Journal of Applied Statistics
Vol. 34, No. 5, 511–527, July 2007

Standardization of Process Norms in Baker's Yeast Fermentation through Statistical Models in Comparison with Neural Networks

PRASUN DAS* & SASADHAR BERA**

**SQC & OR Unit, Indian Statistical Institute, Kolkata, India, **Samserpur, Tarakeswar, Hooghly-712410, India*

ABSTRACT *Achieving consistency of growth pattern for commercial yeast fermentation over batches through addition of water, molasses and other chemicals is often very complex in nature due to its biochemical reactions in operation. Regression models in statistical methods play a very important role in modeling the underlying mechanism, provided it is known. On the contrary, artificial neural networks provide a wide class of general-purpose, flexible non-linear architectures to explain any complex industrial processes. In this paper, an attempt has been made to find a robust control system for a time varying yeast fermentation process through statistical means, and in comparison to non-parametric neural network techniques. The data used in this context are obtained from an industry producing baker's yeast through a fed-batch fermentation process. The model accuracy for predicting the growth pattern of commercial yeast, when compared among the various techniques used, reveals the best performance capability with the backpropagation neural network. The statistical model used through projection pursuit regression also shows higher prediction accuracy. The models, thus developed, would also help to find an optimum combination of parameters for minimizing the variability of yeast production.*

KEY WORDS: Generalized linear model (GLM), multisample bootstrapping, projection pursuit regression, artificial neural network (ANN), yeast, fed-batch fermentation

Introduction

An Artificial Neural Network (ANN) is an information-processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements, known as neurons, working in unison to solve specific problems. An ANN is configured for a specific problem like pattern recognition, data classification, prediction etc through a learning process. Learning in a biological system involves adjustments to the synaptic connections that exist between the neurons. This is true for an ANN as well. All neural networks have some set of processing units that receive inputs from the outside world, which can be referred to

Correspondence Address: Prasun Das, SQC & OR Unit, 203 B.T. Road, Kolkata –700 108, West Bengal, India. Email: dasprasun@rediffmail.com
0266-4763 Print/1360-0532 Online/07/050511–17
DOI: 10.1080/02664760701234793

appropriately as the 'input units' or 'input nodes'. It does have one or more layers of 'hidden' processing units that receive inputs only from the other processing units. The set of processing units that represent the final result of the neural network computation is designated as the 'output units'.

In this paper, the authors are interested in moving towards robust control of a fermentation process using multivariate statistical techniques and neural net systems. The fermentation process is a biological base system. This system runs smoothly depending on effective operator supervisory and formulated rules from knowledge based systems (Lennox *et al.*, 2001). However, maintaining the consistency of the growth pattern over the batches of commercial yeast production through the addition of water, molasses and other chemicals creates problem to the manufacturer. An attempt has been made to find a robust control system for the yeast fermentation process to predict future response under various process constraints. The data used in this context are obtained from an industry producing baker's yeast. Relevant information on Fermenter parameters (such as Air CFM, Temp, pH, Dip (M), Vol(L), Amps, ALC %, Spin), Yeast in fermenter (kg, increment, G.M.), Wort and other chemical additions are collected from a Brew sheet for consecutive batches of yeast production.

A univariate model for predicting the response from real-life data fails in practical situations due to deviations caused by the interaction between process parameters, the unsteady state in batch operation etc (Lennox *et al.*, 1999). A more rigorous approach to monitor a fermentation process is Multivariate Process Control through Multiway principal component analysis (PCA), Multiway partial least squares (PLS), ANN etc. The PCA algorithm is able to project highly correlated process data into a low dimensional space defined by the principal component. The condition monitoring system for fed-batch fermentation systems and algorithms is presented in another paper (Lennox *et al.*, 2001). However, PCA and PLS have their limitation in non-linear system. Artificial Neural Networks can handle effectively and efficiently non-linear and complex systems. The principal difference between ANN and statistical approaches is that an artificial neural network makes no assumption about the statistical distribution or properties of the data.

This paper is organized as follows. In the next section, a brief description of the fermentation process of yeast is described. The third section gives a general discussion on the prediction tools, such as Linear Regression, Stepwise Regression, Generalized Linear Model (GLM), and Projection Pursuit Regression (PPR), along with the Boostrapping and Backpropagation algorithms used by ANN for learning. The implementations and results are given in the fourth section followed by a comparison of results. A brief discussion on the results and conclusions are stated in the fifth and sixth sections respectively.

Fermentation Process

It was in the pioneering scientific work of Louis Pasteur in the late 1860s that yeast was identified as a living organism and the agent responsible for alcoholic fermentation and dough leavening. Shortly following these discoveries, it became possible to isolate yeast in pure culture form. With this new knowledge, the stage was set for commercial production of baker's yeast that began around the turn of the 20th century. Baker's yeast is used to leaven bread throughout the world. Baker's yeast products are made from strains of this yeast selected for their special qualities relating to the needs of the baking industry. The quality of baker's yeast is often discussed in terms of microbiological purity and gas producing activity.

The baker's yeast production process flow can be divided into four basic steps, namely, molasses and other raw material preparation, culture or seed yeast preparation, fermentation and harvesting, and filtration and packaging. Yeast can grow in the presence or absence of

air. For instance, in bread dough, yeast grows very little under anaerobic conditions; instead, the sugar that can sustain either fermentation or growth is used mainly to produce alcohol and carbon dioxide. This means that the baker who is interested in the leavening action of the carbon dioxide works under conditions that minimize the presence of dissolved oxygen. In contrast, under aerobic conditions, in the presence of a sufficient quantity of dissolved oxygen, yeast grows by using most of the available sugar for growth and producing only negligible quantities of alcohol. So, a yeast manufacturer that wants to produce more yeast cell mass, works under aerobic conditions by bubbling air through the solution in which the yeast is grown.

