INFERENTIAL SENSOR FOR THE OLIVE OIL INDUSTRY

Carlos Bordons* and Manuel L. Zafra †

Abstract

This paper shows an inferential sensor that has been developed to be used in the olive oil industry. This sensor has been designed to measure two variables that appear in the elaboration of olive oil in a mill which are very difficult to be measured on line by a physical sensor. The knowledge of these variables on line is crucial for the optimal operation of the process, since they provide the state of the plant, allowing the development of a control strategy that can improve the quality and yield of the product. This sensor measures variables that in other case should come form laboratory analysis with large processing delays or from very expensive and difficult to use on line analysers.

The sensor has been devised based upon artificial Neural Networks (NN) and has been implemented as a routine running on a Programmable Logic Controller (PLC) and successfully tested on a real plant.

Keywords: Inferential Sensors, Neural Networks, Food Industry.

1 Introduction

Inferential or soft sensors are a way of indirectly measuring values when it is not possible to take a direct measurement. Inferential sensors differ from real or *physical* sensors in that they calculate a value by measuring other values. They are programs that calculate difficult to measure values by using formulae with inputs based on other, easier to measure values.

The most interesting use of this kind of sensors is to measure variables for which a sensor does not exist or, although it exists, it is very difficult or expensive to be used for on-line measurements. For instance, some attributes of manufactured products such as polymer melt index or resistance to thermal flow in insulation can only be measured by laboratory analysis, while other variables as moisture content need expensive and frequently calibrated sensors to be measured on-line. Such kind of variables appear in the process of elaboration of olive oil.

*Depto. de Ingeniería de Sistemas y Automática, Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, 41092-Sevilla (Spain), Phone:+34 95487348 Fax: +34 954487340. E-mail: bordons@esi.us.es.

†Department of Electrical and Computer Engineering. Northwestern University, 2145 Sheridan Road, Evanston, IL, 60208 Phone: 1 (847) 491 5410, Fax: (847) 491 4455. E-mail: zafra@ece.northwestern.edu.

The automatic control of the extraction of oil out of olives is still an open field where many problems related with measurement have to be studied. As olive oil mills are becoming bigger the chances for automation are increasing, therefore it is important to acquire the necessary knowledge of the process behaviour in order to design the appropriate control strategies. A survey of automation practices in the food industry can be found in [5].

In order to operate the process, it is needed to know on line two basic properties of the by-product: moisture content and oil content. This is usually done by means of laboratory measurements which, due to the large delay involved, are not suitable for automatic control. The inclusion of this inferential sensor in the control system allows the development of feedback controllers such as done in [6].

The paper is organized as follows. In section 2 a review of inferential sensors is presented and description of the process is done in section 3. Section 4 is dedicated to sensor development, whose results of use in a real mill are shown in section 5 and finally the major conclusions to be drawn are given.

2 Inferential sensors

Development of a soft sensor requires data collection and pre-processing, variable and time delay selection and model training and validation. The kind of model used depends mainly on the nature of the process. Partial Least Squares (PLS) or Principal Component Analysis (PCA) techniques can be used where significant correlations exist among measured variables, although most soft sensor applications describe non-linear behaviour and for this reason they use artificial Neural Nets (NN) technology to predict the parameter of interest.

Once the kind of model has been chosen, the sensor has to be trained with real data before starting operation. Data collection is essential since quality data are the only base for building a quality model. It is not so much the volume as the information content of the data. The main question here is how to define usable data and what constitutes a sufficient amount of data.

Network training automatically adjusts the weighting factors in the neural net, based on well-conditioned training data. The training tool must present information to the model, compare the output to a target value, automatically modify an input's weighting and repeat the training cycle until an acceptable output is achieved. After the ini-

tial training, the model must be validated with a different data set. A good verification indicates that the sensor is ready to go online.

The real time operation of the sensor, once it has been correctly trained, does not require great efforts, since it reduces to the evaluation of the model with fresh data that, even with non-linear models, is a low-consuming process. This makes it possible to be executed on a simple PLC.

What makes soft sensors useful is the ability of the underlying model to be trained to infer the physical property measurement otherwise available only from analysers or laboratory testing.

There are many inferential sensor applications, mainly is petrochemical refineries. One example is reported in [4], where an inferential sensor estimates distillation column top composition in an aeromatics separation unit using inputs for temperature, pressure and flow.

