Improving Microbial Growth Prediction by Product Unit Neural Networks

CESAR HERVÁS-MARTÍNEZ, ROSA M. GARCÍA-GIMENO, ALFONSO C. MARTÍNEZ-ESTUDILLO, FRANCISCO J. MARTÍNEZ-ESTUDILLO, AND GONZALO ZURERA-COSANO

ABSTRACT: This article presents a new approach to the Artificial Neural Networks (ANN) modeling of bacterial growth; using Neural Network models based on Product Units (PUNN) instead of on sigmoidal units (multilayer perceptron type [MLP]) of kinetic parameters (lag-time, growth rate, and maximum population density) of *Leuconostoc mesenteroides* and those factors affecting their growth such as storage temperature, pH, NaCl, and $NaNO_2$ concentrations under anaerobic conditions. To enable the best degree of interpretability, a series of simple rules to simplify the expression of the model were set up. The new model PUNN was compared with Response Surface (RS) and MLP estimations developed previously. Standard Estimation Error of generalization (SEP_G,) values obtained by PUNN were lower for lag-time and growth rate but higher for maximum population density than MLP when validated against a new data set. In all cases, bias factors (B_f) and accuracy factors (A_f) were close to unity, which indicates a good fit between the observations and predictions for the 3 models. In our study, PUNN and MLP models were more complex than the RS models, especially in the case of the growth rate parameter, but they described lower SEP_G. With this work we have attempted to propose a new approach to neural networks estimations for its application on predictive microbiology, searching for models with easier interpretation and with a great ability to fit the data on the boundaries of variables range. We consider that still there is a lot left to do but PUNN could be a very valuable instrument for mathematical modeling.

Keywords: artificial neural networks, product units, growth model, *Leuconostoc mesenteroides*, spoilage bacteria

Introduction

Factors that most affect microorganism growth are, among others, pH, storage temperature, water activity, preservatives, and the modification of the atmosphere during packaging (Gibson and others 1988). Given an adequate database, the response of many microbes in food could be predicted from knowledge of the food's formulation, processing, and storage conditions and afterward can be applied in food product development and food safety risk assessment. There is a growing interest in microbial growth modeling as an alternative to time-consuming, traditional, microbiological enumeration techniques.

The models most frequently used for this purpose are usually polynomial regressions (such as Response Surface, RS), which provides a great simplicity and availability of user-friendly software. In this type of model, the coefficients of the equation can be estimated by least square or through local search algorithms based on the gradient, such as that of Levenberg-Marquardt (Levenberg 1944; Marquardt 1963). Moreover, RS requires the model order to be specified a priori (that is, if it is of 1st, 2nd, or 3rd order). They also need initial values for the coefficients of the model, so that the local search algorithm can obtain the model in the most efficient way. More recently, new models have been introduced, some involving the application of Artificial Neural Networks (ANN) of the multilayer perceptron type (MLP). These have amply demonstrated their capacity for predicting the parameters associated with microbial growth (Hajmeer and others 1997; Geeraerd and others 1998; Hervás and others 2001; Jeyamkondan and others 2001; Lou and Nakai 2001; García-Gimeno and others 2002, 2003, 2005).

This article presents a new approach for the ANN modeling of bacterial growth using Neural Network models based on Product Units (PUNN) instead of on sigmoidal units (MLP) like the one described previously. The use of this new type of neuronal network using product base functions is justified in the search for more easily interpretable models without decreasing either the prediction capacity or the robustness of models. On the other hand, this type of model can be proposed in predictive microbiology because it is logical to suppose a priori that a strong interaction exists between the factors that affect the microbe growth parameters. Thus the use of PUNN has 2 major advantages: these product units are more effective in picking up the interactions between the factors and they are easier to interpret than MLP. In contrast to the usual black-box model or neural network based on sigmoidal functions, we can consider these networks as being "gray-box" models. The product units have the ability of implementing higher-order functions and therefore they can also implement polynomial functions as a particular case (Gurney 1992). Moreover, PUNN models approaches more easily that MLP to complex decision making because its potential function is not so smoothing.

Despite these advantages, PUNNs have a major drawback. Their training is more difficult than that of standard sigmoidal-based networks using a back-propagation local algorithm (Durbin and Rumelhart 1989) because small changes in the exponents can cause great changes in the total error. The main reason for this difficulty is that the PUNN tend to more local minima and plateaus (Ismail and Engelbrecht 2000). It is a well known issue that back-propagation is not efficient in training product units. Several efforts have been made to develop learning methods for PUNN (Janson and Frenzel 1993; Ismail

MS 20050267 Submitted 5/10/04, Revised 7/11/04, Accepted 8/4/05. *Author Hervás-Martínez is with Dept. of Computing, Univ. of Córdoba, Campus Univ. de Rabanales, 14014 Córdoba, Spain. Authors García-Gimeno and Zurera-Cosano are with Dept. of Food Science and Technology. Univ. of Córdoba, Campus Univ. de Rabanales, Córdoba, Spain. Authors Martínez-Estudillo and Martínez-Estudillo are with ETEA, Dept. of Applied Mathematics, Univ. of Cordoba, Córdoba, Spain. Direct inquiries to author Hervás-Martínez (E-mail: chervas@uco.es).*

Table 1—Average of observed (OBS) and estimated growth rate (Gr [/h]), lag time (Lag [h]), and maximum population density (yEnd [OD]* by product unit neural networks (PUNN) of *Leuconostoc mesenteroides* for model development[a]