The problem posed to the yeast manufacturer, however, is not just as simple as adding air during the fermentation process. If the concentration of sugar in the fermentation growth media is greater than a very small amount, the yeast will produce some alcohol even if the supply of oxygen is adequate or even in abundance. Adding the sugar solution slowly to the yeast throughout the fermentation process can solve this problem. The rate of addition of the sugar solution must be such that the yeast uses the sugar fast enough so that the sugar

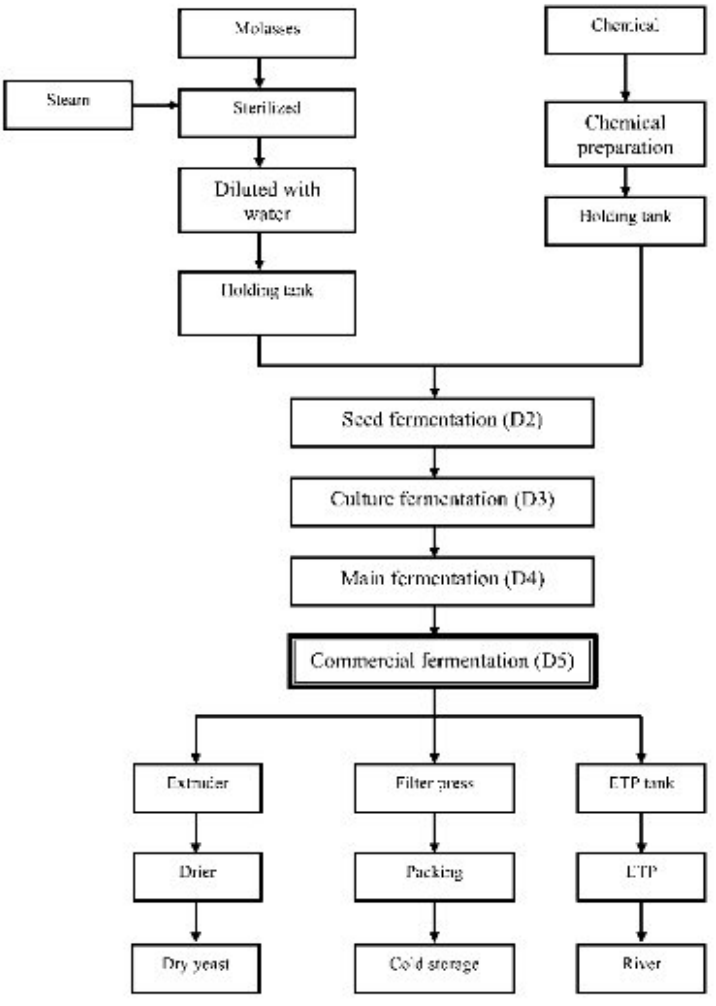

Figure 1. Yeast fermentation process – schematic

concentration at any one time is practically zero. This type of fermentation is referred to as fed-batch fermentation. The flowchart of the fermentation process is shown in Figure 1.

Materials and Methods

Bootstrapping

The bootstrap is a method of Monte Carlo simulation where *no parametric assumptions* are made about the underlying population that generated the random sample. Instead, we use the sample as an estimate of the population. This estimate is called the empirical distribution $\hat{F}$ where each x_i has probability mass $1/n$. Thus, each x_i has the same likelihood of being selected in a new sample taken from $\hat{F}$.

When we use $\hat{F}$ as our pseudo-population, then we resample *with replacement* from the original sample $x = (x_1, x_2, \ldots, x_n)$. We denote the B new sample obtained in this manner by $x^{*b} = (x_1^{*b}, x_2^{*b}, \ldots, x_n^{*b})$, $b = 1, 2, \ldots, B$. Since we are sampling with replacement from the original sample, there is a possibility that some points x_i will appear more than once in x^{*b} or may be not at all. The statistic is first calculated using the observed data and then recalculated using each of the new samples, yielding a bootstrap distribution. The resulting replicates are used to calculate the bootstrap estimates of bias, mean, and standard error for the statistic (Martinez & Martinez, 2002).

Basic Bootstrap Methodology

Step 1. Given a random sample, $x = (x_1, x_2, \ldots, x_n)$, calculate statistic $\hat{\theta}$.

Step 2. Generate a sample with replacement from the original sample to get $x^{*b} = (x_1^{*b}, x_2^{*b}, \ldots, x_n^{*b})$.

Step 3. Calculate the same statistics using the bootstrap sample in Step 2 to get, $\hat{\theta}^{*b}$

Step 4. Repeat Steps 2 through 3, $b = 1, 2, \ldots, B$ times. B is number of bootstrap resamples to be drawn.

Step 5. Use this estimate of the distribution of $\hat{\theta}$ to obtain the desired characteristic (standard error, bias, confidence interval).

Multisample Bootstrapping

Multisample bootstrapping is supported through the group argument. Group arguments allow stratified sampling and bootstrapping multi-sample problems. The unique values of this vector determine groups. For each resample, a bootstrap sample is drawn separately for each group, and the observations are combined to give the full resample. The statistic is calculated for the resample as a whole (S-PLUS Guide, 1999).

Generalized Linear Model

A generalized linear model (GLM) provides a way to estimate a function (called the link function) of the mean response as a linear function of the values of some set of predictors. This is written as

$$g\left(E\left(\frac{Y}{X}\right)\right) = g(\mu) = \beta_0 + \sum_{i=1}^{p} \beta_i X_i = \eta(X)$$

where g is the link function. The linear function of the predictors, $\eta(X)$, is the linear predictors. For GLM, the variance of Y may be a function of the mean response μ like, $\text{Var}(Y) = \varphi \text{Var}(\mu)$.

