Charles University in Prague

Faculty of Social Sciences

Institute of Economic Studies

DIPLOMA THESIS

2006 Jozef Baruník

Charles University in Prague

Faculty of Social Sciences

Institute of Economic Studies

DIPLOMA THESIS

On the predictability of Central European stock returns

"Do Neural Networks outperform modern econometric techniques?"

Author: Jozef Baruník

Supervisor: PhDr. Filip Žikeš

Academic year: 2005/2006

Declaration:

Hereby I claim that I elaborated this diploma thesis on my own, and that the only literature and sources I used are those listed in references.

July the 14th, 2006

Author's signature

ABSTRACT

In this thesis we apply neural networks as nonparametric and nonlinear methods to the Central European stock markets returns (Czech, Polish, Hungarian and German) modelling. In the first two chapters we define prediction task and link the classical econometric analysis to neural networks. We also present optimization methods which will be used in the tests, conjugate gradient, Levenberg-Marquardt, and evolutionary search method. Further on, we present statistical methods for comparing the predictive accuracy of the non-nested models, as well as economic significance measures. In the empirical tests we first show the power of neural networks on Mackey-Glass chaotic time series followed by real-world data of the daily and weekly returns of mentioned stock exchanges for the 2000:2006 period. We find neural networks to have significantly lower prediction error than classical models for daily DAX series, weekly PX50 and BUX series. The lags of time-series were used, and also cross-country predictability has been tested, but the results were not significantly different. We also achieved economic significance of predictions with both daily and weekly PX-50, BUX and DAX with 60% accuracy of prediction. Finally we use neural network to learn Black-Scholes model and compared the pricing errors of Black-Scholes and neural network approach on the European call warrant on CEZ. We find that networks can be used as alternative pricing method as they were able to approximate the market price of call warrant with significantly lower error then Black-Scholes itself. Our last finding was that Levenberg-Marquardt optimization algorithm used with evolutionary search provides us with significantly lower errors than conjugate gradient or gradient descent.
Keywords: emerging stock markets, predictability of stock returns, neural networks, optimization algorithms, derivative pricing using neural networks
JEL classification: C22, C32, C45, C53, E44, G14, G15

ABSTRAKT (in Czech)

V této práci jsou aplikovány neuronové sítě jako neparametrická, nelineární metoda modelování na středoevropské trhy (Český, Polský, Maďarský a Německý). V prvních dvou kapitolách je definováno prognózování v kontextu klasické ekonometrické analýzy ve spojení s neuronovými sítěmi. Dále jsou prezentovány optimalizační metody použité při testování – konjugovaný gradient, Levenberg-Marquardt a genetické algoritmy, a nakonec statistické metody pro srovnání přesnosti předpovědí různých modelů a jejich ekonomickou signifikaci. V empirickém modelování je nejdřív ukázána výkonnost neuronové sítě na chaotické časové řadě Mackey-Glass. Dále následuje analýza reálných denních a týdenních časových řad středoevropských indexů pro období let 2000 až 2006, kde je ukázáno, že Neuronové sítě predikují denní výnosy DAX a týdenní výnosy PX50, BUX se signifikantně nižší chybou pomocí časových řad historických výnosů než ostatní ekonometrické metody. Podobných výsledků bylo dosaženo při predikci národního výnosu pomocí zpožděných výnosů alespoň jednoho z ostatních indexů. Dále je taky ukázáno, že s Neuronovou sítí byla dosažena ekonomická signifikace predikce denních i týdenních výnosů PX-50, BUX i DAX. Přesnost předpovědí testovaných řad se pohybuje kolem 60%, co považujeme za dobrý výsledek. V poslední kapitole je použita neuronová síť pro ocenění Evropského nákupního warrantu na ČEZ za pomoci časové řady historických cen. Je ukázáno, že síť je možné použít i jako alternativu pro oceňování, jelikož dokáže aproximovat tržní cenu lépe než Black-Scholesův model. Poslední testy ukázaly, že Levenberg-Marquardtova optimalizační metoda použita s genetickým algoritmem vykazuje signifikantně nižší chyby odhadů než ostatní metody.
Klíčová slova: výnosy akcií a jejich predikce pomocí neuronové sítě, optimalizační algoritmy, oceňování derivátů pomocí neuronové sítě
JEL klasifikace: C22, C32, C45, C53, E44, G14, G15

Contents

Acknowledgments

First and foremost I would like to thank Filip Žikeš from the Faculty of Social Sciences, Charles University for his guidance, many useful suggestions and valuable comments, and for supervising my work on this thesis. I also owe a great deal to people from Brokerjet a.s. (Prague) for giving me the chance to understand the market behavior from its "inside", specially to Petr Ondřej and Tomáš Provazník for various discussions on the trading issues for past three years.

Last, but not least, I would like to thank to my parents for their never-ending love and support.

Introduction

"One of the earliest and most enduring questions of financial econometrics is whether financial asset prices are forecastable. Perhaps because of the obvious analogy between financial investments and games of chance, mathematical models of asset prices have an unusually rich history that predates virtually every other aspect of economic analysis. The fact that many prominent mathematicians and scientists have applied their considerable skills to forecasting financial securities prices is a testament to the fascination and the challenges of this problem. Indeed, modern financial economics is firmly rooted in early attempts to "beat the market", an endeavor that is still of current interest, discussed and debated in journal articles, conferences, and at cocktail parties!"

Campbell, Lo and MacKinlay (1997), p.27

Life must be understood looking backwards, but must be lived looking forward. The past is helpful for predicting the future, but we have to know which approximating models to use, in combination with past data, to predict future events. Žikeš (2003) finds that European stock returns do not follow random walk, thus contains predictable components, and presents modern econometric techniques which helps us to uncover part of the pattern. We would like to link these methods with neural networks research and provide a useful bridge which lacks in most of the literature. This thesis is an extension of previous work aimed on the predictability of Central European stock markets returns, presenting the neural network approach to the problem.

On the basis of universal approximation theorem, we use the neural networks with hope they will improve the prediction task as they are able to approximate any function as Hornik, Stinchcombe, and White (1989) shows. Thus, we will aim on comparison of results of econometric modelling and neural network modelling to see whether neural networks brings us closer insight into the patterns of stock returns or not. The readers shall see that the neural network

is a very useful nonparametric econometric technique. Criticisms rise mainly from the fact that neural networks drew their motivation from biological phenomena, from physiology of nerve cells, they have become part of a separate literature (see Hertz, Krogh and Palmer (1991), Hutchinson, Lo, and Poggio (1994), Poggio and Girosi (1990), and White (1988) resp. (1992) for the overview). We will also append this discussion in this thesis. The structure will be as followed:

We start with theoretical framework of stock returns predictability in the first chapter, where we present Efficient Market Hypothesis, define the prediction task, and present linear regression models and GARCH modelling.

In the second chapter we move further on to neural networks. We discuss methodology problems first to avoid confusion, then we present basic forms of networks and transformation functions which will be tested further in the next two chapters. We also discuss the most important - optimization methods used. Starting with quasi-Newton stochastic gradient search, through conjugate gradient and Levenberg-Marquardt we get to stochastic evolutionary searches and discuss nonlinear estimation problem. At the end of the chapter we pay attention to the evaluation of estimated models, and to statistical methods of predictive accuracy and economic significance. We close the chapter with Black-Box criticism discussion where we comment on its irrelevance.

In the third chapter we apply presented methods to central European stock market returns. We start with the modelling of Mackey-Glass's chaotic time series to show how neural network perform on artificial data. On the basis of general approximation theorem we expect the neural network to approximate the process very well. We will also compare it to common techniques presented in the first chapter to illustrate the power of the networks. In the rest of the chapter we model the PX-50, BUX, DAX and WIG daily and weekly returns. On the in-sample and more important out-of-sample criteria we test classical autoregressive models, ARIMA (p,I,q) and GARCH with neural networks. For the comparison we use statistical tests described in the theoretical part, and also tests of economic relevance of the prediction model.

In the last chapter we examine the usage of neural networks to derivatives pricing. If the price of derivative is determined by the Black-Scholes formula, neural network can be used to estimate the Black Scholes formula with sufficient degree of accuracy. If the assumptions of Black-Scholes model are violated, the neural networks can be used as better and more efficient derivative pricing models. We follow this analysis as the logical implication from findings in the third chapter, while assumptions of Black Scholes as lognormal distribution of stock

prices, geometric Brownian motion, constant volatility or frictionless markets are nonrealistic, we expect the neural network to be able to price the derivatives more efficiently. We conduct the empirical analysis on the European call warrant on the CEZ, the second most liquid security on the Czech stock market. The methodology is simple. Firstly we test if the neural network is able to approximate the Black-Scholes on the artificial data on the call warrant on CEZ. Then we will use real market prices and test if the neural networks can be used as the nonparametric derivative pricing method effectively than Black-Scholes itself.

The thesis concludes with summary of the empirical results we achieve and suggestions for further research.

Chapter 1

Stock returns predictability using modern econometric methods

Predictability of stock returns have been attracting the attention of many academics and professionals for a long time[1]. It concerns forecasting future returns from the past – observed – returns as well as cross-sectional forecasting from other - financial or macroeconomic - variables[2] that relates to the returns. The basic assumption is that history tends to repeat itself, meaning that past patterns of price behavior in individual stocks will tend to repeat in future. Thus the way to predict the future of returns is to develop and uncover those patterns. The economic rationale for doing so is very strong: abnormal returns. At a first glance, the problem seems to be simple. All we need is historical prices of the returns which we want to forecast, and *"user-friendly"* econometric software which will do the work for us and recognize the patterns in the data. Costs are negligible even to a common investor and possible results of correctly modeled returns are very attractive.

This chapter outlines commonly used techniques for time series prediction, and presents enhanced modern econometric methods for modelling of time series and detecting the presence of regular patterns. Although it presents most of the

[1] Campbel, Lo, MacKinlay (1997) can be used to find references addressing almost any question of the problem. Hellstrom, Holmstrom (1998), Hawanini and Keim (1993).

[2] Main reference to this research are Fama and French (1988, 1989, 1990), Chen, Roll and Ross (1986), Barro (1990)

important concepts and brings the reader in the problem, it serves just as an introductory chapter to the main concept – neural networks presented in this thesis.

The organization is as follows. Firstly, the Efficient Market hypothesis, an idea which stands at the beginning of this research is presented in its three forms in (1.2). Martingale and Random Walk processes helps to close the basic framework of stock returns predictability. In (1.4) we present Classical Linear regression modelling with more general autoregressive and ARIMA (p,1,q) models. Subchapter (1.5) follows with exploring nonlinear, time-varying models which stands on the generalized autoregressive conditional heteroskedasticity, GARCH.

1.1 Properties of stock returns time-series

First of all we present basic properties of stock returns as the motivation. All of the problems will be discussed in detail in following subchapters thus the reader can find references there. Also statistical and distributional properties (i.e. heavy tails) will not be mentioned here as we will discuss them further in empirical testing of the presented models. This part should only serve as an essential introduction of the basic concepts of stock returns predictability.

i) Stock returns time series often behave nearly like a random-walk process, which means that from a theoretical point of view there are no predictable regular patterns. Predictability of stock returns have also been questioned in scope of the efficient market hypothesis.
ii) Statistical properties of the time series are different at different points in time.
iii) Financial time series are very noisy, meaning that there is a large amount of random day-to-day variations.

1.2 Efficient market hypothesis

The efficient market hypothesis (EHM) has been one of the most important concepts in modern financial theory as it has found broad acceptance[3]. As summarized by Fama (1970), "a market in which prices always 'fully reflect'

[3] Anthony and Biggs (1995), Malkiel (1987), White (1998)

available information is called 'efficient'." As Campbel, Lo, MacKinlay (1997) remarks, quotation marks 'fully reflects' are prompting that the formulation needs to be explained in detail. Malkiel (1987) expands the Fama's definition with the idea of judging efficiency of market by measuring the profits that can be made by trading on the available information. He writes: "If the market is efficient, it is impossible to make economic profit by trading on the information." Thus if the current price reflected all information available at the market, no prediction of future changes would be possible. As new information enters the market, it is immediately reflected and new market price is developed. Depending on the type of information set, Roberts (1967) distinguishes

- **Weak-form Efficiency:** The information set includes only the history of the prices or returns themselves. In other words, technical analysis[4] is of no use.
- **Semistrong-form Efficiency:** The information set includes all *publicly available* information known to all market participants. In other words, fundamental analysis[5] is of no use.
- **Strong-form Efficiency:** The information set includes all *privately available* information known to any market participant. In other words, even insider information is of no use.

As we consider stock returns predictability at this work, we will work only with weak-form efficiency which enables us to hope that we will be able to predict the future returns from the past ones.

1.2.1 Martingale model

Martingale model was perhaps the earliest idea of financial asset pricing models, which grew from the history of game of chances and probability theory. Girolamo Cardano (1565) proposed that the "most fundamental principle of

[4] Technical analysis is based on creating various basic indicators as trend-lines, support and resistance, volatility, momentum indicators etc. from past prices and volume. Indicators are used to produce trading (buy/sell) signals or rules. This is done mainly graphically by comparing the price and a trading rule.

[5] Fundamental analysis is mainly based on the financial analysis of the company's value aiming on profitability, efficiency and true value of company's stock.

gambling is equal conditions." Thus by the means of a fair game, the stochastic process $\{P_t\}_{t=0}^{\infty}$ satisfies the following condition:

$$\mathrm{E}\left[P_{t+1}\middle|\mathcal{F}_t\right]=P_t, \tag{1.1}$$

where P_t is stock price at time t and is $\mathcal{F}_t$-measurable, $\mathrm{E}\left[P_{t+1}\middle|\mathcal{F}_t\right]$ are conditional expectations defined on the probabilistic space $\left(\Omega,\mathcal{F},\{\mathcal{F}_t\},\mathbb{P}\right)$, where Ω is the space of market situations, $\mathcal{F}$ is σ-algebra of the subsets of Ω, $\{\mathcal{F}_t\}$ is the usual filtration, $\mathcal{F}_t=\sigma\{P_t,P_{t-1},...,P_1\}$, which is also called information set, and $\mathbb{P}$ is a probability measure on $\mathcal{F}$. Then tomorrow's price is expected to be equal to today's price given the historical prices as information set. Martingale hypothesis implies that the expected return is zero as:

$$\mathrm{E}\left[P_{t+1}\middle|\mathcal{F}_t\right]=P_t+\mathrm{E}\left[r_{t+1}\middle|\mathcal{F}_t\right], \tag{1.2}$$

or if equation (1.1) holds,

$$\mathrm{E}\left[P_{t+1}-P_t\middle|\mathcal{F}_t\right]=\mathrm{E}\left[r_{t+1}\middle|\mathcal{F}_t\right]=0, \tag{1.3}$$

where r_t is stock price change. The reader should note that martingale hypothesis implies that price changes are uncorrelated at all lags. Increments in value (changes in price) are unpredictable and conditional on the information set which is fully reflected in prices. Hence any attempt of linear and nonlinear forecasting rules is ineffective, as

$$Cov_t\left[f\left(r_t\right),g\left(r_{t+j}\right)\right]=0, \tag{1.4}$$

where $f(.)$ and $g(.)$ are two arbitrary functions $\forall f,g:\mathbb{R}\to\mathbb{R}$, r_t and r_{t+j} are stock price changes, or returns in two periods for all t and $j\neq 0$.

In fact, the martingale was considered to be a necessary condition for an efficient market. Roberts (1967) considers it to be a weak-form market efficiency.

Main drawback of the martingale model is that it does not allow a trade-off between risk and expected return. If the expected return was zero, no one would invest in the security. It has been shown that martingale is neither a necessary nor a sufficient condition for rational markets[6].

[6] i.e. Leroy (1973)

1.2.2 Random Walk model

The martingale model given by (1.1) resp. (1.2) can be rewritten equivalently as

$$P_{t+1} = P_t + \varepsilon_t , \qquad (1.5)$$

where $\{\varepsilon_t\}$ is a martingale difference sequence. In this form, it is nearly identical with the random walk model, the forerunner of the theory of efficient capital markets. The martingale, however, is less restrictive than the random walk. It requires only independence of the conditional expectation of price changes from the information available. Random walk model requires, furthermore, independence involving the higher conditional moments of the probability distribution of price changes.

Campbel, Lo and MacKinlay (1997) distinguish between three versions of the random walk hypothesis. The simplest one is *Random Walk 1* or *RW1,* the independently and identically distributed - *iid*[7] increments in which the dynamics of $\{p_t\}$ [8]is given by:

$$p_t = \mu + p_{t-1} + \varepsilon_t, \qquad \varepsilon_t \sim \left(0, \sigma^2\right), \qquad (1.6)$$

where ε_t is an random variable with zero mean, variance σ^2 and μ is the expected price change or *drift*. Conditional mean and variance are linear functions of time[9], which implies that random walk is nonstationary. We will assert that natural logarithm of prices follows random walk with *iid* increments to avoid the problem of limited liability of stock returns. If the $\{P_t\}$ was normally distributed, there would always be positive probability of $P_t < 0$ which is unrealistic.

Random Walk is thus sufficient but not necessary condition for market efficiency in its weak-form. Hence rejecting the null hypothesis H_0 that stock returns follow random walk does not mean market inefficiency. The second version, *RW2,* also relaxes the identical distribution assumption which allows time-varying, unconditional volatility. RW1 is thus a special case of RW2 which contains more general price processes and allows for unconditional

---

[7] *iid* will be used from this point as standard notation for independently and identically distributed variable

[8] Continuously compounded returns $r = p_t - p_{t-1}$, where p_t is natural logarithm of price $p_t = \ln P_t$.

[9] $\mathrm{E}\left[p_t \middle| p_0\right] = p_0 + \mu t, \;\; Var\left[p_t \middle| p_0\right] = \sigma^2 t.$

heteroskedasticity[10]. RW3 is an even more general version – one most often tested in the literature – which relaxes the independence assumption and includes price processes with dependent but uncorrelated increments. Lo, MacKinlay (1988) exploits simple Random walk tests in detail. We will not describe the tests here as the reader can follow the reference if needed.

Now, when we have discussed the basic idea of stock return predictability, we can move on to more sophisticated methods, but before we do so, a short conclusion of EHM framework will be carried out. The paradox of efficient markets is that if every investor believed a market was efficient, then the market would not be efficient because the participants would not want to trade as they would not expect the profit. In effect, efficient markets depend on market participants who believe the market is inefficient and trade securities in an attempt to outperform the market. For deeper analysis, see Grossman, Stiglitz (1980)

Although market efficiency is not really testable because of *joint hypothesis*[11], it provides a basic framework of stock returns prediction. It started the discussion, and non-rejecting Random Walk hypothesis implies that there are no patterns to be found in the stock returns.

Even we can not test the market efficiency, in reality we find most of the markets to be neither perfectly efficient nor completely inefficient. For evidence, Cambazoglu (2003), Hellstrom, Holstrom (1998), Lo, MacKinlay (1988), Žikeš (2003) and much more researchers found predictable patterns at various world stock markets and provided an evidence that tested markets are predictable to some extent. From the other point of view, we can say that all markets are efficient to a certain extent, some more so than others. *"Rather than being an issue of black or white, market efficiency is more a matter of shades of gray"* [12]. In markets with substantial impairments of efficiency, more knowledgeable investors can outperform less knowledgeable ones. Hence, abnormal returns, even if small ones, will necessarily exist to compensate participants for taking their risk, even if predictable patterns will not be found. This debate is the starting point for predictability models which will be discussed in next chapters.

---

[10] In recent literature reader can find dozens of empirical evidence that returns are conditional heteroskedastic. i.e. Campbel, Lo and MacKinlay (1997) contains the reference

[11] Any test of efficiency must assume an equilibrium model that defines normal returns. Rejecting market efficiency implies that market is truly inefficient or an incorrect equilibrium model has been assumed. Hence, market efficiency as such can never be rejected, Fama (1991)

[12] Lo, MacKinlay (1988)

1.3 Definition of the prediction task

Prediction problem can be formulated in various ways. We will restrict on defining the stock returns prediction, as it is the primary concern of the thesis, even if the stock prices are not the only financial time-series of the general economist's interest. General prediction can be defined as follows:

Let P_t *be a random variable defined on a probability space* $\left(\Omega, \mathcal{F}, \{\mathcal{F}_t\}, \mathbb{P}\right)$, *where* Ω *is space of outcomes,* $\mathcal{F}$ *is* σ*-algebra of the subsets of* Ω*, and* $\mathbb{P}$ *is a probability measure on* $\mathcal{F}$ *and* $\{\mathcal{F}_t\}$ *is the usual filtration. A conditional probability* $\mathbb{P}\left[P_{t+1} \middle| \mathcal{F}_t\right]$ *is conditional probability of the set* P_t *being evaluated with the information available in the* σ*-algebra* $\mathcal{F}$*.*

Now let us assume following economic agent's utility functions:

$$u\left(W_{t+h}\right) = g\left(P_{t+h}, \gamma\left(\hat{P}_{t+h}\right)\right), \qquad (1.7)$$

where agent's utility $u(.)$ depends on the variable P in time $t+h$, decision function $\gamma(.)$ and forecast $\hat{P}$ with forecasting horizon $h \geq 1$, and w is an reward variable. For illustration, let us set $h=1$. At time $t+1$, agent's utility depends on the realization of p_{t+1}, and accuracy of it's forecast, $\hat{p}_{t+1}$. Forecasting is defined as major factor of a decision rule.

Let $E\left[P_{t+h} \middle| \mathcal{F}_t\right] = \hat{P}_{t+h|t} = h\left(X_t, \theta\right)$ *be an expectation of* P_{t+h} *conditional on the information set* $\mathcal{F}_t$*, where* $\theta \in \Theta$ *is unknown vector of parameters, where* $\Theta \subseteq \mathbb{R}^k$ *is compact and observable at time* t*,* X_t *is an* $\mathcal{F}_t$*-measurable vector of variables.*

X_t may include P_{t-n} information, but also some exogenous variables, indicators, etc. Thus the reader may note that an optimal forecast from our definition does not exclude misspecification or failure to include relevant information in X_t, which may have crucial impact on the predictions. Under this imperfect setting, utility function will be negatively correlated with forecast error which can be defined as $\varepsilon_{t+h|t} = p_{t+h} - \hat{p}_{t+h|t}$.

Maximizing utility function requires to find optimal forecast $\hat{P}^*_{t+h}$ and to establish optimal decision $\gamma(.)$ based on this forecast. Optimality here can be achieved by minimizing expected loss function $L:\mathbb{R}\to\mathbb{R}^+$:

$$\hat{P}^*_{t+h|t} \equiv \arg\min_{\theta\in\Theta} E\left[L\left(P_{t+h}, X, \theta, \alpha\right)\middle|\mathcal{F}_t\right], \tag{1.8}$$

were α is a degree of asymmetry. The reader can find in-depth discussions of possible error functions with assumptions in Patton, Timmermann (2004, 2006) reference as general definition of Loss function is sufficient for our definition of prediction task. Rigorous discussion of prediction task can also be found in Hamilton (1994). For illustration, we define just optimal forecast depending on loss function which depends only on forecast errors. This form[13] will be also used further in our tests:

$$\hat{P}^*_{t+h|t} \equiv \min E\left[L\left(P_{t+h}-\hat{P}_{t+h|t}\right)\middle|\mathcal{F}_t\right] = \min E\left[L\left(\varepsilon_{t+h|t}\right)\middle|\mathcal{F}_t\right]. \tag{1.9}$$

Later in the chapter (2.8) - Statistical Comparison of Predictive Accuracy, we will present an optimal forecast under the different loss functions.