				Gr (/h)		Lag (h)		yEnd (OD)	
T (°C)	pH	NaCl (%)	$NaNO_2$ (ppm)	OBS	PUNN	OBS	PUNN	OBS	PUNN
10.5	6.5	3.25	100	0.106	0.102	16.919	16.583	0.382	0.354
14	6	1.75	50	0.161	0.164	7.535	7.087	0.706	0.733
14	6	1.75	150	0.149	0.160	7.855	8.428	0.286	0.308
14	6	4.75	50	0.139	0.130	12.446	13.802	0.297	0.313
14	6	4.75	150	0.120	0.123	21.743	20.868	0.094	0.083
14	7	1.75	50	0.180	0.181	6.571	6.631	0.978	0.930
14	7	1.75	150	0.168	0.177	7.854	7.603	0.487	0.451
14	7	4.75	50	0.142	0.140	12.817	10.929	0.617	0.636
14	7	4.75	150	0.130	0.133	11.168	11.220	0.305	0.317
17.5	5.5	3.25	100	0.103	0.102	12.914	10.543	0.544	0.451
17.5	7.5	3.25	100	0.169	0.164	6.122	5.479	0.824	0.832
17.5	6.5	3.25	0	0.191	0.201	5.335	5.361	1.028	0.932
17.5	6.5	3.25	200	0.157	0.161	9.419	9.273	0.233	0.220
17.5*	6.5	3.25	100	0.172	0.172	6.475	6.663	0.548	0.576
17.5*	6.5	3.25	100	0.172	0.172	6.602	6.663	0.529	0.576
17.5*	6.5	3.25	100	0.170	0.172	6.356	6.663	0.539	0.576
17.5*	6.5	3.25	100	0.176	0.172	6.498	6.663	0.537	0.576
17.5*	6.5	3.25	100	0.178	0.172	6.679	6.663	0.536	0.576
17.5*	6.5	3.25	100	0.167	0.172	6.063	6.663	0.542	0.242
17.5	6.5	6.25	100	0.141	0.142	14.864	6.663	0.269	0.644
17.5	6.5	0.25	100	0.363	0.352	3.589	6.663	0.632	0.794
21	6	1.75	50	0.336	0.338	3.793	14.720	0.783	0.334
21	6	1.75	150	0.312	0.317	4.259	3.670	0.366	0.397
21	6	4.75	50	0.323	0.309	9.088	3.863	0.371	0.120
21	6	4.75	150	0.269	0.274	12.648	4.598	0.129	0.955
21	7	1.75	50	0.363	0.360	3.630	8.641	1.049	0.634
21	7	1.75	150	0.337	0.339	4.272	13.122	0.634	0.716
21	7	4.75	50	0.313	0.323	5.880	3.419	0.696	0.373
21	7	4.75	150	0.296	0.288	5.301	3.864	0.367	0.504
24.5	6.5	3.25	100	0.409	0.411	3.658	5.924	0.480	0.354

* OD = Optical density.
[a]* = Center point conditions.

and Leerink and others 1995; Engelbrecht 1999, 2000). Martinez-Estudillo and others (2005) proposed a model of evolutionary computation of PUNN to overcome this difficulty that evolves both the weights and the structure of these networks by using an algorithm based on evolutionary programming.

This article presents a new approach to the ANN modeling of bacterial growth; using Neural Network models based on Product Units (PUNN) instead of on sigmoidal units (MLP) of kinetic parameters (lag-time, growth rate and maximum population density) of *Leuconostoc mesenteroides* and those factors affecting their growth such as storage temperature, pH, NaCl, and $NaNO_2$ concentrations under anaerobic conditions.

Materials and Methods

Experimental data

The specific growth rate (Gr), lag-time (Lag), and maximum population density (yEnd) data of *L. mesenteroides* subsp. *mesenteroides* ATCC 8293 (Spanish Collection of Strain Types, Valencia) were taken from Zurera-Cosano and others (2005).

A Central Composite Design (CCD) was used, incorporating the following variables and levels: temperature (10.5 °C, 14 °C, 17.5 °C, 21 °C, and 24.5 °C), pH (5.5, 6, 6.5, 7, and 7.5), concentrations of sodium chloride (0.25%, 1.75%, 3.25%, 4.75%, and 6.25%) and concentrations of sodium nitrite (0, 50, 100, 150, and 200 ppm) under anaerobic conditions shown in Table 1. Each of the 25 different factor combinations thus obtained was replicated 7 times, and 6 center point replications were performed to estimate experimental variance.

Models development

PUNN is a powerful basis for its application in modeling, and we will try to explain how these models are carried out. To start processing data, we avoided saturation problems in the product basis functions by preventing the driving of the weights to infinity and by improving the learning process. Each of the input and output variables should be scaled in the rank [0.1, 1.1] and [1, 2] respectively. The new scaled variables are named t*, p*, c* and n*, for the input variables and l*, g* and y* for the output variables. For example, T and l is calculated as follows:

$$t^* = \frac{T - T_{min}}{T_{max} - T_{min}} + 0.1 \quad (1)$$

$$l^* = \frac{l - l_{min}}{l_{max} - l_{min}} + 1 \quad (2)$$

where T and l are the original temperature and Lnlag respectively. T_{min}, l_{min}, T_{max}, and l_{max} are the minimum and maximum values, and t*, l* are the scaled temperature and Lnlag. Once obtained, model estimations should be de-scaled following the same equation.

We begin by defining the family of functions to be used in the modeling process and their representation through the corresponding PUNN model. The general mathematical description of a family with this type of function is as follows:

Let Y^k be a n dimensional Euclidean space and K a compact subset of it defined by $K = \{(x_1, x_2, \ldots, x_k) \in \mathbb{h}^k : x_i \in \mathbb{h}^+, i = 1, 2, \ldots, k\}$. We represent by $F(K)$ the family of functions $f : K \subset \mathbb{h}^k \to \mathbb{h}$ given by

$$f(x_1, x_2, \ldots, x_k) = \sum_{j=1}^{p} \beta_j \left(\prod_{i=1}^{k} x_i^{w_{ji}} \right) \quad (3)$$

where β_j, $w_{ji} \in \hat{\mathbb{h}}$, with $w_{ji} \geq 0$ and $p, k, \in i$.

This typology of functions can be viewed as a polynomial with real exponents, and by appropriately choosing the exponents of the function f, it is easy to observe that the polynomial regression models are subsets of $F(K)$. For example, by the appropriate selection of the exponents, $w_{ji} \in \{0, 1, 2\}$ a 2nd-order polynomial regression model or quadratic response surface can be obtained:

$$f(x) = \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_{ii} x_i^2 + \sum_{i=1,\, i<j}^{k} \beta_{ij} x_i x_j \quad (4)$$

Let now the data set be $D_E \{x_l, y_l\}$, for $l = 1, 2, \ldots, n$ for which the regression model can be expressed by means of a lineal potential base function topology or PUNNs as Eq. 1. In these models, the product units can be defined as follows:

$$\prod_{i=1}^{k} x_i^{w_{ji}} \quad (5)$$

where $w_j = (w_{j1}, w_{j2}, \ldots, w_{jk})$ is a parameter set for the potential base functions.