3 Plant description

The process is composed of several operations: reception of raw material (olives), washing, preparation, extraction, and storage of the produced oil [2]. Figure 1 shows the most important phases of the process.

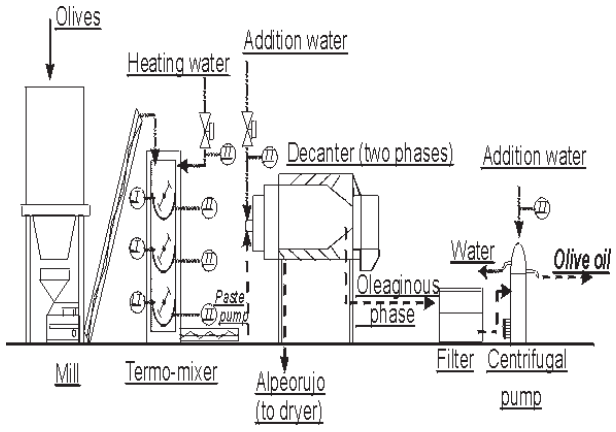

Figure 1: Olive oil mill - process description

The preparation phase consists of two subprocesses. The first one is olive crushing by an special mill, whose objective is to destroy the olive cells where oil is stored. The second one aims at homogenizing the paste by revolving it while its temperature is kept constant at a specified value (around 35^oC). This is performed in a machine called *thermomixer*, which homogenizes the three phases of the paste (oil, water and by-product (*alpeorujo*)) while exchanges energy with surrounding pipes of hot water. This is done in order to facilitate oil extraction in the following process: mechanical separation, or extraction, in the *decanter*.

Homogenization is really important in the whole process, because bad operation conditions in the *thermomixer* can dramatically reduce the quality and quantity of the final product. The paste is heated in order to facilitate mixing since the paste turns more fluent when temperature rises. However, there exists an upper temperature limit behind which olive oil loses quality (flavour, fragrance, etc.) due to the oxidation process and the loss of volatile components. Therefore, keeping low values of temperature will be a high-priority objective. Experiences of modeling and predictive control of this phase are described in [1].

The next stage is based on the separation of the product phases by means of a centrifuge. This is a continuous process which separates the different components that constitute the paste by means of centrifugal force. This separation is made in the horizontal centrifuge or *decanter*, that separates olive oil from by-product. In order to perform a good separation, the paste that enters the *decanter* must be accommodated. Its flow must be controlled to a setpoint that depends on operating conditions and some water must be added depending on the properties of the raw material.

Finally, the last stage of the system consists of the storage and the conservation of the obtained oil.

The main variable to be controlled is olive oil flow. The objective is to obtain the maximum yield but without affecting product quality. It implies that some operational constraints have to be fulfilled. The main indication of the way the process is operating is the oil content of the byproduct (*alpeorujo*). If this value is too high, it means that part of the extractable oil is being wasted. Therefore, this value is used to manage the plant.

Another important variable is the moisture content of the byproduct. On one hand, it is used to calculate the quantity of process water that must be added to the paste before it enters the decanter and on the other hand it has to comply with operational limits that are imposed on this stuff since it is re-processed later.

A block diagram of the process showing the place where these variables should be measured is depicted in figure 2.

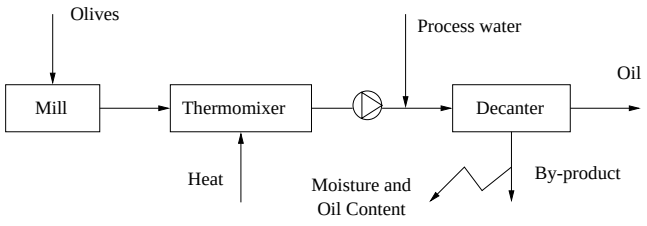

Figure 2: Olive oil mill - process description

Up to now, the usual way of measuring these variables is by laboratory tests. Samples are taken and sent to the laboratory (either in the same plant or outside) which gives the results some hours later. Obviously, this value is not appropriate for a real time control, since the information is received when no correcting actions can be taken. This delay in knowing crucial information gives place to production losses, which could be corrected with information

available on time.

This can be solved by the use of on-line analysers which give real time measurements. There exist devices based on Near Infrared (NIR) spectroscopy that can supply these values (see for instance, [3]). But they are high-cost, complex devices which need a delicate calibration procedure done by an expert. They also need complex auxiliary devices to take samples, which can affect the measured value.