In next sections we will consider classical linear and nonlinear regression models as common choices of estimating $E\left[P_{t+h}\middle|\mathcal{F}_t\right]$, through which we will get to another possibilities, neural network models

1.4 Linear regression models

Mounting evidence in the literature can be found, that stock prices do not follow random walk. Lo, MacKinlay (1988) decisively reject the null hypothesis that U.S. stock weekly returns are the random walk process. Žikeš (2003) finds that Central European markets also do not follow random walk. Filacek et al. (1998) find that daily returns of PSE's[14] main index PX-50 are significantly positively autocorrelated. In this subchapter we will introduce basic linear and nonlinear regression models, so the principle of the modern forecasting techniques can be extended in next chapters by Neural Network models.

[13] i.e. MSE – mean squared error, MAE – Mean absolute error has this form

[14] Prague Stock Exchange, Czech Republic

1.4.1 Classical regression model

When predicting, we usually start with a linear regression model, where a given output variable y is predicted from information on a set x of observed variables. In time series, input variables might include lagged output variable or contemporaneous exogenous variables. The model is defined by following equation:

$$y_t = \sum_{i=1}^{p} \beta_i x_{i,t} + \varepsilon_t \,, \qquad (1.10)$$

$$\varepsilon_t \sim N\left(0, \sigma^2\right),$$

where ε_t is random disturbance term, $E\left[\varepsilon_t \middle| x_t\right] = 0$. $\left\{\beta_p\right\}$ are parameters to be estimated, while $\left\{\widehat{\beta_p}\right\}$ represents estimated set of coefficients and $\left\{\widehat{y_p}\right\}$ denotes estimated (predicted) output variables. The main goal is to find $\left\{\widehat{\beta_p}\right\}$ to minimize the sum of squared differences, or residuals ψ between the observed y variable and the model-predicted $\hat{y}$ variable. There are a various ways and estimation methods[15] of the problem:

$$Min\psi = \sum_{t=1}^{T} \hat{\varepsilon}_t^{\,2} = \sum_{t=1}^{T} \left(y_t - \hat{y}_t\right)^2 , \qquad (1.11)$$

where

$$y_t = \sum_{i=1}^{p} \beta_i x_{i,t} + \varepsilon_t \,,$$

$$\hat{y}_t = \sum_{i=1}^{p} \beta_i x_{i,t} \,,$$

$$\varepsilon_t \sim N\left(0, \sigma^2\right).$$

[15] with different assumptions about distribution of the disturbance term ε_t, or about the constancy of its variance σ^2, as well as about the independence of the input variable, reader can find these methods at any standard econometric textbook, i.e. Greene (1993) or Baltagi (2002)

1.4.2 Autoregressive model

Commonly used linear model which enhances classical regression is an autoregressive model:

$$y_t = \sum_{i=1}^{p} \beta_i y_{t-i} + \sum_{j=1}^{q} \gamma_j x_{j,t} + \varepsilon_t \ , \qquad (1.12)$$

where are $\varepsilon_t \sim N\left(0, \sigma^2\right)$, and where there are q exogenous x variables with coefficients γ_j, p lags of the dependent variable y and $p+q$ coefficients to be estimated. In the time-series model this is known as the linear ARX model, since the autoregressive components are given by lagged y variables and it incorporates exogenous x variables.

1.4.3 The ARIMA (*p*,1,*q*) model

Generalization of simple Random Walk Model and Autoregressive Model is allowing for serial correlation in the disturbances ε_t. *Autoregressive integrated moving average* model - ARIMA (*p*,1,*q*) - is the most applied linear model for approximation of stock returns processes. It puts together three processes for modelling the serial correlation in the disturbances: AR (*p*), MA (*q*) and integration order term. The processes are as follows.

AR (*p*) process includes *p* lagged values of the returns in the forecasting equation for the unconditional residual. An autoregressive model of order *p* has the form:

$$r_t = \sum_{i=1}^{p} \rho_i r_{t-i} + \varepsilon_t \ , \qquad (1.13)$$

or represented using lag operator ***L***. $\forall n \in \{1, ..., p\} : L^n r_t = r_{t-n}$:

$$\left(1 - \left(\sum_{i=1}^{p} \rho_i L^i\right)\right) r_t = \varepsilon_t \ . \qquad (1.14)$$

The second, integration order term corresponds to differencing the values being forecast. In this model, the first difference is enough as the stationarity can be achieved. Third, MA (*q*) process uses lagged values of forecast error to improve the current forecasts. For the *q* order it has the form:

$$r_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \,, \tag{1.15}$$

or

$$r_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right) \varepsilon_t \,. \tag{1.16}$$

Thus ARIMA (p,1,q)[16] model can be generally represented by:

$$\left(1 - \sum_{i=1}^{p} \rho_i L^i\right)(1-L) r_t = \mu + \left(1 + \sum_{j=1}^{q} \theta_j L^j\right) \varepsilon_t \,. \tag{1.17}$$

A common way to estimate the ARIMA (*p*,1,*q*) was proposed by Box and Jenkins (1976). Time series needs to be differenced to achieve stationarity. Then the guess of *p* and *q* is made by observing autocorrelation and partial correlation functions. Nonlinear least squares or Maximum likelihood method is then applied to estimate the model, and diagnostic tests are run to see if the guess of *p* and *q* orders was appropriate. Box-Jenkins methodology is widely used and the reader can find the details in Box and Jenkins (1976).

While choosing p,q as a "let the data speak" process is being attacked by researchers because it is a process of guessing, ARIMA model still helps the researchers in understanding of behavior of the stock prices. Linear models may become of very good use mainly on the markets with long-term trends with only small symmetric changes in the variable. However, for the volatile markets, nonlinear processes in the returns may come into the researcher's sight. Thus, linear models may fail to capture the turning points, bubbles and unexpected moves in the prices. For this reason, we will present nonlinear forecasting techniques.

1.5 GARCH models

There are many types of nonlinear functional forms to use as an alternative to linear ones. The main approach is the GARCH-type models[17]. These models are based on the main principles of the modern finance – risk which is related to an expected stock returns. To measure the risk of an asset, the standard deviation of returns from unconditional mean is used. This measure is also interpreted as the volatility of a stock returns hence main use of GARCH

[16] Note that ARIMA (0,1,0) is a random walk which is a special case of this general process.

[17] GARCH stands for generalized autoregressive conditional heteroskedasticity. The model was introduced by Engle (1982) who received the Nobel price in 2003 for his work on this model and generalized by Bollerslev (1986).

models is for volatility prediction. Following describes a general GARCH(*r*,*p*) model:

$$r_t = \beta_0 + x_t^T \beta_1 + \varepsilon_t \,, \tag{1.18}$$

$$\varepsilon_t \approx \phi\left(0, \sigma_t^2\right),$$

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{n} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{m} \delta_j \sigma_{t-j}^2 \,, \tag{1.19}$$

where r is rate of return, ε_t is normally distributed with zero mean and conditional variance σ^2. α's and δ's represent evolution of conditional variance. Condition $\sum_{i=1}^{\max(p,q)} \left(\alpha_i + \delta_i\right) < 1$ is imposed so the unconditional variance is finite, whereas its conditional variance evolves over time.

For the demonstrative purposes we set r, p to 1 and present GARCH (1,1) model, which is most common in financial time series predictions.

$$\sigma_t^2 = \alpha_0 + \delta_1 \sigma_{t-1}^2 + \alpha_1 \varepsilon_{t-1}^2 . \tag{1.20}$$

GARCH-M type model is another useful alternative, while it accounts for the possibility that returns are dependent on the volatility. In GARCH-M models, the variance of the disturbance term directly affects the mean of the dependent variable. Thus it includes volatility in the return equation:

$$r_t = \beta_0 + \beta_1 \sigma_t^2 + \varepsilon_t \,, \tag{1.21}$$

$$\sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \delta_1 \sigma_{t-1}^2 . \tag{1.22}$$

The GARCH-M model is a stochastic recursive system, given the initial conditions σ_0^2 and ε_0^2, as well as estimates. Random shock is drawn from the normal distribution, hence we can use maximum likelihood estimation. The likelihood function *L* is the joint likelihood of observing $\{y_t\}$, for $t = 1, \ldots, T$ and has following form:

$$L = \prod_{t=1}^{T} \sqrt{\frac{1}{2\pi\hat{\sigma}_t^2}} \exp\left[-\frac{\left(y_t - \hat{y}_t\right)^2}{2\hat{\sigma}_t^2} \right], \tag{1.23}$$

$$\hat{y}_t = \beta_0 + \beta_1 \hat{\sigma}_t \,, \tag{1.24}$$

$$\varepsilon_t = y_t - \hat{y}_t \,, \tag{1.25}$$

$$\hat{\sigma}_t^2 = \alpha_0 + \delta_1 \sigma_{t-1}^2 + \alpha_1 \varepsilon_{t-1}^2 . \tag{1.26}$$

The usual method of obtaining the parameter estimates $\hat{\alpha}_0, \hat{\alpha}_1, \hat{\beta}_0, \hat{\beta}_1, \hat{\delta}_1$ is maximizing he logarithm of the likelihood function wrt. parameters and restriction that variance is greater than zero and $\alpha > 0$, $\delta > 0$:

$$\underset{\left\{\hat{\alpha}_0,\hat{\alpha}_1,\hat{\beta}_0,\hat{\beta}_1,\hat{\delta}_1\right\}}{Max} \sum_{t=1}^{T} \ln\left(L_t\right) = -\frac{1}{2}\left(\ln\left(2\pi\right) + \sum_{t=1}^{T} \ln\left(\hat{\sigma}_t\right) + \sum_{t=1}^{T} \frac{\left(y_t - \hat{y}_t\right)^2}{\hat{\sigma}_t} \right), \quad (1.27)$$

where $\forall t = 1,...,T; \hat{\sigma}_t^2 > 0$.

What is nice about the GARCH approach is that it captures the source of nonlinearity. Conditional variance is nonlinear function of past values, variance is the function of past prediction errors. Thus the risk factor in the forecasting/predicting the dynamics of asset returns is captured well by the model. GARCH models are also able to capture well-observed phenomenon in stock returns time series, volatility clustering. Periods of high volatility are followed by high volatility and the same with periods of low volatility. Thus we have a specific set of parameters to be estimated with well-defined meaning, interpretation, and rationale. But the model is restrictive, because we are limited to these well-defined sets of parameters and distribution, and specific form.

Possibility for reduction of this restrictiveness is to follow Bollerslev (1986) and use his proposed Student's t-distribution which better captures to financial time-series as they are often leptokurtic[18] and fat-tailed. Bollerslev and Wooldridge (1988) also derive the quasi-maximum likelihood estimation method.

A interested reader should look for the details in the mentioned references as our main interest of this thesis is neural network models and we just outline the principles of the modern econometric tools for predicting time-series so we can compare and link it to the neural network approach in next sections. Even though it's not the main aim of this thesis, it can also serve to some extend as an overview of all main methods, linear and nonlinear regression-types and also neural network-types. By starting the thesis with this first chapter where a reader could find not only the framework for the prediction in form of EHM and Random Walk but also the preview of approaches, we can do so. After this brief introductory chapter to the problem, we will continue with the neural networks.

[18] Variable is called leptokurtic when the standardized fourth moment, *kurtosis*, is higher than 3, sometimes referred to as *excess kurtosis*. This also results in "fatter tails" of the density function.

Chapter 2

Neural Networks

Neural Networks learning methods provide a robust approach to approximating real-valued, vector-valued, and discrete-valued functions. The study of artificial neural networks (ANNs) has been inspired by the observation that biological learning systems are built of very complex webs of interconnected neurons. ANNs, are analogically built webs of interconnected set of simple units, or inputs which may be possible outputs of other units, to produce simple output, which may become input in other units, Mitchell (1997). The interested reader is recommended to use the reference for further details, as we will put the neural networks in use with financial time series, mainly stock returns. By referring to "neural networks" we will consider mainly research targeting development of systems capable to approximate complex functions efficiently and robustly in the manner of the definition (1.3).

The main motivation of neural networks usage in predicting stock returns, or other financial time-series, is the same as presented in the first chapter. As classical econometric models provide us some insights into the behavior of stock returns, we believe that neural network will do better. We believe that the learning process of neural networks will help approximate the learning process of agents or investors more efficiently resulting in finding a better understanding of stock prices. Contrary to the EMH, several researchers claim the stock market exhibit chaos[19]. Chaos is a nonlinear deterministic process which appears random, but can not be easily expressed. With the neural network's ability to

[19] Hsieh (1991), Barkoulas, travlos (1998), Peters (1994)

learn nonlinear, chaotic systems, it may be possible to outperform traditional analysis presented in previous chapters.

McNelis (2005) shows very good results on predicting artificial data and chaos process by neural networks and shows how artificial intelligence could shed more light on the time-series processes more then econometric tools presented in the first chapter. He tests predicting power of the models also on industry data, inflation, but the test on stock markets and volatility are missing. In the following chapters, we will follow his and other works with empirical research on Central European Markets while we believe that emerging markets, in particular, or markets with a great innovation and changes, represent great opportunity for the use of neural networks for the prediction task. The reasons are intuitive:

The data are often very noisy either because of thinness of the markets or information or discontinuous trading[20] gaps. Thus we have to deal with lots of asymmetries and nonlinearities which can not be assumed. The other reason is that agents in these markets are themselves in process of learning, mainly by trial and error. Often they can not assume impact of policy news or legal changes to the market simply because they did not see any real examples in their past. Thus, information set for the prediction task is very limited. As we will show, parameter estimates of neural networks are themselves a result of "learning by mistake" and the search process and can be compared to parameters used by agents to forecast and make decisions.

In this chapter we will present theoretical framework of neural networks used further in the work for empirical modelling. We begin with methodology problems, introducing the basic definitions of neural networks, feedforward and multilayered feedforward neural networks. On the basis of *universal approximation theorem*, these forms can approximate any continuous real function as Hornik, Stinchcombe, and White (1989) show. We show that neural network is not black-box instrument by describing transformation functions, neurons and defining the system mathematically. Then we follow with crucial learning algorithms discussion, as tool for optimalization in terms of error minimization. We discuss basic gradient descent search, more sophisticated conjugate gradient method, Levenberg-Marquardt method which seems to be most efficient. We close the discussion with presenting a stochastic evolutionary search and the discussion of the nonlinear estimation problem.

[20] Often there are many stocks with no or very low volume trades at these markets

Finally, we turn into the crucial data preprocessing and testing statistics for comparison of the analysis conducted in following chapters. We introduce nonlinear Principal Component Analysis as an tool for dealing with curse of dimensionality.

After the exhaustive introduction of neural network estimation procedure, we close the chapter with attending Black-box criticism discussion and try to argue in favor of neural network usage in econometric modelling.

2.1 The methodology problems

Much of the early development and work on neural network analysis has been within psychology, neuroscience related to the pattern recognition problems. Genetic algorithms used for empirical implementation of neural networks have followed similar pattern of development in applied mathematics in optimization of dynamic nonlinear and discrete systems, moving into data engineering.

Thus these systems have been developed in different surroundings that econometrical and statistical models which results in confusion in literature, mainly from the simple technical and naming conventions. A *model* is known as an *architecture*, and we *train* rather than *estimate* the network architecture. A Researcher uses *training set* and *test set* of data instead of *in-sample* and *out-of-sample* data, and the confusion should disappear whenever the reader expects *coefficients* instead of *weights*.

If we consider the application of neural networks, or Artificial Intelligence itself, the gap is almost widening. Broad literature on neural networks is simply not relevant to financial professionals or academics. Also mounting publications and empirical works on usage of neural networks in finance does not link to preceding theoretical financial literature which is probably the reason why the most of this literature is not taken seriously by the broader financial and economic academic community. As McNelis (2005) remarks: "The appeal of the neural network approach lies in the assumption of *bounded rationality*: when we forecast in financial markets, we are forecasting the forecasts of others, or approximating the expectations of others." Thus, market participants are continuously learning and adapting their beliefs from the past mistakes.

The basic is that reactions of market participants are not linear and proportionate, but asymmetric and nonlinear to changes in variables. Neural networks approximate this behavior in a very intuitive way, while our definition

from (1.3) still holds. A Very important point is approximation through the learning process. As market agents are continuously learning, the neural network is trying to capture the learning process and base on it. The difference between Neural Network models and presented econometric models is also that researchers are not making hypothesis about the coefficients to be estimated, or about functional form of the model. The coefficients, or as mentioned *weights,* are not able to be interpreted. In this manner, the methodology of prediction is different while in econometrics one is striving to obtain consistent, accurate, unbiased estimates of parameters to be interpreted.

2.2 What is a Neural Network?

Like linear or nonlinear methods, a neural network relates a set of input variables, say, $\{x_i\}, i = 1,...,k$ to a set of one or more output variables, say, $\{y_j\}, j = 1,...,k*$. Let us recall the definition of the stock returns prediction problem from chapter (1.3). It defines the prediction problem in the very similar manner. The only difference between network and other approximation methods is, that the approximating function uses one or more so called hidden layers, in which the input variables are squashed or transformed by a special function, known as logistic or logsigmoid transformation. While this approach may seem "*esoteric*" or maybe "*mystical"* on at the first glance, the reader will soon see that it may be used as a very efficient way to model nonlinear processes.

The reason we turn into neural network is straightforward. It is the goal of the prediction problem to find an approach or method that forecasts the data best, generated by unknown, nonlinear processes, with as few parameters as possible, which is as simple as achievable and as easy to estimate as it can be. Even if it seems impossible now, we may be surprised by the findings of next chapters. Moreover, it has been shown that "neural networks can approximate any function with finitely many discontinuities to arbitrary precision"[21]. This is known as *the universal approximation theorem*.

[21] Hornik, Stinchcombe, White (1989)

2.2.1 Feedforward Networks

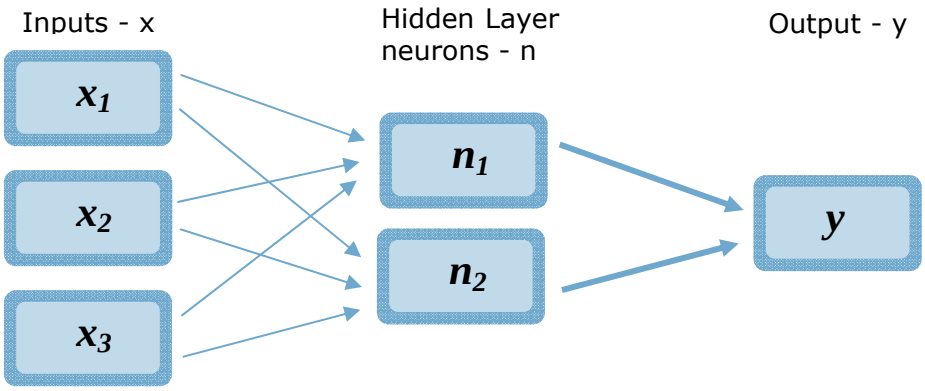


FIGURE 2.1. : Feedforward Neural network

Structure of the most basic and commonly used neural network in finance with one hidden layer[22] containing two neurons, three input variables and one output is schematically shown in FIGURE 2.1. We can see that in comparison with classical linear models, there are two more neurons which process inputs to improve the predictions. It should be mentioned here that the connection between input variables and neurons, also called *input neurons*, and connections between neurons and output, *output neurons* are called *synapses*.

The reader might note that the simple linear regression model is just a special case of the feedforward neural network, namely network with one neuron which contains a linear approximation function. The simples example of an artificial neural network is the *binary threshold model*, McCulloch and Pitts (1943), in which an output Y can either be zero or one related to I input variables. The model may be formalized as follows[23]:

$$Y = f\left(\sum_{i=1}^{I} \beta_i X_i - \mu\right), \tag{2.1}$$

$$f(u) = \begin{cases} 1 \; if \;\; u \geq 0 \\ 0 \; if \;\; u < 0. \end{cases}, \tag{2.2}$$

where $f(u)$ is the activation function, hidden layer which transforms the inputs into the neuron, and if the weighted sum of inputs is greater than μ, neuron is activated. Now, we can discuss in detail most common functional forms of the "*mystic*" neurons work

[22] Sometimes referred to as multiperceptron network

[23] We include this simple example here because it is very illustrative connection between classical regression models and neural network models and we fell that this connection is often being forgotten to explain in the neural networks financial research papers. This results in confusion and refusing of this approaches.

2.2.2 Transformation functions – *logsigmoid, tansig and Gaussian*

Maybe the most confusion about neural networks comes from the hidden layer presence and the function of neurons. They process inputs by forming linear combinations of them and then squashing these combinations using the logsigmoid function. In this part we will describe these squasher or transformation functions, but for the illustrative purposes, we start with the figure of a typical logistic function which will transform inputs, say $\{x_i\}, i = -5,...,5$ before transmitting their effects to the output.

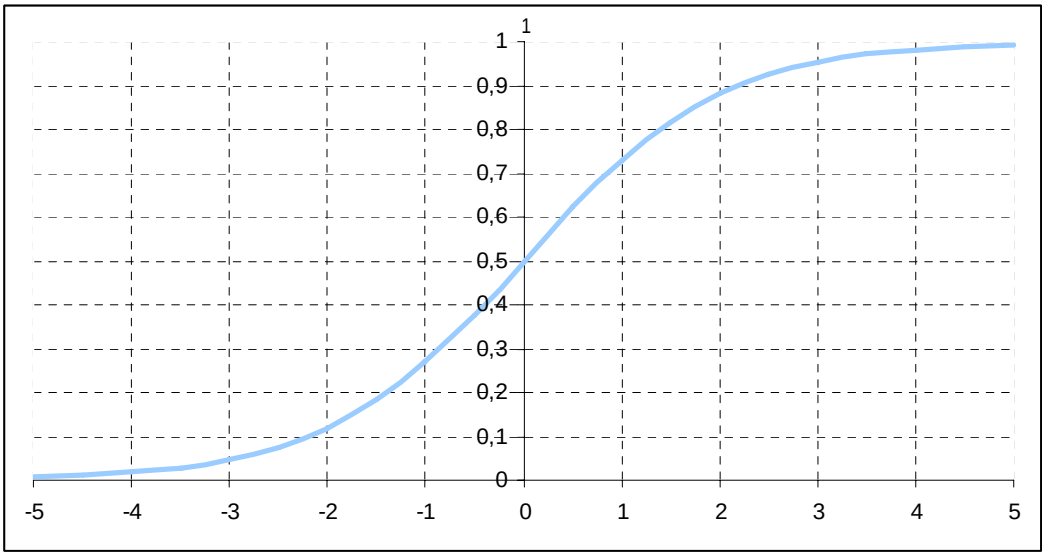


FIGURE 2.2. : Logsigmoid function

This function reflects the learning behavior of the networks, more precisely, "learning by doing". The function is increasingly steep until the inflection point from which it becomes increasingly flat and its slope moves exponentially to zero. Nonlinear sigmoid function captures learning process in the formation of expectations characterized by *bounded rationality*. Kuan, White (1994) describes it as "tendency of certain types of neurons to be quiescent of modest levels of input activity, and to become active only after the input activity passes a certain threshold, while beyond this, increases in input activity have little further effect". The *feedforward* or *multilayered perception* (MLP) network can be described by following equations:

$$n_{k,t} = \omega_{k,0} + \sum_{i=1}^{i^*} \omega_{k,i} x_{i,t} \,, \tag{2.3}$$

$$N_{k,t} = \Lambda\left(n_{k,t}\right) = \frac{1}{1+e^{-n_{k,t}}} \,, \tag{2.4}$$

$$y_t = \gamma_0 + \sum_{k=1}^{k^*} \gamma_k N_{k,t} \,, \tag{2.5}$$

where $\Lambda\left(n_{k,t}\right)$ is the logsigmoid activation function. There is i^* input variables $\{x\}$, and k^* neurons. $\omega_{k,i}$ represents coefficient vector or *input weights* vector. Variable $n_{t,k}$ is squashed by the logsigmoid function, and becomes a neuron $N_{t,k}$ at time t. Then the set of k^* neurons are combined linearly with the vector of coefficients $\{\gamma_k\}, k = 1,...,k^*$ and forms the final output which is forecast $\hat{y}_t$. This model is the workhorse of the neural networks forecasting approach as almost all researchers start with this network as the first alternative to the linear models.