This kind of function topology can be represented by a neural network architecture, as shown in Figure 1, with the following features: 1 input layer for the input variables, 1 hidden layer with a suitable number of nodes, and 1 output layer, expressed as: "n° of input neurons":"n° of hidden neurons":"n° of output neurons." Furthermore, the nodes of one layer cannot be connected with each other and there are no direct connections between the input and output layers. In the microbial growth model addressed in this study, the independent variables $(x_1, x_2, \ldots, x_k)$ are the 4 environmental conditions considered, the P nodes in the hidden layer represent the term numbers of the model and therefore the number of product units considered, and the node in the output layer corresponds to the microbial kinetics parameters, the *Lag* or *Gr* or *yEnd*

The transfer function of the j-th node of the hidden layer is given by Eq. 3, in which $w_{ji} \in [0, L]$ is the weight for the connection between the i-th node of the input layer and the j-th ones of the hidden layer. The linear transfer function of the node of the output layer is given by Eq. 1, in which $\beta_{ji} \in [-M, M]$ is the weight for the connection between the j-th node of the hidden layer and the node of the output layer. In summary, the topology for the functions defined in Eq. 1 can readily be represented by a PUNN model.

Evolutionary algorithm

The general structure of the evolutionary algorithm, which is applied to an initial population of Np individuals, can be supported in the following steps:

1. Generate initial population with randomly generated networks.
2. Evaluate the fitness score for each individual of the population on the basis of the objective function.
3. Copy the best individual to the next generation.
4. The best 10% of a population substitutes the worst 10% of individuals.
5. Apply parametric mutation operators to the best 10% of the population.
6. Apply structural parametric mutation to the rest of the population.

These steps should be repeated until the population converges or a previous fitted number of generation is reached.

The evolution of product-unit networks uses the operations of replication and 2 types of mutation: parametric and structural. Parametric mutation alters the values of the exponents and coefficients of the functions of the population and structural mutation alters the architecture if the net (connections and nodes). Parametric mutations are applied to each parameter w_{ji} and b_j of a function f with gaussian noise (a normal random variable is added to the weights), where the variance of normal distribution depends on the function's T. The severity of a mutation to an individual f is dictated by the adaptive function $T(f)$ given by:

$$T(f) = 1 - A(f) \quad 0 \geq T(f) \leq 1 \quad (6)$$

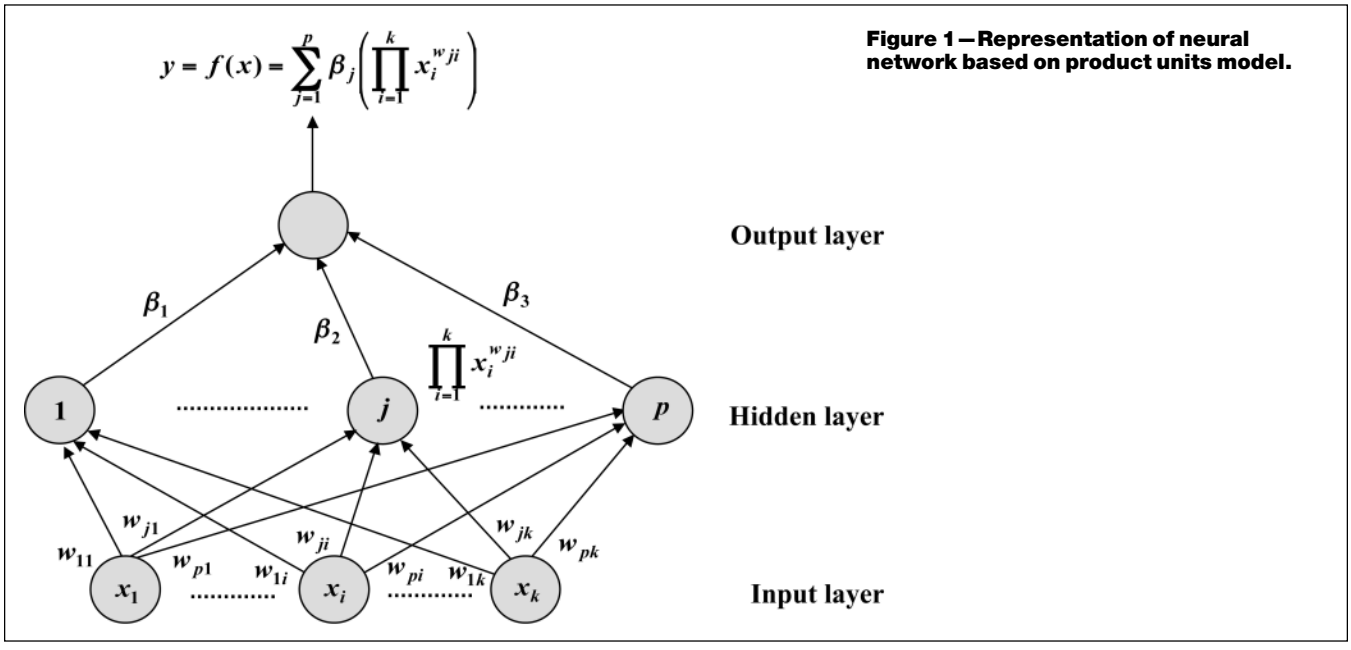

Figure 1—Representation of neural network based on product units model.

Table 2—parametric values used by the evolutionary algorithm for product unit neural networks for the estimation of *Leuconostoc mesenteroides*

Population parameters		Structural mutation parameters: interval [D_m, D_M]		Parametric mutation Parameters of Eq. 16	
Size, N_P	1000	Add nodes	[1, 2]	$a_1(0)$	1
Maximum nr of hidden nodes, r	8	Delete nodes	[1, 2]	$a_2(0)$	5
Nr of independent variables, k	4	Add connections	[1, 6]	b	0.5
Exponent interval, [$-M$, M]	[0, 3]	Delete connections	[1, 6]	r	10
Coefficient interval, [0, L]	[–5, 5]				

where $A(f)$ is the fitness function. Thus, the adaptive function $T(f)$ is determined by how close the function is to the solution to the problem. So networks with a high adaptive function are mutated severely, and those with a low adaptive function only slightly. This allows a coarse-grained search initially, and progressively finer-grained ones as the network approaches the solution of the problem. More details on parametric mutation are shown in Appendix A.

Structural mutation is more complex because it implies a modification of the structure of the network. There are 5 different structural mutations:

(1) Addition of a node: The node is added with no connections to others layers to enforce the behavioral link with its parents. (2) Deletion of a node: A node is selected randomly and deleted together with its connections. (3) Addition of a connection: A connection is added, with weight 0, to a randomly selected node. There are 2 types of connections: from an input node to a hidden node and from a hidden node to the output node. (4) Deletion of a connection: A connection is selected and removed. (5) Joint node: 2 hidden nodes a and b selected randomly were replaced by another node c.

