Neural and Fuzzy Neural Networks in Prediction of Natural Gas Consumption

Nguyen Hoang Viet[1] and Jacek Mańdziuk[2]

[1]Institute of Fundamental Technological Research,
Polish Academy of Sciences,
Swietokrzyska 21 Str., 00-049 Warsaw, Poland
[2]Faculty of Mathematics and Information Science,
Warsaw University of Technology,
Plac Politechniki 1, 00-661 Warsaw, Poland

Abstract
In this work several approaches to prediction of natural gas consumption with neural and fuzzy neural systems are analyzed and tested. The data covers daily natural gas load in two different regions of Poland. Prediction strategies tested in the paper include: single neural net module approach, combination of three neural modules, temperature context based method, and application of fuzzy neural networks. The results indicate the superiority of temperature context based method and the modular approach over single neural net and fuzzy neural approaches. One of the interesting issues observed in the paper is relatively good performance of tested methods in the case of long-term (four week) prediction compared to mid-term (one week) prediction. Generally, the results are superior to those obtained by linear and quadratic regression models and by statistical methods currently used for this task in the gas company under consideration.
Keywords - Feedforward networks, fuzzy neural networks, context networks, time series prediction, gas consumption prediction

1. INTRODUCTION

The paper discusses several neural and fuzzy neural approaches to the problem of prediction of natural gas consumption in two regions of Poland. The first one is mostly rural with several villages and small cities and therefore the consumers are mostly individual or they belong to small industry (bakeries, restaurants, laundries, etc.). The other one covers highly industrialized and densely inhabited urban area.

Predicting consumption of natural gas is an interesting, non-trivial and highly economically-motivated task (Browna et al., 1996; Khotanzad et al., 2000). The conventional approach to solving it is based on applying statistical methods which are usually efficient enough only if the large amount of historical data is available. An alternative approach is to exploit soft-computing methods - and especially neural and fuzzy neural network models.

Neural networks are well known to be universal non-linear approximators (Kolmogorov, 1957; Kurkova, 2000) and therefore are in general capable of close approximation of the prediction model without the need of its explicit (mathematical) formulation in contrast to statistical approaches. The other advantage of using neural networks over statistical methods in the application domain considered in this paper is the problem of gas market volatility in Poland in recent years. On one hand several new consumers are attracted by relatively low cost of this type of energy, but on the other hand some gas consumers, especially in the rural regions, switch to alternative energy sources, like wood or coal. This type of *structural instability* of natural gas consumers is very harmful for statistical approaches, which do not implement adaptation mechanisms.

Three types of prediction are considered in this work: one day prediction (denoted by **D-type**), one week prediction (denoted by **W-type**) and four week prediction (denoted by **4W-type**). From the gas company point of view, the long term prediction is the most valuable. Nevertheless, the short-term and the mid-term ones also play significant role, for example in gas distribution planning.

The main research goal of this paper is to compare the efficacy of proposed neural and fuzzy neural methods, with special attention put on checking their effectiveness in the rural versus industrialized areas.

The paper is organized as follows: in the next section network architectures and training methods used in this work are introduced. Data files and preprocessing methods are described in section 3. Experimental results are presented in section 4. Conclusions and directions for future development of this work are placed in the last section.

2. NETWORK ARCHITECTURES AND TRAINING METHODS

In this section, two neural models for gas load prediction are presented: the feedforward network and the fuzzy one.

Available data set contains the daily cumulative loads of natural gas provided by the telemetric system as well as the average daily temperatures. We denote by $G(t)$ the actual gas consumption and by $T(t)$ the average temperature on day t. It should be noted that throughout the paper all references to temperature values concern the

observable average temperatures in a given period in the past. Due to seasonality of the data the time factor defining the period of interest is also of major importance.

All the above values are taken as inputs for the neural networks. More details on time coding and data preprocessing are presented in section 3.

2.1. Feedforward network

2.1.1. Network architecture

The first network architecture tested in this work is the common feedforward one with sigmoidal neurons (see Fig. 1).

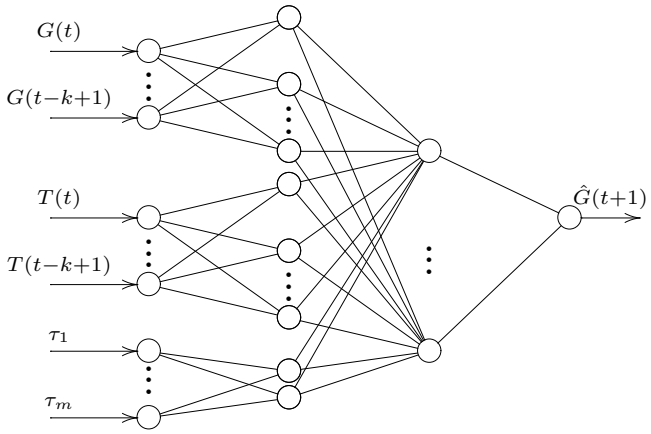

Figure 1: Feedforward network architecture used in prediction.

Neurons in the input layer can be divided into three groups. The first k neurons in the input layer represent daily gas loads. The second group of k neurons refers to average daily temperatures from k previous days. The last input group $\tau_1, \ldots, \tau_m$ denotes the time factor defining the period under consideration.

The first hidden layer also consists of three groups of neurons. Neurons belonging to the first group are fully connected to those in the first group of the input layer (i.e. *gas units*). Similarly, neurons in the second group are fully connected to those in the second input group (*temperature inputs*). The third group neurons are connected to the rest of the inputs. All the first hidden layer neurons are fully connected to the neurons in the second hidden layer, which are in turn connected to a single output neuron. The real value produced by the output neuron, denoted by $\hat{G}(t+1)$ for prediction made on day t, represents the next day load in case of one day prediction or the average daily load in a given time period in the case of one week or four week prediction.

The goal of a partition of the first hidden layer neurons is to aggregate the input data of each type (i.e. gas, temperature and time inputs) independently from the

rest of the input data. This information is then aggregated in the second hidden layer, where no restrictions are imposed on connections to the previous layer. No explicit temperature prediction was made since this kind of prediction was virtually embedded in the gas load prediction network, due to its specific architecture.