An alternative to a logsigmoid activation function is *tansig* or *tanh* hyperbolic tangent function. The behavior is very similar to the logsigmoid function, but it squashes the linear combinations within the wider interval of $[-1,1]$ rather then $[0,1]$. Formalization of the network with *tansig* squasher functions is as follows:

$$n_{k,t} = \omega_{k,0} + \sum_{i=1}^{i^*} \omega_{k,i} x_{i,t} \,, \tag{2.6}$$

$$N_{k,t} = \mathrm{T}\left(n_{k,t}\right) = \frac{e^{n_{k,t}} - e^{-n_{k,t}}}{e^{n_{k,t}} + e^{-n_{k,t}}} \,, \tag{2.7}$$

$$y_t = \gamma_0 + \sum_{k=1}^{k^*} \gamma_k N_{k,t} \,, \tag{2.8}$$

where $\mathrm{T}\left(n_{k,t}\right)$ is the tansig activation function.

Another activation function is *cumulative Gaussian function*, commonly referred to as the normal function. FIGURE 2.3 plots this activation function against logsigmoid function.

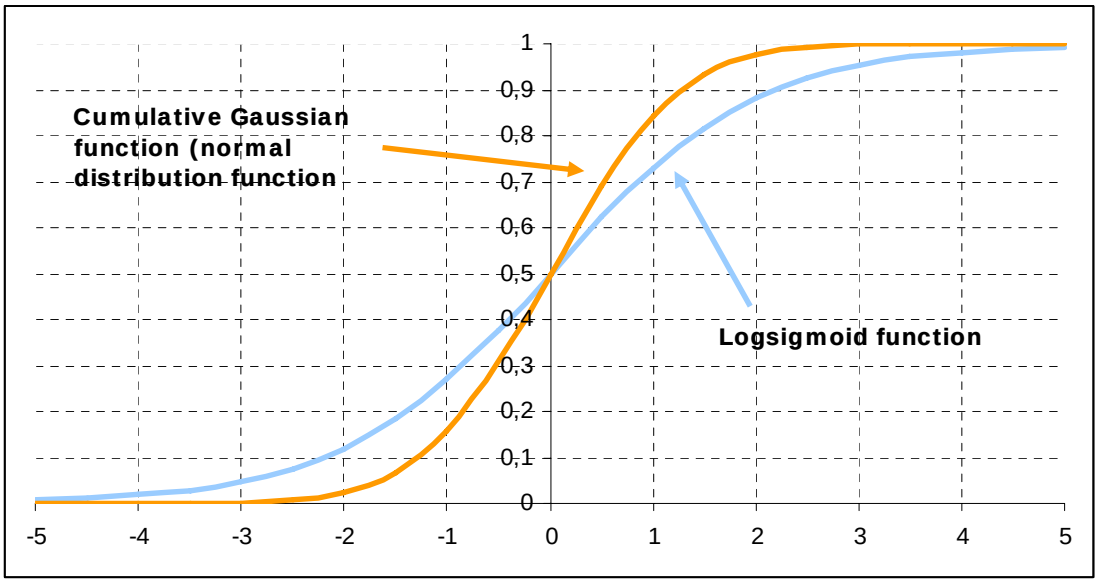


FIGURE 2.3: Gaussian function

The advantage of usage the Gaussian function is that has thinner tails, thus it does not respond to some extreme values. It can be observed from the figure, that it shows very little or no response to extreme values below -2 and above +2, while logsigmoid responds to them much more. Mathematical formalization of the neural network using Gaussian activation function can be represented by following system:

$$n_{k,t} = \omega_{k,0} + \sum_{i=1}^{i^*} \omega_{k,i} x_{i,t} \,, \tag{2.9}$$

$$N_{k,t} = \Phi\left(n_{k,t}\right) = \int_{-\infty}^{n_{k,t}} \sqrt{\frac{1}{2\pi}} e^{-\frac{1}{2}n_{k,t}^2} \,, \tag{2.10}$$

$$y_t = \gamma_0 + \sum_{k=1}^{k^*} \gamma_k N_{k,t} \,, \tag{2.11}$$

where $\Phi\left(n_{k,t}\right)$ is the standard cumulative Gaussian function.

We described basic functional forms of neural networks with most commonly used transformation functions. The reader is now probably asking the questions: "OK but, what transformation function should I use?", or "Are there any other transformation functions?". There are many other possible transformation functions in fact. The reason we describe these few is that they performed best in our tests and are also used in each of the references used in this paper.

The answer to the first question is not as simple as answer to the second one. Each transformation function transforms inputs in a different manner. Some respond to extreme values, some do not, thus they do not serve equally well in approximating the unknown function. Hence, choosing the form of squasher function is often up to the researcher and the data used. The best way is to perform tests with different transformation functions used in the neurons and use the one which performs best. This is one of the main drawbacks of neural networks, which will be discussed in further detail at the end of this chapter, while it takes time.

2.3 Multilayered Feedforward Networks

By making use of two or more hidden layers, we may be able to approximate more complex systems. FIGURE 2.4 illustrates neural network with two hidden layers, each consisting of two neurons. In the figure we also illustrate an example of time series modelling with neural network. Say we have returns $\{x_t\}$ through time t and we want to forecast them. Then we simply use inputs $\{x_{t-2}, x_{t-1}, x_t\}$ to produce output $\{x_{t+1}\}$. For generality of the illustration, we denote y as output variable.

Mathematical representation of the system with i^* input variables, k^* neurons in one hidden layer, and l^* neurons in the second hidden layer follows:

$$n_{k,t} = \omega_{k,0} + \sum_{i=1}^{i^*} \omega_{k,i} x_{i,t} \,, \tag{2.12}$$

$$N_{k,t} = \frac{1}{1+e^{-n_{k,t}}} \,, \tag{2.13}$$

$$p_{l,t} = \rho_{l,0} + \sum_{k=1}^{k^*} \rho_{l,k} N_{k,t} \,, \tag{2.14}$$

$$P_{l,t} = \frac{1}{1+e^{-p_{l,t}}} \,, \tag{2.15}$$

$$y_t = \gamma_0 + \sum_{l=1}^{l^*} \gamma_l P_{l,t} \,. \tag{2.16}$$

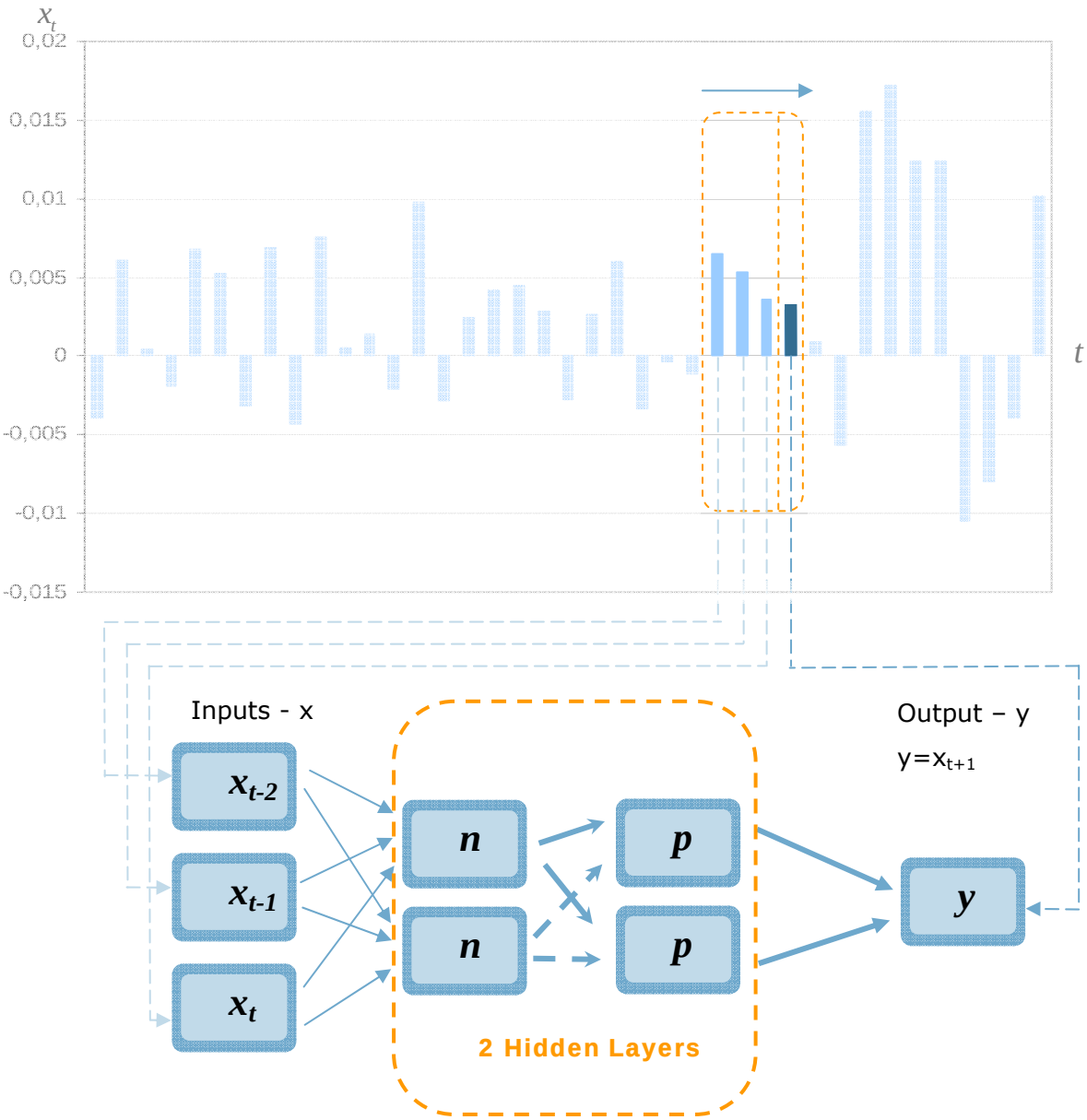


FIGURE 2.4: Feedforward network with two hidden layers

Adding a second hidden layer increases the number of parameters to be estimated and this is basically the cost of complexity which is gained by using more hidden layers. Researchers should note that with more parameters not only greater training time is a problem, there is a much greater probability that the parameter estimates will converge to a local, rather that global optimum. This problem is further discussed in chapter (2.5). As shown by Dayhoff and DeLeo (2001), simplicity of network brings better results and we will probably manage with smaller networks in our tests also:

"A general function approximation theorem has been proven for three-layer neural networks. This result shows that artificial neural networks with two layers of trainable weights are capable of approximating any nonlinear function. This is powerful computational property that is robust and has ramifications for many different applications of neural networks. Neural networks can approximate a multifactor function in such a way that creating the functional form and fitting the function are performed at the same time, unlike nonlinear regression in which a fit is forced to a pre-chosen function. This capability gives neural networks a decided advantage over traditional statistical multivariate regression techniques."

(Dayhoff and DeLeo(2001, p.1624)

2.4 Learning algorithms

In order to be able to approximate the target function – in our case stock returns, the neural network has to be able to *"learn"*. The process of learning is defined as adjustment of weights using a learning algorithm. We present common backpropagation algorithm and two more specific, conjugate gradient algorithm, and Levenberg-Marquardt algorithm. These two are presented mainly because they provided most impressive results in comparison to other common methods as the reader can see in next chapters.

The most common way to train neural network is by learning an algorithm called *"backpropagation"* or *"error-backpropagation"*. Let us assume following error function:

$$\Psi(\varpi) = \frac{1}{T}\sum_{t=1}^{t^*}(y_t - \hat{y}_t)^2 , \qquad (2.17)$$

where $\hat{y}_t$ is the estimated output variable of the network - or forecast, y_t is variable being forecasted, or input variable in time $t \in \{1,...,T\}$. Then according to our definition of prediction task in (1.3), the main goal of the learning process is to minimize $\Psi(\varpi)$ - the sum of prediction errors for all training examples. Training phase is thus unconstrained nonlinear optimization problem, where the goal is to find optimal set of weights of parameters by solving minimization problem.

$$\min\{\Psi(\omega) : \omega \in \Re^n\}, \qquad (2.18)$$

where $\Psi : \Re^n \to \Re$ is continuously differentiable.

2.4.1 Stochastic gradient descent backpropagation learning algorithm

There are several ways of achieving minimization of the $\Psi(\varpi)$, but basically the algorithm is as follows[24]:

(i) choose random initial values for the model – weights ϖ

(ii) calculate the gradient G of the error function $\Psi(\varpi)$ with respect to each weight

(iii) adjust the model weights so we move a short distance in the direction of the greatest rate of decrease of the error, i.e. in the direction of (–G)

(iv) repeat steps (ii) and (iii) until G is zero and $\Psi(\varpi)$ is minimized.

So we are searching for the gradient $G = \nabla\Psi(\varpi)$ of function Ψ which is the vector of first partial derivatives of the error function $\Psi(\varpi)$ with respect to the weight vector ϖ

$$\nabla\Psi(\varpi) = \left(\frac{\partial\Psi(\varpi)}{\partial\varpi_1}, \frac{\partial\Psi(\varpi)}{\partial\varpi_2}, ..., \frac{\partial\Psi(\varpi)}{\partial\varpi_n} \right). \qquad (2.19)$$

Further more, the gradient specifies the direction that produces the steepest increase in Ψ. The negative of this vector thus gives us the direction of steepest decrease.

FIGURE 2.5 [25] the behavior of $\Psi(\varpi)$ with respect to one weight ϖ. In order to find minimum, we always have to increase/decrease w in opposite direction to the slope, by $\Delta\omega = \eta\delta_j x_{ji}$, where $\eta \in \Re,$ but most commonly[26] $0 < \eta \leq 0.5$ is learning rate that determines size of steps for the algorithm, the rest is the partial derivative of $\Psi(\varpi)$ with respect to weights. Thus:

---

[24] Schraudolph and Cummins (2002)

[25] Please note that the figure is only schematic and in real neural network we will work with much more weights then one.

[26] Note that this is usual interval used by rule of thumb. If η is too small near zero, it may take huge time to converge to optimal weights. If η is too big it may happen that it will "jump" from positive to negative gradient and optimum will not be found at all.

$$\Delta\omega = \eta\delta_j x_{ji} = -\frac{\partial\Psi(\varpi)}{\partial\varpi_{ji}}, \qquad (2.20)$$

and finally the algorithm will find the final weights with minimum the error function by

$$\omega_{ji}^{t+1} \leftarrow \omega_{ji}^{t} + \Delta\omega\,. \qquad (2.21)$$

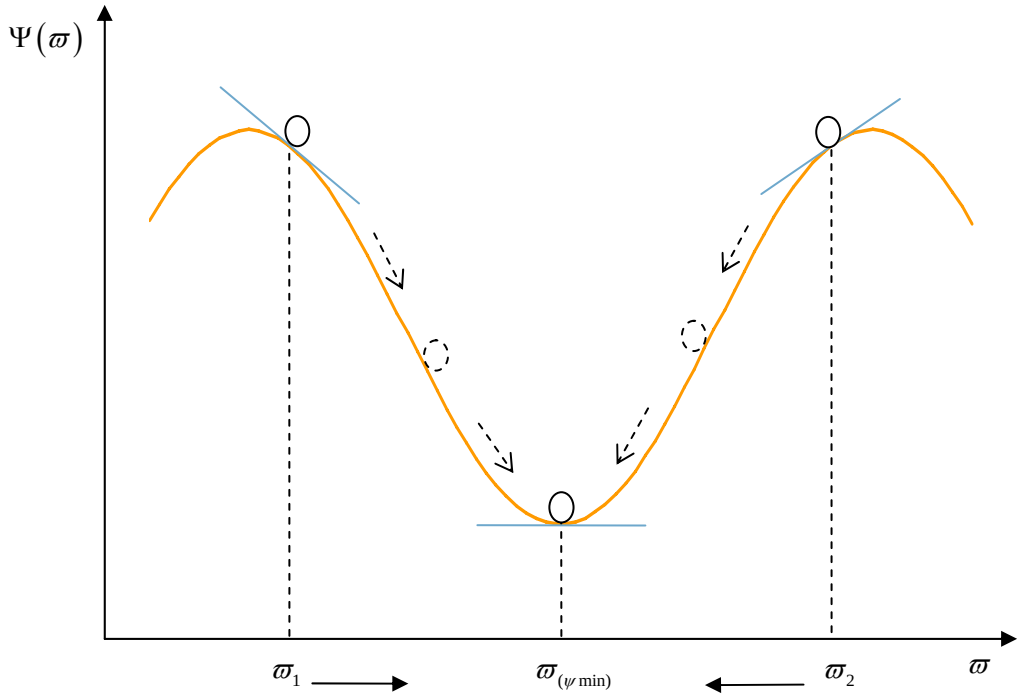


FIGURE 2.5 : Gradient descent

So if we find negative gradient in step (ii) of algorithm, we will increase w in step (iii) and vice versa. In this way we will move towards the minimum $\nabla\Psi(\varpi) = 0$ by repeating the algorithm in N steps.

Important feature of this algorithm is that is assumes a quadratic error function, hence there exist only one minimum. In practice the error function will have apart from the global minimum multiple local minima. At this point the reader probably knows what will follow – the alert that algorithm can converge to local minimum and will not find global one. Other drawbacks of this method are that there is a need to specify η and much worse, it's slow convergence.

2.4.2 Conjugate Gradient Learning Algorithm

Besides popular steepest descent algorithm, conjugate gradient algorithm is another search method that can be used to minimize the network error function $\Psi(\varpi)$ in conjugate directions. This method puts into the use orthogonal and linearly independent non-zero vectors and in some cases brings better convergence results then previous method.

Definition: *Two vectors* d_i *and* d_j *are mutually* $G-$*conjugate if* :

$$d_i^T G d_j = 0\,. \tag{2.22}$$

Then to minimize error function $\Psi(\omega_0)$ we begin with initializing the parameter vector ω of n elements at any random value $\omega_0 : \Psi(\omega_0) = c$. Then we iterate on the weights set ω until minimum of $\Psi(\omega)$ is found. Error function is represented by following second-order Taylor expansion:

$$\Psi(\omega) = c - \nabla\omega + \frac{1}{2}\omega^T G\omega\,, \tag{2.23}$$

where ∇ is gradient of the error function wrt. weights set ω and G is Hessian of the error function, an $n\times n$ symmetric and positive definite matrix. Name conjugate[27] comes from the fact that in this iteration, weights vectors are conjugates of Hessian.

Choosing $\omega_0 = (\omega_{0,1},...,\omega_{0,k})$ as set of k initial parameters, we search for direction $d_0 = -\nabla_0$. The gradient vector is defined as:

$$\nabla_0 = \begin{pmatrix} \dfrac{\Psi(\omega_{0,1}+h_1,...,\omega_{0,k}) - \Psi(\omega_{0,1},...,\omega_{0,k})}{h} \\ \dfrac{\Psi(\omega_{0,1},...,\omega_{0,i}+h_i,...,\omega_{0,k}) - \Psi(\omega_{0,1},...,\omega_{0,k})}{h_i} \\ \vdots \\ \dfrac{\Psi(\omega_{0,1},...,\omega_{0,k}++h_k) - \Psi(\omega_{0,1},...,\omega_{0,k})}{h_k} \end{pmatrix}. \tag{2.24}$$

The h_i is set as $\max(\varepsilon, \varepsilon\omega_{0,i})$ with $\varepsilon = 10^{-6}$. Hessian G_0 is matrix of second-order partial derivatives of $\Psi(\omega)$ wrt. to ω_0 and is computed similarly as Jacobian or gradient vector:

[27] Method was originally proposed by Hestens, Stiefel (1952)

$$G_0 = \begin{pmatrix} \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,1}^2} & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,1}, \partial \omega_{0,2}} & \cdots & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,1}, \partial \omega_{0,k}} \\ \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,2}, \partial \omega_{0,1}} & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,2}^2} & \cdots & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,2}, \partial \omega_{0,k}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,k}, \partial \omega_{0,1}} & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,k}, \partial \omega_{0,2}} & \cdots & \frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,k}^2} \end{pmatrix}. \qquad (2.25)$$

Off-diagonal elements of the matrix will be given by:

$$\frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,i}, \partial \omega_{0,j}} = \frac{1}{h_j h_i} \times \begin{bmatrix} \Psi(\omega_{0,1}, ..., \omega_{0,i} + h_i, \omega_{0,j} + h_j, ..., \omega_{0,k}) - \Psi(\omega_{0,1}, ..., \omega_{0,i}, .., \omega_{0,j} + h_j, ..., \omega_{0,k}) \\ -\Psi(\omega_{0,1}, ..., \omega_{0,i} + h_i, \omega_{0,j}, ..., \omega_{0,k}) - \Psi(\omega_{0,1}, ..., \omega_{0,k}) \end{bmatrix} \qquad (2.26)$$

And diagonal elements are given by:

$$\frac{\partial^2 \Psi(\omega)}{\partial \omega_{0,i}^2} = \frac{1}{h_i^2} \times \left[\Psi(\omega_{0,1}, ..., \omega_{0,i} + h_i, ..., \omega_{0,k}) - 2\Psi(\omega_{0,1}, ..., \omega_{0,k}) + \Psi(\omega_{0,1}, ..., \omega_{0,i} - h_i, ..., \omega_{0,k}) \right] \qquad (2.27)$$

We found direction d_0 thus we can follow iteration process to solve the minimization problem of $\Psi(\omega)$.

$$\omega_{k+1} = \omega_k + \alpha_k d_k , \qquad (2.28)$$

$$d_{k+1} = -\nabla_{k+1} + \beta_k d_k , \qquad (2.29)$$

α and β are momentum terms to avoid oscillations. Let $\mu_k = \frac{1}{1+\beta_k}$. Equation (2.29) can be rewritten as follows

$$d_{k+1} = \frac{1}{\mu} \left[\mu(-\nabla_{k+1}) + (1-\mu) d_k \right], \qquad (2.30)$$

which allows us to look at the search direction as a convex combination of the current steepest descent direction and the direction of last move. The search distance of each direction is varied. Value of α_k can be found by line search techniques such as Brent's Algorithm[28] so that $\Psi(w_k + \alpha_k d_k)$ is minimized given fixes ω_k and d_k.