All the previously mentioned mutations are made sequentially in the same generation on the same network. For each mutation there is a minimum value, Δ_m and a maximum value Δ_M and the number of elements (nodes and connections) involved in the mutation is calculated as follows:

$$P(\Delta A) = \exp\left(-\frac{\Delta A}{T}\right) \tag{7}$$

where U(0, 1) is an uniform distribution un the interval [0, 1]

Finally, the system evolves until the average fitness of the network population stops growing; that is, if during 20 generations there is no improvement in the average performance of the best 20% of the population, or until a number when functions $\alpha_1(t)$ and $\alpha_2(t)$ are near zero, or when a number of generations decided a priori is reached (3000).

The values of parameters used by the evolutionary algorithm for PUNN are shown in Table 2. It should be pointed out that the algorithm is quite robust to the modification of these parameters.

To evaluate the fitting and prediction accuracy of each model, Root-Mean-Squares Error (RMSE) and Standard Error of Prediction percentage (% SEP) were used:

$$\%\text{SEP} = 100 \times \frac{\sqrt{\frac{\sum_{i=1}^{n}(g_i - \hat{g}_i)^2}{n}}}{|\overline{g}|} \tag{8}$$

and

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(g_i - \hat{g}_i)^2}{n}} \tag{9}$$

where g_i is the value of the growth parameter observed; $\hat{g}_i$ is the predicted value obtained with our PUNN model, and $\overline{g}$ is the mean of observed values.

The Standard Error of Prediction percentage (% SEP), is a relatively typical deviation of the mean prediction values and has the advantage, compared with other error measurements, of not being dependent on the magnitude of the measurements while it can be used to compare the error of the different growth parameters for different ranges and scales.

To increase the interpretability of the PUNN models found, a series of rules are set up to simplify the number of addends in the original model for certain domains of the input variables (Setieno and others 2002). Through these rules we will try to get simpler models (in certain subregions of the definition domain of the factors used as net input) that are therefore more easily interpretable without losing their generalization capacity. Rules for function approximation normally take the following form: if (a condition or restriction in the input variables, x, is satisfied), then the output predicts y = f(x), where f(x) is a constant or a linear or nonlinear simple function of x. This kind of rule is acceptable if we take into account its similarity to nonlinear classification and statistical regression methods. Thus, in this study we use heuristic rules in our best PUNN models to quantify the growth parameters as a function of the environmental factors used.

The new model PUNN will be compared with RS (Zurera-Cosano and others 2005) and MLP (García-Gimeno and others 2005) estimations developed previously.

Model validation

The models were tested against a growth data set obtained under the same experimental conditions (30% of the total data set), but not included in the development of the model (internal validation, test, or generalization). They were also contrasted with a new data set obtained under different experimental conditions, but included in the range of experimental design (external validation) (Table 3), which would be the equivalent of what has been called by other predictive microbiology authors "mathematical validation" (Van Impe and others 1998). To evaluate the predictive capacity of the proposed model, the aforementioned error criteria, RMSE and SEP (%), were calculated together with bias factors (B_f) and accuracy factors (A_f) (Ross 1996).

$$B_f = 10^{\left[\frac{\sum_{i=1}^{n}\log\left(\frac{\hat{g}_i}{g_i}\right)}{n}\right]} \tag{10}$$

$$A_f = 10^{\left[\frac{\sum_{i=1}^{n}\left|\log\left(\frac{\hat{g}_i}{g_i}\right)\right|}{n}\right]} \tag{11}$$

where g_i is the observed i-th value and $\hat{g}_i$ is the predicted i-th value.

Table 3—Average of observed (OBS) and estimated growth rate (Gr [/h]), Lag time (Lag [h]), and maximum population density (yEnd [OD]* by Product Unit Neural Networks (PUNN) of *Leuconostoc mesenteroides* for external validation in anaerobic conditions

				Gr (/h)		Lag (h)		yEnd (OD)	
T (°C)	pH	NaCl (%)	$NaNO_2$ (ppm)	OBS	PUNN	OBS	PUNN	OBS	PUNN
10.5	6.5	3.25	50	0.112	0.102	11.01	14.65	0.562	0.508
10.5	6.5	3.25	100	0.106	0.102	15.06	16.58	0.385	0.354
14	7	1.75	0	0.214	0.187	3.53	6.43	1.154	1.025
14	7	4.75	0	0.149	0.150	14.36	10.02	0.979	0.748
17.5	6	1.75	50	0.274	0.241	3.24	5.14	0.853	0.760
17.5	6	3.25	50	0.157	0.166	3.66	6.78	0.768	0.570
17.5	6.5	0.25	50	0.374	0.355	2.42	3.64	0.993	0.763
17.5	6.5	1.75	50	0.305	0.257	4.06	4.86	0.912	0.904
17.5	6.5	1.75	100	0.297	0.250	4.41	5.14	0.529	0.684
17.5	6.5	3.25	50	0.175	0.181	5.93	6.10	0.890	0.804
21	6	1.75	0	0.369	0.367	3.11	3.61	1.128	0.862
21	6	3.25	50	0.332	0.213	5.24	5.00	0.540	0.622
24.5	6	1.75	150	0.402	0.438	4.08	3.55	0.343	0.360
24.5	6.5	3.25	50	0.416	0.439	2.66	3.25	0.763	0.878
24.5	6.5	3.25	150	0.389	0.393	3.56	4.04	0.382	0.44

* OD = Optical density.

Results and Discussion

The best model from the entire PUNN model nets generated have been selected (that is, those with the lowest Standard Error of Prediction for generalization set, SEP_G value) for each of the kinetic parameters.

For the growth rate, the best model had a 4:5:1 architecture net and was as follows:

$$Gr^* = 2.9118\ (T^*)^{1.91} \quad (S1)$$

$$+6.0509\ (T^*)^{6.55}\ (NaCl^*)^{5.25} \quad (S2)$$

$$+1.8178\ (T^*)^{0.17}\ (pH^*)^{0.13}\ (NaCl^*)^{0.11} \quad (S3)$$

$$-0.2551\ (T^*)^{0.11}\ (pH^*)^{1.59}\ (NaCl^*)^{0.40} \quad (S4)$$

$$-4.3718\ (T^*)^{1.82}\ (NaCl^*)^{0.56}\ (NaNO_2^*)^{0.04} \quad (S5)$$

For the lag-time, the best model had a 4:5:1 architecture net and was as follows:

$$Lnlag^* = 2.1146\ (NaCl^*)^{0.19} \quad (S1)$$

$$+1.3187\ (NaCl^*)^{1.79}\ (NaNO_2^*)^{2.09} \quad (S2)$$

$$-1.5026\ (T^*)^{0.75}\ (pH^*)^{0.08}\ (NaCl^*)^{0.58} \quad (S3)$$

$$+2.6521\ (T^*)^{0.80}\ (NaCl^*)^{2.65}\ (NaNO_2^*)^{0.58} \quad (S4)$$

$$-3.6252\ (T^*)^{0.39}\ (pH^*)^{0.28}\ (NaCl^*)^{2.66}\ (NaNO_2^*)^{1.46} \quad (S5)$$

