the measuring time can be a significant source of delay.

In the case of measuring instrumentation used for monitoring purposes, this corresponds to a delay in the time in which data are presented to the operator, with no relevant consequences, unless this information is important for safety issues.

When information about a variable value is needed in a closed-loop control scheme the effects of delay can decrease system performance to the point that the measuring hardware is not suitable for the control application.

In this class of applications the variable measurement is always available, albeit with a relevant delay. This allows the use of Auto-Regressive with eXogenous inputs (ARX) or Nonlinear ARX (NARX) model structures, which perform finite (and small) step-ahead prediction of the variable.

The real-time estimation obtained by the soft sensor can be used by the controller, while the corresponding delayed measurements allow the soft sensor performance to be improved, by avoiding the error propagation effect mentioned in the previous subsection.

2.2.4 Sensor Validation, Fault Detection and Diagnosis

An industrial control system can be seen as a hierarchy of at least three levels: the first level is the control level, which implements the actual control loop by means of feedback and feedforward controllers, state observers, parameter estimators, and so on. Above the control level, the supervision level accomplishes the task of continuously monitoring the operational life of the process, making the process operation almost independent from the presence of human operators. The highest level is dedicated to management, coordination and optimization activities, which provide the control system with high-level directives in order to maximize the performance of the system with respect to certain criteria.

Fault detection and diagnosis are part of the supervision functions accomplished by modern industrial control systems. In the past, the supervision function was essentially limited to checking important variables, and the consequent raising of alarms if some safety thresholds were trespassed. This was

actually an early stage of fault detection. On the other hand, at present, fault detection and diagnosis is performed by means of advanced techniques of mathematical modeling, signal processing, identification methods, computational intelligence, approximate reasoning, and many others. The main goals of modern fault detection and diagnosis systems are to:

- perform early detection of faults in the various components of the system, possibly providing as much information as possible about the fault which has occurred (or is occurring), like size, time, location, evaluation of its effects;
- provide a decision support system for scheduled, preventive, or predictive maintenance and repair;
- provide a basis for the development of fault-tolerant systems.

Fault detection and diagnosis strategies always exploit some form of redundancy. This is the capability of having two or more ways to determine some characteristic properties (variables, parameters, symptoms) of the process, in order to exploit more information sources for an effective detection and diagnosis action. The main idea underlying all fault detection strategies is to compare information collected from the system to be monitored with the corresponding information from a redundant source. A fault is generally detected if the system and the redundant source provide two different sets of information. There can be three main kinds of redundancy: physical redundancy, which consists of physically replicating the component to be monitored; analytical redundancy, in which the redundant source is a mathematical model of the component; knowledge redundancy, in which the redundant source consists of heuristic information about the process. When dealing with industrial applications, an effective fault detection and diagnosis algorithm must usually exploit a combination of redundancy sources, rather than a single one.

Sensor validation is a particular kind of fault detection, in which the system to be monitored is a sensor (or a set of sensors). At a basic level, the aim of sensor validation is to provide the users of a measurement system (that can be human operators, measurement databases, other processes, control systems, *etc.*) with an evaluation about the reliability of the measurement performed. At a higher level, a sensor validation system may also provide an estimate of the measurement in the case in which the actual sensor is out of order. In this framework, soft sensors are a valuable tool to perform sensor validation. Their usefulness is twofold. First, they can be exploited as a source of analytical redundancy. They can in fact be paralleled with actual sensors, and faults can be detected by comparison between the outputs of actual and soft sensors. Second, they can be exploited to provide an estimate of the sensor output in the case of sensor fault. Therefore, they can be used as a back-up device once a fault has been detected.

2.2.5 What-if Analysis

The design process of control systems requires the process behavior to be described via adequate theoretical/data-driven models that might be able to predict the system output corresponding to suitable input trends, for a given time span.

A model is used in this case to perform simulation of the system dynamics corresponding to input trends that are of interest, with the aim of obtaining both a deeper understanding of system behavior and/or designing suitable control policies. This particular use of process models to perform simulation is called what-if analysis.

Though first principle models could be a better choice due to their capability of describing the phenomena ruling the process, the difficulty of obtaining accurate enough models in a reasonable time can lead experts to adopt data-driven inferential models.

In the case of what-if analysis, inputs are therefore synthetic quantities, *i.e.* they are designed in order to analyze system reactions on a time span that makes sense, in accordance with system dynamics.

In this case, NARX models can be a suitable choice, due to the finite time span used in simulations. In fact, in this way, model error effects propagate only for a small number of iterations that must, however, be carefully fixed by the designer. It is also worth noting that, in the case of what-if analysis, input variables are noise-free, thus improving simulation performances.

On the other hand, much attention must be addressed to a careful choice of input trends. Much more than in the cases described in previous subsections, data used during soft sensor design must represent the whole system dynamics.

Also, the usual model validation should be followed by a further test phase in which canonical signals are used to force the real plant, and recorded plant reactions are compared to model simulations. A case study describing the design of a soft sensor to perform the what-if analysis of a real process will be reported in Chapter 8.