[28] Brent (1973)

β_k is then calculated by following three formulae:

Hestens and Stiefel's formula[29] $$\beta_k = \frac{\nabla_{k+1}^T \left[\nabla_{k+1} - \nabla_k\right]}{d_k^T \left[\nabla_{k+1} - \nabla_k\right]}. \qquad (2.31)$$

Polak and Ribiére's formula[30] $$\beta_k = \frac{\nabla_{k+1}^T \left[\nabla_{k+1} - \nabla_k\right]}{\nabla_k^T \nabla_k}. \qquad (2.32)$$

Fletcher and Reeve's formula[31] $$\beta_k = \frac{\nabla_{k+1}^T \nabla_{k+1}}{\nabla_k^T \nabla_k}. \qquad (2.33)$$

Shanno's inexact line search[32] considers the conjugate method as memoryless quasi-Newton method and derives following formula for computing d_{k+1}:

$$d_{k+1} = -\nabla_{k+1} - \left[\left(1 + \frac{y_k^T y_k}{p_k^T y_k}\right)\frac{p_k^T \nabla_k}{p_k^T y_k} - \frac{y_k^T \nabla_k}{p_k^T y_k}\right] p_k^T + \frac{p_k^T \nabla_k}{p_k^T y_k} y_k, \qquad (2.34)$$

where $p_k = \alpha_k d_k$ and $y_k = \nabla_{k+1} - \nabla_k$

Conjugate gradient method finds optimal vector ω along the current gradient by doing the li-search, and converges to the solution faster than steepest gradient. Method computes gradient at the new point and projects it onto the subspace defined by the complement of the space defined by all previously chosen gradients. New direction is orthogonal to all previous search directions.

Before moving to Levenberg-Marquardt algorithm, we will sum up the conjugate gradient algorithm, by putting it into few simple steps:

(i) set k=1, initialize ω_0

(ii) compute $\nabla_0 = \nabla\Psi(\omega_0)$

(iii) set $d_0 = -\nabla_0$

(iv) compute α_k by line search where $\alpha_k = \arg\min_\alpha \left[\Psi\left(w_k + \alpha_k d_k\right)\right]$

(v) update weight vector by $\omega_{k+1} = \omega_k + \alpha_k d_k$

(vi) if network error $\Psi(\omega)$ is less than a pre-set minimum value of the maximum number of iterations has been reached, stop
else go to next step

(vii) if $k+1 > n$, then $\omega_1 = \omega_{k+1}$, k=1 and go to step (ii)

---

[29] Hestens, Stiefel (1952)

[30] Polak (1971)

[31] Dai, Yuan (1996)

[32] Shanno (1978)

else 1) set k=k+1

2) compute $\nabla_{k+1} = \nabla\Psi\left(\omega_{k+1}\right)$

3) compute $\hat{\alpha}_k$

4) compute new direction $d_{k+1} = -\nabla_{k+1} + \beta_k d_k$

5) go to step (iv)

We do not expect from conjugate gradient approach to minimize error function better, but we do expect more efficiency while it should provide faster results. Next, we introduce last, Levenberg-Marguard algorithm, an we will expect also better level of minimization from it.

2.4.3 Levenberg-Marquardt Learning Algorithm

Gradient descent works for simple models, but is too simplistic for more complex models. So we may want to use more sophisticated methods to obtain better results. The technique invented by Levenberg[33] involves blending between the introduced steepest gradient and the quadratic approximation. It uses the steepest gradient to approach minimum, and then switch to the quadratic approximation. We can formalize it as follows. Let λ be a "blending factor", constant which will determine the mix between the two methods. The update rule here is:

$$\omega_{k+1} = \omega_k - \left(H + \lambda \mathrm{I}\right)^{-1} d\ , \qquad (2.35)$$

where again ω is weight vector, H is Hessian matrix of the error function and I is identity matrix. Depending on the value of λ we can approach to following forms. With $\lambda \to 0$, we get $\omega_{k+1} = \omega_k - H^{-1}d$, which is basically quadratic approximation and with growing λ we get $\omega_{k+1} = \omega_k - \frac{1}{\lambda}d$ which the reader can compare to equation (2.21) and find that it is steepest gradient.

Algorithm adjusts value of λ according to whether $\Psi\left(\omega\right)$ is increasing or decreasing as follows:

(i) do update according to equation (2.35)

(ii) evaluate the error at the new weight vector

[33] Levenberg (1944)

(iii) if the error has *increased* as result of the step (i), retract weights to previous values and increase λ by[34] 10. Then go to (i)

Else (if the error decreased), accept the weights and decrease λ by factor 10.

If error is increasing, quadratic approximation is not working well and we are far from the minimum. Thus we need to approach simple descent by increasing λ to locate the minimum. Conversely, if we locate the minimum and the error is decreasing, approximation is working well. Hence, we expect that we are closer to minimum so we try to incline to Hessian by decreasing the λ.

Marquardt (1963) improved this method with a clever incorporation of estimated local curvature information. His insight was that when λ is high and we are doing essentially gradient descent, we can still benefit from Hessian matrix that we estimated. He suggested that we should move further in the directions in which the gradient is *smaller* in order to get around the error valley problem. Marquardt replaced identity matrix from equation (2.35) with diagonal of Hessian:

$$\omega_{k+1} = \omega_k - \left(H + \lambda diag\left[H\right]\right)^{-1} d\,. \qquad (2.36)$$

We can see that this method does not require other computations then previous methods. All we need is $\Psi(\omega)$ as error function of estimated output and desired output, and it's gradient $\nabla\Psi(\omega)$.

It is important to notice that it is nothing more than a heuristic method. It is not optimal for any defined criterion of speed or final error. What is so appealing is that it works extremely well in practice. Its only drawback is that it requires matrix inversion step, thus becomes much slower than *backpropagation* or *conjugate gradient* in more complex models. On the other hand, it has a much better results as the reader will see in further chapters.

2.5 The Nonlinear Estimation Problem

As we saw in previous subchapters, finding the coefficient values of nonlinear models is not that easy job as neural network is highly complex nonlinear system. We can hit several locally optimal solutions, but none of these

[34] Or other significant factor. 10 was originally proposed by Levenberg.

can be the best solution in terms of minimizing error between our model prediction $\hat{y}$ and actual value y.

In any nonlinear system, we start the estimation with initial conditions as we saw in previous chapter. These are meant to be a guess or random variable, and we get to the problem of some parameters being guessed better than others. This may end in converging to local rather than global optimum and of course to best forecast in local neighborhood of initial guess, but not best forecast ahead of the "*initial area*". This can be very intuitively illustrated in following FIGURE 2.6 :

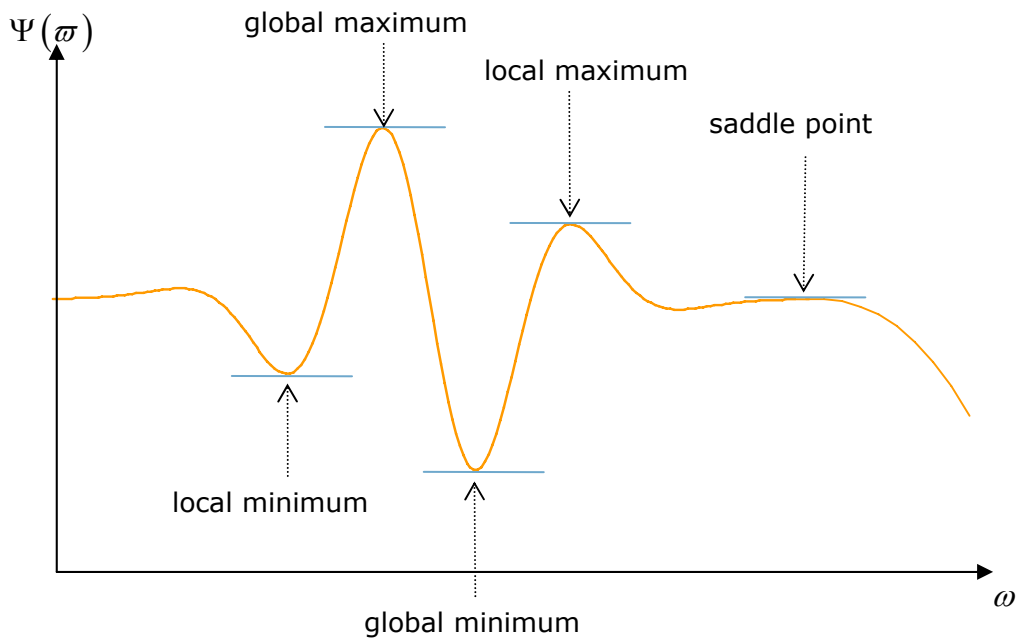


FIGURE 2.6: Problem of search for local optima

As we can see, initial set weights may rather lie near to a local maximum than a minimum, or near a saddle point while our search of minimum of error function is using derivatives of error function. Thus we have to recognize also curvature around our point by second-derivatives which will provide us better insight. If the change of gradient or second-derivative is positive, we know that we are near minimum and vice versa for maximum.

So as we adjust weights by presented algorithms, we can easily get stuck at any of the positions from FIGURE 2.6 where derivative is zero or function has a flat slope (blue lines on the figure). If we are adjusting weights by too large steps, algorithm can easily converge from near-global minimum to maximum or other point. If we adjust by too small steps in contrary, the algorithm may get stuck in a saddle point for a long time during the training period and may not converge to a minimum at all.

Maybe the reader is asking the question: "but what can we do to avoid this problem?" There are several techniques of minimizing the chance of converting to "the wrong" optimum. A very intuitive way is re-estimation of whole model, another way is stochastic evolutionary search presenting in following subchapter.

2.5.1 Stochastic evolutionary search

Genetic algorithm reduces the likelihood of landing in a local minimum. We do not need to approximate Hessian, we start with "population" of p initial guesses, $\{\omega_{0,1}, \omega_{0,2}, ..., \omega_{0,p}\}$ and update them by genetic selection, breeding, and mutation, for many generations, until the best coefficient vector is found. Let us have a closer look at this process.

(i) **Population creation**

We start with a population $N*$ of random vectors ω. Let p be the size of each vector representing the total number of parameters to be estimated. Then we create following population:

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \\ \omega_p \end{pmatrix}_1 \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \\ \omega_p \end{pmatrix}_2 \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \\ \omega_p \end{pmatrix}_i \dots \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \\ \omega_p \end{pmatrix}_{N*} . \qquad (2.37)$$

(ii) **Selection**

The next step is the selection of two pairs from population at random, with replacement, and evaluation the fitness of them according to sum of squared errors. Weights with lower error receive better fitness values. Two winning vectors (i,j) with best fitness are then chosen for "breeding"

(iii) **Crossover**

now, these two vectors (i,j) will "breed children" meaning they will be associated with another pair of vectors C1(i) and C2(j) by one of three methods to be chosen randomly with same probability equal to 1/3. *Shuffle crossover* for which random draws from a binomial distribution are made and new vectors are swapped or no change is made , *Arithmetic crossover* for which the random value of $c \in (0,1)$ is chosen and then new vectors are linear combination of old ones:

$c\omega_{i,p}+(1-c)\omega_{j,p}, (1-c\omega_{i,p}+c)\omega_{j,p}$ or last one, *Single-point crossover*, where integer I is randomly chosen from set $[1, k-1]$. The vectors are then cut at this integer and parameters are swapped.

(iv) **Mutation**

now "children" C1(i) and C2(j) have to mutate in generations $G = 1, 2, ..., G^*$ with probability[35], say, $\tilde{p} = 0.15 + 0.33/G$ assigned to them. Randomly drawing real numbers $r_1, r_2 \in (0,1)$ and random number s from a standard normal distribution, mutated weight $\tilde{\omega}_{i,p}$ is given by

$$\tilde{\omega}_{i,p} = \begin{cases} \omega_{i,p} + s\left(1 - r_2^{\left(1-\frac{G}{G^*}\right)^b}\right) & if \;\; r_1 > 0.5 \\ \omega_{i,p} - s\left(1 - r_2^{\left(1-\frac{G}{G^*}\right)^b}\right) & if \;\; r_1 \le 0.5 \end{cases}, \qquad (2.38)$$

where G is the number of generations, G^* is the maximum number of generations and b is the degree to which the mutation is nonuniform. Usually b=2. Probability of creating new coefficient which is far from the current coefficient diminishes as G approaches G^*. This allows more precise search of weights approaching to a global optimum.

(v) **Election tournament**

The last step is "tournament" in which all chosen weights are competing for the best fitness criterion. Again, two vectors with the best fitness "survive" and pass to next generation. Even if the older pair has better fitness, it wins the tournament and the younger one is eliminated.

The process is repeated from (i) through (v) for G^* generations. Convergence is obtained if we do not see improvement in fitness of the last – optimal weights. Unfortunately, literature does not provide us with the optimal value of G^* as for each problem it will be different. What we can do is to add simple if-then rule of no improvement in sum of squared errors, or fitness. If there is no improvement seen, the algorithm will stop.

[35] Probability here is just an example

2.5.2 Hybrid learning as a solution?

One of the main drawbacks of genetic algorithms is its extreme slowness. Even for reasonable dimension of weights vector ω, the various combinations and permutations of elements that the genetic algorithm might find optimal may become very large. In the next sub-chapter we will discuss the *course of dimensionality* problem, but even if we manage to reduce the dimension significantly the time taken to converge to a global optimum may be extremely long. On the other hand, it has been mathematically proved[36], that convergence occurs.

The *hybrid approach* solves partially the problem of slowness of the genetic algorithm. We may run genetic algorithm for a reasonable number of generations, say 50 or 100 which will take little time and then use obtained vector of weights as initial weights in gradient searching algorithms.

Problems arise even with usage of the hybrid approach because of the nature of neural networks. The Neural network structure can give different results with some kind of data, as initial guess may fall in the local optimum trap as we saw in previous chapters. We can use repeated estimations for the robustness of results. Granger and Jeon (2002) have suggested a simple idea of thick modelling. The framework of this idea is to repeatedly estimate a given data set with different specifications and then use the mean of the obtained information. They mainly use this method for forecasting, thus they find a mean of repeated forecasts to be an optimal one. They find this method outperformed simple linear models, while it also outperformed individual network results on macroeconomic data modelling.

2.6 Preprocessing the data

One of the first steps of research when modelling time series is adjusting, scaling the data and removing nonstationarity. These procedures are known as data preprocessing and are often crucial for the results. In this subchapter we will discuss the problems of preprocessing the data including curse of dimensionality.

[36] See Hartl (1990), or Mitchel(1997)

2.6.1 Curse of dimensionality

One of the most important steps in designing a neural network is the choice of appropriate data pre- and post-processing. The first problem arrives with choosing the variables that may explain our observations best. In forecasting stock market prices, there may be many variables that may have influence on the price. If we use all possible candidates as a regressors in the model, we will face the *curse of dimensionality*, first mentioned by Bellman (1961). It simply means that the number of sample sizes needed to estimate a model with a given degree of accuracy grows exponentially with the number of variables in the model.

Thus, intuitive assumption – "more data will provide greater insight into the process" does not necessarily hold and reduction of dimensionality is often necessary for good, simple predictive model, as it is crucial for the model to choose variables that influence the observations most. In other words, to reduce the number of regressors to a manageable subset if we want to have sufficient degree of freedom for any meaningful conclusions.

2.6.2 Principal Component Analysis

Principal component analysis (PCA) is basically an approach to reducing a large set of variables into a smaller subset – reduction of dimensionality while preserving as much information contained in the data as possible. PCA identifies linear combinations of data that explain most of the variation of the original data. For N vectors, N linearly independent combinations will explain total variation of the data. However, what if only two or three linear combinations, or *principal components* explains most of the variation of the total data set? We can then significantly reduce the dimension of the model. This should be done with caution because it can happen that we reduce important information away.

2.6.2.1 Karhunen-Loeve Transformation

The goal of principal component analysis is to map d-dimensional vectors x_i to m-dimensional vectors z_i with $m < d$. We can express vector x as linear combination of a set of d orthonormal vectors u_i

$$x = \sum_{i=1}^{d} z_i u_i \,, \tag{2.39}$$

where the vectors u_i satisfy the orthonormality relationship

$$u_i^T u_j = \delta_{ij}\,, \tag{2.40}$$

where δ_{ij} is the Kronecker delta[37]. Explicitly, coefficients z_i can be found as

$$z_i = u_i^T x\,. \tag{2.41}$$

So the dimensionality reduction works as follows: $m : m > d$ coefficients z_i are replaced by constant, say b_i so vector x can be best approximated as follows:

$$\tilde{x} = \sum_{i=1}^{m} z_i u_i + \sum_{i=m+1}^{d} b_i u_i\,. \tag{2.42}$$

So again we are solving problem of minimization of sum of squares errors of data set of N samples, which is defined as:

$$\Psi(u) = \frac{1}{2}\sum_{n=1}^{N}\left\|x_n - \tilde{x}_n\right\|^2 = \frac{1}{2}\sum_{n=1}^{N}\sum_{i=m+1}^{d}\left(z_{n,j} - b_i\right)^2\,. \tag{2.43}$$

If we set $\partial\Psi \big/ \partial b_i = 0$, then

$$b_i = \frac{1}{N}\sum_{n=1}^{N} z_i^n = u_i^T \overline{x}\,, \tag{2.44}$$

with $\overline{x}$ being arithmetic mean and using (2.41) we can rewrite Ψ as:

$$\Psi(u) = \frac{1}{2}\sum_{i=m+1}^{d}\sum_{n=1}^{N}\left(u_i^T\left(x_n - \overline{x}\right)\right)^2 = \frac{1}{2}\sum_{i=m+1}^{d} u_i^T \sum u_i\,. \tag{2.45}$$

Where $\sum = \sum_{i=1}^{n}\left(x_n - \overline{x}\right)\left(x_n - \overline{x}\right)^T$ is covariance matrix of x_i. As shown in Bishop (1996), minimum can be found when the basis vector satisfies condition $\sum u_i = \lambda_i u_i$ so they are eigenvectors of the covariance matrix. Note that since covariance matrix is real and symmetric, its eigenvectors can be orthonormal as assumed. Thus value of error in minimum is equal to:

$$\Psi(u_{\min}) = \frac{1}{N}\sum_{i=m+1}^{d}\lambda_i\,, \tag{2.46}$$

and minimum can be found by choosing $d - m$ smallest eigenvalues and their corresponding eigenvectors u_i - or *principal components* - to discard.

---

[37] Kronecker delta is a function of two variables, usually integers, which is 1 if they are equal, and 0 otherwise. $\delta_{1,2} = 0$, but $\delta_{3,3} = 1$. It can be formalized as follows: $\delta_{i,j} = \begin{cases} 1 & if \quad i = j \\ 0 & if \quad i \neq j \end{cases}$.

2.6.3 Nonlinear Principal Components using neural networks

Neural networks can also be used for reduction of dimensionality problem. Network is trained to map the d-dimensional input space onto itself over a m-dimensional $(m < d)$ hidden layers. Let us consider four input variable network encoded by two logsigmoid functions under neurons n in a dimensionality reduction mapping as shown in FIGURE 2.7.

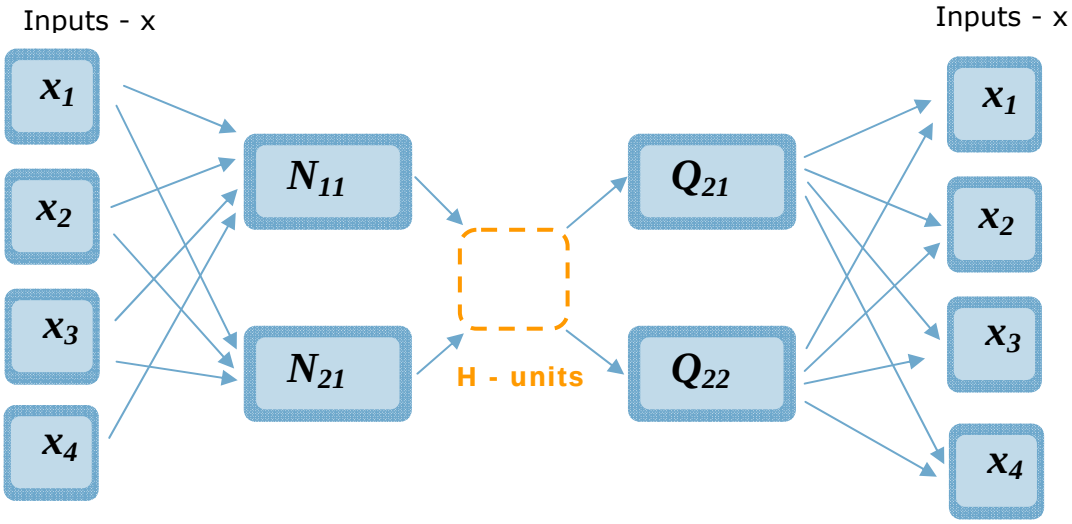


FIGURE 2.7: Neural Principal components

First two N-neurons for dimensionality reduction mapping are linearly combined to form H neural principal components. Then these are decoded by another logsigmoid Q-neurons for reconstruction mapping which are linearly combined to generate inputs as the output layer. Thus inputs $x_1,...,x_n$ are mapped into themselves. Letting X be a matrix with k columns, there is j neurons and p, model can be formalizes by following system of equations:

$$n_j = \sum_{k=1}^{K} \alpha_{j,k} X_k ,$$

$$N_j = \frac{1}{1+\exp(-n_j)},$$

$$H_p = \sum_{j=1}^{J} \beta_{p,j} N_j ,$$

$$q_j = \sum_{p=1}^{P} \gamma_{j,p} H_p ,$$

$$Q_j = \frac{1}{1+\exp\left(-q_j\right)},$$

$$\hat{X}_k = \sum_{j=1}^{J} \delta_{k,j} Q_j \,. \qquad (2.47)$$

And naturally, this system of equation can be optimized by solving minimization of sum of squared errors problem $\min\left\{\Psi\left(x\right): x \in \Re^n\right\}$ where $\Psi\left(x\right)$ is a loss function.

McNellis(2005) shows that nonlinear principal component analysis outperforms linear one in much better accuracy. The main drawback is again the time needed to find the optimum.

2.6.4 Stationarity: Dickey—Fuller Test

Most of the time series considered in this thesis are time dependent and before starting to work with them, we need to difference the data to gain *covariance stationarity* time series. Series is said to be (weakly or covariance) *stationary* if the first and second moments[38] are constant through time.