For the maximum density population, the best model had a 4:6:1 architecture net and was as follow:

$$LnyEnd^* = 7.1439\ (T^*)^{1.19}\ (pH^*)^{2.17}\ (NaCl^*)^{0.89}\ (NaNO_2^*)^{3.65} \quad (S1)$$

$$-0.8340\ (T^*)^{0.07}\ (pH^*)^{0.19}\ (NaCl^*)^{1.80} \quad (S2)$$

$$+2.0445\ (NaCl^*)^{0.03} \quad (S3)$$

$$-2.2889\ (T^*)^{0.23}\ (pH^*)^{0.51}\ (NaCl^*)^{0.54}\ (NaNO_2^*)^{3.00} \quad (S4)$$

$$+2.8170\ (T^*)^{1.28}\ (pH^*)^{2.19}\ (NaCl^*)^{1.31} \quad (S5)$$

$$-11.3193\ (T^*)^{3.89}\ (pH^*)^{4.83}\ (NaCl^*)^{1.27}\ (NaNO_2^*)^{0.90} \quad (S6)$$

Because these models give an impression of being complex from the start, and to enable the best degree of interpretability possible, we are going to set up a series of simple rules to simplify the expression of the model. S1 to S6 are the 6 addends of the equations and they are associated to each node of the hidden layer of the net. There are addends in some areas of the factors that do not contribute significant values for the prediction of the corresponding growth parameter (Table 4).

Studying how the different addends affect the estimation of the models, we observed that the *Gr* equation has S3 as the base or tendential addend, expresses the obvious interaction between the temperature, pH, and salt, and over it the other 4 addends are accommodated, 2 by 2. S1 and S5 reaches high values, the 1st with a positive sign associated only with the temperature and the 2nd with a negative 1 associated with temperature and chloride. It is observed that the effect of the temperature is compensated, while when NaCl increases, the Gr decreases significantly. The addends S2 and S4 have a relative value lower than the previous ones, the 1st with a positive sign and the 2nd negative and only results in somewhat significant values for S2 when the temperature and the NaCl are high and for S4 when the pH is high. This means that when the temperature increases along with the salt, the values of the 2 addends are compensated, whereas when the pH increases, the Gr decreases somewhat. Nitrate hardly affects the Gr because the temperature deprives it of protagonism.

Under refrigeration conditions ($T \leq 14$ °C) and/or with little salt, the equation can be simplified to only 2 addends (Table 4). When the pH is low ($pH \leq 5.5$), the S4 addend can be eliminated.

In the *lag* equation, the primary (S1) and 3rd (S3) addends always appear in the model with different values of environmental factors, which is why they are its base addends. This indicates that the lag value depends directly and basically on the amount of salt, and inversely on the interaction of salt with the temperature and the pH, and to a lesser extent on the amount of nitrate. The relationship between the S1 values is approximately double those of S3 when the values of the factors are high. This means that a strong direct relationship exists between NaCl and lag. Table 4 shows that when the salt hits minimum values, the equation can be simplified to 2 addends.

The effect of the temperature and NaCl is compensated between S3, S4, and S5, although when pH rises the lag is decreased.

When the temperature increases and the pH goes down, the lag is seen to decrease due to S3 because the temperature is compensated by S4 and S5 addends and the pH continues to show an inverse relationship with lag in the S5 addend. Increasing the nitrate increases the lag because the sum of the values of the S2, S4, and S5 addends is always positive where nitrate appears.

In the yEnd model, the addend 3 is the base addend around which the other addends are accommodated, showing the great influence salt holds on this kinetic parameter. The interaction of all the factors in the S1, S4, and S6 addends results in a negative effect added onto the value of yEnd. For temperature and high pH, the effect on yEnd through the added effect of the S2 and S5 addends is positive. For small values of some factors, some of the addends in the model are not significant enough. So if the pH is low, the S1, S5, and S6 addends are insignificant, allowing simplification of the equation as can be seen in Table 4. When nitrates are absent ($NaNO_2 = 0$), then the S1 and S4 addends do not contribute significant values for the calculation of yEnd. When salt is 0.25%, the S2 addend is not significant (Table 4). The S6 addend loses relevance when pH and T are at minimum values (Table 4). On observing the base addends of yEnd, these are similar to the base addends of lag, which shows that the salt is the most influential element in both.

The estimations errors found in PUNN are in general significantly inferior to those found in the other 2 models. Table 5 shows the SEP, RMSE B_f and A_f found in the best model for each variable. It is important to mention that the experimental design carried out with (CCD) was originally meant for the treatment with RS that later was applied to MLP models and, in this study, to PUNN models. This design entails a series of disadvantages. The low error must be accompanied by a model that is not too complex and that has a good generalization capacity. New data in the same conditions (internal validation) and, later, data in different conditions but always within the range of the design (external validation) were compared with the estimations. This estimation should also be contrasted with other authors' data but in our knowledge there is no *L. mesenteroides* model published to compare with.

SEP_G, B_f, and A_f values obtained for the 3 kinetic parameters by the best model of PUNN, RS, and MLP models (Table 5) during the process of internal validation (generalization) were compared.

The model with least SEP_G was PUNN (5.59% SEP_G) for lag. The number of connections or equation parameters of this model (18) was smaller than MLP (22) although in this aspect RS wins with only 8 equation parameters (Table 6). In this case, based on internal validation, as RS SEP_G (6.58) is just above PUNN error, we consider that for this kinetic parameter the RS model would be the best one.

For Gr a wide difference between SEP_G is observed: PUNN (2.91%), MLP (3.77), and RS (9.91). In this case, although the PUNN model is more complex (17 parameters) versus RS (8) but simpler than MLP (19), the 1st 1 could be the model chosen due to its considerably lower error of prediction.

For yEnd the best SEP_G was obtained by PUNN (12.22%). Although it is considerably more complex (25 parameters) than MLP (13) or RS (10), the differences in error estimation could compensate this.

These 1st draft conclusions must be corroborated by the analysis of what happens in the external validation.