Therefore, the measurement of these byproduct properties presents two basis difficulties: the processing delay when it is done in laboratories and the complexity and cost when a continuous measuring with NIR devices want to be done.

The work done here tries to solve these problems by the use of a soft sensor that supplies the inferred values of moisture and oil content in real time using measurements supplied by physical sensors such as temperature and flow that are already located in the plant.

4 Sensor development

In order to implement the neural network (NN) that is going to function as the inferential sensor, certain issues have to be pertinently addressed, such as: variables that are going to be used as inputs and how each of them affect the output variables, topology of the NN (number of nodes in each layer of the NN), how many samples of each variable are going to be used, training algorithm to be used, previous treatment of the input data and how the results are going to be evaluated. These previous work requires a deep knowledge of the process to be modeled and will determine in a drastic way the future performance of the inferential sensor.

4.1 Variables to be included in the model

The outputs of the NN are the variables which we want to estimate with the inferential sensor, e.g., the oil content and the moisture content of the byproduct (*alpeorujo*). Therefore, the NN will have two outputs. Instead of implementing only one NN with two outputs, we proved that the results are improved if two independent NN were trained separately, given that the weights have to adapt to estimate only one variable at a time.

After careful analysis of the performance of the plant and taking into account the already available sensors present in the plant, the selected input variables were the flow and temperature of the olive paste and the flow and temperature of the water that is added after the paste leaves the thermo-mixer. Other variables that might have influence on the outputs, such as the moisture content and oil content of the incoming olives were not included in the model since they are not currently being measured, but its inclusion in the model would be easy and could only make our results improve.

For each of the inputs, two important issues have to be determined: the delay due to transportation phenomena and the characteristic variation time of the output when there is a change in the input. The delay is important because in order to estimate the output at time t the input to be considered should be at time $t-D_i$, $i = 1 \ldots$ number of inputs, where D_i is the delay associated with each input. The characteristic time T_i is essential in the sense that it provides quantitative information of the dynamics of the plant, so that the best inputs can be chosen. The way of choosing these two times is by looking at the plant behaviour. The input vector is, therefore, $x(t) = [x_i(t - D_i), x_i(t-D_i-T_i/10)]^T$ supposing that we take 10 samples during the characteristic time.

Table 4.1 shows the values of the delays and characteristic times for the four variables included in the model.

Variable	Delay	Char. Time
Flow of water	7	7
Flow of paste	7	4
Water Temperature	4.5	6
Paste Temperature	8	7

Table 1: Times of the chosen variables (in minutes)

4.2 Topology of the neural network

The optimization of the topology of the NN is also a very important topic in this paper, since there is a compromise between obtaining good results in the estimation and the computational effort. A neural network is theoretically able to estimate almost any function with arbitrary small error if a large number of parameters are left free. Those parameters are the result of the NN training, so the estimation of too many parameters leads to complexity and time consuming training algorithms.

For our purpose, a two-layered NN was enough to capture the dynamic of the plant, and the number of nodes of the first layer and the number of cycles of training were subjects of deep study. In this study, 225 neural networks with different topologies and number of cycles were trained and evaluated. Since the starting weights of the NN are chosen arbitrarily at the beginning of the training algorithm, this experiment helped us understand the random behavior of the performance of the NN as well. Figure 3 shows some results for several number of nodes and number of cycles.

4.3 Training algorithm and treatment of input data

The training algorithm used in this paper was Levenberg-Marquardt Backpropagation [7]. It was chosen for functionality and easy implementation with Matlab. Training stops when one of these conditions occurs: the maximum number of epochs has been reached, the maximum amount

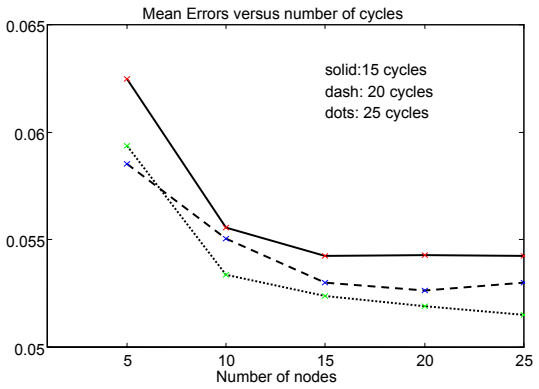

Figure 3: Error versus number of nodes and cycles

of time has been reached, the performance gradient falls below a certain number or the validation performance has increased more than a certain number.