2.1.2. Network training

Let us denote by $\mathcal{N}$ a particular feedforward neural network. Each instance $\mathcal{N}_{\mathrm{w}}$ of network $\mathcal{N}$ refers to a particular weight configuration w, $\mathrm{w} \in \mathbb{R}^{\mathrm{N}}$, where N is the number of network parameters. Providing a training sample set $\mathbf{T} = \left\{(x_i, d_i) : i = \overline{1, \mathrm{Q}}\right\}$, where x_i, d_i are input and output vectors, resp., Q is the number of training samples, the network training problem can be considered as an optimization problem:

$$\text{Find } \mathrm{w}^*, \text{ which minimizes some cost function } \mathrm{E}(\mathrm{w}, \mathbf{T}). \tag{1}$$

The cost function $\mathrm{E}(\mathrm{w}, \mathbf{T})$ in general can be defined in various ways, depending on particular application. In the gas load prediction problem considered in the paper, the following cost function is assumed:

$$\mathrm{E}(\mathrm{w}, \mathbf{T}) = \mathrm{E}(\mathrm{w}) \;=\; \frac{1}{2} \sum_{(x_i, d_i) \in \mathbf{T}} ||\mathcal{N}_{\mathrm{w}}(x_i) - d_i||^2 \tag{2}$$

where $||.||$ is the Euclidian norm in $\mathbb{R}^{\mathrm{N}}$. The optimization problem (1) can be solved by various techniques, including genetic algorithms, simulated annealing and many others. The most popular one is perhaps the gradient-based technique, which makes use of the Taylor expansion of the cost function. The minimization process can be considered as recursively constructing a series of weight vectors $\{\mathrm{w}_k\}$:

$$\mathrm{w}_{k+1} = \mathrm{w}_k + \alpha_k \mathrm{p}_k \tag{3}$$

where $k \geq 0$, $0 \neq \mathrm{p}_k \in \mathbb{R}^{\mathrm{N}}, \alpha_k \in \mathbb{R}$, so that $\{\mathrm{E}(\mathrm{w}_k)\}$ converges to a satisfactory local minimum hopefully being the global one. Vector p_k represents the local *search direction* and $\alpha_k \in \mathbb{R}^{\mathrm{N}}$ represents the *learning rate* at step k. The technique of choosing p_k and α_k is essential for each gradient-based algorithm. In this paper, the Scaled Conjugate Gradient Algorithm (Moller, 1993) is applied to train the prediction networks (see Appendix 1).

2.2. Fuzzy neural networks (FNNs)

Several FNN models have been developed and successfully used in various applications (Buckley & Hayashi, 1994; Gupta & Rao, 1994; Vuorimaa, 1994; Karayiannis, 1996). The advantages of fuzzy over conventional neural networks are their high generalization abilities and the capability of dealing with imprecise data. Some fuzzy

neural networks can process fuzzy inputs directly. In some cases, fuzzy rules can be extracted from trained fuzzy neural networks (Pal & Pal, 1999; Shann & Fu, 1995). In the gas load prediction problem, the temperature input plays significant role. The use of fuzzy neural network for this task is, among other things, motivated by the fact that the available average daily temperature is only an approximation of the exact temperature which varies throughout the day. In this section, first a fuzzy neural network model is described together with its training algorithm. The rest of the section discusses the use of this model in our prediction task.

2.2.1. The FNN model

The fuzzy neural network implemented in this work can be represented by a one hidden layer feedforward architecture with N input units, K hidden units and M output ones (Figure 2).

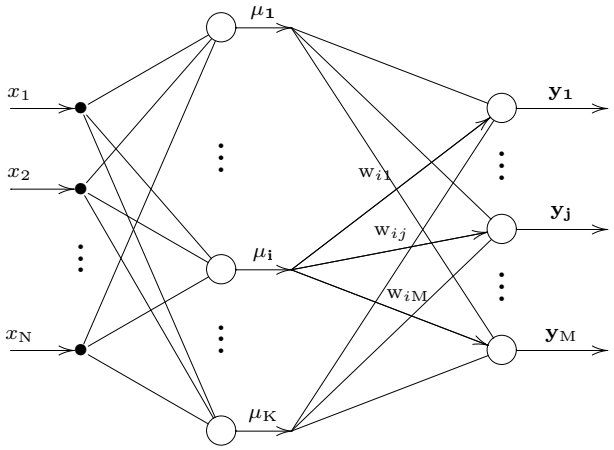

Figure 2: The architecture of fuzzy-neural network used in simulations.

Each input neuron represents a crisp input value. Each connection between input and hidden units has a weight equal to 1. Each unit in the hidden layer represents a fuzzy set over the input space $\mathbb{R}^N$. The output value of the i-th hidden neuron for a given input vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$ can be interpreted as the degree of membership of $\mathbf{X}$ to the fuzzy set represented by this neuron. Neurons in the hidden layer are fully connected with neurons in the output layer. Information of the membership degree of vector $\mathbf{X}$ to each fuzzy set is aggregated in the output layer.

Assuming the Gaussian form of the membership function in the fuzzy sets represented by hidden neurons, the output of the i-th hidden neuron given $\mathbf{X}$ as input is equal to:

$$\mu_i \;=\; \exp\left\{-\left(\frac{\parallel \mathbf{X} - \mathbf{C}_i \parallel}{\sigma_i}\right)^2\right\}, \tag{4}$$

where $\| \cdot \|$ is the Euclidian norm, $\sigma_i \in \mathbb{R}$, $\mathbf{C}_i = [c_{i1}, c_{i2}, \ldots, c_{iN}]^T \in \mathbb{R}^{\mathrm{N}}$ are parameters associated with a given neuron.

Denoting by $\mathbf{W} = [\mathrm{w}_{km}]^{\mathrm{K}\times\mathrm{M}}$ the weight matrix of connections between the hidden layer and the output layer, the input value for the j-th output neuron is equal to:

$$u_j = \sum_{k=1}^{\mathrm{K}} \mathrm{w}_{kj}\mu_k. \tag{5}$$

If we denote

$$s = \sum_{k=1}^{\mathrm{K}} \mu_k, \tag{6}$$

then the output value of that neuron is defined as:

$$y_j = \frac{u_j}{s} = \frac{\sum_{k=1}^{\mathrm{K}} \mathrm{w}_{kj}\mu_k}{\sum_{k=1}^{\mathrm{K}} \mu_k}. \tag{7}$$

The FNN model described here can be interpreted in terms of an equivalent fuzzy system. For the i-th neuron in the input layer (the fuzzification neuron), a fuzzy IF-THEN rule $\mathbf{R}_i$ can be extracted:

$$\mathbf{R}_i \ : \ \text{IF X is } \mu_i \text{ THEN } y_1 \ = \ \mathrm{w}_{i1} \text{ AND } \ \ldots \ \text{AND } y_{\mathrm{M}} \ = \ \mathrm{w}_{i\mathrm{M}}.$$

It is clear that fuzzy properties μ_i as well as the (crisp) outputs w_{ij} of the fuzzy rules are determined in the training process. Certainly the notion of μ_i indicates only a fuzzy set over the input space, not any linguistic value. The fuzzy rule set is then defined as:

$$\mathrm{S} = \left\{ \mathbf{R}_i : \ i = \overline{1, \mathrm{K}} \right\},$$

and the inference engine for such fuzzy system is specified by (7).