Furthermore, in all cases, B_f and A_f were close to unity, which indicates a good fit between the observations and predictions, as was also the case for RS and MLP. For the parameter Gr, B_f must be greater than 1 because in the case of spoilage microorganisms, this indicates that the model creates accurate shelf life predictions, because it will estimate beforehand any sensorial alterations in the product. In this

Table 4—Rules derived from the best Product Unit (PUNN) models

IF	THEN
T ≤ 14 °C, and/or NaCl ≤ 1.75	Gr* = S1 + S3
pH ≤ 5.5	Gr* = S1 + S2 + S3 + S5
$NaNO_2 = 0$	LnLag* = S1 + S3 + S4 + S5
NaCl = 0.25	LnLag* = S1 + S3
NaCl = 0.25	LnyEnd* = S1 + S3 + S4 + S5 + S6
pH = 5.5 o	LnyEnd * = S2 + S3 + S4
$NaNO_2 = 0$	LnyEnd * = S2 + S3 + S5 + S6

study, the Gr parameter had a very good A_f value (1.03), within the range of acceptability criteria described by Ross and others (2000) who considered an A_f value to be acceptable with an increase of up to 0.15 (15%) for each variable included in the model. Therefore, in our study, with 4 variables (temperature, pH, and concentration of salt and nitrites), we should expect A_f values of up to 1.6.

Comparison with other authors' studies was impossible because no model of *L. mesenteroides* has been achieved. Scientific literature contains few references to the internal validation of predictive models of other bacteria, and the results found were very similar to those determined in our own study. This is the case for the research carried out by Hervás and others (2001), who obtained SEP values of around 9% for Gr in an Artificial Neural Network for *Salmonella* spp., and García-Gimeno and others (2002), who observed values of between 11% to 17% for Gr and Lag in *Lactobacillus plantarum*.

All of this demonstrates that the model has good generalization ability when it comes to accurately estimating the growth response of *L. mesenteroides*.

For external validation, comparison with other authors' studies would be advisable but was impossible in this case because no model of *L. mesenteroides* has been achieved. Thus the model estimation was validated against a new data set in different conditions but within the range of the model. SEP_G values obtained by PUNN were lower for lag and GR but higher for yEnd than MLP (Table 5).

In all cases, B_f and A_f were close to unity, which indicates a good fit between the observations and predictions for the 3 models. During the process of external validation of the predictive models, several authors have accepted B_f values of between 0.75 and 1.25 as being acceptable for spoilage microorganisms (Dalgaard 2000). According to these criteria, each of the models elaborated can be considered suitable to describe the growth of *L. mesenteroides*.

For the parameter Gr, the B_f is below 1, which means that the models underestimated this parameter, although the A_f value indicates an acceptable estimation error (<1.60). Other studies describe values for bias and accuracy factors similar to those obtained in our study. Lebert and others (2000) observed good fit when they applied mathematical validation to models that estimate *generation time* for *Pseudomonas* spp. (B_f = 0.82 to 1.16 and A_f = 1.13 to 1.24). In a different study on the same microorganism, values were produced that were similar to the predictions for these parameters, B_f = 0.84 and A_f = 1.23 (Neumeyer and others 1997). Another study undertaken by Valík and Pieckovä (2001) with spoilage moulds produced values very close to unity, B_f = 1.01 and A_f = 1.07, showing the goodness of fit and the accuracy of the RS model elaborated. Arinder and Borch (1999) observed similar values for these factors, B_f = 1.02 and A_f = 1.36, for the growth rate of *Pseudomonas* spp.

Lag models also underestimate their value because B_f is higher than 1, meaning that it predicts higher times of adaptations of the microorganism than observed. A_f values indicate acceptable values, all below the criteria. The prediction of the lag-time poses more problems for our models than the other parameters because it depends on several factors, such as the physiological stage and size of the inoculum and pre-

Table 5— Standard errors of prediction (%SEP_G), Bias (B_f), and accuracy factors (A_f) for the best models of Product Units (PUNN), Response Surface (RS)[a], and multilayer perceptron (MLP)[b] for the lag-time (Lag), growth rate (Gr), and maximum population density (yEnd) of *Leuconostoc mesenteroides*

		PUNN				RS				MLP			
		RMSE	SEP	BF	AF	RMSE	SEP	BF	AF	RMSE	SEP	BF	AF
Lag-time (h)	Model	0.0934	4.34	1.00	1.04	0.120	6.02	1.02	1.1	0.109	5.51	0.98	1.09
	Internal validation	0.1201	5.59	0.99	1.03	0.120	6.58	1.01	1.09	0.134	6.55	0.97	1.11
	External validation	0.3156	20.14	1.14	1.19	0.1321	35.08	1.17	1.33	0.5110	32.61	1.18	1.31
Growth rate (/h)	Model	0.0073	3.04	1.01	1.04	0.022	10.48	1.00	1.09	0.009	4.13	1.00	1.04
	Internal validation	0.0062	2.91	1.00	1.03	0.0239	9.91	0.98	1.11	0.009	3.77	1.00	1.04
	External validation	0.0390	14.37	0.94	1.10	0.0415	15.31	0.95	1.12	0.0423	15.59	0.94	1.14
Maximum population density (OD)	Model	0.0616	11.57	0.99	1.10	0.087	16.35	0.98	1.14	0.078	14.60	0.99	1.12
	Internal validation	0.0668	12.22	0.94	1.11	0.0892	16.31	0.95	1.13	0.107	14.15	0.94	1.13
	External validation	0.1416	18.99	0.94	1.17	0.1418	19.02	0.90	1.19	0.1104	14.80	0.99	1.13

[a]RS = data taken from Zurera-Cosano and others (2005).
[b]MLP = data taken from García-Gimeno and others (2005).

vious growth conditions (Robinson and others 1998; Ross and others 2000). Other authors, for example García-Gimeno and others (2003), conducted a study using *Escherichia coli* O157:H7, and obtained accurate estimations using an MLP model (B_f = 0.95 and A_f = 1.24). For *Staphylococcus aureus,* Zurera-Cosano and others (2004) observed values of B_f = 0.87 to 1.54 and A_f = 1.52 to 2.22 using a Surface Response model in aerobic and anaerobic conditions. The values obtained in our study are within the range described by other authors and, in fact, are even better because they are closer to unity.

The kinetic parameter yEnd is not often modeled in predictive microbiology and is included in only a few models, such as those developed by McCann and others (2003) and Nauta and others (2003). In our study, we obtained values close to unity for B_f, although they did slightly underestimate the growth response of *L. mesenteroides*. The yEnd parameter has a poorer generalization SEP for PUNN, the MLP being the model with the best SEP_G, B_f and A_f.