The most commonly used test for *stationarity* is one proposed by Dickey and Fuller (1979). For a given series $\left\{y_t\right\}$:

$$\Delta y_t = \rho y_{t-1} + \sum_{i=1}^{k} \alpha_i \Delta y_{t-i} + \varepsilon_t \,, \qquad (2.48)$$

where $\Delta y_t = y_t - y_{t-1}$, ρ, α_i are coefficients to be estimated, and ε_t is a random disturbance term with $E\left(\varepsilon_t\right) = 0$ and $E\left(\varepsilon_t^2\right) = \sigma^2$. Under the null hypothesis, $\rho = 0$. From equation (2.48) we can see that if this holds, y_t at any time will be equal to y_{t-1} plus/minus effects of the remaining terms. Thus long-run expected value of the series is uncertain if $y_t = y_{t-1}$ and $E\left(y_t\right) = E\left(\varepsilon_t\right) = 0$. Series with $\rho = 0$ are called *nonstationary*, or a *unit root process*.

If there is some persistence in the model, with ρ falling in the interval $\left(-1,0\right)$, the relevant regression changes to:

$$y_t = \left(1+\rho\right) y_{t-1} + \sum_{i=1}^{k} \alpha_i \Delta y_{t-i} + \varepsilon_t \,. \qquad (2.49)$$

[38] First moment - mean, second moment – variances and covariances

In the long run it is still valid that $\Delta y_{t-i} = 0$ for $i = i,...,k$. But long-run mean reduces to the following, with $\rho^* = (1+\rho)$:

$$y_t(1-\rho^*) = \varepsilon_t. \tag{2.50}$$

Then, expected value of y_t is $E(y_t) = E(\varepsilon_t) \big/ (1-\rho^*)$.

For stationarity, it is necessary that coefficient ρ is significantly less than zero. Dickey and Fuller tests are modified, one-sided t-tests of hypothesis $\rho < 0$ in a linear regression and allow presence of constant and trend terms in the regressions.

Most of the time financial series are nonstationary themselves and needs to be first-differenced to achieve stationarity. Logarithmic first differencing usually helps and it is nothing else then transforming the financial series into the returns:

$$\Delta r_t = \ln(P_t) - \ln(P_{t-1}), \tag{2.51}$$

where r_t is return of the series, P_t is series itself. After transforming the series we should use proposed Dickey-Fuller test[39] to assure that our testing series are stationary.

2.6.5 Data scaling

Sometimes we use data with very high or low numbers, or outliers which may cause a computer to assign zero to values being minimizes. Sometimes we want to test differently scaled data, i.e. if we want to test effects of interest rates changes on the market, or sometimes our data simply contains too many zero values which cause errors in the estimation process. For all of these cases, it might be crucial to scale the data right after we gained its stationarity.

The reader should also note that using i.e. logsigmoid functions for estimation might cause that large value will be simply assigned[40] 1 and low values 0. Then it is very likely that we might loose information. Thus there may be a need in transforming the data. The most simple is *linear* scaling function to range (0,1) or(-1,1). It uses maximum and minimum values of the series x. Following equations represent scaling to intervals (0,1) and (-1,1) respectively:

[39] Or an alternative to it

[40] For illustration see FIGURE 2.2. : Logsigmoid function, p. 19

$$x_{i,t}^{*} = \frac{x_{i,t}^{*} - \min\left(x_i\right)}{\max\left(x_i\right) - \min\left(x_i\right)}, \tag{2.52}$$

$$x_{i,t}^{*} = 2.\frac{x_{i,t}^{*} - \min\left(x_i\right)}{\max\left(x_i\right) - \min\left(x_i\right)} - 1. \tag{2.53}$$

There are also nonlinear methods of scaling data which transforms series x_i say to z_i i.e. in following way. Firstly we standardize a series, and then use nonlinear transformation:

$$x^{*} = \frac{1}{1 + \exp\left(-z\right)}, \tag{2.54}$$

$$z = \frac{x - \overline{x}}{\sigma_x}. \tag{2.55}$$

Of course it is often very hard to say which of the transformation should be used. It depends only on the results obtained, so researcher is left with trial – error method. Luckily for us, most of the financial series does not need scaling while first differencing most of the times help us to "keep" data in the narrow ranges. In other words, how many times we see 1 representing 100% return in two consequent time periods x_t and x_{t+1}? On the other hand, we should always keep in mind this possibility of data pre-processing.

2.7 Evaluation of estimated models

Until now we presented complex procedure of estimation with neural networks. In this section we will briefly present a few criteria which will help us with interpreting the results. We will work with in-sample criteria, or training period results interpretation which is in fact evaluation for information on how well the estimated data fits our modeled data. We will see that model which explains most of the variation of the training data may turn to be inapplicable for forecasting purposes, or better said out-of-sample data which model "did not see before". They are also called testing data or out-of-sample criteria which will be most important for us in the testing part.

So the framework of empirical testing is the following: After preprocessing the data we divide it into 2 or 3 samples – *training*, *cross-validation* and *testing sets*. The Neural network will be estimated using the training data and optimal

weights will be found at this stage. Then the weights are put to cross-validation data and might be slightly adapted to changes if we find that in-sample criteria deteriorated. Just after that, the last set of data is put to test. Coefficients obtained from training will be used to perform with new data which had no impact on calculating the coefficients, which is the most important part. The reader should also be aware of the proportion of training to testing data set. In most of the studies they cut 20-25% for testing purposes, but it can be crucial for our results to do this with patience. Imagine we want to model AAA stock returns and 15.01.2002 there had been huge reforms at the company leading to consistently higher than expected profits. This would also have impact on returns of our AAA Company and if we train network on the data until 15.01.2002 and try to test them further on, we may be extremely disappointed. Our model will know just pattern from the pre-reform period. Hence according to changes, also pattern of returns changed after the date and our model will not be capable to deal with it.

2.7.1 Normality

It is a common practice that residuals are assumed to come from a Gaussian or normal distribution in econometric modelling. Assumption may be needed for efficiency, and we often do not release it also in neural network modelling. Well-known test, Jarque-Bera (1980) statistics, starts from the assumption that a normal distribution has zero third moment - skewness[41] S and fourths moment - kurtosis[42] K of 3 and measures the difference from the normal distribution.. Given the residual vector $\hat{\varepsilon}$, the Jarque-Bera statistics is formalized as follows:

$$JB(\hat{\varepsilon}) = \frac{N-k}{6}\left(S^2 + \frac{(K-3)^2}{4} \right). \qquad (2.56)$$

---

[41] *Skewness* is a measure of asymmetry of the distribution of the series around the mean. We compute it as $S = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \overline{y}}{\hat{\sigma}}\right)^3$ where $\hat{\sigma}$ is estimator of the standard deviation. Positive skewness means long right tail, negative skewness implies long left tail.

[42] *Kurtosis* measures the peakedness or flatness of the distribution of the series. We compute is as follows: $K = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \overline{y}}{\hat{\sigma}}\right)^4$. If kurtosis exceeds 3, the distribution is peaked, said to be *leptokurtic* and if it's less than 3 it is flat – *platykurtic* relative to normal distribution.

Under the null hypothesis of a normal distribution, the Jarque-Bera statistics is distributed as $\chi^2(2)$. Reported probability is probability, that Jarque-Bera statistic exceeds in absolute value the observed value under the null hypothesis. Thus i.e. JB=4.32 (p<0.039523) tells us that we reject the null hypothesis of normal distribution at the 5% significance level but not at the 1% significance level.

2.7.2 Goodness of fit

R-squared coefficient (multiple correlation coefficient) is probably the most commonly used measure of overall goodness of fit of a model. It can be simply interpreted as the fraction of variance of the dependent variable explained by the independent variables. Value of statistics fall into the $(0,1)$ interval[43] while if it's 0,we can assume that model fits the data no better that simple mean of dependent variable if it's 1, model explains the variance perfectly. Statistics is represented by:

$$R^2 = \frac{\sum_{t=1}^{T}\left(\hat{y}_t - \bar{y}_t\right)^2}{\sum_{t=1}^{T}\left(y_t - \bar{y}_t\right)^2}. \qquad (2.57)$$

One problem with using R^2 as a measure of goodness of fit is that it will never decrease as we add regressors. As an extreme case, we can obtain $R^2 = 1$ if we include as many independent regressors as there are sample observations. Thus we adjusted R^2 measure is used, $\bar{R}^2$ or $adjR^2$ which penalizes R^2 for addition of regressors which do not contribute to the explanatory power of the model. It can be computed as follows:

$$\bar{R}^2 = 1 - \left(1 - R^2\right)\frac{T-1}{T-k}. \qquad (2.58)$$

and is naturally never larger then R^2 In all our tests, if we refer to R^2, we refer to adjusted statistics.

[43] Please note that for number of reasons this coefficient can be also negative in standard econometric modelling. For example if regression does not have an intercept or constant, if it contains restrictions, or if the model is two-stage least squares or ARCH.

2.7.3 Schwarz Information Criterion

One way to modify the R^2 statistic is to make use of Schwarz (1978) information criterion which corrects the performance of a model for the number of parameters, k, it uses. The statistic is used simply to prefer model with lowest value.

$$SC = \ln\left(\sum_{t=1}^{T}\frac{\left(y_t-\hat{y}_t\right)^2}{T}\right)+\frac{k\ln\left(T\right)}{T}. \qquad (2.59)$$

Alternatively, Akaike or Hannah-Quinn statistics may be used which punishes a given model by factor of $2k/T$ or $k\left[\ln\left(\ln\left(T\right)\right)\right]/T$ respectively. Schwarz criterion punishes model more than others by factor of $k\left(\ln\left(T\right)\right)/T$.

2.7.4 Q-Statistics

Besides having properties of constancy of variances, assumption of normality of residuals, serial independence is next step of evaluating whether there is some information content in residuals. If model is well-specified, residuals should not contain any pattern in their first and second moments. Thus we need to test for serial independence and homoskedasticity, or constancy of variance.

If the autocorrelation is absent, residuals are unpredictable from past data. The autocorrelation function is defined by the following equation, for different lag lengths m:

$$\rho_m\left(\hat{\varepsilon}\right)=\frac{\sum_{t=m+1}^{T}\hat{\varepsilon}_t\hat{\varepsilon}_{t-m}}{\sum_{t=1}^{T}\hat{\varepsilon}_t^2}. \qquad (2.60)$$

Following statistics proposed by Ljung and Box (1978) is than used for examining the joint significance of the first M residual autocorrelations, with asymptotic Chi-squared distribution with M degrees of freedom $\chi^2\left(M\right)$:

$$Q\left(M\right)=T\left(T+2\right)\sum_{m=1}^{M}\frac{\rho_m^2\left(\hat{\varepsilon}\right)}{\left(T-m\right)}. \qquad (2.61)$$

2.7.5 Root Mean Squared Error Statistic

The most common statistic for evaluating out-of-sample fitness under quadratic loss function is the root mean squared statistic derived from Mean squared error:

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(y_t - \hat{y}_t\right)^2}\,, \tag{2.62}$$

where T is number of observations. Normalized Mean Squared Error is also used and is given by:

$$NMSE = \frac{\sum_{t=1}^{T}\left(y_t - \hat{y}_t\right)^2}{\sum_{t=1}^{T}\left(y_t - \overline{y}_t\right)^2}. \tag{2.63}$$

Please note that NMSE can also be expressed by $1-R^2$. See equation (2.57).

2.8 Statistical Comparison of Predictive Accuracy

The key question in forecasting is measurement of accuracy of different forecasts, as we are interested in the model producing most accurate forecasts. As we will compare performance of various econometric models and neural network models, we have to consider statistical methods for comparing the results so we are able to identify if neural network models help us in producing more accurate results or not. This needs to be done on out-of-sample model valuation.

Let us consider two *h-step* forecasts, $\left\{\hat{p}^i_{t+h|t}\right\}_{t=1}^{T}$ and $\left\{\hat{p}^j_{t+h|t}\right\}_{t=1}^{T}$, of the time series $\left\{p_{t+h}\right\}_{t=1}^{T}$ with forecast errors of $\left\{\varepsilon^i_{t+h|t}\right\}_{t=1}^{T}$ and $\left\{\varepsilon^j_{t+h|t}\right\}_{t=1}^{T}$. To choose model with significantly lower prediction error, thus better accuracy, we wish to compare the expected loss associated with both forecasts. Of course this will depend on the chosen loss function as defined in (1.3). We will restrict on the loss function dependent on the forecast error here, $L\left(\varepsilon_{t+h|h}\right)$, and we will try to find optimal *h-step* prediction:

$$\hat{P}^*_{t+h|t} \equiv \arg\min E\left[L\left(\varepsilon_{t+h|t}\right)\middle|\mathcal{F}_t\right] \tag{2.64}$$

Thus we will test the null hypothesis of equal forecast accuracy for two forecasts against the alternative hypothesis of unequal forecast accuracy:

$$H_0 : E\left[L\left(\varepsilon^i_{t+h|t}\right) - L\left(\varepsilon^j_{t+h|t}\right)\right] = 0, \tag{2.65}$$

$$H_1 : E\left[L\left(\varepsilon^i_{t+h|t}\right) - L\left(\varepsilon^j_{t+h|t}\right)\right] \neq 0, \tag{2.66}$$

where $L\left(\varepsilon_{t+h|t}\right)$ is positive loss function and $L\left(\varepsilon^i_{t+h|t}\right) - L\left(\varepsilon^j_{t+h|t}\right)$ is the loss differential.

In the testing of the null hypothesis, the choice of the loss function is needed. In the next subchapter, we will present quadratic loss function, when we basically chose model j, if $\left(\varepsilon^j_{t+h|t}\right)^2 < \left(\varepsilon^i_{t+h|t}\right)^2$, mean absolute loss function, when we choose model j if $\left|\varepsilon^j_{t+h|t}\right| < \left|\varepsilon^i_{t+h|t}\right|$, and also asymmetric loss functions, when we are more concerned about positive errors than negative, or vice-versa.

2.8.1 Optimal forecast under different loss functions

Quadratic loss function – under the quadratic loss function, we can define optimal *h-step* forecast as follows:

$$\hat{P}^*_{t+h|t} \equiv \arg\min E\left[\left(p_{t+h} - \hat{p}_{t+h|t}\right)^2 \left|\mathcal{F}_t\right.\right], \tag{2.67}$$

where the prediction is conditional expectation $E\left[P_{t+h}\left|\mathcal{F}_t\right.\right] = \hat{P}_{t+h|t}$ on information set $\mathcal{F}$. Thus considering two forecasts $\left\{\hat{p}^i_{t+h|t}\right\}_{t=1}^T$ and $\left\{\hat{p}^j_{t+h|t}\right\}_{t=1}^T$, we will choose forecast $\left\{\hat{p}^j_{t+h|t}\right\}_{t=1}^T$ if it satisfies $E\left[\left(p_{t+h} - \hat{p}^j_{t+h|t}\right)^2\right] < E\left[\left(p_{t+h} - \hat{p}^i_{t+h|t}\right)^2\right]$

Quadratic loss function is the most popular in the literature, it is monotonically increasing, symmetric, homogenous of degree 2 and differentiable everywhere.

Mean absolute loss function – under the mean absolute loss function, the optimal *h-step* forecast will be:

$$\hat{P}^*_{t+h|t} \equiv \arg\min E\left[\left(\left|p_{t+h} - \hat{p}_{t+h|t}\right|\right)\left|\mathcal{F}_t\right.\right], \tag{2.68}$$

where loss function $L\left(\varepsilon_{t+h|t}\right)=\left|p_{t+h}-\hat{p}_{t+h|t}\right|$ is monotonically increasing, symmetric, homogenous and differentiable everywhere except $\varepsilon_{t+h|t}=0$.

Asymmetric loss functions - sometimes the researcher is more concerned about positive errors $\left(p_{t+h}-\hat{p}_{t+h|t}>0\right)$, than about negative errors $\left(p_{t+h}-\hat{p}_{t+h|t}<0\right)$ as they may be more costly. Two well known asymmetric loss functions are linear exponential loss function – *Linex*, and linear-linear loss function – *Lin-Lin*:

<u>*Linex loss function*</u> – under the linear exponential loss function, the optimal *h-step* forecast will be:

$$\hat{P}^{*}_{t+h|t}\equiv\arg\min E\left[\left(\exp\left(a\left(p_{t+h}-\hat{p}_{t+h|t}\right)\right)+a\left(p_{t+h}-\hat{p}_{t+h|t}\right)-1\right)|\mathcal{F}_{t}\right], \qquad (2.69)$$

for $a\neq 0$. Function is asymmetric as for $a>0$ it is almost linear to the left of the *y*-axis, and almost exponential to the right, and vice versa for $a<0$.For this loss function, we will try to find the $\left\{\hat{p}^{j}_{t+h|t}\right\}_{t=1}^{T}$ which will satisfy following condition:

$$\begin{aligned}&E\left[\left(\exp\left(a\left(p_{t+h}-\hat{p}^{j}_{t+h|t}\right)\right)+a\left(p_{t+h}-\hat{p}^{j}_{t+h|t}\right)-1\right)\right]\\&<E\left[\left(\exp\left(a\left(p_{t+h}-\hat{p}^{i}_{t+h|t}\right)\right)+a\left(p_{t+h}-\hat{p}^{i}_{t+h|t}\right)-1\right)\right],\end{aligned}\quad a\neq 0.$$

<u>*Piecewise asymmetric loss functions*</u>

$$L\left(\varepsilon_{t+h|t};a,b,\rho\right)=\begin{cases}aL_{1}\left(\varepsilon_{t+h|t};\rho\right) & \varepsilon_{t+h|t}>0\\ bL_{2}\left(\varepsilon_{t+h|t};\rho\right) & \varepsilon_{t+h|t}<0\end{cases}\qquad a,b,\rho>0, \qquad (2.70)$$

where typically $L_{1}\left(\varepsilon_{t+h|t};\rho\right)=L_{2}\left(\varepsilon_{t+h|t};\rho\right)=\left|\varepsilon_{t+h|t}\right|^{\rho}$. Special cases are: $\rho=1$: Lin-Lin loss function and $\rho=2$: Quad-quad loss function, both non-differentiable at zero, but continuous, and asymmetric for $a\neq b$

2.8.2 Diebold-Mariano Test

The most important question is, how can we determine, if the out-of-sample fit of one model is significantly better than the out-of-sample fit of another model. Diebold and Mariano (1995) have proposed a test for the null hypothesis of equal predictive ability, against the alternative of non equal predictive ability. For two *nonnested* [44] models, let the $\left\{\varepsilon^i_{t+h|t}\right\}_{t=1}^{T}$ and $\left\{\varepsilon^j_{t+h|t}\right\}_{t=1}^{T}$ be the *h-step* ahead prediction errors. Under the assumption that errors are strictly stationary, the null hypothesis of equal predictive accuracy is specified as $H_0 : E\left[L\left(\varepsilon^i_{t+h|t}\right)-L\left(\varepsilon^j_{t+h|t}\right)\right]=0$, and $H_1 : E\left[L\left(\varepsilon^i_{t+h|t}\right)-L\left(\varepsilon^j_{t+h|t}\right)\right]\neq 0$. The statistic is based on loss differential,

$$d_t = L\left(\varepsilon^i_{t+h|t}\right)-L\left(\varepsilon^j_{t+h|t}\right), \tag{2.71}$$

is following:

$$DM_\tau = \frac{\frac{1}{T}\sum_{t=1}^{T} d_t}{\sqrt{\frac{1}{T}\sum_{\tau=-(T-1)}^{T-1} 1\left\{\frac{\tau}{S(T)}\right\}\hat{\gamma}(\tau)}} \overset{a}{\sim} N(0,1), \tag{2.72}$$

where $\hat{\gamma}(\tau)=\frac{1}{T}\sum_{t=|\tau|+1}^{T}\left(d_t-\bar{d}\right)\left(d_{t-|\tau|}-\bar{d}\right)$ and $1\left\{\tau/S(T)\right\}$ is the lag window, and $S(T)$ is the truncation lag. The statistics is based on the idea that for large samples the mean loss differential, which is the numerator in (2.72), is approximately normally distributed with mean μ and variance $2\pi f_d(0)$. In the denominator of (2.72), there is an consistent estimate of $2\pi f_d(0)$, which is *weighted* sum of the available sample autocovariances. For further details please see Diebold, Mariano (1995).

Thus we will test if the competing neural network model with out-of-sample prediction errors $\left\{\varepsilon^j_{t+h|t}\right\}_{t=1}^{T}$, is significantly better than a benchmark model with prediction errors $\left\{\varepsilon^i_{t+h|t}\right\}_{t=1}^{T}$. The DM_τ statistics is approximately normally distributed under the null hypothesis of no significant differences in predictive accuracy of the models. Thus if the neural network's predictive errors will be

[44] neither one is a special case of the other

significantly lower than for example ARIMA(p,I,q), the DM_τ should be below the critical value of -1.96 at the 5% critical level. Thus we will report the statistics and the p-values for it.

2.9 Economic significance tests

In the final analysis, the criteria will rest on the question: "how does the results of a neural network lend themselves to interpretations that make economical sense and give us better information for decision making?".

Let $z_{t+1}^{\kappa} \equiv E\left[r_{t+1}^{\kappa} \middle| \mathcal{F}_t \right]$ be the expected return on an optimal portfolio κ for period $t+1$, and r_{t+1} the rate of return on a risk-free asset at $t+1$, whose value is known at time t. For this study, portfolio κ will always consist of an asset being predicted. Simple asset allocation strategy is formed[45]:

$$\theta_{t+1} = \begin{cases} 1 & if \quad z_{t+1}^{\kappa} > r_{t+1} \\ 0 & otherwise \end{cases}, \tag{2.73}$$

where θ_{t+1} is the fraction of asset invested in the portfolio κ. So we will invest to an asset being predicted if the expected return is greater that a risk-free return, and vice versa. Thus realized return on this trading strategy x_{t+1} will be $x_{t+1} = \theta_{t+1} r_{t+1}^{\kappa} + \left(1 - \theta_{t+1}\right) r_t$.

2.9.1 The Henriksson-Merton measure

Henriksson and Merton (1981) proposed a non-parametric measure to evaluate the performance of the trading strategy described above. Let p_1 denote the probability of a correct forecast in an "down" market and p_2 be the probability of a correct forecast in an "up" market:

$$p_1 = \Pr ob\left[\theta_t = 0 \middle| r_t^{\kappa} \le r_t \right],$$
$$p_2 = \Pr ob\left[\theta_t = 1 \middle| r_t^{\kappa} > r_t \right].$$

[45] See Henriksson and Merton (1981), Lo and MacKinlay (1997).

$p_1 + p_2$ is a sufficient statistic for assessing the predictions[46]. A sufficient condition for forecast to have a positive economic value is $p_1 + p_2 > 1$, while the null hypothesis of no predictability can be formed as:

$$H_0 : p_1 + p_2 = 1,$$

against

$$H_1 : p_1 + p_2 > 1. \qquad (2.74)$$

Under the null hypothesis, n_1 - number of successful predictions in a "down" market has hypergeometric distribution that can be asymptotically approximated by normal distribution:

$$n_1 \overset{a}{\sim} \left(\frac{nN_1}{N}, \frac{n_1 N_1 N_2 (N-n)}{N^2 (N-1)} \right), \qquad (2.75)$$

where $N = N_1 + N_2$ is total number of observations with N_1 observations where $r_t^{\kappa} \le r_t$, $n = n_1 + n_2$ is total number of predictions that $r_t^{\kappa} \le r_t$, while n_1 is number of successful predictions, given $r_t^{\kappa} \le r_t$, and n_2 number of unsuccessful predictions. Thus null hypothesis can be tested with this statistics by referring n_1 to the critical values of normal distribution.