A fundamental idea is that there are two components to a GLM; the response distribution (also called the error distribution) and the link function. Another concept underlying GLM is the exponential family of distribution, which includes the normal, Poisson, Binomial, Exponential and Gamma distributions as members. Since, the normal error in the linear model is just a special case of the GLM, the GLM, therefore, can be thought of as a unifying approach to many aspects of empirical modeling and data analysis.

The estimates of the regression parameter in GLM are maximum likelihood estimates, produced by iteratively reweighed least squares (IRLS) (Myers *et al.*, 2002).

Stepwise Regression

Evaluating all possible regressions can be computationally burdensome; various methods have been developed for evaluating only a small number of subset regression models by either adding or deleting regressors one at a time. These methods are generally referred to as stepwise-type procedures. These are *forward selection*, *backward elimination* and *stepwise regression*. The last one is a popular combination of procedures of the first two. It is a process in which at each step all regressors, entered into the model previously, are reassessed via their partial F-statistic and considered as current members in the model (Ronald, 1996).

Projection Pursuit Regression

This method computes an exploratory nonlinear regression method that models *Y (response)* as a sum of non-parametric functions of projections of the *X (predictor)* variables. Projection pursuit regression (PPR) constructs a model of the regression surface based on projections of the data on planes spanned by the response and linear combination of predictors. It has the ability to pick up model interaction.

Let Y and $X = (X_1, X_2, \ldots, X_p)^T$ denote the response and explanatory vector respectively. Suppose you have observations Y_i and corresponding predictors $X_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^T, i = 1, 2, \ldots, n$.

Let $a_1, a_2, \ldots, a_p$ denote p-dimensional unit vectors, as 'direction' vectors. The projection pursuit function finds $M = M_0$, direction vectors $a_1, a_2, \ldots, a_{M_0}$ and good nonlinear transformations $\phi_1, \phi_2, \ldots, \phi_{M_0}$ such that $Y = \mu_Y + \sum_{m=1}^{M_0} \beta_m \phi_m(a_m^T X) + \varepsilon$ where $\beta_m =$ term weight on mth term and $\mu_Y = E(Y) = \overline{Y} = 1/n \sum_{i=1}^{n} Y_i$ provides a 'good' model for the data $Y_i, X_i, i = 1, 2, \ldots, n$.

The 'projection' part of the term *projection pursuit regression* indicates that the carrier vector X is projected onto the direction vectors $a_1, a_2, \ldots, a_{M_0}$ to get the lengths $a^T X, i = 1, 2, \ldots, n$ of the projections and the 'pursuit' part indicates that an optimization technique is used to find 'good' direction vectors $a_1, a_2, \ldots, a_{M_0}$.

Y and X are presumed to satisfy the conditional expectation model

$$E[Y|x_1, x_2, \ldots, x_p] = \mu_Y + \sum_{m=1}^{M_0} \beta_m \phi_m(a_m^T X)$$

and ϕ_m have been standardized to have mean zero and unity variance:

$$E\{\phi_m(a^T X)\} = 0 \quad \text{and} \quad E\{\phi_m^2(a^T X)\} = 0, \quad m = 1, 2, \ldots, M_0$$

The observations Y_i and corresponding predictors $X_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^T$, $i = 1, 2, \ldots, n$ are assumed to be independent and identically distributed random variables such as Y and X.

The true model parameters β_m, ϕ_m, a_m $m = 1, \ldots, M_0$ minimize the mean squared error

$$E\left[Y - \mu_Y + \sum_{m=1}^{M_0} \beta_m \phi_m (a_m^T X)\right]^2 \quad \text{over all possible} \quad \beta_m, \phi_m, a_m$$

When $M_0 = p$ and $a_1 = (1, 0, 0, \ldots, 0)^T, a_2 = (0, 1, 0, \ldots, 0), \ldots, a_p = (0, 0, 0, \ldots, 1)$, β_ms are absorbed into the ϕ_ms.

This model builds up the smooth regression surface, which is a projection defined surface by the smooth estimators. However it is difficult to interpret surface for m larger than 2. When $M_0 = 1$ assuming predictors X are independent with mean 0 and variance 1

$$a^T = \frac{(b_1, b_2, \ldots, b_p)}{\sqrt{b_1^2 + b_2^2 + \cdots + b_p^2}}, \phi_1(t) = t \quad \text{and} \quad \beta_1 = \sqrt{b_1^2 + b_2^2 + \cdots + b_p^2}$$

where b_j are regression coefficients.

Projection Pursuit Model Selection Strategy

For each order m, $1 \leq m \leq M$, PPR will evaluate the fraction of unexplained variance

$$e^2(m) = \frac{SSR(m)}{\sum_{i=1}^n w_i (Y_i - \overline{Y})^2} = \frac{\sum_{i=1}^n w_i [Y_i - \overline{Y} - \sum_{l=1}^m \hat{\beta}_1 \hat{\phi}_l (\hat{a}_1^T x_i)]^2}{\sum_{i=1}^n w_i (Y_i - \overline{Y})^2},$$

$w_i =$ Observation weights

(1) A plot of $e^2(m)$ versus m, which is decreasing in m, may suggest a good choice of $m = M_0$.
(2) Often, $e^2(m)$ will decrease relatively rapidly when m is smaller than a good model of order M_0 and then flatten out. $e^2(m)$ will decrease more slowly for m larger than M_0. M_0 will have to be chosen keeping this in mind.

This technique is highly computer intensive and is also useful in multivariate responses (Friedman & Stuetzle, 1981; Hastie *et al.*, 2001).

Backpropagation Neural Network

Generalizing the Widrow-Hoff learning rule (Widrow & Hof, 1960) to multiple-layer networks and nonlinear differentiable transfer functions creates Backpropagation. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by users. Standard backpropagation is a gradient descent algorithm in which the network weights are moved along the negative of the gradient of the performance function. The term backpropagation refers to the manner in which the gradient is computed for nonlinear multilayer networks. There are a number of variations on the basic

algorithms that are based on other standard optimization techniques, but Scaled Conjugate Gradient (Moller, M.F., 1993) and Levenberg-Marquardt algorithms (Marquardt, 1963) are most effective and appreciable.