When choosing a predictive model, it is not only important to bear in mind estimation errors, but also the complexity of the model (number of connections or coefficients), which is another decisive factor in the comparison of the models developed, even though in other publications this information is not specified (Lou and Nakai 2001).

Several authors highlight that MLP models produce better estimations of kinetic parameters than other models such as the RS (Hajmeer and others 1997; Hervás and others 2001; Lou and Nakai 2001; García-Gimeno and others 2002, 2003). In the studies conducted by García-Gimeno and others (2002, 2003) on *L. plantarum* and *E. coli* O157:H7, respectively, the MLP models were chosen instead of RS models based on the lower SEP, although the MLP models had a greater degree of complexity. Hajmeer and others (1997) reported on an MLP for *Shigella flexneri* with lower error values (4% to 12% mean absolute relative error) but with a considerable degree of complexity (142 parameters). Some researchers do not agree with the use of MLPs to predict growth parameters, due to their complexity. However, thanks to genetic algorithm pruning, MLPs have been shown to be even simpler than regression in certain cases (Hervás and others 2001; García-Gimeno and others 2002).

In our study, PUNN and MLP models were more complex than the RS models, especially in the case of the parameter Gr, but described lower SEP_G. In several publications, MLP models have been chosen over others although these models are more complex because they produce fewer errors in prediction values (Hajmeer and others 1997; García-Gimeno and others 2002, 2003).

The possibility of describing the development of spoilage bacteria in foods by predictive microbiology, and relating the spoilage of the product to a certain level of microorganisms would allow us to estimate the shelf life of different products. Of course, the model should include microorganism behavior data throughout the general shelf life of that type of product to estimate realistic shelf life duration of any of the products. The more accurate the models are, the more accurate our predictions will be, and this is an advantage for their practical application.

Table 6—Statistical comparison (*P* values) for Levene and student *t* tests of the SEP_G and number of connections for MLP and PUNN for Lag, Gr, and yEnd[a]

	%SEP_G		Nr of connections	
MLP versus PUNN	Levene test	test	Levene test	test
Lag (h)	0.000	0.118	0.000	0.000
Gr (/h)	0.009	0.753	0.000	0.178
yEnd (OD)	0.073	0.023	0.000	0.000

[a]Gr = growth rate; Lag = lag-time; MLP = multilayer perceptron; PUNN = Product Units; SEP_G = Standard Error of Prediction for generalization; yEnd = maximum density population.

Conclusions

We have defined neuronal net models of potential base, PUNN and of sigmoidal base MLP, as models that use the same methodology to approximate functions of a continuous type, where the problem is obtaining the optimum number of base functions that best adjust to a specified function, as well as the coefficients of these models. To do this we have used algorithms of evolutive computation to optimize the search for the best designs and coefficients of neuronal net models. The analysis of growth predictions under experimental conditions showed that the MLP and the PUNN satisfactorily represent the experimental data, although the best models are obtained with the PUNN models, which are much easier to interpret than the MLP ones. And here rests our treatise on the balance between the complexity of the model and the greater accuracy of the estimations.

With this work we have proposed a new approach to neural nets estimations for its application on predictive microbiology, searching for models with easier interpretation, and that has the advantage of having a great ability to fit the boundaries of the range of the input factors. We consider that there is still a lot left to do, but PUNN could be very valuable instrument for mathematical modeling.

Acknowledgments

This work was partly financed by the CICYT ALI98-0676-C02-01, MCYT AGL2001-2435, CAL01-032; TIC2002-04036-C05-02; TIN2005-08386-C05-02 and the Research Group AGR-170 HIBRO and TIC-148.

The authors would also like to thank the Plan Andaluz de Investigación for a Research Staff Training grant. FEDER also provided additional funding.

References

Arinder P, Borch E. 1999. Validation of mathematical models for predicting growth of Pseudomonas spp. Predictive microbiology applied to chilled food preservation. In: Refrigeration Science and Technology Proceedings. p 185–93.

Dalgaard P. 2000. Fresh and lightly preserved seafood. In: Man CMD, Jones AA, editors. Shelf life evaluation of foods. 2nd ed. Md.: Aspen Publishing. p 110–39.

Durbin R, Rumelhart D. 1989. Product units: a computationally powerful and biologically plausible extension to backpropagation networks. Neural Comput 1:133–42.

García-Gimeno RM, Hervás-Martínez C, Barco-Alcalá E, Zurera-Cosano G, Sanz-Tapia E. 2003. An artificial neural network approach to *Escherichia coli* O157:H7 growth estimation. J Food Sci 68:639–45.

García-Gimeno RM, Hervás-Martínez C, de Silóniz MI. 2002. Improving artificial neural networks with a pruning methodology and genetic algorithms for their application in microbial growth prediction in food. Int J Food Microbiol 72:19–30.

García-Gimeno RM, Hervás-Martínez C, Rodríguez-Pérez R, Zurera-Cosano G. 2005. Modelling the growth of *Leuconostoc mesenteroides* by artificial neural networks. Int J Food Microbiol 105:317-32.

Geeraerd A, Herremans C, Cenens C, Van Impe J. 1998. Applications of artificial neural networks as a non-linear modular modeling technique to describe bacterial growth in chilled food products. Int J Food Microbiol 44:49–68.

Gibson A, Bratchell N, Roberts T. 1988. Predicting microbial growth: growth responses of salmonellae in a laboratory medium affected by pH, sodium chloride and storage temperature. Int J Food Microbiol 6:155–78.

Gurney KN. 1992. Training nets of hardware realizable sigma-pi units. Neural Networks 5(2):289–303.

Hajmeer M, Basheer I, Najjar Y. 1997. Computational neural networks for predictive microbiology. II. Application to microbial growth. Int J Food Microbiol 34:51–66.

Hervás C, Zurera G, García-Gimeno RM, Martínez JA. 2001. Optimisation of computational neural network for its application to the prediction of microbial growth in foods. Food Sci Technol Int 7:159–63.

Ismail A, Engelbrecht AP. 1999. Training product units in feedforward neural networks using particle swarm optimization. Development and practice of artificial intelligence techniques. In: Bajic VB, Sha D, editors. Proceedings of the Intl. Conference on Artificial Intelligence; Sept 1999; Durban, South Africa. p 36–40.

Ismail AA, Engelbrecht AP. 2000. Global optimisation algorithms for training product unit neural networks. Institute of Electrical and Electronics Engineers (IEEE). International Joint Conference on Neural Networks; July 24–7, 2000; Como, Italy. Paper 032.