2.9.2 The Break-Even Transaction Costs

Another direct measure of the economic significance of stock return predictability can be found in Lo and MacKinlay (1997). Basically, they measure break-even transaction costs equating total return on an active market-timing trading strategy with the total return on a passive investment. The end-of-period value of a dollar investment over the entire period can be defined as:

$$W_T^P = \left(1 + r_r^{\kappa}\right),$$

$$W_T^A = \theta_t \left(1 + r_t^{\kappa}\right) + \left(1 - \theta_t\right)\left(1 + r_t\right),$$

where A,P are active and passive. If we switch between these two portfolios k times, the one-way transaction costs (100 x c) can be found from equation:

[46] Merton (1981)

$$W_T^P = W_T^A \left(1-c\right)^k ,$$

hence

$$c = 1 - \left(\frac{W_T^P}{W_T^A} \right)^{1/k} . \qquad (2.76)$$

(100 x c) are implied transaction costs and if we compare them with the real-world transaction costs, we will get a measure of economic significance of stock return predictability.

2.9.3 Pesaran and Timmerman non-parametric market timing

In financial time series one may be often interested more in the sign of the stock return predictions rather than the exact value. If we have good sign predicting model, we can use it for construction of simple signals. If the model predicts positive change, buy signal would be created, if negative change, sell signal would be created. Furthermore, if the predicted sign is the same as for the previous period, hold signal would be created.

Such statistics was formalized by Pesaran and Timmerman (1992) and is based on the null hypothesis that a given model has no economic value in forecasting the direction. The statistics is defined as follows:

$$PT = \frac{SR - SRI}{\sqrt{\mathrm{var}\left(SR\right) - \mathrm{var}\left(SRI\right)}} \overset{a}{\sim} N\left(0,1\right), \qquad (2.77)$$

where *SR* is success ratio computed as an weighted average of $I_h = 1\left\{ p_{t+h} . \hat{p}_{t+h} > 0 \right\}$, *SRI* is estimate of the probability of correctly predicting the direction of change assuming independence between the actual and the predicted directions, $SRI = D\hat{D} - \left(1-D\right)\left(1-\hat{D}\right)$, where D and $\hat{D}$ are weighted averages of an $I_h^{actual} = 1\left\{ p_{t+h} > 0 \right\}$ and $I_h^{predicted} = 1\left\{ \hat{p}_{t+h} > 0 \right\}$ respectively.

Thus the PT statistics is approximately distributed as standard normal, under the null hypothesis that the signs of the forecasts and the signs of actual variables are independent. Hence, if we will have a model with a very good predictive accuracy, forecasted and actual signs will be statistically dependent, and the forecasting model will have economic significance.

2.10 Black-box criticism

The growth in popularity of neural networks in recent years has led some researchers to make partial judgments in favor or against these models. In this section, we will review a few of these claims and discuss the black-box criticism. Let us start with few statements:

(i) Networks do not require the type of distributional assumptions used in econometrics
(ii) Networks are intelligent systems that learn
(iii) The early stopping procedure requires arbitrary decisions by the researcher

Some researchers, such as Aiken and Bsat (1999), claim that neural networks are not constrained by the distributional assumptions used in other statistical methods. However, as demonstrated by Sarle (1998), neural networks involve *exactly* the same type of distributional assumptions as other statistical methods. For more than a century, statisticians have studied the properties of various estimators and have identified the conditions under which these estimators are efficient, i.e. when they yield consistent unbiased estimates with a minimal variance. They discovered, for example, that efficient results are obtained when the errors are normally distributed with zero mean, are uncorrelated with each other, and have a constant variance throughout the sample. By rigorously identifying these optimality conditions, statisticians have been able to assess the consequences of the violation of these conditions. Since many neural networks are equivalent to statistical methods, they require the exact same conditions to attain an optimal performance. This implies, among others, that the residuals of a neural network should be subjected to the same diagnostic tests that are applied to the residuals of a linear regression model. Researchers who ignore these optimality conditions and proceed to estimate their network weights will obtain sub-optimal estimates. Most empirical studies involving neural networks do not pay attention to these optimality conditions.

Researchers also tend to ignore issues of stationarity when building their network. A prudent researcher should verify that all variables in the network are stationary before experimenting with different architectures. In fact, level variables that are trend stationary but that are not bounded could also pose

problems for the network. Since a hidden unit produces a value that is bounded, the use of input variables that grow continuously over time could eventually lead the hidden units to reach their maximal or minimal value. The contribution of each hidden unit to the network's output (which is given by the value of the hidden unit multiplied by the weight connecting it to the output unit) would then remain constant, even if the boundless input continues to grow over time. This would result in a deterioration of forecasting accuracy for subsequent periods. Similar problems would arise when attempting to forecast a level variable that grows continuously over time. Hence, even trend stationary level variables should be transformed so that they do not grow continuously over time (e.g. by using the first difference, the growth rate, the ratio to GDP, etc.)

Also when implementing the early stopping procedure, the researcher must make a certain number of arbitrary decisions that can have a significant bearing on the estimation results. First, the researcher must divide the sample into training, validation, and test sets. A commonly used "rule of thumb" consists in retaining 25 percent of the sample for the validation set and test set and with the remainder being allocated to the training set. However, this guideline does not have any theoretical or empirical foundations as results vary depending on data used. In addition, the researcher must decide which observations to include in each set. Some researchers assemble their validation set from the most recent observations in their time series, while others randomly select observations from the entire sample. Once again, there is no objective rule to this effect.

This criticism should not be overemphasized since a researcher can estimate the network using a different division of the data into the various sets and thus assess the sensitivity of the results to this allotment. Moreover, it is important to remember that econometricians make similar arbitrary decisions when they withhold observations from their sample in order to make out-of-sample forecasts. Econometricians using time-series data typically withhold an arbitrary number of observations from the end of their sample, since they are interested in assessing the model's capacity to forecast the future. To the extent that researchers in the neural network field assemble their validation and test sets from the last observations of the sample, they will be consistent with standard econometric practice.

The beauty of neural networks is that they can model behavior of agents without in the process of learning without giving them the model according to which they can change their behavior. A nice example is the Black-Scholes option

pricing model[47] which was found to approximate behavior of agents in the markets who are searching for the arbitrage opportunity. Nowadays, the model is used for options pricing and in fact, agents adjusted their decisions to it. Hutchinson, J.M., A.W. Lo and T. Poggio (1994) shown that neural networks can learn Black-Scholes very quickly. The reader can use this reference to learn more about this research. In the last chapter, we will use the neural network to price a warrant on Czech security and compare it to Black-Scholes pricing. And this is the example which shows us that neural networks has great potential in approximating of behavior of agents without "knowing" the model first. Neural network is able to find the price of the option even more efficiently than Black-Scholes, without using it, just by process of learning. Thus, even if philosophical question, black-box criticism can be easily turned down by this argumentation while neural networks perform in very efficient way of learning. Just as economic agents are in learning process.

2.11 Concluding remarks

We discussed the process of modelling series by neural networks in this chapter in depth so we can move further to test the theory on real data as *"Gray is the theory, green is the life" [48].*We Defined neural networks, discussed learning processes of finding optimal solutions and formalized it, we also discussed preprocessing data methods and closed the chapter with defining estimation criteria for our modelling. So we are ready to put the theory to test in next chapter.

We saw that when facing the task of estimating a model we have a large number of choices at all stages of the modelling process. We can assign different weights to in-sample and out-of-sample performance. We also have to decide e.g. whether to take logarithms and first-difference the data, deseasonalize or scale them, what type of network specification to use, which diagnostics should have more weight for and so on.

Most of these questions generally take care of themselves in the process of modelling. In general, we want to find out and compare the performance to linear models, we use the same data preprocessing and lags as we would use in linear models. Thus sometimes, linear models can help us in choosing the input

[47] Black and Scholes (1973), Merton (1973)

[48] Mephistopheles words from Goethe's tragedy *Faust*, Erster Teil, Studierzimmer.

variables of the network by estimating in-sample performance of it. Of course if we have linear model which is poorly specified it will not be hard for network to outperform it. Also in-sample performance of the network in comparison to well-specified linear model should be better. Real test of performance is on out-of-samples. After the inputs specifications, we start with simplest networks and search algorithms moving to more complex ones. Always we compare the performance by estimation criteria and if these do not improve by more complex methods, we should stick to the simpler ones. Commonly with more variables and more complexity we can have better feeling of explaining the variance of data, but we may also end up with disappointment when test the model on out-of-samples. Generally, we should not loose the parsimony as parsimonious models often outperform the more complex ones.

So the reader can see that it is a very complex process, we can say "state of the art" when researcher can influence the process in many ways and can directly improve the results by choosing different optimalization algorithms, or transformation functions in neurons. This is also one of the main drawbacks put to a criticism of neural networks – slowness of the estimation process. But as we will see time investment may bring some fruit.

Chapter 3

Application to Central-European Stock Market returns modelling

In this chapter, we will use the presented theory for modelling[49] of the Central-European stock markets with emphasis on the prediction task defined in 1.3. We believe that the emerging markets represent the best ground for the use of neural network models. The data are very often much noisier because the markets are very thin and also due to the speed with which the news spread among the market agents. Thus our assumption is that neural network should be able to help uncover the process.

As the motivation for the good modelling results of emerging markets the reader may be interested in following research that has been carried recently. Almost all results are very impressive. Nygren (2004) examines the predictability of Swedish stock exchange, Mohan, Jha, Laha, and Dutta (2005) examines neural networks predictive power on Bombay stock exchange, Cambazoglu (2003) finds impressive patterns on Turkish stock exchange. Finally, Yao, Tan, and Poh (1999) study Kuala Lumpur Stock Exchange with some impressive results.

Encouraged with previous research, we move to test the power of neural networks in Central-European Markets against linear methods discussed in the first chapter. Outline of this chapter is as follows: firstly we will use artificial Mackey-Glass time series for testing as these are not constrained with the sample

[49] Please note that all tests were carried out using Eviews 4.1 and Neuro Solutions 5.0 software – product that provide environment for neural networks modelling, and development of any learning procedures. Free 60–day, fully functional evaluation copy can be ordered at http://www.neurosolutions.com/ also with MATLAB or EXCELL extensions

length and should prove the ability of networks to discover and learn the pattern of series almost perfectly. Then we will model returns of Central European Stock indices daily and weekly, namely of Prague, Warsaw and Budapest stock exchanges which we believe describe the corresponding stock markets well. For comparison and more complex forecasting model development we will analyze also index of Deutche Boerse which is believed to be most liquid in continental Europe.

Finally, on the basis of cointegration analysis we will develop a robust forecasting model when the indices will be predicted among each others lags, as there has been recent studies of European Stock market cointegration – see Žikeš (2003) – who found European markets to be co integrated.

3.1 Example of a Mackey-glass artificial series

To show the power of neural network approach relative to autoregressive linear models, we start with simple example of artificial data modelling[50]. Very good motivation for use of these data is that there is no size-of-the-sample limits! The data are artificial which means that they are produced by model, and thus we know that there exist functional form. According to general approximation theorem, neural network should be able to learn the system by which the data are generated. For this purposes, we use Mackey-Glass[51] time series produced by a following stochastic time-delay difference system:

$$\frac{dx}{dt} = \beta x(t) + \frac{\alpha x(t-\gamma)}{1 + x(t-\gamma)^{10}}, \qquad (3.1)$$

where $x(t)$ is the value of the time series at the time t. This system is chaotic for $\gamma > 16.8$. We use the value of 30, and α, β values of 0.2 and -0.1 respectively. The data are scaled to (-1,1) interval:

[50] Reader is convinced to use the McNelis (2005) reference for more examples on artificial data modelling.

[51] Mackey and Glass (1977)

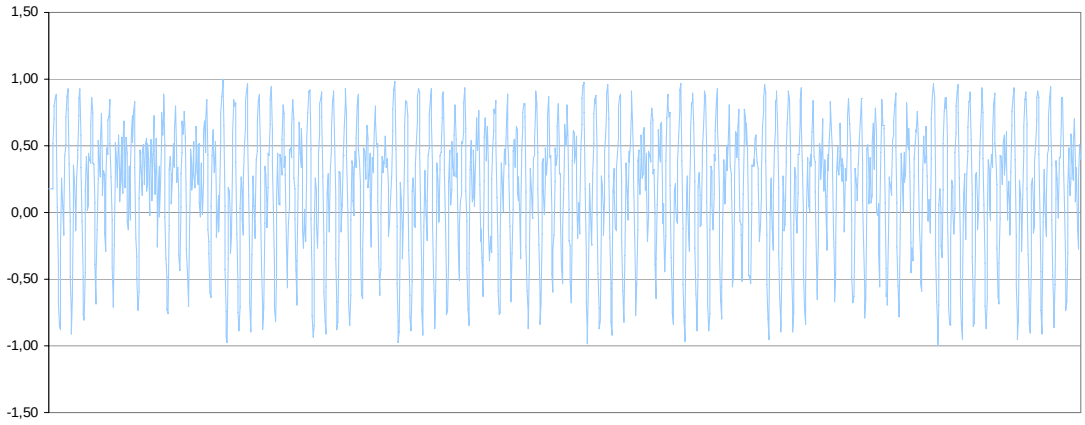


FIGURE 3.1: Mackey-Glass chaotic time series

Firstly, we reject the null hypothesis of normality with help of Jarque-Bera test statistic being equal to 86.3 at 1% significance level[52]. The value of test statistics of Augmented Dickey-fuller test exceeds the critical values so we can reject the null hypothesis of a unit root. Thus series are stationary. We find strong autocorrelation in the data, but we try first with simple regression - y_t being explained by y_{t-1}, y_{t-2} and y_{t-3}. Autocorrelation still remains strong in residuals even after estimating ARMA (p,q) model. We find that ARMA(2,2) best fits the data, but we still can not reject the null hypothesis of serial independence of residuals. ARCH-LM test strongly suggests the presence of heteroskedastic residuals, but we found that even GARCH(1,1) model did not help.

Table 1: Estimation results: Mackey-Glass chaotic time-series

Statistics	**data**	**Autoregression**	**ARMA(2,2)**	**NN**
adjR^2		0.8	0,84	**0.99**
Q-stats		165*	155*	
Schwarz criterion		-0,234583	-0,431651	**-7.6489**
ARCH-LM		80.729*	50,39*	
Dickey-Fuller	-7.289867*			
Jarque-Bera	86.73115*			
Out-of-sample results				
RMSE		0.212	0.1916	**0,0503969**
NMSE		0.162	0.132	**0,0100132**
			AR vs. NN	**ARMA vs. NN**
DM(0)			**-14.88***	**-14.11***
DM(1)			**-17.86***	**-14.42***
DM(2)			**-14.33***	**-12.26***
DM(3)			**-16.53***	**-13.39 ***

*1% significance level, DM statistics are comparing NN models versus benchmark linear models.

[52] For the distributions of time-series and all other results of tests see Appendix A

The in-sample performance of the models is quite good. Classical regression, ARMA(2,2) explains 80% and 84% of the variance in data respectively. Feedforward Neural network with one layer and 3 neurons with logsigmoid function and Levenberg-Marquardt optimization was chosen as an alternative to linear models. As we can observe from results, it explains 99% of an in-sample data. Schwarz information criterion is much better also. Results are very good as for linear models and network, but real test will be out-of-sample data testing[53].

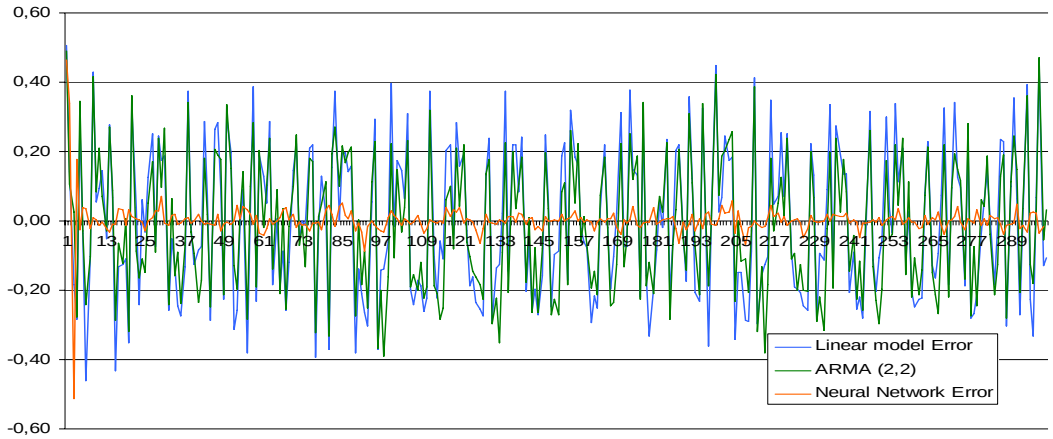


FIGURE 3.2: Out-of-sample prediction error comparison

For out-of-sample, we use Diebold – Mariano (chapter 2.8.2) to compare simple autoregression and ARMA (2,2) with neural network errors. DM statistics strongly rejects the null of no significant differences in predictive accuracy at 1% significance levels for all tested lags. Neural network also managed to explain 98% of the data. Errors can be compared in figure (3.2).

Of course we were testing artificial data thus data which were "created" and obviously must contain pattern. One would expect that if the data are artificial, good predicting model should recognize the pattern and use it for powerful predictions. As we see, linear models managed to uncover the pattern of artificial data well (ARMA little better than simple regression), but still neural model was much better in this task, when it predicted with better accuracy much more significantly than other models. We chose this example to show power of neural networks and their ability to learn the pattern. Clearly, if the underlying data were generated by a stochastic process, networks will be preferred over other tested models. Thus we showed that the general approximation theorem is valid, and we will see how the models will perform on the real data in next sections, or maybe better said, if the data are generated by any process which is to be uncovered or not.

[53] We divided 20% of observations for real-time forecasting.

3.2 European Stock markets

3.2.1 Data description

In the prediction task, we focus on sample of 1566 daily returns[54] from January 2000 until April 2006 and 382 weekly returns from January 1999 until April 2006 of value-weighted indices PX-50, WIG, BUX and DAX[55]. All the data were downloaded and regularly uploaded from Bloomberg during the research. Monthly returns were omitted because the sample size very small even for neural network. The descriptive statistics of the series is summarized in the following table.

Table 2: The descriptive statistics

	Daily (1564 observations)				Weekly (381 observations)			
	BUX	**DAX**	**PX-50**	**WIG**	**BUX**	**DAX**	**PX-50**	**WIG**
Mean	0,00067	-0,00005	0,00075	0,00056	0,00293	0,00337	0,00028	0,00329
Median	0,00049	0,00045	0,00081	0,00047	0,00240	0,00525	0,00332	0,00507
Maximum	0,06004	0,07553	0,04179	0,05593	0,09569	0,08719	0,12887	0,11501
Minimum	-0,07433	-0,08875	-0,06000	-0,08468	-0,13579	-0,09876	-0,13919	-0,18100
Std, Dev,	0,01410	0,01690	0,01248	0,01281	0,02967	0,02748	0,03383	0,03402
Skewness	-0,14797	-0,01262	-0,27616	-0,12427	-0,20928	-0,23586	-0,17928	-0,40852
Kurtosis	4,88697	5,61569	4,38258	5,54571	4,61303	3,69753	4,27986	5,35253
Jarque-Bera	237,74*	445,90*	144,45*	426,35*	44,09*	11,26*	28,05*	98,46*

*Significant at the 1% level.

Jarque-Bera test statistics tells us that all indices for daily and weekly returns deviate from normal distribution. This is no surprise to us because financial time series are well known to be *leptokurtic*, but we will have a closer look to an distribution to learn more about the shape of it. We will report histogram and non-parametric Epanechnikov kernel density estimator – which has the form of $K(u) = \frac{3}{4}\left(1-u^2\right) I\left(|u| \le 1\right)$ - for all series. The bandwidth h was selected according Silverman's rule of thumb, $h = 0.9kN^{-1/5}\min\left(s, R/1.34\right)$. See Silverman (1986, equation 3.31).

[54] To achieve stationarity all the data are first difference of log series $r_t = \ln P_t - \ln P_{t-1}$

[55] PX-50 – Prague Stock Exchange, WIG – Warsaw Stock Exchange, BUX – Budapest Stock Exchange and DAX – Deutche Boerse

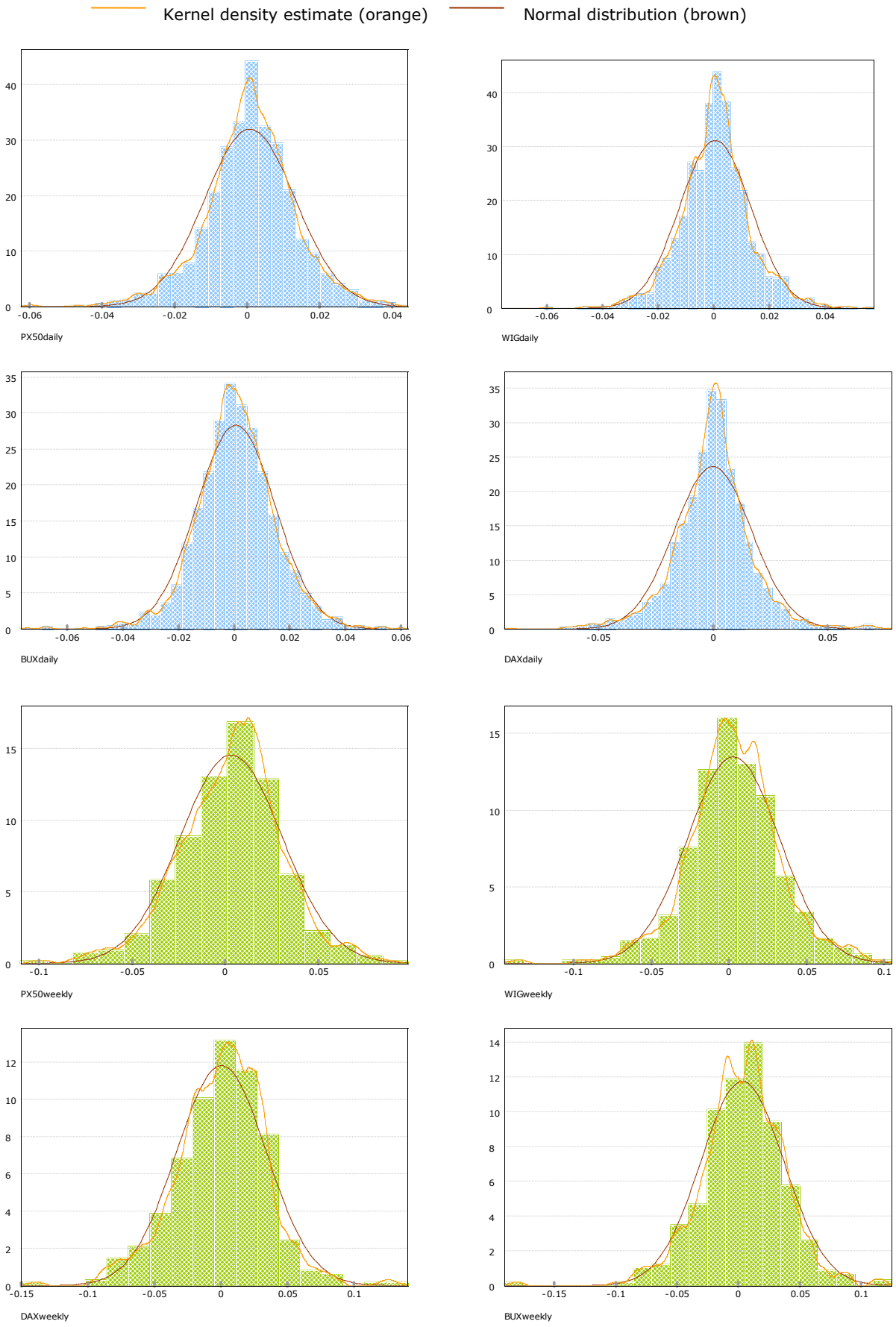


FIGURE 3.3: Histograms and Kernel density functions compared to normal distribution

Distributions of central European stock markets are in line with the developed stock market distributions. They are *leptokurtic* as expected which means that they are said to have heavy or fat tails. This may be attributed to conditional heteroskedasticity, so it is important to notice this before estimation.