2.2.2. FNN training

The fuzzy neural network described above can be trained in supervised mode. For a single training sample $(\mathbf{X}, \mathbf{d})$, $\mathbf{X} = [x_1, x_2, \ldots, x_{\mathrm{N}}]^T$, $\mathbf{d} = [d_1, d_2, \ldots, d_{\mathrm{M}}]^T$, the error function defined as:

$$\mathrm{E} \ = \ \frac{1}{2} \sum_{j=1}^{\mathrm{M}} (d_j \ - \ y_j)^2 \tag{8}$$

is to be minimized w.r.t. σ_i, $\mathbf{C}_i$ $\left(i = \overline{1, \mathrm{K}}\right)$ and $\mathbf{W}$. This minimization task can be solved by any gradient based optimization algorithm. The partial derivatives of E can be computed in the following way:

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{ij}} \ = \ -(d_j - y_j) \frac{\partial y_j}{\partial \mathrm{w}_{ij}},$$

$$\begin{aligned}\frac{\partial \mathrm{E}}{\partial c_{il}} &= -\sum_{j=1}^{\mathrm{M}} (d_j - y_j) \frac{\partial y_j}{\partial c_{il}}, \\ \frac{\partial \mathrm{E}}{\partial \sigma_i} &= -\sum_{j=1}^{\mathrm{M}} (d_j - y_j) \frac{\partial y_j}{\partial \sigma_i}\end{aligned}$$

for $i = \overline{1, \mathrm{K}}$, $j = \overline{1, \mathrm{M}}$ and $l = \overline{1, \mathrm{N}}$. Using (7) one can write:

$$\begin{aligned}\frac{\partial y_j}{\partial \mathrm{w}_{ij}} &= \frac{\mu_i}{\sum_{k=1}^{\mathrm{K}} \mu_k} = \frac{\mu_i}{s}, \\ \frac{\partial y_j}{\partial c_{il}} &= \frac{\partial y_j}{\partial \mu_i} \frac{\partial \mu_i}{\partial c_{il}} = \frac{\mathrm{w}_{ij} s - u_j}{s^2} \frac{\partial \mu_i}{\partial c_{il}}, \\ \frac{\partial y_j}{\partial \sigma_i} &= \frac{\partial y_j}{\partial \mu_i} \frac{\partial \mu_i}{\partial \sigma_i} = \frac{\mathrm{w}_{ij} s - u_j}{s^2} \frac{\partial \mu_i}{\partial \sigma_i}.\end{aligned}$$

From (4) we have:

$$\begin{aligned}\frac{\partial \mu_i}{\partial c_{il}} &= 2 \frac{\mu_i}{\sigma_i^2} (x_l - c_{il}), \\ \frac{\partial \mu_i}{\partial \sigma_i} &= 2 \frac{\mu_i}{\sigma_i^3} \sum_{n=1}^{\mathrm{N}} (x_n - c_{in})^2.\end{aligned}$$

Finally:

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{ij}} = \delta_j^{(1)} \mu_i, \tag{9}$$

$$\frac{\partial \mathrm{E}}{\partial c_{il}} = \delta_i^{(2)} (x_l - c_{il}), \tag{10}$$

$$\frac{\partial \mathrm{E}}{\partial \sigma_i} = \delta_i^{(2)} \frac{1}{\sigma_i} \sum_{n=1}^{\mathrm{N}} (x_n - c_{in})^2, \tag{11}$$

where

$$\begin{aligned}\delta_j^{(1)} &= -\frac{d_j - y_j}{s}, \\ \delta_i^{(2)} &= 2 \frac{\mu_i}{\sigma_i^2} \sum_{j=1}^{\mathrm{M}} \left(\delta_j^{(1)} \left(\mathrm{w}_{ij} - \frac{u_j}{s} \right) \right)\end{aligned}$$

are the error signals in output and hidden neurons, resp. which can be computed effectively in a back-propagation manner. Having the partial gradients determined as in (9)-(11), the network's parameters (i.e. w_{ij}, c_{il} and δ_i) can be then adapted, e.g. using the standard steepest descent algorithm. In this work the scaled conjugate gradient algorithm (see Appendix 1) was applied.

One important issue worth mentioning here is the method of initializing FNN's parameters before training. Since each hidden neuron in such network represents a

fuzzy set of Gaussian membership function, the parameter vectors $\mathbf{C}_i$, $i = \overline{1, \mathrm{K}}$, i.e. the medians of these fuzzy sets, should be initialized *close* to the centers of some clusters that *cover* the input space. Then the training process *moves* these fuzzy sets to an optimal configuration. In order to initialize parameters $\mathbf{C}_i$ before training, a self-organizing map was first built from the training data set. In this paper, the maps were built using the neural gas technique presented in (Martinetz et al., 1993).

3. DATA COLLECTING AND PREPROCESSING

3.1. Data collecting

The main problem in this particular prediction task is the lack of historical data. The two available data sets - one refers to a rural region (denoted by **RR**) and the other refers to a highly industrialized area (denoted by **IR**) - provided by the gas company range only from Jan. 01, 2000 to Dec. 31, 2002. Since for these two data sets the same preprocessing techniques were applied and the same types of tests were performed, for the sake of simplicity, henceforth, in our description we shall refer to these two sets as to one set (being either **RR** or **IR** set).

The data set was initially divided into two sets: the first one covering years $2000 - 2001$ to compose the training set and the second one with 2002 year data to form the test set. Furthermore 10% of the training data was randomly chosen for validation. Each data record represents the daily cumulative load of natural gas provided by the telemetric system as well as the average daily temperature. The daily load distributions registered during the years 2000 and 2001 in **RR** and **IR** are presented in Figure 3.

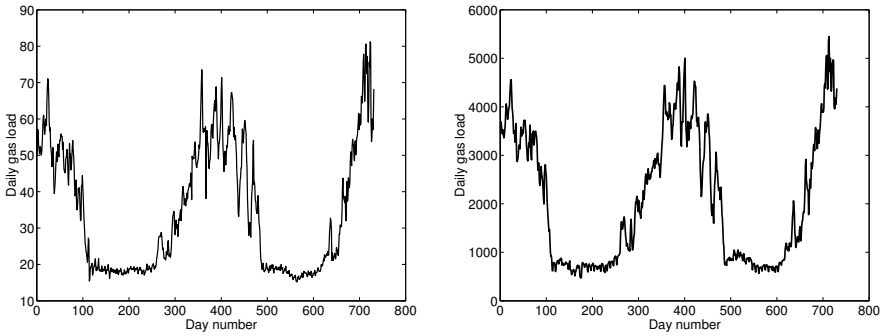

Figure 3: Daily gas load data from Jan. 01, 2000 to Dec. 31, 2001 used as the training and validating set for the **RR** (left) and the **IR** (right) areas, resp.

Due to a relatively small amount of past data, the sliding window mechanism was used in order to artificially enlarge the data sets. Namely, for W and 4W-type predictions the target periods are overlapping and subsequent target periods are

defined as $[t+1, t+n]$ - for prediction made on day t, $[t+2, t+n+1]$ - for prediction made on day $t+1$, $[t+3, t+n+2]$ - for prediction made on day $t+2$, etc., where $n = 7$ or $n = 28$ for W-type and 4W-type predictions, resp.