Janson DJ, Frenzel JF. 1993. Training product unit neural networks with genetic algorithms. Institute of Electrical and Electronics Engineers (IEEE) Intelligent Systems 8(5):26–33.

Jeyamkondan S, Jayas D, Holley R. 2001. Microbial growth modeling with artificial neural networks. Int J Food Microbiol 64:343–54.

Lebert I, Robles-Olvera V, Lebert A. 2000. Application of polynominal models to predict growth of mixed cultures of Pseudomonas spp. and Listeria in meat. Int J Food Microbiol 61:27–39.

Leerink LR, Giles CL, Horne BG, Jabri MA. 1995. Learning with product units. Tesauro G, Touretzky D, editors. In: Advances in neural information processing systems 7. Cambridge, Mass.: MIT Press. p 537–44.

Levenberg K. 1944. A method for solution of certain non-linear problems in least squares. Q J Appl Math II 2:164–8.

Lou W, Nakai S. 2001. Artificial neural network based predictive model for bacterial growth in a simulated medium of modified-atmosphere-packed cooked meat products. J Agric Food Chem 49:1799–804.

Marquardt DW. 1963. An algorithm for least-squares estimation of non-linear parameters. J Soc Ind Appl Math 11(2):431–41.

Martínez AC, Martínez FJ, García N, Hervás C. 2002. Modelado de Sistemas mediante modelos híbridos de redes neuronales y computación evolutiva. Garijo F, Riquelme JC, Toro M, editors. Workshop Aprendizaje y minería de datos. Iberamia 2002; Nov 12-5, 2002; Sevilla, Spain. p 67–76.

Martínez-Estudillo AC, Martínez FJ, Estudillo M, Hervás C, García N. 2005. Model fitting by evolutionary computation using product units. Neural Netwoks (Forthcoming).

McCann TL, Eifert JD, Gennings C, Schilling MW, Carter WH. 2003. A predictive model with repeated measures analysis of *Staphylococcus aureus* growth data. Food Microbiol 20:139–47.

Nauta MJ, Litman S, Barker GC, Carlin F. 2003. A retail and consumer phase model for exposure assessment of *Bacillus cereus*. Int J Food Microbiol 83:205–18.

Neumeyer K, Ross T, Thomson G, McMeekin TA. 1997. Validation of a model describing the effects of temperature and water activity on the growth of psychrotrophic pseudomonads. Int J Food Microbiol 38:55–63.

Robinson TP, Ocio MJ, Kaloti A, Mackey BM. 1998. The effect of the growth environment on the *Lag* phase of *Listeria monocytogenes*. Int J Food Microbiol 44:83–92.

Ross T. 1996. Indices for performance evaluation of predictive models in food microbiology. J Appl Bacteriol 81:501–8.

Ross T, Dalgaard P, Tienungoon S. 2000. Predictive modeling of the growth and survival of *Listeria* in fishery products. Int J Food Microbiol 62:231–45.

Setieno R, Leow WK, Zurada JM. 2002. Extraction of rules from artificial neural networks for nonlinear regression. IEEE Trans Neural Networks 13:564–77.

Valík L, Pieckova E. 2001. Growth modeling of heat-resistant fungi: the effect of water activity. Int J Food Microbiol 63:11–7.

Van Impe JF, Versyck KJ, Geeraerd AH. 1998. Validation of predictive models: definitions and concepts. In: COST 914. Predictive modeling of microbial growth and survival in foods. Brussels: Luxembourg Commission. p 31–8.

Zurera-Cosano G, Castillejo-Rodríguez AM, García-Gimeno RM, Rincón-León F. 2004. Response surface vs Davey model for the estimation of *Staphylococcus aureus* growth under different experimental conditions. J Food Protect 67:1138–45.

Zurera-Cosano G, García-Gimeno RM, Rodríguez-Pérez MR, Hervás Martínez C. 2005. Performance of Response Surface model for prediction of *Leuconostoc mesenteroides* growth parameters under different experimental conditions. Food Control 17:429-38.

Appendix

Exactly, the exponents w_{ji} of the function, which represent the weights of the connections between an input and hidden nodes, are modified as follows:

$$w_{ji}(t+1) = w_{ji}(t) + \xi_1(t) \quad , \quad 1 \le i \le k \quad , \quad 1 \le j \le p \qquad (1)$$

where $\xi_1 \in N(0, \alpha_1(t)\, T(f))$ represents a normally distributed 1-dimensional random variable with mean 0 and variance $\alpha(t)T(f)$. The coefficients β_j of the function f representing the weights of the connections between a hidden node and the output node, are modified as follows:

$$\beta_j(t+1) = \beta_j(t) + \xi_2(t) \quad , \quad 1 \le j \le p \qquad (2)$$

where $\xi_2 \in N(0, \alpha_2(t)\, T(f))$ represents, in a similar way, a normally distributed 1-dimensional random variable with mean 0 and variance $\alpha_2(t)T(f)$.

It should be pointed out that the modification of the exponents is different so that coefficients, that is $\alpha_1(t) << \alpha_2(t)$, are adaptively changed in every generation by some predefined rule.

In essence, the functions $\alpha_1(t)$ and $\alpha_2(t)$ define the mutation strength in each case and specifically, they are defined by:

$$\alpha_i(t) = \begin{cases} (1+\beta)\alpha_i(t) & \text{if } A(s) > A(s-1),\ \forall s \in \{t, t-1, \ldots, t-r\} \\ (1-\beta)\alpha_i(t) & \text{if } A(s) = A(s-1),\ \forall s \in \{t, t-1, \ldots, t-r\} \\ \alpha_i(t) & \text{in the other cases} \end{cases} \quad i = 1, 2 \qquad (3)$$

where $A(s)$ represents the fitness of the best individual in the generation s-th and the parameters β and r are fixed, user-defined parameters.

Taking into account that a generation is defined as successful if the best individual of the population is better than the best individual of the previous generation, if many successful generations are observed, this indicates that the best solutions are residing in a better region in the search space. In this case, we increase the mutation strength in the hope of finding ever better solutions closer to the optimum solution. If the fitness of the best individual is constant in different generations, we decrease the mutation strength. In the other cases the mutation strength is constant.

When the mutations are realized, the fitness of the individual is recalculated and the usual simulated annealing criterion is applied. Being ΔA the difference in the fitness function before and after the random step:

If $\Delta A \ge 0$ the step is accepted

If $\Delta A < 0$ then the step is accepted with a probability

$$P(\Delta A) = \exp\left(-\frac{\Delta A}{T}\right) \qquad (4)$$

where T is the current temperature.