The training algorithm follows the following path:

1. A vector of samples is applied at the input layer.
2. Comparison between the desired output and the one obtained, so that you get an associated error.
3. Determine whether each weight has to be increased or decreased (+ or -)
4. Depending on the value of the gradient of the error function, the exact amount of change of each weight is obtained.
5. Repeat the steps above until you reach the desired average error or a predefined number of times or cycles.

The data have to be preprocessed before following the training algorithm. First of all, the data have to be normalized conforming the following formula:

$$X_{nom} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

where X_{min} and X_{max} are the minimum and maximum values that the variable can take, respectively. In this way, the exponential functions involved in the algorithm do not saturate and, what is even more important, you could extrapolate the results obtained with this method to a another olive oil plant with a different size but similar manufacturing process.

The data need to be filtered as well, so that the NN does not capture the high frequency dynamics but the low frequency dynamics that we desire. Depending on the sample time, data could need to be resampled with the objective of reducing the number of necessary computations.

During the normal operation of the plant, there are several stops due to change of the type of olives that are entering the plant or due to the cleaning of the machinery involved. Therefore, those data should be carefully removed before training the NN. Data used to train the NN should be dynamically rich in the sense that the NN would be able to reproduce its behavior after being trained. Dynamically poor data lead to poor estimations.

4.4 Validation of results and robustness

There are two types of validation. The first one is the average error made in the estimation of the two outputs while training the neural network. Although the dynamic of the plant was many times confusing and not clear, close approximations to the real dynamic of the plant with a reasonable computation time and resource effort were achieved.

The second evaluation type, which is even more important from a practical point of view, is the performance of the NN when it is estimating the outputs in real plant, e.g.., with data that are completely different than those with which it was trained. This performance index is essential and not easy to improve, so that many modifications had to be made to the NN in order to make this index improve even if the error while training the NN would increase. This means that we had to sacrifice the error obtained while training the net in order to get a more robust behavior in practice.

5 Results and Implementation

The main objective of this paper is to evaluate whether the inferential sensor can substitute the real sensor or, in contrast, the estimations are far away from the real values of the outputs.

For most of the simulations done in this paper, a NN with 9 nodes at the first layer was chosen and 10 cycles were used for the training. The NN had 8 inputs, two entries for each of the variables delayed $T_i/30$ (seconds) one from each other.

5.1 Results while training the NN

Figure 4 shows the network output for moisture content.The error made in the estimation of the moisture content was 0.35 (1.8 %) and the maximum was 1.89. Notice that the range of variation for this variable was 45-64.59.

The error made in the estimation of the oil content was 0.18 (2.8%) and the maximum was 0.94. The range of variation for this variable was 2.73-9.19. The evolution is shown in figure 5.

The evolution of the average error with the number of cycles can be observed in the following plot (figure 6).

5.2 Results when extrapolating the NN

Several modifications were introduced in the topology of the NN and in the implementation of the system in order

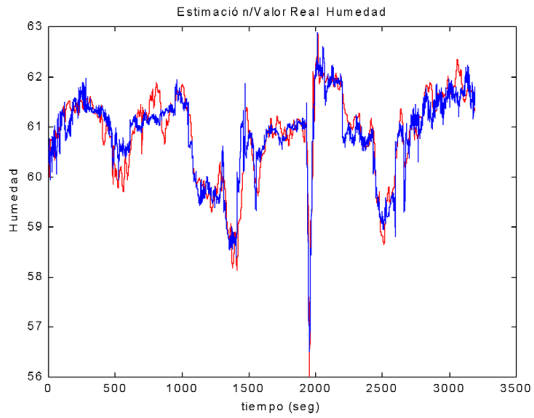

Figure 4: Estimation of moisture content

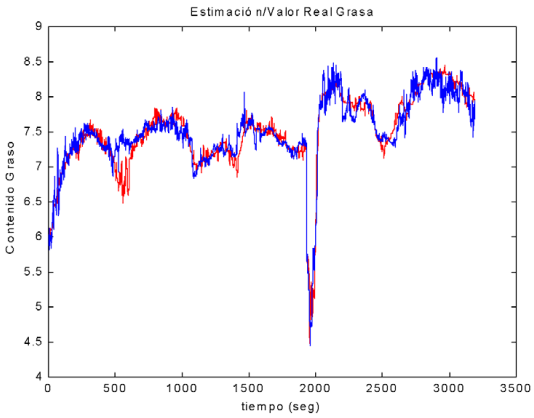

Figure 5: Estimation of oil content

to make the behavior of the NN robust. Some of these modifications were:

- Removal of the temperature of the paste from the model: the great dependence of the measures of the real sensor on the temperature of the paste made it difficult to estimate the outputs when the temperature of the water was not in the range with which the NN had been trained. By doing this, the extrapolation improved significantly.
- Variation of the sample time: With the objective of reducing the computational effort, instead of taking 30 samples per characteristic time, we took only 10, reducing the number of input patrons to a third of the original. By doing this, not only the computational time was significantly reduced but the performance of the NN improved as well.
- Two-output NN: Since both outputs are highly correlated, implementing only one NN for both outputs would be reasonable in the sense that it would save memory in the PLC (less variables have to be created in the data base), time and resources. Nevertheless, the performance is expected to be worse since all the parameters in the NN have to be adapted for both outputs at the same time.

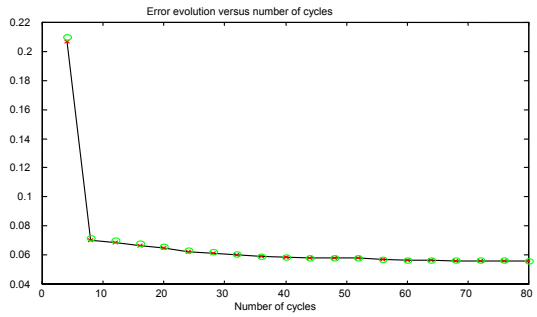

Figure 6: Evolution

- Removal of the temperature of the water from the model: After deep analysis of the performance of the NN while changing the inputs, we got to the conclusion that the addition of the flow of water was nothing but a source of error while extrapolating the NN, since it is a magnitude that depends mainly on the moisture content of the incoming olives. During wet seasons water is hardly added to the paste and while dry seasons this flow can reach very high values. Extrapolation improved.
- Addition of more samples of each variable in the input vector: in order to capture the dynamics of the plant with a higher accuracy, four delayed samples of each variable were introduced in the input vector to the NN. Doing this, the NN would be able to recognize higher order dynamics present in the plant. The computational time and effort does not increase drastically and much better performances are reached.
- Dynamic NN: This case covered the hypothetical case in which you had previous information of the values of the outputs some time before. Two NN would be working in parallel, the first estimating the values of the outputs a certain time delayed while the second NN would use the information given by the first NN to estimate the current values of the outputs. Significant improvement was reached with this approach.
- Addition of laboratory analysis results: The NN should be retrained when new laboratory analysis results are available. This is actually possible adding that information while training the network and it is also possible to weight the importance of the new information versus the one already available.

The results of the extrapolation of the NN can be viewed in figures 7 and 8.

The average error made in the estimation of the moisture content was 1.60 and the maximum was 8.47. The

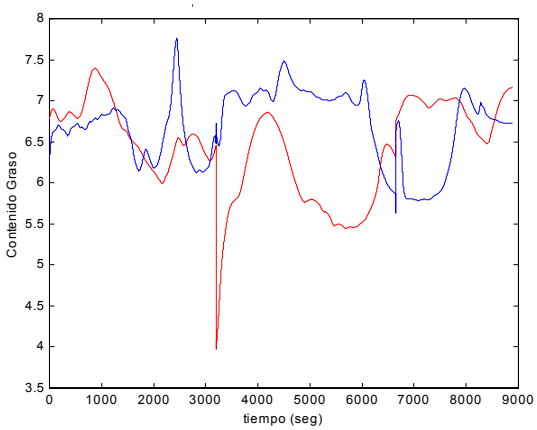

Figure 7: Results of extrapolation

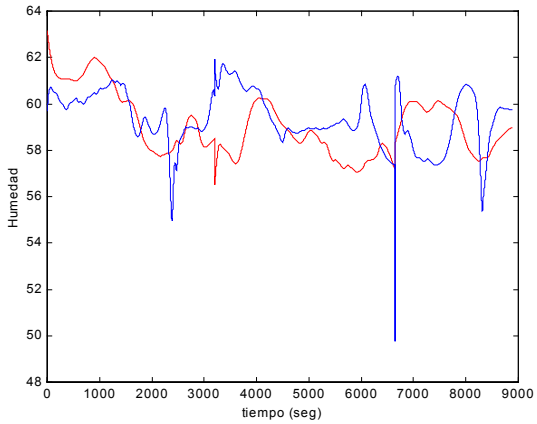

Figure 8: Results of extrapolation

errors while training were 0.65 and 2.80, respectively. The range of variation for this variable was 45-64.59.