3.2. Data scaling

The available data (i.e. the daily gas load and the average daily temperature) must be scaled to some appropriate range before being passed to the network. In the case of temperature inputs, the maximum and the minimum daily temperatures, denoted by t_{max} and t_{min} respectively, were determined from the training set. Afterwards the temperature range $[t_{min},\ t_{max}]$ was uniformly extended by 20% from each side into the range $[\mathrm{T}_{min}, \mathrm{T}_{max}]$, namely:

$$\begin{aligned} \mathrm{T}_{min} &= t_{min} - [t_{max} - t_{min}] \cdot 20\% \\ \mathrm{T}_{max} &= t_{max} + [t_{max} - t_{min}] \cdot 20\%, \end{aligned}$$

in order to prevent some extreme values of the temperature which were not present in the training set. Next, each temperature input t was replaced by a new value $t' \in [-1, +1]$:

$$t' = \begin{cases} \frac{2t - \mathrm{T}_{max} - \mathrm{T}_{min}}{\mathrm{T}_{max} - \mathrm{T}_{min}} & \text{if } t \in [\mathrm{T}_{min}, \mathrm{T}_{max}] \\ -1 & \text{if } t < \mathrm{T}_{min} \\ +1 & \text{if } t > \mathrm{T}_{max} \end{cases} \tag{12}$$

The input daily loads were scaled into $[0, 1]$ by dividing each value $G(t)$ by G_{max}, where G_{max} is the extended - as in the case of temperature - maximum daily load over the training data set. The output (target) values in the training set, which are the average daily loads for a given one-day, one-week or four-week period - depending on the type of prediction - were scaled similarly, i.e. were divided by G_{max}.

3.3. Defining the time factor

By analyzing the training data set, it can be observed (Figure 3) that the gas consumption is highly seasonal: the highest loads occur in winter periods and the lowest ones in summer time. Figure 4 shows a strong dependency of daily gas consumption on the average daily temperature. Since the data is highly seasonal, it is important to properly code the time factor for each prediction period. This statement was confirmed by some additional experimental tests where no time factor was taken into consideration. The results obtained in such a case turned out to be much worse than those obtained when the time factor was concerned.

Denote by t the *day of the year* number of the first day of some n-day period ($n \geq 1, 0 \leq t \leq 365$). The consecutive days in that period are numbered by $t+1$,

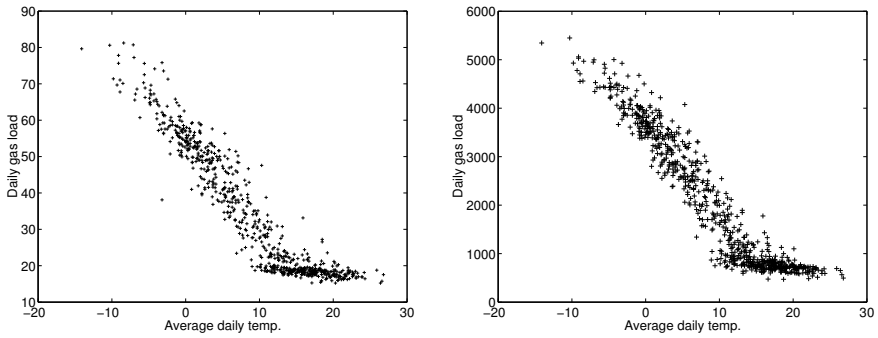

Figure 4: The dependency of daily gas load on the average daily temperature for the **RR** (left) and the **IR** (right) areas, resp. The data covers the period from Jan. 01, 2000 to Dec. 31, 2001.

$t+2, \ldots, t+n-1$. For such a period, the two following time inputs are applied:

$$\tau_1 = \sin \frac{2\pi\ D_c}{366} \qquad \text{and} \qquad \tau_2 = \cos \frac{2\pi\ D_c}{366}, \tag{13}$$

where D_c is a real value indicating the *center day* of the period under consideration, i.e.:

$$D_c = t + \frac{n-1}{2} \tag{14}$$

It can be observed that the use of (13) allows smooth coding of the season of the year, which is especially important in the case of mid-term and long-term prediction. In the case of D-type prediction there is an additional time coding input τ_3, which indicates the type of the next day, where $\tau_3 = -1$ for working days and $\tau_3 = +1$ for non-working days[1]. This input can be interpreted as to define the working and non-working day context of one day prediction networks.

4. EXPERIMENTAL RESULTS

For each prediction horizon the following experiments were performed: naive prediction, prediction using linear and quadratic regression models, single neural network prediction module, prediction using a combination of three neural modules, prediction using three neural modules - each of which was devoted to a predefined temperature range, single neural network prediction performed for the working days only (concerns D-type prediction only) and prediction based on a single fuzzy neural network

[1]Actually, for the sake of simplicity, only weekend days are treated as non-working days. All holidays that occurred in weekdays are treated as working days.

module.

4.1. Experiment description

4.1.1. Naive prediction

In the first experiment, no neural network is involved. The predicted average daily load in a n-day period $[t+1,\ t+n]$ is simply assumed to be the average daily load in the preceding period, i.e. $[t-n+1,\ t]$, where $n = 1,\ 7,\ 28$, resp. for one day, one week and four week predictions.

4.1.2. Prediction using a single feedforward network

In this experiment, a single feedforward network described in section 2.1 was involved in the prediction module. The only difference between the networks for one day, one week and four week predictions are the numbers of input and hidden neurons. Various network configurations were tested for each type of prediction. The following configurations were finally chosen:

- For one day prediction, the gas load and temperature from the last three days were taken as inputs (i.e. $k = 3$). In the first hidden layer, there were three neurons in the first and three in the second groups of neurons. The number of neurons in the third group was equal to two (see section 2.1.1 for details). And finally the number of neurons in the second hidden layer was set to four.

- For one week prediction, the data from the last five days were used as the inputs for the prediction networks. The number of neurons in each of the first two groups in the first hidden layer was equal to four. There were also two neurons in the last group of the first hidden layer. The second hidden layer was composed of four neurons.

- In the case of four week prediction, the input data was taken from the last seven days. The number of neurons in each hidden layer was the same as in the case of one week prediction.

For each prediction horizon, 50 neural networks were trained using the training sets. Next, each single network was used as the predictor. Note that the network sets are separate for the two regions under consideration.

4.1.3. Working days prediction

In the case of highly industrialized area, it could be assumed that there may be some difference between the gas consumption in the week days and the weekend days.

This experiment was performed for one day prediction only. The data corresponding to the weekend days was removed from the training and test sets. The networks used here were analogous to these described in section 4.1.2.