Scaled Conjugate Gradient

The basic backpropagation Scaled Conjugate Gradient (SCG) algorithm adjusts the weights in the steepest descent direction (negative of the gradient). In the conjugate gradient algorithms, a search is performed along conjugate directions, which produces generally faster convergence than steepest descent directions. In case of quadratic functions, exact answers are obtainable without calculating second-order derivatives, as discussed next.

Given a symmetric matrix Q, two vectors d_1 and d_2 are said to be *conjugate* with respect to Q if $d_1^T Q d_2 = 0$. An important result is that when the matrix Q is positive-definite, a set of non-zero conjugate vectors is also linearly independent. The conjugate gradient algorithm for a quadratic problem is defined as follows:

(1) Let $d_0 = -\nabla f(x_o) = b - Qx_0$, where $x_0 \in R^n$ is an arbitrary starting point.
(2) For $k = 0, 1, \ldots, (n-1)$, define $\nabla f(x_k) = Qx_k - b$, and do

a) $x_{k+1} = x_k + \alpha_k d_k$, where $\alpha_k = -\dfrac{(\nabla f(x_k))^T d_k}{d_k^T Q d_k}$

b) $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$, $\beta_k = \dfrac{(\nabla f(x_k))^T Q d_k}{d_k^T Q d_k}$

Commonly used stopping criteria are:

$$f(x_{k+1}) - f(x_k) < \varepsilon \quad \text{and} \quad x_{k+1} - x_k < \delta$$

Levenberg–Marquardt

The Levenberg–Marquardt (LM) algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. When the performance function has the form of a sum of squares (as is typical in training feedforward networks), then the Hessian matrix can be approximated as

$$\mathrm{M} = J^T J$$

and, the gradient can be computed as

$$g = J^T e$$

where, J is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases, and e is a vector of network errors. The Jacobian matrix can be computed through a standard backpropagation technique that is much less complex than computing the Hessian matrix. The Levenberg–Marquardt algorithm uses this approximation to the Hessian matrix in the following update:

$$x_{k+1} = x_k - [J^T J + \mu \mathrm{I}]^{-1} J^T e$$

μ is decreased after each successful step (reduction in performance function) and is increased only when a tentative step would increase the performance function. In this way, the performance function will always be reduced at each iteration of the algorithm.

The network architecture that is most commonly used with the backpropagation algorithm is multilayer feed-forward network (Rumelhart *et al.*, 1986). Here, the hidden nodes are arranged in a series of layers. The only permissible connection between nodes lies in consecutive layers. The connection (synaptic) weights are specified for all connections. Biases and transfer functions are proposed for each of the hidden and output nodes.

Implementation and Results

Data Collection

To model the commercial fermentation process, the input parameters and corresponding output characteristics were selected first. There is a total of 16 time sequences (hours of production) for a particular batch of yeast fermentation. The necessary collected information on complete batch operation with selected parameters for analysis is given below.

(1) Time sequence: X_1
(2) Airflow rate at that particular sequence: X_2
(3) Temperature for this interval: X_3
(4) PH of the liquid at the start of the sequence: X_4
(5) Alcohol (%) of the liquid at the start of the sequence: X_5
(6) Residual sugar at the start of the sequence: X_6
(7) Percentage increase of yeast at the end of the time sequence: Y

In order to carry out the analysis in the next step, the dataset was thought to undergo bootstrapping, first keeping the variable X_1 fixed and independent of variable X_2 over the process within a time sequence. The remaining variables X_3, X_4 and X_5 were considered for bootstrapping since these variables create larger deviation and have a significant influence on the response.

Analysis Procedure

To make the model robust, the following analysis procedure was undertaken.

Step 1. The data set was divided into two parts, one for model selection and another for model validation.
Step 2. Next, bootstrap methodology was used on the first partition data set obtained from Step 1 using multisample bootstrapping supported through the group argument (here, group argument is time sequence variable X_1) considering the input parameters X_3, X_4 and X_5. The statistic used for bootstrapping was the median value of these variables.
Step 3. In this stage, a replication on all other parameters X_1, X_2, X_6 including response Y is made to the same strength as the number of resamples generated in Step 2 for the variables X_3, X_4 and X_5.
Step 4. Next, both the data set (i.e. replicate and resamples through bootstrapping) are arranged with respect to time sequence (X_1).
Step 5. The data sets, thus arranged in Step 4, were divided into two sets, one for training purpose and another for testing purpose. Subsequently, the best model was found through comparison of all the techniques as discussed in the previous section.
Step 6. The models, thus developed, were validated on the second partition dataset as defined in Step 1.

Results

As discussed earlier, the techniques used in these comparative studies are simple Linear regression, Generalized linear model, Stepwise regression, Projection Pursuit regression and ANN modeling. A total of 2400 ($= 16 \times 150$) resamples were generated where the number of time sequences for a batch is 16 and the number of bootstrap resamples generated for each time sequence is 150. For each time sequence, out of 150 resamples, 90 resamples were used for training and 60 resamples were kept for testing purposes. Thus, for 16 time sequences, 1440 ($= 16 \times 90$) data were used for training purposes and a 960 ($= 16 \times 60$) data set was kept for testing purposes.

The results observed on the above set of data are given below.

Linear regression model. The simple linear regression model for the data considered is:

$$\begin{aligned} Y = {} & 5.6038 - 1.6899(X_1) + 0.0006(X_2) + 0.6176(X_3) \\ & + 0.2976(X_4) - 24.6754(X_5) + 0.0010(X_6) \end{aligned} \tag{1}$$

The corresponding ANOVA table for the above model is given in Table 1.