3.2.2 Empirical results – daily returns

We start with modelling the daily returns of each index with ARIMA estimation. Augmented Dickey-Fuller statistics exceed the critical values on 1% significant level, thus we can reject the null of presence of unit root and state that all tested series are stationary. PX50 seems to follow ARIMA (1,0,1) best. BUX returns seems to be explained well by ARIMA(2,0,2), WIG and DAX does not contain AR and MA errors thus the random walk hypothesis can not be rejected for them. Ljung-Box Q statistics shows us the presence of conditional heteroskedasticity in the residuals from ARIMA models. So we will try to model it by GARCH(1,1) model as it turns out that this model rules not only with its parsimony, but also performance with these series. We find these ARIMA-GARCH models to be most appropriately specified. ARIMA(1,0,1)-GARCH(1,1) for PX50, ARIMA(2,0,2)-GARCH(1,1) for BUX, and GARCH(1,1) for DAX and WIG returns. According to results in Table 3 we can see that null hypothesis of no serial correlation can be clearly rejected with PX50 model and also with BUX model. Thus these models do not explain all of the variance and should be used with caution for forecasting prediction task. We will use them only as the representatives of linear modelling against the neural networks, because we did not find any better specification models for the data. This might be explained by use of daily stock returns which are autocorrelated due to the effect of nonsynchronous trading[56]. Thus in next sections the use of weekly data should improve performance of these models.

Table 4: In-sample performance on daily returns

	PX50		**BUX**		**WIG**		**DAX**	
	linear	neural	linear	neural	linear	neural	linear	neural
Adj R-squared	0.004888	**0,19**	0.021550	**0,11**	0.0019618	**0,09**	0.024190	**0,16**
Schwarz criterion	-6.020920	**-9,283**	-5.732452	**-8,58**	-6.002524	**-8,61**	-5.715768	**-6,5**
Ljung-box Q(4)	8,96*		3,8**		7,7		3,31	
Ljung-box Q(8)	13,312**		5,98		10,48		7,18	
Ljung-box Q(12)	16,42**		9,775		13,82		13,481	

*,**,*** significance on 1%, 5% and 10% levels

[56] For more details of this issue see Campbel, Lo, MacKinlay (1997)

Table 5: Out-of sample performance on daily returns

	PX50		BUX		WIG		DAX	
	linear	**neural**	Linear	neural	linear	neural	linear	neural
RMSE	0,0199	**0,00966**	0.0154	**0.149**	0.012	**0.011**	0.086	0.08
NMSE	1,003	**0,965**	0.999	**0.981**	1.012	**0.99**	1.007	1.012
D-M(0)	-1.1 (0.14)		-0.59 (0.25)		-0.98 (0.16)		**-1.72 (0.04)**	
D-M(1)	-0.91 (0.18)		-0.82 (0.2)		-1.09 (0.13)		**-1.78 (0.036)**	
D-M(2)	-0.83 (0.21)		-0.79 (0.22)		-1.2 (0.11)		**-2.02 (0.022)**	
D-M(3)	-0.71 (0.24)		-0.78 (0.21)		-1.16 (0.13)		**-1.72 (0.04)**	
H-M	1 (0.00)	**1.08 (0.00)**	1.01 (0.02)	**1.02 (0.00)**	1 (0.00)	1 (0.1)	1 (0.00)	1.03 (0.15)
P-T	51%(0.2)	**56% (0.07)**	53% (0.12)	**54% (0.02)**	54% (0.5)	54% (0.3)	62% (0.4)	47% (0.13
TC	0.002%	**1.2%**	0.31%	**1.1%**	0.002%	0.2%	0.03%	0.3%

D-M: Diebold-Mariano statistic (p-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs

In comparison to modern econometric tools, we will model stock returns using presented neural network methodology. Simple Feedforward Time-Delayed structure of network will be used in testing with 1 hidden layer, and Levenberg-Marquardt algorithm. Inputs were used 3 lagged variables mapped into 3 neurons as we found it provided best results. From results obtained, we can see that there is very poor pattern to be learned from our data. It seems that although indices returns are predictable to some extend, it is very small. Neural networks perform a little better with explaining the in-sample data. R^2 increases from 0.4% achieved by linear model to 19% achieved by neural net with PX50 index, and similarly with other indices as shown in Table 5. Schwarz criterion also favors to neural networks.

But real test of out-of-samples does not make very big difference between usage of linear and neural network models. We withheld 20% of the data as a rule of thumb for out-of-sample testing. As to the Diebold Mariano test, we can not reject the null hypothesis of equal predictive accuracy of linear and neural network models for all tested series, except DAX. Thus neural network model does not seem to have significantly different errors for the tested daily returns. Economic significance of predictions differs. For all linear models, we can not reject the null hypothesis of no predictability with Henriksson-Merton statistic[57] and neither Pessaran-Timmerman. Thus linear models have no economic value and should not be used for real predictions. Even implied transaction costs are on very low level. Situation is little bit different with neural network models. With PX50 and BUX data, we can reject the null of no predictability, while H-M is significant at 1% level for both data sets. P-T is significant at 10% level for PX50, and BUX also, which means that the null hypothesis of independence of actual

[57] we use PRIBOR as risk-free rate, and it will be used also in following tests

signs and forecasted signs can be rejected at 10% significance level. Implied transaction costs are higher than real world transaction costs. Thus even if neural networks could not beat the linear models with statistically significant lower errors, they seem to have economic value at least for two tested series.

Although we can gain some predictive edge with daily European Stock returns, time-series does not seem to explain themselves very well. It may be caused by autocorrelation which we could not remove, but as to the power of approximation ability of neural networks, we think the tested daily returns can simply be unpredictable, or producing not significant predictions. In the following section we will see if the weekly data will bring us better results and we will be able to gain some more predictive edge using neural network models.

3.2.3 Empirical results – weekly returns

Again we start with the very similar approach with weekly returns. ADF test confirms stationarity of the data, thus we can proceed to Box-Jenkins methodology. PX-50 follows ARIMA (1,0,0). Note that this result is interesting, because the weekly data contains MA errors no more. Other weekly returns are best explained with the same models as daily ones. After the observation of Q statistics we add GARCH(1,1) to model heteroskedasticity in the residuals and we end with ARIMA(1,0,0)-GARCH(1,1) for PX50, ARIMA(2,0,2)-GARCH(1,1) for BUX, and GARCH(1,1) for DAX and WIG returns. It is interesting that the null hypothesis of no serial correlation can not be rejected at 1%, 5% and even 10% significance levels. Thus models seems to explain most of the variance in the data and thus can be used for predicting.

Feedforward Time-delayed neural network architecture with 1 hidden layer, 3 inputs (lagged variables), logsigmoid squasher function and Levenberg-Marquardt algorithm is put to test. From obtained results we can see that in-sample improvement by the neural network seems to be really significant as to the explanatory power and Schwarz criteria.

Table 6: in-sample performance on weekly returns

	PX50		**BUX**		**WIG**		**DAX**	
	linear	neural	linear	neural	linear	neural	linear	neural
Adj R-squared	0,018	**0,48**	0,014	**0,15**	-0,00056	**0,28**	-0,00447	**0,34**
Schwarz criterion	-4,3765	-9,456	-3,95	-11,17	-4,25	-7,2966	-4,12	6,87
Ljung-box Q(4)	0,1034		1,445		7,07		3,236	
Ljung-box Q(8)	5,942		2,2489		16,331***		6,28	
Ljung-box Q(12)	8,087		7,6128		19,147***		10,814	

*,**,*** significance on 1%, 5% and 10% levels

Table 7: Out-of sample performance on weekly returns

	PX50		BUX		WIG		DAX	
	linear	**neural**	Linear	neural	linear	neural	linear	neural
RMSE	0,0206	0,01915	0.0342	0.015	0.025	0.019	0.022	0.02
NMSE	0,9806	0.978	0.993	0.987	1.03	0.97	1.029	0.99
D-M(0)	**-2.01 (0.022)**		**-1.78 (0.0375)**		-0.65 (0.25)		-0.67 (0.25)	
D-M(1)	**-2.03 (0.021)**		**-1.89 (0.029)**		-0.62 (0.26)		-0.54 (0.29)	
D-M(2)	**-1.85 (0.031)**		**-1.98 (0.023)**		-0.68 (0.24)		-0.48 (0.31)	
D-M(3)	**-1.94 (0.025)**		**-1.8 (0.035)**		-0.8 (0.21)		-0.51 (0.29)	
H-M	1.07 (0.04)	**1.09 (0.00)**	0.9 (0.02)	**1.01 (0.05)**	1 (0.2)	**1.1 (0.00)**	1 (0.00)	**1.2 (0.06)**
P-T	58% (0.25)	**60% (0.09)**	58%(0.2)	60%(0.12)	0.55%(0.15)	**58% (0.09)**	55% (0.3)	**58% (0.07)**
TC	0.4%	**0.8%**	-0.6%	0.1%	0.01%	**1%**	0.03%	**0.7%**

D-M: Diebold-Mariano statistic (p-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs

Let us turn to more interesting out-of-sample forecasts. Diebold-Mariano tells us that neural networks have significantly lower error compared to linear models with PX50 and BUX, as null hypothesis of equal predictive accuracy can be rejected at 5% significance level for all lags. For other two tested series, WIG and DAX, the null of equal predictive accuracy can not be rejected, thus for these data, the models performs statistically similar. As to the economic significance of forecasts, we reject the null hypothesis of no predictability using H-M for PX50 and WIG series at 1% significance levels, and for DAX at 10% significance level. According to P-T, the neural networks has also significant sign predictions, as the null hypothesis of independence of signs between predicted series and actual ones can be rejected at 10% significance levels. Implied transaction costs are quite low, but slightly higher than those of real world[58]. Models did not perform well only with BUX series, where we can not reject neither the null hypothesis of no predictability, nor the null of signs independence.

From preceding tests we can conclude that there is a predictive edge in the European-stock markets. Neural networks seem to explain the time series a little better than classical approach. When facing the prediction task, the results are also improved. We can say that with significant chance of 3:2 next week's return can be predicted with use of raw price data with neural network. We use these results as the starting point for development of more robust model in next subchapter. While it is clear that one can gain abnormal returns using presented methods, we will try to propose different model which will use not only the lagged

[58] we found real-world transaction costs for an 10.000 EUR investment in Czech Republic to be cca. 0.05% in average.

variables of the time series itself, but also other variables to gain more explanatory power and robust results even on daily returns.

3.3 PX-50: Gaining the predictive edge

In previous sections we found that the European Stock markets contain predictable components, but use of the models with lagged data does not seem to provide us with strong results[59] on daily data. On the weekly data models performed significantly better in two cases, and we managed to gain economic significance almost for all tested series. In this chapter we will continue with different approach. We will try to find empirical relationship between the European Stock Markets and if we manage to find any, we will use it to build a model which would bring us deeper understanding of the PX-50 stock market returns. In this part, we will use the same daily data as described in previous section[60].

3.3.1 Cointegration of BUX, WIG, DAX and PX-50 markets

Our first hypothesis is that PX50, DAX, BUX and WIG are co-moving and thus the returns of these markets can be used to bring more light into their patterns and to predict each other. Žikeš (2003) provided us with results of Johansen multivariate cointegration analysis and found that all markets are influenced by at least one lagged variable of neighbor markets. Instead of conducting the same research and receiving the same results we will try to use his results in our modelling. Let us firstly examine very illustrative figure FIGURE 3.4 where we plot daily returns of all indices normalized to interval (0,1).

[59] The results are not that bad though. Reader should keep in mind that if we can predict future returns with 55%-60% accuracy we have "3/2 : 1" ratio of winning to loosing trades. If we manage to predict returns with 70% accuracy it is actually excellent result as we have a "7/3 : 1" ratio of winning to loosing trades and we can consistently earn abnormal returns from the market.

[60] We just remind that for all of the tests we divide the tested sample into 70% in-sample, 10% cross-section and 20% out-of-sample for neural nets, and 80% : 20% for regressions.

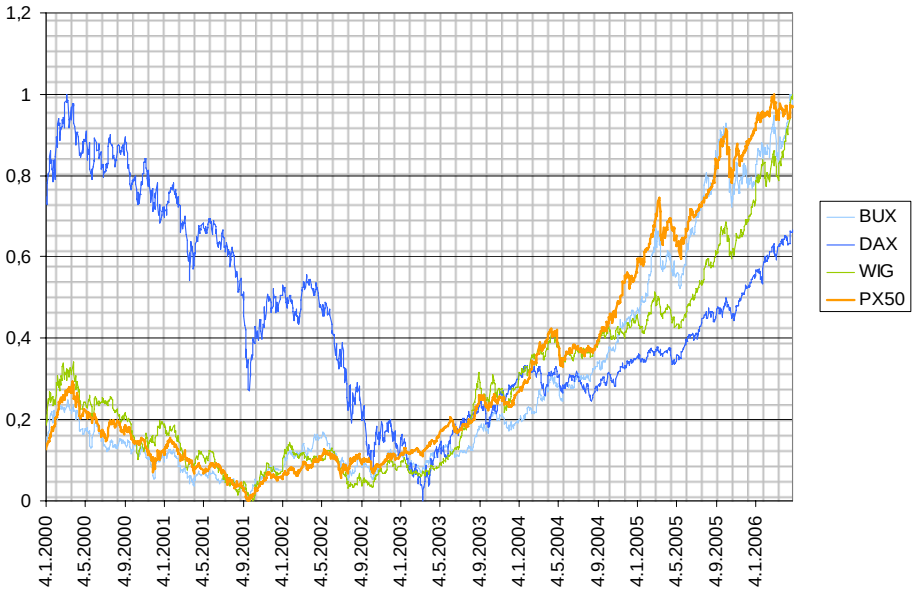


FIGURE 3.4: Daily price of BUX, DAX, WIG, PX-50 scaled to (0,1)

From the figure it is clear that the Czech, Hungarian and Polish markets are moving together, German market was falling much faster in the period of 2002 – 2003 and in the middle of the year 2003 it joined other markets but underperformed them. From this period we can see that the markets are co-moving. With empirical rigorous background of Žikeš's analysis, we can use this information for the PX50 stock market returns prediction.

First of all, we conduct a PCA analysis to find which vectors influence market returns most. We will conduct classical regression PCA analysis and also nonlinear neural network PCA described in section (2.6.3) for all four indices. logsigmoid squasher function and Levenberg-Marquardt optimalization mechanism will be used. The results are in following table:

Table 8: Results of PCA

	PX50		BUX		WIG		DAX	
	classical	neural	classical	neural	classical	neural	classical	neural
Adj R-squared	0,28[1]	0,31	0,35[2]	0,4	0,32[3]	0,34	0,18[4]	0,24
Schwarz criterion	-6,36	-9,13	-6,2	-9,13	-6,47	-9,25	-5,42	-8,2
Ljung-box Q(4)	10,98**		20,2*		8,58***		17,23*	
Ljung-box Q(8)	13,241		30,058*		9,91		55,03*	
Ljung-box Q(12)	15,232		33,97*		14,69		59,95*	

[1] PX50 returns are being explained by BUX, WIG and DAX with coefficients 0.276*, 0.243* and 0.076* resp.
[2] BUX returns are being explained by PX50, WIG and DAX with coefficients 0.322*, 0.376* and 0.117* resp.
[3] WIG returns are being explained by PX50, BUX, and DAX with coefficients 0.2189*, 0.290* and 0.1075* resp.
[4] DAX returns are being explained by PX50, BUX and WIG with coefficients 0.191*, 0.258* and 0.30* resp.
*,**,*** significance levels of 1%, 5% and 10% resp.

Thus we can see that really all markets are influencing each other and are moving in tight range. Thus we may try to use lags of PX-50, BUX and WIG for explaining their variance and follow the analysis from previous chapters. The reader noticed that DAX coefficients are smallest thus DAX surprisingly does not have such big influence on the three market indices. They explain themselves best and this information can be used also for their prediction in following text.

Not so surprisingly, DAX is not explained well with PX-50, BUX and WIG returns. This is caused mainly by the fact that half of the tested period the DAX was moving faster against remaining markets. If we divide the sets to 2 subsets of pre-2003 and after-2003 we would find much better results in the second period. But this is obvious from the FIGURE 3.4 so we will leave this part as an exercise for interested readers as we will provide the division to the sub-periods in next out-of sample forecasting tests. Thus for now the results are clear and we will move further to use them for real forecasting of the market returns.

3.3.2 Cross-market predictions

In previous subchapter we found that the PX-50, BUX, WIG and DAX returns are co-moving thus now we will be interested if this information can be used for the forecasting. The methodology here will be quite different. We will try to forecast the one day return of the market with use of the lags of 3 remaining markets. For this purpose we apply correlation analysis[61] to find which lags influences the returns most. Than we will use linear OLS estimate and Feed Forward Neural Network again with best performing logsigmoid transformation function and Levenberg-Marquardt search algorithm. Following models were developed[62]:

$$PX50_{t+1} = \beta_0 + \beta_1 PX50_{t-1} + \beta_2 PX50_{t-5} + \beta_3 BUX_{t-1} + \beta_4 BUX_{t-3} + \beta_5 DAX_t \quad (3.2)$$

$$BUX_{t+1} = \beta_0 + \beta_1 BUX_{t-3} + \beta_2 BUX_{t-5} + \beta_3 DAX_t + \beta_4 DAX_{t-2} \quad (3.3)$$

$$WIG_{t+1} = \beta_0 + \beta_1 WIG_t + \beta_2 WIG_{t-5} + \beta_3 PX50_t + \beta_4 PX50_{t-3} + \beta_5 DAX_t + \beta_6 DAX_{t-2} \quad (3.4)$$

$$DAX_{t+1} = \beta_0 + \beta_1 DAX_{t-3} + \beta_2 DAX_{t-4} + \beta_3 PX50_{t-5} + \beta_4 WIG_{t-4} + \beta_5 BUX_{t-1} \quad (3.5)$$

---

[61] We use sample correlation coefficient - Pearson product moment correlation coefficient which is the best estimate of the correlation between two series to determine the potential explanatory variables. We pick all variables with correlation coefficient statistically significant at 1%, 5% and 10% levels.

[62] Estimates can be found in appendix B

Table 9: In-sample performance of the daily models for whole tested period

in-sample	PX50		BUX		WIG		DAX	
	classical	neural	classical	neural	classical	neural	classical	Neural
Adj R-squared	0,0234	**0,12**	0,015	**0,202**	0,023	**0,17**	0,019	**0,11**
Schwarz criterion	-6,038	**-8,99**	-5,78	**-8,85**	-6,08	**-9,409**	-5,23	**-7,98**
Ljung-box Q(4)	1,31		5,53		0,96		2,8	
Ljung-box Q(8)	2,99		5,94		1,302		24,69	
Ljung-box Q(12)	3,016		6,53		4,05		30,51*	

*,**,*** significance levels of 1%, 5% and 10% resp.

As we can see, PX50, BUX, WIG and DAX returns seems to be explained to some extend with their mutual lags. As to the comparison of the autoregressive model with neural network, neural network leaves the autoregressive models far behind. As to the explanatory power, neural network explains 12%-20% of variance of the returns in individual model, while autoregression only 1.5%-2.34%. Schwartz criterion is also preferring networks much better. So Implication for the modelling would be very intuitive – use linear regression model to identify significance of the variables and then improve the estimates with neural networks. The reader can observe very interesting thing – there is no autocorrelation present in the models. Ljung-box Q statistics were not significant at any level for any Q(k). So the results suggests us, that we could gain some predictive edge from these models.

Again, we will be concerned with out-of-sample testing more than in-sample. In Table 10 we have the results for whole testing period.

Table 10: Out-of-sample performance of the daily models for whole tested period

	PX50		BUX		WIG		DAX	
	Linear	**neural**	linear	neural	linear	neural	linear	neural
RMSE	0.09	**0,009**	0,0152	**0.014**	0.012	0.0095	0.0086	0.008
NMSE	0,985	**0.978**	0.994	**0.96**	1.014	0.999	1.014	0.9878
D-M(0)	-0.71 (0.23)		-0.26 (0.59)		-0.06 (0.52)		**-1.9 (0.02)**	
D-M(1)	-0.71 (0.22)		-0.23 (0.59)		-0.08 (0.52)		**-1.99 (0.02)**	
D-M(2)	-0.68 (0.24)		-0.23 (0.59)		-0.08 (0.53)		**-1.91 (0.02)**	
D-M(3)	-0.65 (0.25)		-0.24 (0.59)		-0.077 (0.53)		**-1.8 (0.036)**	
H-M	1.03 (0.06)	**1.07 (0.00)**	1.04 (0.2)	**1.06 (0.05)**	0.98 (0.00)	1 (0.05)	1.02 (0.15)	**1.07 (0.00)**
P-T	56%(0.056)	**59%(0.06)**	52%(0.27)	**57%(0.05)**	52%(0.61)	56% (0.21)	53% (0.22)	57% (0.12)
TC	0.6%	**1.2%**	1%	**1.7%**	-0.62%	-.46%	-0.45%	0.2%

D-M: Diebold-Mariano statistic (p-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs

In our final tests, Diebold-Mariano tells us that for almost all series the errors of linear models and neural ones are almost identical, while we can not

reject the null hypothesis of equal predictive accuracy for PX50, BUX, WIG. But we can reject the null hypothesis of no predictability for PX50, BUX and DAX at 1%, 5% and 1% significance levels resp. We also reject the null hypothesis of independence of directional change of actual and predicted series for PX50 and BUX at 10% significance level. Implied transaction costs are also in line with H-M and P-T statistic, while they confirm economic significance. But again, we were not able to gain consistently and significantly better predictive power for all tested series, even with usage of neural network models. This may imply that the daily European stock market returns are simply unpredictable, as the lags of surrounding markets did help to explain the variance very little.