4.1.4. Combination of feedforward neural networks

It is interesting to combine the networks described in sect. 4.1.2 into one prediction module. Since the networks were trained independently with different initial weights, their responses may be different for the same input vector. Each network may be treated as an independent agent and combination of several agents may reduce prediction error.

In this experiment, for each prediction horizon, all combinations of three different networks among the above 50 ones were tested. The output value was set to be the average value of the outputs from each component network. Namely, for the three networks $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$, the output given by the combined predictor $\mathcal{C}$ for a particular input vector X was:

$$\mathcal{C}\left(\mathrm{X}\right) \;:=\; \frac{1}{3}\sum_{i=1}^{3}\mathcal{N}_i\left(\mathrm{X}\right) \tag{15}$$

4.1.5. Temperature context networks

It has been stressed in the paper that temperature plays a major role in the task of gas load prediction. So far the networks described in section 4.1.2 were trained based on the whole training set covering the whole year. The main idea of the temperature context approach is to train and evaluate each network within some particular range of temperatures. This kind of partition may facilitate the learning task for prediction networks (Bortolan & Pedrycz, 2002).

In this experiment the training set $\mathbf{T}$ was divided into three equipotent, overlapping subsets. For each training sample p, the average daily temperature $\bar{t}_p$ of the days covered by this sample was calculated. Let t_1, t_2 and t_3 be some temperature values where:

$$t_1 \;<\; t_2 \;<\; t_3.$$

Let:

$$\begin{aligned}
\mathbf{L} &= \left\{p \in \mathbf{T} \;:\; \bar{t}_p \;<\; t_2\right\} \\
\mathbf{M} &= \left\{p \in \mathbf{T} \;:\; t_1 \;\leq\; \bar{t}_p \;<\; t_3\right\} \\
\mathbf{H} &= \left\{p \in \mathbf{T} \;:\; \bar{t}_p \;\geq\; t_2\right\}.
\end{aligned}$$

The values t_1, t_2 and t_3 were chosen so that:

$$\mathrm{card}\left(\mathbf{L}\right) \;=\; \mathrm{card}\left(\mathbf{M}\right) \;=\; \mathrm{card}\left(\mathbf{H}\right).$$

In other words, the training set was divided according to the temperature context. The subsets **L**, **M** and **H** can be regarded as containing sample data corresponding to "low", "medium" and "high" temperatures. Certainly, such notions are subjective, hence the use of quotation marks.

For each of these subsets, a set of 20 neural networks was independently trained. The context-based partition of the training data could facilitate the prediction task within each temperature range. In the test phase, all possible combinations of the three modules (one of each type) were tested (8000 combinations in total). Please note that due to some overlapping areas in the training sets, the effect of gluing the modules was achieved in a straightforward way (c.f. Sharkey, 1997; Waibel, 1989). For a given test sample, the corresponding average input temperature was calculated and then depending on this value, one or (usually) two appropriate networks were activated. The ultimate prediction was the average value of the outputs of the modules involved in prediction.

4.1.6. Prediction with a single fuzzy neural network

Analogously to the case of feedforward networks, the fuzzy neural networks used here received the daily gas load and the average daily temperature from the last few days as the input: three days for short-term, five days for mid-term and seven days for long term prediction.

In all FNNs used here, the number of hidden units was arbitrarily set to four[2]. Each network contained one output neuron that produced the predicted average daily gas load (c.f. section 2.1). For each prediction horizon, an ensemble of 50 FNNs was trained and tested analogously as in the experiment with single neural modules.

4.1.7. Linear and quadratic regression models

In order to compare the efficiency of all neural network based approaches described above with some statistical methods of gas consumption prediction, linear and quadratic regression models were built for each type of prediction using statistical toolbox in MATLAB. The input data for those regression models were the same as for neural networks. The regression models' parameters were computed based on the same data set which was used for training prediction networks.

4.2. Results

The test data set, in the case of one week and four week predictions was artificially enlarged by using the sliding window mechanism in the same manner as for the

[2]This parameter was selected experimentally based on preliminary tests.

training set. In order to examine the capabilities of each prediction model, several experiments with various neural configurations were performed. In all tests the same error measure - the Mean Absolute Percentage Error (MAPE) defined as:

$$\text{MAPE} = \frac{1}{\text{Q}} \sum_{i=1}^{\text{Q}} \left| \frac{y_i - d_i}{d_i} \right| \times 100\%, \tag{16}$$

commonly used in prediction tasks (Khptanzad et al., 2000; Zhang et al., 1998), was applied. Here y_i is the predicted value, d_i the expected output and Q the number of samples in the test set.

For each experiment defined is section 4.1 the *average, min, max* (in percent) and *standard deviation* of MAPE over all tested networks (or all their combinations, where applicable) were calculated. The results for both **RR** and **IR** regions are presented in Tables 1-3.

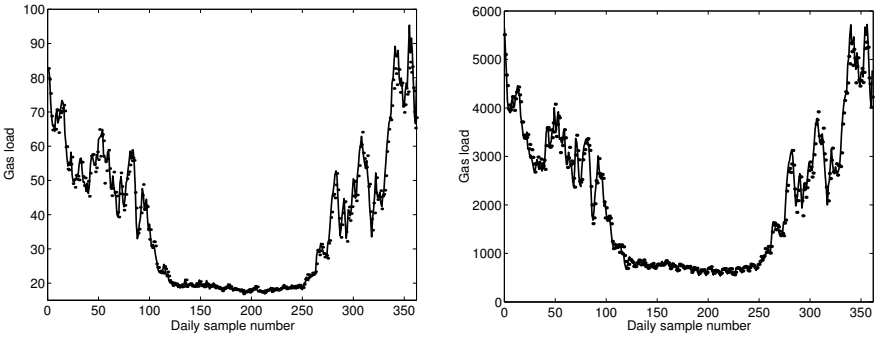

Figure 5: Test results for D-type prediction in the period Jan. 01, 2002- Dec. 31, 2002 in the **RR** (left) and the **IR** (right) areas, resp.

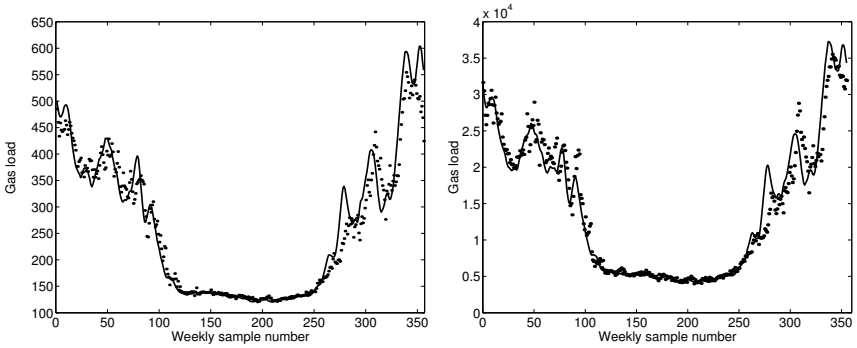

Figure 6: Test results for W-type prediction in the period Jan. 01, 2002- Dec. 31, 2002 in the **RR** (left) and the **IR** (right) areas, resp.