The MSE (train data) and Multiple-R^2 found for this model are 1.715 and 0.9587. The MSE for test data and validation data are computed as 1.773 and 2.219 respectively.

Generalized linear model. The generalized linear model (considering the second order polynomial) is

$$\begin{aligned} Y = {} & 3.09\text{E}+02 - 3.73\text{E}+00(X_1) + 3.20\text{E}-02(X_2) - 9.33\text{E}+00(X_3) \\ & - 1.03E+02(X_4) + 1.91\text{E}+02(X_5) + 3.07\text{E}-01(X_6) - 2.20\text{E}-04(X_1 : X_2) \\ & - 5.59\text{E}-02(X_1 : X_3) + 8.07\text{E}-01(X_1 : X_4) + 1.47\text{E}+00(X_1 : X_5) \\ & + 1.48\text{E}-03(X_1 : X_6) + 3.66\text{E}-04(X_2 : X_3) - 8.89\text{E}-03(X_2 : X_4) \\ & - 2.09\text{E}-02(X_2 : X_5) + 6.59\text{E}-06(X_2 : X_6) + 3.24\text{E}+00(X_3 : X_4) \\ & - 1.66\text{E}+01(X_3 : X_5) - 9.79\text{E}-03(X_3 : X_6) + 8.97\text{E}+01(X_4 : X_5) \\ & + 3.27\text{E}-04(X_4 : X_6) - 1.48\text{E}-02(X_5 : X_6) \end{aligned} \tag{2}$$

The Analysis of Deviance table for the above model is given in Table 2.

The MSE for train data, test data and validation data are computed as 1.527, 1.612 and 2.586 respectively.

Table 1. ANOVA table for Response Y

Source	Df	Sum of Sqr	Mean Sqr	F-value	P-value
X_1	1	56529.40	56529.40	32798.68	0.000
X_2	1	138.21	138.21	80.19	0.000
X_3	1	201.87	201.87	117.13	0.000
X_4	1	172.33	172.33	99.98	0.000
X_5	1	229.62	229.62	133.23	0.000
X_6	1	7.11	7.11	4.13	0.0424
Residual	1433	2469.81			
Total	1440	59748.35			

Table 2. Analysis of Deviance (Gaussian Model)

Source	Df	Deviance	F Value	Pr(F)
X_1	1	56529.4	36447.61	0
X_2	1	138.21	89.11	0
X_3	1	201.87	130.16	0
X_4	1	172.33	111.11	0
X_5	1	229.62	148.05	0
X_6	1	7.11	4.59	0.032418
X_1:X_2	1	76.31	49.2	0
X_1:X_3	1	0.09	0.06	0.810827
X_1:X_4	1	86.8	55.96	0
X_1:X_5	1	4.49	2.9	0.088904
X_1:X_6	1	8.75	5.64	0.01766
X_2:X_3	1	13.63	8.79	0.003082
X_2:X_4	1	31.04	20.01	8.3E-06
X_2:X_5	1	12.07	7.78	0.005357
X_2:X_6	1	1.87	1.2	0.272713
X_3:X_4	1	0.52	0.34	0.562255
X_3:X_5	1	0.4	0.26	0.610682
X_3:X_6	1	24.54	15.82	7.32E-05
X_4:X_5	1	9.72	6.27	0.012403
X_4:X_6	1	0.03	0.02	0.893318
X_5:X_6	1	0.27	0.18	0.675189
Residual	1418	2199.285		
Total	1439	59748.35		

(Dispersion Parameter for Gaussian family: 1.550977).

Stepwise regression. The best-fitted second order linear model found through stepwise regression analysis is

$$\begin{aligned} Y = {} & 2.27\text{E}+02 - 5.20\text{E}+00(X_1) + 3.82\text{E}-02(X_2) - 6.66\text{E}+00(X_3) \\ & - 9.60\text{E}+01(X_4) + 4.40\text{E}+02(X_5) + 3.19\text{E}-01(X_6) - 1.77\text{E}-04(X_1 : X_2) \\ & + 7.67\text{E}-01(X_1 : X_4) + 1.10\text{E}-03(X_1 : X_6) - 8.40\text{E}-03(X_2 : X_4) \\ & + 9.25\text{E}-06(X_2 : X_6) + 3.03\text{E}+00(X_3 : X_4) - 2.47\text{E}+01(X_3 : X_5) \\ & - 1.03\text{E}-02(X_3 : X_6) + 8.40\text{E}+01(X_4 : X_5) \end{aligned} \tag{3}$$

The corresponding ANOVA table for the above model is given in Table 3.

The MSE (train data) and Multiple-R^2 found for this model are 1.531 and 0.9631. The MSE for test data and validation data are computed as 1.587 and 2.652 respectively.

Projection pursuit regression. The plot of $e^2(m)$ versus m is drawn with respect to train MSE and test MSE to find the optimum value of m. The graph is shown in Figure 2.

From the above plot, the value of m has been selected as 9. The estimates for β_m and a_m are obtained next. The minimum MSE for this model (where, $m = 9$) is found as 1.047. The MSE for test data and validation data are computed as 1.175 and 1.882 respectively. The values of MSE versus m, for train data, test data and validation data are tabulated in Table 4. The estimated values of β_m and a_m are given in Table 5.

Backpropagation network. The ranges of the variables used in the present work are listed in Table 6.