The average error made in the estimation of the oil content was 0.87 and the maximum was 2.75. The errors while training were 0.26 and 1.02, respectively. The range of variation for this variable was 2.73-9.19.

5.3 Implementation on a Commercial Control System

Before implementing the inferential sensor in a SCADA/PLC packet it is necessary to distinguish between the NN propagating the information forward and the NN being trained. These two different states of the NN should be considered independently and generate some important issues that need to be pertinently addressed:

- Memory issues: The larger the NN is, the more variables you need in the data base to implement the NN. With two NN working simultaneously, 2×(No. inputs × No. nodes + No. nodes) variables are needed just to store the values of the weights of the NN. If only one NN with two outputs is chosen, (No. inputs × No nodes + 2 × No. nodes) variables are needed. In some applications, this issue could limit the size of the NN to implement.

- Programming languages that are used in SCADA/PLC packets are usually relatively simple, what makes it easy to program an application but constrains the complexity that can be achieved with those packets.

- Training of the NN consumes a long time if compared to the time other routines in the PLC take. Usually, one routine is executed after other and each of them have an execution time assigned, so including the training of the NN could collapse the normal execution of the system.

- The two different status of the NN have different timing sequence. While the NN should be propagating forward the information in a constant basis, the NN should only be retrained when new lab analysis are available.

All this implies that the best way to implement the training routines are outside the PLC, in the computer where the SCADA is running in a high level programming language, like C or C++.

This is what has been done in this application. The sequence of actions is:

1. Reception of lab analysis at the olive oil plant. The operator in charge will be responsible for introducing the new information in the training program through the appropriate interface (Graphic Monitoring System of the SCADA or equivalent Visual Interface)

2. The training program will read the current weights of the network which are contained in the data base of the SCADA.

3. The program will read the historical data base intervals to train the NN which should be chosen with the precaution that they contain no unusual states of functioning and that they are dynamically rich enough to obtain good estimates of the outputs.

4. The training algorithm starts and when it is done it will rewrite the new weights of the NN over the old ones. The NN propagation routine runs in the PLC.

6 Conclusions

An inferential sensor to provide moisture and oil content in an olive oil mill has been developed and tested. This sensor solves the existing problems associated to the real-time measurement of these values, whose knowledge is crucial for a good plant control.

The sensor has been devised based upon artificial Neural Networks and has been implemented as a routine running on a PLC and successfully tested on a real plant. The low computational requirements of the sensor allow its use in a simple control system, as is the case of an olive mill where it has been tested.

A feedback procedure for updating NN weights is used in the way that the NN is retrained every time a new laboratory measurement is available. This improves the inferential sensor behaviour since it avoids possible drift.

7 Acknowledgments

The authors would like to acknowledge Manuel R. Arahal for collaborating in the project and people from PROCISA for their help in carrying out the tests. This work has been supported by the Spanish Ministry of Science and Technology under grant DPI2001-2380-C02-01.

References

[1] C. Bordons and J.R. Cueli. Modelling and Predictive Control of an Olive Oil Mill. In *Proceedings European Control Conference, Porto*, september 2001.

[2] L. Civantos. *Obtencin del aceite de oliva virgen (In Spanish)*. Ed. Agrícola Española, S.A, 1999.

[3] D.M. Haaland and D.K. Melgaard. New augmented classical least squares methods for improved quantitative spectral analyses. *Vibrational Spectroscopy*, pages 171–175, 2002.

[4] D. Harrold. Process Control's latest tool: soft sensors. *Control Engineering Europe. June*, pages 42–45, 2001.

[5] S.V. Ilyukhin, T.A. Haley, and R.K. Singh. A survey of automation practices in the food industry. *Food Control*, 12:285–296, 2001.

[6] C.B. Scheffer-Dutra, A. Nú nez Reyes, and C. Bordons. Predictive Control of an Olive Oil Mill with Multiobjective prioritisation. In *Proceedings of the IFAC Triennal World Congress*, Barcelona, Spain, 2002.

[7] L. Zhang and G. Subbarayan. An evaluation of backpropagation neural networks for the optimal design of structural systems: Part II. Numerical evaluation. *Computer Methods in Applied Mechanics and Engineering*, pages 2887–2904, 2002.