Examples of D-type, W-type and 4W-type predictions obtained using a combination of three networks, each of which was trained for some temperature range are presented in Figures 5-7, resp. The solid lines indicate the exact values taken from the test set, the dotted ones the predicted values produced by neural modules.

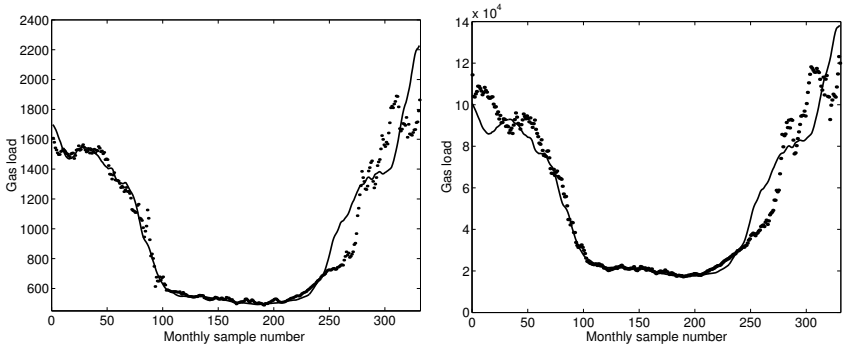

Figure 7: Test results for 4W-type prediction in the period Jan. 01, 2002 - Dec. 31, 2002 in the **RR** (left) and the **IR** (right) areas, resp.

The following general conclusions can be drawn from Tables 1-3:

- Comparing the average MAPE values the best performance is observed for the combination of three temperature context based modules (in case of D-type and W-type predictions) and the combination of three non-temperature-based networks (in case of 4W-type predictions). These two outperform the single neural and the single fuzzy neural approaches.

- The efficiencies of single fuzzy neural network and single neural network modules are comparable. It is important to note, however, that for each type of prediction, the size of a fuzzy neural network module was much smaller compared to the corresponding crisp neural module.

- In the case of D-type prediction an interesting issue is to compare the working days prediction *WorkD* versus all days prediction *SingleN*. As can be observed in Table 1, such a comparison is inconclusive. In case of the **RR** region *WorkD* results slightly outperform the *SingleN* ones, however in case of **IR** area the situation is the opposite.

- In all experiments, the naive approach as well as linear and quadratic regression models appeared to be visibly worse than neural-based methods. This observation confirms that prediction of natural gas consumption is a difficult, most probably highly nonlinear task.

- The quality of results in all tested methods degrades along with the length of prediction horizon. An interesting phenomenon is that the degradation between W-type and 4W-type predictions is relatively much less significant than between D-type and W-type ones. This suggests that at some point prediction error for longer periods may stabilize at a certain level.

D-type	RR				IR			
	AVG	MIN	MAX	SD	AVG	MIN	MAX	SD
$Naive$	4.56	$--$	$--$	$--$	6.13	$--$	$--$	$--$
$LinRg$	4.75	$--$	$--$	$--$	6.04	$--$	$--$	$--$
$QuadrRg$	4.38	$--$	$--$	$--$	5.69	$--$	$--$	$--$
$SingleN$	4.33	3.94	4.80	0.0018	5.44	4.93	6.20	0.0027
$3AvgN$	4.16	3.90	4.55	0.0009	5.28	4.93	5.82	0.0014
$3TempN$	4.04	3.97	4.60	0.0013	5.15	4.82	5.50	0.0012
$WorkD$	4.29	4.02	4.66	0.0018	5.68	5.13	6.67	0.0028
$FuzzyN$	4.24	3.97	4.60	0.0013	5.63	5.32	6.28	0.0016

Table 1: The average, the minimum and the maximum MAPE (in percent) and its standard deviation for D-type predictions.

W-type	RR				IR			
	AVG	MIN	MAX	SD	AVG	MIN	MAX	SD
$Naive$	10.64	$--$	$--$	$--$	12.74	$--$	$--$	$--$
$LinRg$	8.43	$--$	$--$	$--$	11.49	$--$	$--$	$--$
$QuadrRg$	8.34	$--$	$--$	$--$	11.52	$--$	$--$	$--$
$SingleN$	7.60	6.63	8.80	0.0049	10.32	9.16	11.60	0.0057
$3AvgN$	7.17	6.30	8.34	0.0028	9.69	8.50	10.91	0.0034
$3TempN$	7.04	6.34	7.78	0.0023	9.38	8.36	10.77	0.0033
$FuzzyN$	7.62	7.07	8.87	0.0037	10.44	9.57	12.27	0.0059

Table 2: The average, the minimum and the maximum MAPE (in percent) and its standard deviation for W-type predictions.

5. CONCLUSIONS

The quality of results achieved by neural networks and fuzzy neural networks is encouraging and acceptable from the natural gas company's viewpoint. Statistical methods used so far by the company yielded the average MAPE error for monthly predictions in the **RR** and **IR** regions in the period $01.2000-12.2001$ equal to 12.86% and 13.93%, resp. At the moment a β-version of the program written based on the methods described in the paper is tested in the company. The future goal is to develop an application capable to practically support the process of natural gas purchasing.

In the numerical evaluation of the results it should be taken into account that winter of the year 2002 was unusually cold (c.f. Fig. 7) and therefore making predictions for the period Nov.-Dec. was much more difficult in the year 2002 than in the two previous years.

4W-type	RR				IR			
	AVG	MIN	MAX	SD	AVG	MIN	MAX	SD
$Naive$	21.64	$--$	$--$	$--$	32.90	$--$	$--$	$--$
$LinRg$	11.19	$--$	$--$	$--$	17.14	$--$	$--$	$--$
$QuadrRg$	11.80	$--$	$--$	$--$	14.87	$--$	$--$	$--$
$SingleN$	8.30	6.69	9.94	0.0071	10.95	9.87	12.61	0.0062
$3AvgN$	7.73	5.88	9.06	0.0048	10.37	9.37	11.62	0.0035
$3TempN$	7.86	6.08	9.89	0.0054	10.43	9.30	11.57	0.0034
$FuzzyN$	8.16	7.29	9.10	0.0037	11.37	9.80	14.35	0.0092

Table 3: The average, the minimum and the maximum MAPE (in percent) and its standard deviation for 4W-type predictions.

Another comparison can be fairly made with the naive prediction approach, which was outperformed by neural methods, especially in the case of long-term prediction.

The comparison of our results with the literature is ambiguous since the data sets used in other works are different from ours and also the consumer profiles may be different. One example is the work of Khotanzad et al. (2000) where the MAPE error of 3.78% for D-type prediction involving temperature data is reported.