Each variable X_i is normalized within the range of 0 to 1 for ANN modeling by the 'minimax normalization' technique given below and used in the same form for other

Table 3. ANOVA table for response Y

Source	Df	Sum of Sq	Mean Sq	F Value	Pr(F)
X_1	1	56529.4	56529.4	36522.85	0
X_2	1	138.21	138.21	89.29	0
X_3	1	201.87	201.87	130.43	0
X_4	1	172.33	172.33	111.34	0
X_5	1	229.62	229.62	148.35	0
X_6	1	7.11	7.11	4.59	0.032239
X_1:X_2	1	76.31	76.31	49.3	0
X_1:X_4	1	52.26	52.26	33.77	0
X_1:X_6	1	16.44	16.44	10.62	0.001144
X_2:X_4	1	44.05	44.05	28.46	1E-07
X_2:X_6	1	1.48	1.48	0.96	0.328419
X_3:X_4	1	18.65	18.65	12.05	0.000534
X_3:X_5	1	9.15	9.15	5.91	0.015139
X_3:X_6	1	33.3	33.3	21.51	3.8E-06
X_4:X_5	1	14.14	14.14	9.13	0.002555
Residual	1424	2204.04			
Total	1439	59748.35			

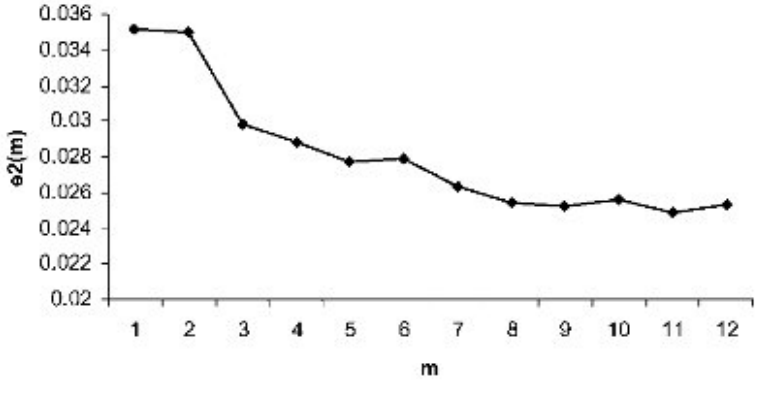

Figure 2. Plot of $e^2(m)$ versus m

Table 4. Maximum number of terms to choose for the model ($M = 12$)

Min. no. of terms to include in the model (m)	Unexplained Variance [$e^2(m)$]	Train Data (MSE)	Test Data (MSE)	Validation Data (MSE)
1	0.0352	1.461	1.433	2.360
2	0.0350	1.454	1.455	2.560
3	0.0298	1.237	1.251	2.697
4	0.0288	1.195	1.227	2.914
5	0.0277	1.148	1.221	2.669
6	0.0279	1.157	1.245	2.464
7	0.0263	1.093	1.179	2.241
8	0.0254	1.052	1.186	2.037
9	**0.0252**	**1.047**	**1.175**	**1.882**
10	0.0256	1.061	1.174	1.878
11	0.0249	1.031	1.167	2.016
12	0.0253	1.052	1.174	1.840

Table 5. List of direction vectors and weight estimates ($m = 1$ to 9)

m	Direction vectors	X_1	X_2	X_3	X_4	X_5	X_6	β_m
[1]	A_1	−0.624697	0.000141	0.302474	0.535657	−0.480971	0.000547	6.4943012
[2]	A_2	0.119828	0.000087	−0.551978	−0.540756	−0.623330	0.001985	0.5557497
[3]	A_3	0.278629	−0.002695	0.035853	0.015936	−0.959583	0.004484	0.6905427
[4]	A_4	0.033914	0.000243	−0.124856	−0.575697	−0.807362	−0.000121	0.4715563
[5]	A_5	−0.042924	0.001303	−0.539979	−0.481792	−0.688807	−0.000377	0.6685696
[6]	A_6	−0.008782	−0.000169	0.109180	−0.377303	−0.919589	0.000951	0.6160745
[7]	A_7	0.077260	−0.001288	0.230818	0.267690	−0.932252	0.000471	0.4273261
[8]	A_8	0.036572	0.000394	−0.056819	0.201547	−0.977143	−0.001928	0.8037670
[9]	a_9	−0.014487	−0.000224	0.192671	−0.977099	−0.089114	0.002111	0.5500938

Table 6. Summary statistics of variables

Sl. No.	Variable description	Coded variable	Unit	Minimum (X_{min})	Maximum (X_{max})
1	Time sequence	X_1	–	1	16
2	Airflow rate	X_2	CFM	2000	4000
3	Temperature	X_3	^{0}C	30.750	36.420
4	pH	X_4	–	4.040	5.235
5	Alcohol	X_5	%	0.087	0.190
6	Residual Sugar	X_6	Kg.	97.00	1069.00
7	% Increase of yeast	Y	%	1.64	24.710

statistical techniques as well.

$$X_N = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{4}$$

where X_N is the normalized value of the variable X_i, X is the actual value and X_{max} and X_{min} are the maximum and the minimum values of X_i, respectively.

The architecture of ANN used here is a multilayered feed-forward network, trained with both the supervised SCG and LM algorithms. The six input variables, namely, X_1, X_2, X_3, X_4, X_5, X_6, are defined as input neurons, and the percentage growth (Y) of yeast is considered as an output variable. In order to obtain the optimum network, several architectures with single hidden layer (6-N_1-1) and double hidden (6-N_1 – N_2-1) layers, N_1 and N_2 being the number of neurons in the hidden layers, are designed, trained and tested with training data and test data respectively. The tan-sigmoid transfer function is used in the hidden layers while purelin transfer function is used at the output layer. The learning parameter and the momentum parameter are kept fixed at their default values. It is clear during training that both the algorithms have reasonably good capability to train the network, but the network trained by the LM algorithm has slightly better prediction ability. The best network architecture from a total of 18 networks is selected as (6-14-10-1) on the basis of combined performance of MSE, both for train data and test data simultaneously. The 'train MSE' and 'test MSE' with respect to this optimum architecture are found as 0.8122 and 0.8321 respectively. This architecture, when validated on a new set of data, performs with a MSE level of 1.088. A sample plot of 'train mse' performance, is shown in the Figure 3. The parameters (weights and biases) of the optimum architecture are listed in Tables 7(a), 7(b), 7(c) and 8 respectively.