There are several directions in which this work can be continued. In the future we plan to test the efficacy of partially recurrent and RBF networks as well as other models of fuzzy neural networks for this task. Another issue is to test the quality of prediction models in case of other temperature clusterization methods.

ACKNOWLEDGEMENT

The authors would like to thank Mr R. Galbarczyk and Mr M. Łowczykowski from the Mazovian Gas Company - Poland for their help with collecting and preparing the data.

Jacek Mańdziuk was supported by the grant no. 503G 1120 0009 002 from the Warsaw University of Technology

REFERENCES

1. Bortolan, G., & Redrycz, W. (2002). Linguistic neurocomputing: the design of neural networks in the framework of fuzzy sets. *Fuzzy Sets and Systems*, v. 128, pp. 389–412.

2. Browna, R. H., Martin, L., Kharouf, P., & Piessens, L.P. (1996). Development of artificial neural-network models to predict daily gas consumption. *A.G.A. Forecasting Review*, v. 5, pp. 1–22.

3. Buckley, J.J., & Hayashi, Y., (1994). Fuzzy neural networks: A survey. *Fuzzy Sets and Systems*, v. 66, pp. 1–13.

4. Fine, L.T., (1999). *Feedforward Neural Network Methodology*, Springer-Verlag, New York.

5. Gupta, M.M., & Rao, D.H., (1994). On the principles of fuzzy neural networks. *Fuzzy Sets and Systems*, v. 61, pp. 1–18.

6. Karayiannis, N.B., (1996). Fuzzy Algorithms for Learning Vector Quantization. *IEEE Transactions on Neural Networks*, v. 7(5), pp. 1196–1211.

7. Khotanzad, A., Elragal, H., & Lu, T-L. (2000). Combination of artificial neural-network forecasters for prediction of natural gas consumption. *IEEE Transactions on Neural Networks*, v. 11(2), pp. 464–473

8. Kolmogorov, A.N., (1957). On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Dokl. Akad. Nauk ZSRR*, v. 114, pp. 953–956.

9. Kurkova, V., (2000). Rates of approximation by neural networks. In: Sincak, P. & Vascak., J., (Eds.) *Quo Vadis Computational Intelligence*, Springer, Berlin, pp. 23–26.

10. Martinetz, M., Berkovich, S., & Schulten, K. (1993). Neural-gas network for vector quantization and its application to time series prediction. *IEEE Transactions on Neural Networks*, v. 4, pp. 558–569.

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

12. Pal, N.R. & Pal, T., (1999). On rule pruning using fuzzy neural networks. *Fuzzy Sets and Systems*, v. 106, pp. 335–347.

13. Shann, J.J. & Fu, H.C., (1995). A fuzzy neural network for rule acquiring on fuzzy control systems. *Fuzzy Sets and Systems*, v. 71, pp. 345–357.

14. Sharkey, A.J.C., (1997). Modularity, combining and artificial neural nets. *Connection Science*, v. 9(1), pp. 3–10.

15. Vuorimaa, P., (1994). Fuzzy self-organizing map. *Fuzzy Sets and Systems*, v. 66, pp. 223–231.

16. Waibel, A., (1989). Consonant recognition by modular construction of large phonemic time-delay neural networks. In: Touretzky, D., (Ed.), *Advances in NIPS*, Morgan Kaufmann, v. 1, pp. 215–223.

17. Zhang, G., Patuwo, B.E., & Hu, M.Y., (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, v. 14, pp. 35-62.

APPENDIX 1. THE SCALED CONJUGATE GRADIENT ALGORITHM

Let us now assume that $\mathrm{E}(\mathrm{w})$ is a quadratic function with a positive hessian H. The idea of the conjugate gradient approach is to choose the search directions p_k and the learning rate α_k so that at step k, the gradient $\mathrm{g}(\mathrm{w}_k)$, denoted hereby as g_k, of the cost function is orthogonal to each previous search direction, i.e.:

$$\forall\, i : 0 \;\leq\; i \;\leq\; k,\; \mathrm{p}_i^{\mathrm{T}}\, \mathrm{g}_{k+1} \;=\; 0. \tag{17}$$

Let us assume that (17) is fulfilled. Note that:

$$\mathrm{g}_{k+1} \;=\; \mathrm{g}_k + \mathrm{H}\;(\mathrm{w}_{k+1} - \mathrm{w}_k) \;=\; \mathrm{g}_k + \alpha_k\, \mathrm{H}\, \mathrm{p}_k \;=\; \mathrm{g}_k + \alpha_k\, \mathrm{H}\, \mathrm{p}_k, \tag{18}$$

hence (17) can be rewritten as follows:

$$\forall\, i : 0 \;\leq\; i \;\leq\; k,\; \mathrm{p}_i^{\mathrm{T}}\; (\mathrm{g}_k \;+\; \alpha_k\, \mathrm{H}\, \mathrm{p}_k) \;=\; 0 \tag{19}$$

For $i = k$ we have:

$$\begin{aligned} \mathrm{p}_k^{\mathrm{T}}\; (\mathrm{g}_k \;+\; \alpha_k\, \mathrm{H}\, \mathrm{p}_k) \;&=\; 0 \\ \Leftrightarrow\; \alpha_k \;&=\; \frac{-\mathrm{p}_k^{\mathrm{T}}\, \mathrm{g}_k}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k} \quad (\text{note that } \mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k \;>\; 0), \end{aligned} \tag{20}$$

for $i < k$:

$$\begin{aligned} \mathrm{p}_i^{\mathrm{T}}\; (\mathrm{g}_k \;+\; \alpha_k\, \mathrm{H}\, \mathrm{p}_k) \;&=\; \mathrm{p}_i^{\mathrm{T}}\; \left(\mathrm{g}_k \;+\; \alpha_k\, \mathrm{p}_i^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k\right) \\ &=\; \alpha_k\, \mathrm{p}_i^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k. \end{aligned} \tag{21}$$

Thus:

$$\alpha_k \;=\; \frac{-\mathrm{p}_k^{\mathrm{T}}\, \mathrm{g}_k}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k} \;\text{ and } \mathrm{p}_i^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k \;=\; 0 \;\; \forall\, i \;<\; k \tag{22}$$

We have shown that (22) is the necessary condition for (17). We will now show by induction that it is sufficient as well. For $k \;=\; 0$, by choosing any $\mathrm{p}_0 \;\neq\; 0$ and then by applying (20) we can achieve $\mathrm{p}_0^{\mathrm{T}}\, \mathrm{g}_1 \;=\; 0$. Let (17) be true for all $k \;<\; k^*$ for some $k^* \;\geq\; 1$, i.e.:

$$\forall\, i : 0 \;\leq\; i \;<\; k^*,\; \mathrm{p}_i^{\mathrm{T}}\, \mathrm{g}_{k^*} \;=\; 0 \tag{23}$$