Discussion

It is shown in an earlier section that an attempt has been made to model the growth of yeast over hours of production with as much accuracy as feasible. The comparative study (Table 9), based on various statistical techniques for modeling and also with neural network approach, which is a distribution-free topology, reveals that the backpropagation neural network, specifically with LM training algorithm, exhibits the better prediction capability. The comparative performance of the techniques used is done based on the commonly used characteristic, i.e. mse.

The ANN model gives an accurate relationship among incremental growth of yeast and the contribution of process parameters under the existing infrastructure. The shop-floor people can look for necessary corrective measures through the addition of residual sugar (i.e. the addition of wort/molasses), air circulation, chemical addition etc to achieve the desired growth of yeast by adopting the ANN model. This model would also help to find

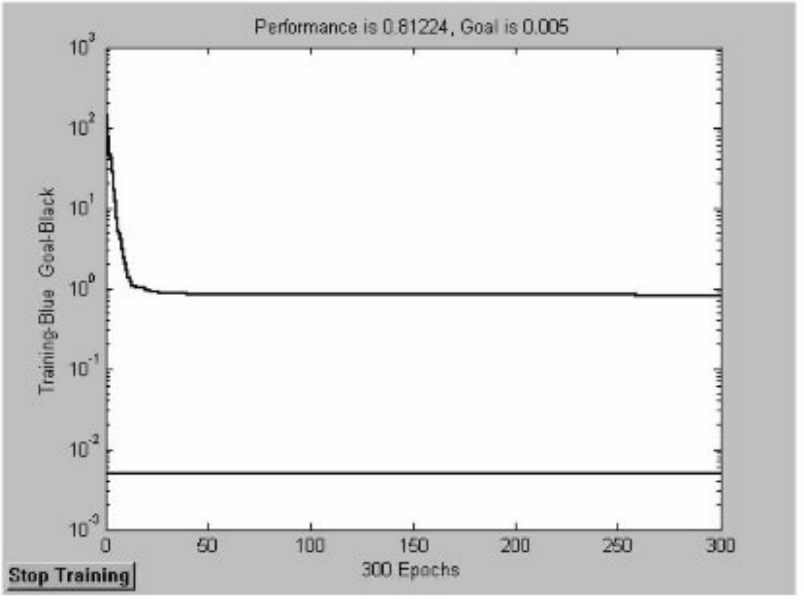

Figure 3. Training performance (6-14-10-1) using LM algorithm

Table 7. (a) Estimated layer weights (6-14-10-1 architecture) (Input layer to 1st Hidden Layer: 6×14 weigh estimates)

			Input Layer					
			1 X_1	2 X_2	3 X_3	4 X_4	5 X_5	6 X_6
1st Hidden Layer	1	N_{11}	0.0161	−0.4403	−0.1949	0.3160	9.9284	−0.0733
	2	N_{12}	0.3087	0.0107	−0.4134	−0.4081	14.4407	−0.0179
	3	N_{13}	3.0737	−0.0071	0.2172	−0.5412	13.6462	−0.0100
	4	N_{14}	−3.0684	0.0224	−1.1623	−2.1539	24.1122	−0.0051
	5	N_{15}	−2.2602	0.0092	−0.5111	4.8289	3.2817	0.0004
	6	N_{16}	−1.5514	0.0192	−0.0433	−0.4035	−0.6908	−0.0878
	7	N_{17}	2.4654	−0.0124	−0.1410	−0.6048	−3.0233	0.0085
	8	N_{18}	−4.2006	0.0083	0.1653	0.2737	11.7613	0.0663
	9	N_{19}	−4.4405	0.0592	−2.2176	−3.2615	−18.4598	-0.0467
	10	$N_{1,10}$	−0.3788	0.0073	0.0150	0.1155	1.7599	−0.0187
	11	$N_{1,11}$	−0.4707	0.1149	−2.7279	−4.0309	−15.5528	−0.4139
	12	$N_{1,12}$	−0.6100	0.1389	1.7794	2.6152	−15.0296	−0.0159
	13	$N_{1,13}$	4.3891	−0.0196	−0.0445	0.7051	−11.1431	0.0338
	14	$N_{1,14}$	3.9308	−0.0284	−0.8342	−1.8448	18.7700	0.2355

an optimum combination of parameters for minimizing the variability of yeast production around its average level of production. This would ease the following planning activities at strategic, tactical and operational levels respectively to the management.

- planning of the requirements of molasses, the major and costliest raw material for yeast production, while adding in commercial phase of fermentation;
- scheduling the post processes for dry yeast production in an effective way; and
- long-term inventory planning through accurate estimation of productivity.

Table 7. (b) Estimates of layer weights (6-14-10-1 architecture) (1st Hidden Layer to 2nd Hidden Layer: 14×10 weigh estimates)

		2nd Hidden Layer									
		1 N_{21}	2 N_{22}	3 N_{23}	4 N_{24}	5 N_{25}	6 N_{26}	7 N_{27}	8 N_{28}	9 N_{29}	10 $N_{2,10}$
1st Hidden Layer	N_{11}	−1.7730	1.1827	−2.6408	−1.6259	2.1408	−2.3020	−0.9617	−0.6096	−1.1665	−0.1661
	N_{12}	−2.2074	−0.1652	1.8238	2.1665	−0.8718	−0.9353	−0.8124	−0.0168	2.5769	−0.1061
	N_{13}	−0.0566	2.0458	−0.0937	−0.1904	0.0703	−3.3095	−3.3476	−1.9355	−1.1129	0.1595
	N_{14}	−1.0052	0.8783	−0.3360	1.2638	1.0252	0.2084	3.6583	1.1694	−1.1228	−3.3614
	N_{15}	−4.0598	0.0078	4.3316	3.4302	0.0882	0.2857	2.2372	2.7073	2.0860	5.6749
	N_{16}	−0.4334	−0.9257	−2.5694	0.6502	−0.2154	−1.5372	−2.1552	2.5125	1.5950	4.2979
	N_{17}	−1.7532	−0.0558	−6.3908	−0.9917	0.4632	−1.8161	0.3118	−2.0565	0.4667	1.7495
	N_{18}	−0.3013	−1.5068	−3.7066	−8.3049	4.1703	−1.6502	−2.4554	−1.8761	1.4908	0.5241
	N_{19}	0.4390	0.0145	−0.2012	1.0027	5.5353	0.1106	−1.1497	−0.9206	3.7640	0.3338
	$N_{1,10}$	0.1425	0.0463	6.8955	−5.5041	3.0353	−1.8027	−0.5419	−6.9686	0.2195	−6.2542
	$N_{1,11}$	0.5668	0.5918	1.0127	2.8113	−0.4921	4.2171	−2.1523	−2.0875	0.8241	1.8550
	$N_{1,12}$	0.7405	−0.3555	1.7076	0.7216	−1.3994	1.1292	0.7906	1.0997	1.2513	1.1029
	$N_{1,13}$	0.8810	2.0427	4.4021	0.2131	−3.1736	2.4230	−4.9481	−0.8086	−2.6820	−3.3038
	$N_{1,14}$	−5.5617	0.6463	0.1540	−3.2563	3.2555	1.3425	−3.5465	−2.6568	0.6489	0.0360

Table 7. (c) Estimates of layer weights (6-14-10-1 architecture) (2nd Hidden Layer to Output Layer: 10×1 weight estimates)

	2nd Hidden Layer									
	N_{21}	N_{22}	N_{23}	N_{24}	N_{25}	N_{26}	N_{27}	N_{28}	N_{29}	$N_{2,10}$
Output Layer	1	2	3	4	5	6	7	8	9	10
Y	3.3093	−1.4984	0.5416	0.2515	1.3638	2.2483	1.4231	2.4565	1.94	3.1634

Table 8. Bias estimates

Bias Values		
1st Hidden Layer	2nd Hidden Layer	Outpur Layer
16.2578	2.9046	4.3649
−3.491	−1.8661	
−16.6832	3.5901	
13.4649	1.7119	
−16.511	−2.2469	
12.1356	1.4557	
20.6148	−0.2378	
−21.0586	−0.7141	
16.7883	2.9524	
−11.8416	2.1367	
20.4581		
−6.2372		
−6.5882		
4.0316		

Table 9. Comparative Performance of MSEs

Techniques	Train Data	Test Data	Validation Data
Linear regression	1.715	1.773	2.219
Generalized Linear Model	1.527	1.612	2.586
Stepwise regression	1.531	1.587	2.652
Projection Pursuit Regression	1.047	1.175	1.882
Backpropagation (NN)	0.8122	0.8321	1.088

Conclusions

This paper has shown a robust model with higher predictive accuracy. The following conclusions, both specific and in general, are drawn from this work.

- Among the statistical methods the Projection Pursuit Regression model is found to have a better capacity for predicting the growth pattern of yeast.
- The feedforward neural network with supervised learning exhibits much better prediction capability than the statistical methods. This is due to the ability of ANN to tackle system non-linearity.
- Among the neural networks, the network using the LM algorithm for error optimization was found to be superior to the other most common (SCG) algorithm.

- The approach can easily be converted for the purpose of any other complicated process modeling that involves large numbers of independent variables, including more than one output (response) variable.
- ANN can be applied effectively and efficiently with a significant prediction capability in the non-manufacturing business sector, since it does not make any assumption on the distribution and the properties of the data. However, situations with noisy data would definitely create problems in building a 'good' model through this approach.

References

Friedman, J. H. & Stuetzle, W. (1981) Projection pursuit regression, *Journal of the American Statistical Association*, 76, pp. 817–823.

Hastie, T., Tibshirani, R. & Friedman, J. (2001) *Elements of Statistical Learning* (New York: Springer-Verlag).

Lennox, B., Hiden, H. G., Montague, G. A., Kornfeld, G. & Goulding, P. R. (2001) Process monitoring of an industrial fed-batch fermentation, *Biotechnology and Bioengineering*, 74(2), pp. 125–135.

Lennox, B., Montague, G. A., Hiden, H. G. & Kornfeld, G. (1999) Case study investigating multivariate statistical techniques for fermentation supervision, *Computers and Chemical Engineering*, 23S, pp. 827–830.

Marquardt, D. (1963) An algorithm for least-squares estimation of nonlinear parameters, *SIAM Journal of Applied Mathematics*, 11(2), pp. 431–441.

Martinez, W. L. & Martinez, A. R. (2002) *Computational Statistics Handbook with MATLAB* (Chapman & Hall, CRC Press).

Moller, M. F. (1993) A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks*, 6, pp. 525–533.

Myers R. H., Montgomery D. C. & Vining G. G. (2002) *Generalized Linear Models* (New York: Wiley).

Ronald R. H. (1996) *Methods and Application of Linear models* (New York: Wiley).

Rumelhart, D., Hinton, G. E. & Williams, R. J. (1986) Learning representation by BP errors, *Nature*, 323, pp. 533–536.

S-PLUS 2000 User's Guide (1999) Data Analysis Products Division, MathSoft, Seattle, WA.

Widrow, B. & Hoff, M. E. (1960) *Adaptive Switching Circuits*, WESCON Convention Record: Part 4, pp. 96–104 (New York: IRE).