Let now $k = k^*$. Again by applying (20) we have:

$$\mathrm{p}_{k^*}^{\mathrm{T}}\, \mathrm{g}_{k^*+1} \;=\; \mathrm{p}_{k^*}^{\mathrm{T}}\, (\mathrm{g}_{k^*} \;+\; \alpha_{k^*}\, \mathrm{H}\, \mathrm{p}_{k^*}) \;=\; 0 \tag{24}$$

Furthermore, for $i < k = k^*$, according to (22) we have $\mathrm{p}_i^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_{k^*} = 0$, then taking under consideration (23) we can write:

$$\begin{aligned} \mathrm{p}_i^{\mathrm{T}}\, \mathrm{g}_{k^*+1} \;&=\; \mathrm{p}_i^{\mathrm{T}}\, (\mathrm{g}_{k^*} \;+\; \alpha_{k^*}\, \mathrm{H}\, \mathrm{p}_{k^*}) \\ &=\; \mathrm{p}_i^{\mathrm{T}}\, \mathrm{g}_{k^*} \;+\; \alpha_{k^*}\, \mathrm{p}_i^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_{k^*} \;=\; 0, \end{aligned}$$

which ends our proof.

We have shown that the gradient g_{k+1} of the quadratic function $\mathrm{E}\,(\mathrm{w})$ at step $k+1$ will be orthogonal to all search directions p_i, $i \leq k$ if and only if the search directions are chosen to be H orthogonal and the learning rate is chosen as specified in (22). Moreover, as shown in (Fine, 1999), in the case of quadratic error function, this algorithm leads to the global minimum in no more than N steps, where N is parameters' space dimension.

We will now discuss how to iteratively choose the H-orthogonal search directions. The first vector $\mathrm{p}_0 \neq 0$ can be chosen arbitrarily. For $k > 0$ we choose:

$$\mathrm{p}_k \;:=\; -\mathrm{g}_k \;+\; \beta_{k-1}\, \mathrm{p}_{k-1}, \tag{25}$$

where:

$$\beta_k \;=\; \frac{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{g}_{k+1}}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k} \tag{26}$$

It has been proven in (Fine, 1999) that the search direction vectors chosen in such way fulfill the conditions given in (17). The hessian H can be eliminated from (26) as follows (the Polak-Ribiere formula):

$$\beta_k \;=\; \frac{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{g}_{k+1}}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k} \;=\; \frac{\mathrm{g}_{k+1}^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, \mathrm{p}_k} \;=\; \frac{\mathrm{g}_{k+1}^{\mathrm{T}}\, \alpha_k\, \mathrm{H}\, \mathrm{p}_k}{\mathrm{p}_k^{\mathrm{T}}\, \alpha_k\, \mathrm{H}\, \mathrm{p}_k} \;=\; \frac{\mathrm{g}_{k+1}^{\mathrm{T}}\, (\mathrm{g}_{k+1} \;-\; \mathrm{g}_k)}{\mathrm{p}_k^{\mathrm{T}}\, (\mathrm{g}_{k+1} \;-\; \mathrm{g}_k)}, \tag{27}$$

which leads to:

$$\mathrm{p}_k \;=\; -\mathrm{g}_k \;+\; \frac{\mathrm{g}_k^{\mathrm{T}}\, (\mathrm{g}_k \;-\; \mathrm{g}_{k-1})}{\mathrm{p}_{k-1}^{\mathrm{T}}\, (\mathrm{g}_k \;-\; \mathrm{g}_{k-1})}\, \mathrm{p}_{k-1} \tag{28}$$

Up to now we have been assuming that the cost function $\mathrm{E}\,(\mathrm{w})$ is quadratic with a (constant) positive hessian H. In fact, the cost function $\mathrm{E}\,(\mathrm{w}, \mathbf{T})$ defined by eq. (2) can only be approximated by some quadratic function $\overline{\mathrm{E}}\,(\mathrm{w})$ (e.g. by using the Taylor series expansion), and its hessian need not always be positive. Hence the computation of the learning rate:

$$\alpha_k \;=\; \frac{-\mathrm{p}_k^{\mathrm{T}}\, \mathrm{g}_k}{\mathrm{p}_k^{\mathrm{T}}\, \mathrm{H}\, (\mathrm{w}_k)\; \mathrm{p}_k}, \tag{29}$$

which involves $\mathrm{H}(\mathrm{w})$, may be problematic. In the scaled conjugate gradient approach, it can be avoided providing:

$$\mathrm{H}(\mathrm{w}_k)\ \mathrm{p}_k \approx \mathrm{s}_k := \frac{\mathrm{g}(\mathrm{w}_k + \sigma_k \mathrm{p}_k) - \mathrm{g}(\mathrm{w}_k)}{\sigma_k} + \lambda_k \mathrm{p}_k, \tag{30}$$

where $\lambda_k \mathrm{p}_k$ is a *regularization* factor and $0 < \sigma_k \ll 1$. The value of λ_k must be chosen so that $\delta_k := \mathrm{p}_k^{\mathrm{T}} s_k > 0$. Let us assume that at step k, for a given value λ_k, δ_k is negative. In this case, we must find another value of λ_k so that $\delta_k > 0$. Let us assume that such a value exists and denote it by λ_k^*. Let us denote the corresponding value for s_k by s_k^*. Observe that:

$$s_k^* = s_k + (\lambda_k^* - \lambda_k)\,\mathrm{p}_k, \tag{31}$$

then we have:

$$\begin{aligned} 0 < \delta_k^* = \mathrm{p}_k^{\mathrm{T}} s_k^* &= \mathrm{p}_k^{\mathrm{T}} s_k + (\lambda_k^* - \lambda_k)\,||\mathrm{p}_k||^2 \\ \Rightarrow \lambda_k^* &> \lambda_k + \frac{-\delta_k}{||\mathrm{p}_k||^2}. \end{aligned} \tag{32}$$

Though we should increase λ_k by some factor which is greater than $\frac{-\delta_k}{||\mathrm{p}_k||^2}$ to obtain a positive value of δ_k. In (Moller, 1993) λ_k^* is chosen as follows:

$$\lambda_k^* = 2\left(\lambda_k - \frac{\delta_k}{||\mathrm{p}_k||^2}\right), \tag{33}$$

which leads to

$$\delta_k^* = \delta_k + (\lambda_k^* - \lambda_k)\,||\mathrm{p}_k||^2 = -\delta_k + \lambda_k ||\mathrm{p}_k||^2, \tag{34}$$

and finally:

$$\alpha_k^* = \frac{-\mathrm{p}_k^{\mathrm{T}}\,\mathrm{g}_k}{-\mathrm{p}_k^{\mathrm{T}} s_k + \lambda_k ||\mathrm{p}_k||^2}. \tag{35}$$

As it can be observed in (35), the coefficient λ_k has some influence on the learning rate, and consequently on the minimization process itself. In practice, values for λ_k are chosen adaptively depending on the training progress.