3.4 Concluding remarks

At the very beginning of this chapter we illustrate the power of neural network modelling. Our hypothesis was that if the neural network can approximate any function, it must be capable of approximating artificial chaotic time series. And we showed that it performed very well on the Mackey-Glass chaotic time series, even in the prediction of them. We compared classical econometric approaches to model the Mackey-Glass chaotic time series with neural network, and showed that neural network performs much better in the task with significantly lower errors. Thus we showed that neural network is capable of approximating any process, hence it is very strong instrument for our prediction task.

Next we moved to real world data, the Central European stock market returns represented by PX-50, BUX, WIG and DAX indices. We described the data first and found no deviation from distributional properties of other developed and more liquid markets which was no surprise to us. More interestingly, we conducted the in-depth analysis of daily returns, followed by weekly returns and found that neural networks can be used to improve predictive power of the classical models only slightly. For daily returns, neural networks improves only economic significance, but the prediction are not significantly different from linear models. We conclude that daily European stock market returns may not contain any significant pattern to be uncovered when using historical prices. With weekly returns neural networks performed significantly better than linear models on PX50 and BUX markets. Economical significance was also gained for 3 out of 4

markets, while networks achieved around 60% directional accuracy on weekly data, which is quite good result.

Thus finally on the basis of cointegration analysis we modeled the returns with lagged variables of all four indices as we found they are significant to the returns. In fact, it is logical step as the markets are moving together, and mainly in these days of globalization, world markets are trading very tightly. In the times when this research had been conducted, NASDAQ[63] unsuccessfully bid for the London Stock Exchange. Few months later in late may 2006, NYSE[64] and EURONEXT[65] bourse announced the merger and creation of first transatlantic exchange behemoth, the largest stock exchange in the world. Thus markets are no more depending only on local economical issues, but surly weaker exchanges follow stronger ones.

But the analysis did not bring the fruit, as the daily lags of surrounding markets did not improve our results. Again we could not reject the null hypothesis of equal predictive accuracy of the used models, and economic significance was very similar to the analysis conducted in the chapters before. Thus we conclude that daily European Stock markets may not contain any predictable pattern even if the lags of surrounding markets are used.

An attentive reader will note that one can try to improve or modify the model for real trading and use indicators, or smoothed prices. We obtained the results with raw stock markets returns, but for instance, if exponential moving averages are used to smooth the stock market returns, the prediction of short-term direction is even stronger. We showed that a good predictive model can be build from raw data and we will leave the exercise of using other inputs of moving averages, or indicators to the reader. For example lagged moving averages of 5 days may predict a one week ahead return well as they smooth the series. And there is much more models to be used depending on the strategy we want to achieve. But we also draw attention to the problem of relevance of the data used. Neural network can approximate any process but when building the model, bear in mind that if you input data which are of no importance into the model, it will return nothing else but forecasts which will be not be applicable. The relevance of the inputs is crucial for good results. In next chapter we will induce implications for derivative pricing methods.

---

[63] USA Technological stock Exchange

[64] New York Stock Exchange

[65] second largest European bourse – integration of Bruxelles, Paris and Amsterdam

Chapter 4

Application to pricing derivatives

In previous chapter we concluded that with the use of neural networks we are able to gain a predictive edge. Of course this is very strong implication for the markets and traders, but still, it is of quite speculative usage. And of course there are many problems of using these models in real trading. The main drawback is for example that most of the models are behaving in the manner that they tend to predict the movement with some lag. This is fine if the markets are steady and the model captures the short-term trends well, but if there are unexpected exogenous moves or crushes of the stock market, the models very often fail to warn us. Of course it depends on the input variables used, but still one should never base his/her trading strategy only on the predictive model as other part of the success is understanding the market and proper reaction to economic news. Of course, the modelling of the market returns and uncovering the pattern serves to a trader very well in gaining abnormal returns in the market if combined also with understanding of the markets.

Much stronger implications of our findings can be made for another very interesting area – pricing and hedging of the derivatives. Well known Black-Scholes[66] model for pricing of European call options is based on assumptions which are unrealistic. Stock prices under the log-normal distribution follow geometric Brownian motion, volatility is constant over time and returns are normally distributed. But these assumptions are nonrealistic. Our study only extends the empirical literature which shows that based on this assumptions, Black-Scholes can not be used for rational pricing of the options. We just showed that the returns are strongly predictable, thus are far away from random walk,

[66] Black and Scholes (1973), Merton (1973)

and the biggest problem is constancy of volatility. One solution to the problem is to re-estimate the model every day with "new" - updated volatility which will be set to constant, but this approach for example does not decrease hedging error which is crucial for big institutional traders.

In the following chapter we will show how neural networks can be used to option pricing much efficiently on the basis of universal approximation theorem. We will start with very brief theoretical introduction, which will be followed with application to an pricing of an warrant which underlying is the largest and second most liquid stock on the Prague Stock exchange – CEZ.

4.1 Theoretical framework proposed by Black and Scholes

Much of a growth of the market for options and other derivatives is linked to the famous papers by Black and Scholes (1973) and Merton (1973) in which closed-form option pricing formulas were obtained through a dynamic hedging argument and no-arbitrage condition. This approach has been generalized to pricing of an array of securities, and even if there is no close-form solution, pricing formulas can be obtained numerically.

The basics of the model lies on the assumption of the hedging/no-arbitrage approach, underlying price dynamics $S(t)$ which is assumed to follow geometric Brownian motion:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t), \qquad (4.1)$$

where μ is expected gain or constant drift, σ volatility and $W(t)$ is Wiener process[67]. Let $C(S,t)$ be the value or price of the European[68] call option on non-paying dividend stock. For $t < T$ the pay-off is following:

$$e^{-r(T-t)} \max\left(S(t) - X, 0\right), \qquad (4.2)$$

Thus under the assumption of lognormal distribution of stock prices where

[67] Continuous-time Gaussian stochastic process with independent increments.

[68] Basic divisions of options is call option (right to buy underlying security for given strike price in given time) and put option (right to sell underlying security for a given strike price in the the given time). *European options* can be exercised only at the expiry date, while *American option* can be excercised at any time before the expiry date.

$$d\ln S(t) = \left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma dz\,, \tag{4.3}$$

$$\ln S(T) - \ln S(t) \sim \Phi\left[\left(\mu - \frac{\sigma^2}{2}\right)*(T-t), \sigma\sqrt{(T-t)}\right], \tag{4.4}$$

the Black-Scholes formula is derived[69]:

$$C(t) = S(t)\Phi(d_1) - Xe^{-r(T-t)}\Phi(d_2)\,, \tag{4.5}$$

$$d_{1,2} = \frac{1}{\sigma\sqrt{T-t}}\left(\ln\frac{S(t)}{X} + r \pm \frac{1}{2}\sigma^2\right)(T-t)\,, \tag{4.6}$$

where $\Phi(.)$ represents cumulative normal distribution function, $S(t)$ is price of underlying asset, X is strike price or exercise price, r risk-free interest rate, σ volatility and $(T-t)$ time to expiration. To be complete, we just note that price of put option can be obtained from put-call parity $S(t) + P(t) = e^{-r(T-t)}X + C(t)$.

This approach to option pricing led to great boom of derivatives trading in 1970's and 80's respectively. Of course from that time there was an mounting evidence that this solution leads to an errors in pricing of the derivatives, but until now no-one came with appropriate substitute of the model. Main drawbacks are misspecification of process of Stock price $S(t)$ leading to systematic pricing and hedging error of derivatives. Another crucial assumption is constant volatility which is not realistic at all. Another issue is also pricing of *American options*, the ones which can be executed any time, not only at time of expiration.

4.2 Neural network approach to derivatives pricing

Purpose of this chapter is to introduce another – data driven – method of derivative pricing, where the data will determine the dynamics of the $S(t)$ and its relation to the derivative security. Assumptions of constant volatility and lognormal distribution of the underlying process can also be relaxed. On the basis of the assumption of *universal approximation property* of neural networks we assume that network must be capable to learn the Black-Scholes formula. If it is

[69] We advice to use the references for exact derivation and for better understanding of the model while it is not our intention to repeat what has been written in thousands of publications.

true, than it can also be trained on the real data and optimal model with optimal weights "becomes" the derivative pricing model. Thus we expect that the neural network can better approximate the price of derivative through learning process than Black-Scholes formula, and can be used to minimize error of hedging or pricing of the derivatives.

Methodology of neural network has been presented in-depth in previous chapters, thus we have strong theoretical background for the testing. All we need to do at this point is to "let the data speak" and move to most interesting part – empirical results. Before we do so, we would like to draw attention to advantages and disadvantages of the neural network usage to derivatives pricing. Firstly, networks does not rely on restrictive parametric assumptions described above, they are robust to the specification errors that plague parametric models, and more important, they are also adaptive and respond to structural changes in the data generating process. Finally they are flexible enough to encompass a wide range of the price dynamics.

On the other hand the advantages comes to cost of large amounts of data needed to best optimalization of weights. Therefore the approach would not be appropriate for newly issued instruments. Another cost is that if the underlying asset's prices is well understood and can be analytically expressed, network will probably not outperform the Black-Scholes. But we have to say that this case is very unlikely on today's markets. Also first drawback turns out to diminish if we consider that there are always amounts of derivatives available to the same asset on the market, thus the newly issued derivative can often be replicated using these data as the underlying process is identical.

In the next section we will put our hypothesis to test. We will try to learn and price the call warrant on CEZ, currently second most liquid stock on Prague Stock exchange which forms 25% of the base of PX-50 index[70]. Czech market is considered as an emerging market, and the liquidity can not be compared to biggest world markets. What is more important, the warrant on CEZ is not directly traded in the Czech stock exchange and in the times this thesis was being finished, there was also no legal regulation of this derivative on the Czech stock

[70] CEZ has been largest stock on the PSE until Erste Bank placed its stock emissions to the market few months before this thesis was finished, 2.2.2006

markets[71]. Thus the derivatives based on the Czech stocks are being traded tightly and the pricing of them is more difficult as pricing of much more liquid options in united states. Now we know the methodology, chosen warrant will be closely described in next section, but to be complete, we need to define warrant first as we refer to it without definition in previous text. Definition:

Warrant is a security that entitles the holder to buy or sell a certain quantity of an underlying security under agreed price and exercise period. The right to buy is referred to as an call warrant, right to sell as an put warrant.

The reader might be confused with the difference between warrant and option, as the definition might seem identical. But when warrant is exercised, a new share of a stock is created while this does not happen with options. Main difference is also that options are being issued by independent parties, such as Chicago Board Options Exchange, while warrants are issued and are guaranteed by companies, such as Special Purpose acquisition company, or large banks which find warrants to be very good dynamic investment instrument nowadays. Another difference is in lifetime of the derivative. We talk about years in warrants, but months in options. The last inequality is in the basis of the derivative. While talking about options we talk about 100 options in one contract and 1 option means right to buy/sell 1 stock of underlying asset, in warrant we can often meet ratios from 0.01 or 0.1 meaning that you need 100 resp. 10 warrants to have and right to buy/sell 1 stock. But the reader is right if he/she does not see the difference in pricing of warrants and options, because they are identical as to this issue.

4.3 Pricing of CEZ Call warrant

4.3.1 The data

In this section we will perform an empirical testing where we want to compare the price of Black-Scholes model, learned neural network and real market price of an call warrant with underlying security CEZ, strike price 500 CZK and maturity 14.6.2006. Holder of one warrant has the right to buy one CEZ

[71] new legal regulations and also vast of derivative securities such as warrants, certificates, turbo-certificates are being prepared for the Czech stock market to be issued while this thesis is being finished.

stock for 500 CZK at the expiration, thus this is European warrant. Warrant was issued by Deutsche Bank 22.10.2004 and is traded in Stuttgart, EUWAX bourse, or directly with eminent in EUR. ISIN of the security is DE000DB21187. We have the data of daily closing prices of CEZ security denoted in EUR, and closing price of warrant from 26.4.2005 until 24.4.2006. Thus we will test one year of data, meaning 253 observations which should be enough. The data were downloaded directly from EUWAX.

First of all, let us have a look at the distributional properties of underlying, CEZ security.

Table **11**: the descriptive statistics of daily CEZ returns

Mean	**Median**	**Maximum**	**Minimum**	**Std. Dev.**
0.002936	0.003744	0.066613	-0.083199	0.018375
Skewness	Kurtosis	**Jarque-Bera**		
-0.529039	5.847853	97.29744*		

*Significant at the 1% level.

Jarque-Bera statistics rejects normality of CEZ returns at 1% significance level. Thus we can again conclude that returns are *leptokurtic* which can be well observed from following figure. Epanechnikov kernel density[72] (orange) line has excess kurtosis over the Gaussian normal distribution (brown), and has also fatter tails.

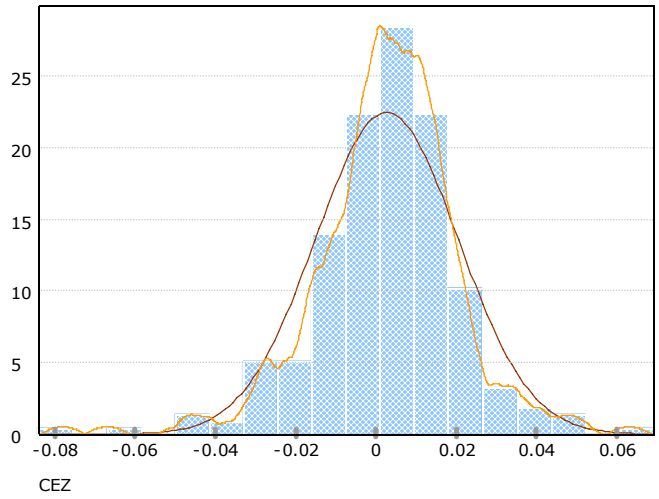


FIGURE 4.1: Histogram, Kernel density function of daily CEZ returns

[72] The same as with previous data in section 3.2.1.

Even if this should be no surprise to any researcher in quantitative finance, we again remind that basic assumption of log-normality is violated, and this property should be in favor for neural networks, and of course, against Black-Scholes.

4.3.2 Learning the Black Scholes formula

Given the power and flexibility of the networks to approximate any complex nonlinear relation, we begin with learning the Black-Scholes price of the described CEZ call warrant. This means that we compute the prices of the warrant using the (4.5) model on the daily basis. To be more realistic, we relax the assumption of the constant volatility and compute volatility on the daily basis as standard deviation for last 20 trading days[73]. We then estimate the price which is generated by differential Black-Scholes equation using Feedforward neural network with one hidden layer, sigmoid transformation function and Levenberg-Marquardt optimalization algorithm. We shall note that the 80% of the data were used to training the network, and rest to testing, or as an out-of-sample.

Table 12: estimation results for Black-Scholes learned by network

In-sample	**Neural Network**
Adj R^2	**0.9999963**
Schwarz criterion	**-8.15**
Out-of-sample results	
RMSE	**0,0504**
NMSE	**0,0202**

From the results we can conclude that the network is able to efficiently approximate the Black-Scholes pricing formula, which is no surprise to us as mentioned before. While network performed very well on the artificial data of Mackey-Glass chaotic time series, it is logical that it could learn the Black-Scholes also very well. Hence we can conduct more interesting test, use the neural network to the pricing of warrant and compare with real data. Then we will clearly see the errors of Black-Scholes and errors of neural network, compare them and see if the neural network can approximate the derivative price more efficiently or not.

[73] As this approach is widely used among traders and financial theory.

4.3.3 Performance of Neural Network in warrant pricing

In the final tests we will aim on comparing the real price collected from EUWAX[74], theoretical Black-Scholes pricing and neural network pricing. The method is simple. We will use the derived Black-Scholes price for each day from previous part, learned neural network price and compare their errors to real market price for the day.

Inputs to neural network used will be price of the underlying CEZ in EUR and time (T-t). While interest rate and volatility are assumed constant in Black-Scholes model, we will not include them. Moreover, we will hope that the network will learn also changes in these parameters and can capture it from the data, as the assumption is unrealistic as discussed earlier. Thus two inputs will be confronted with the real market price, and then the obtained network model will be used to real pricing at out-of-sample data. We should note that we use raw data, no differencing as we try to approximate the price of the derivative, not to predict the return. This may result in worse results as if we used derived, e.g. normalized data. Moreover we will compare feedforward network with one layer with conjugate gradient search and Levenberg-Marquardt search as we did not attach the comparison in previous tests. We note that in all previous tests the two algorithms were used and results were similar – Levenberg-Marquardt performed much better on stock market data. The results are following:

Table 13: in-sample performance comparison

insample			
	BS	***NNconj***	**NNlevenberg**
Adj R^2	0,979	0,999	0,996
r	0,97	0,999	0,998

Table 14: out-of-sample performance comparison

Outofsample			
	BS	***NNconj***	**NNlevenberg**
RMSE	0,224	0,198	0,078
NMSE	0,458	0,358	0,092
r	0,76	0,802	0,93
	BS vs. NNconj	**BS vs. NNlevenberg**	**NNconj vs. NN levenberg**
D-M (0)	-1.17 (0.12)	**-1.71 (0.04)**	**-1.76 (0.035)**
D-M (1)	-1.13 (0.12)	**-2.29 (0.01)**	**-1.78 (0.038)**
D-M (2)	-0.99 (0.15)	**-1.98 (0.02)**	**-1.61 (0.053)**
D-M (3)	**-1.38 (0.08)**	**-2.94 (0.00)**	**-1.68 (0.045)**
D-M (4)	**-1.72 (0.04)**	**-2.59 (0.00)**	**-1.63 (0.049)**

[74] Trading platform where the warrant is traded

From the results we can see that the in-sample performance is very good determined with very high coefficient of determination. Small r refer to linear correlation coefficient between the estimated output vector and real vector output. Out-of-sample results are very impressive also. We can see that Neural network outperforms classical Black-Scholes approach far, as NMSE is much lower. Diebold Mariano statistic is not significant only for lags 0,1,2 when comparing BS and NN with conjugate gradient. Thus the null hypothesis of equal error functions can not be rejected for these lags. When comparing NN used with Levenberg-Marquardt algorithm and Black-Scholes, we see that D-M is significant, thus the null hypothesis of equal errors can be rejected at 5% level for all lags. Among networks, Levenberg-Marquardt algorithm approximates the price much better than conjugate gradient as in all previous tests, while we can reject the null hypothesis of equal errors at 5% significance levels. Thus Levenberg-Marquardt has significantly lower errors than conjugate gradient. It has significantly lower errors also when compared to Black-Scholes.

So let us have a look at an comparison of the out-of-sample period of the error functions of all three tested models.

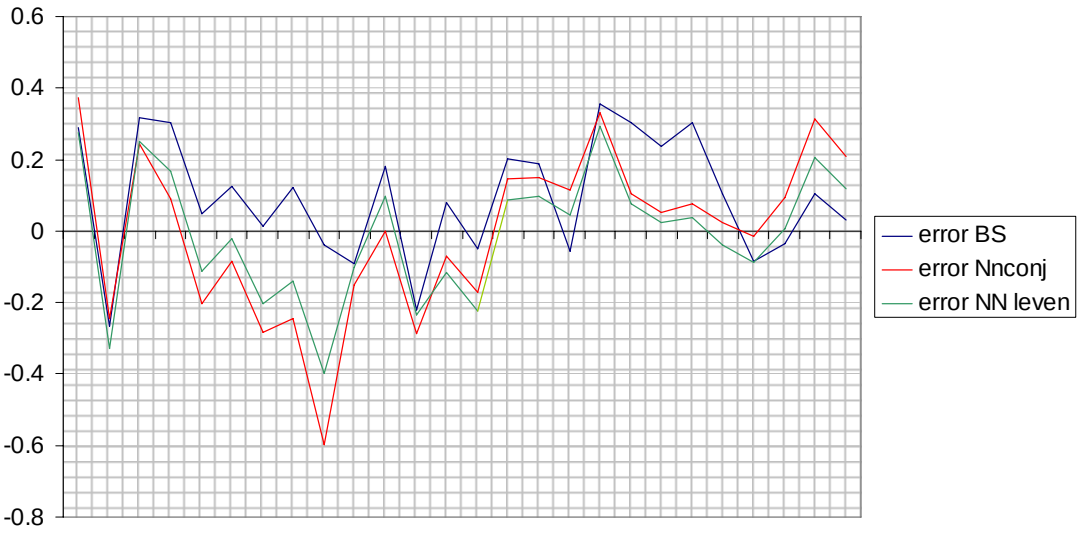


FIGURE 4.2: out-of-sample errors comparison

To conclude, previous results show that even if we relax the strict assumptions of Black-Scholes model, take only price of the underlying security and time value of it, we can estimate the optimal weights and use the obtained model to an option pricing better then Black-Scholes itself.

4.4 Concluding remarks

To conclude the chapter, we applied neural network to approximation of warrant price. Firstly we introduce briefly the Black-Scholes approach to the derivatives pricing and its main drawbacks of unrealistic assumptions. Then we show that neural network can learn the Black-Scholes formula very well.

Finally, we compare Black-Scholes pricing, Neural Network pricing and real price of the market on the out-of-sample testing. Neural networks clearly outperforms Black-Scholes pricing method. Simulated data training not only produces statistically lower error which has crucial implications for delta-hedging strategies, but we have to note once again that with neural network approach, we do not need to worry about volatility at all, nor about interest rates or log-normality of returns.

Even if the results are promising, we used only data for one warrant and one security, thus no generalization can be claimed here. But there is mounting evidence mainly on the more liquid options exchanges in USA in favor of our research which finds that neural networks can be use to pricing of the derivatives as an substitute if other analytical methods fail. The reason why we conducted this analysis was to show that neural networks are able to help to price derivatives on the emerging markets where the liquidity of underlying stock is lower if compared to developed markets, derivatives are not traded in the "home" country of the origin, it is traded in different currency so the exchange rate enters the formula, and most important of all – the liquidity of warrant is very low as Czech investors are not familiar with this forms of investing. Thus it seems very difficult to price such instrument and catch the behavior of market participants in these conditions.

Thus most important implication is that we showed that even such non-liquid derivatives can be priced even without considering and worrying about volatility problem, which is threatening investors most. We may consider to use also other inputs as general market volatility, market returns if correlated with the underlying security or others. We believe that if we can train the network to price the derivative with only the price of underlying asset and time value of the derivative, as we showed, few more inputs might help to explain the remaining part of the variance, hence this analysis set forward the research. The problem becomes also very actual at Czech Stock market as derivatives are to become traded at the market soon. Thus we hope that our research will help to move further in understanding the market processes.

Conclusion

In this thesis we present neural network approach and its application on the European stock market returns modelling. We show that there is no black box behind the networks, but robust mathematical model and we view the analysis as nonparametric econometric method. Thus we provided a link between theoretical approach of classical econometric with neural networks, and then use it to empirical test on the Czech, Hungarian and Polish returns from 1999:2006 period to see if the networks will help us in uncovering the returns process. We also present an optimization algorithms and statistical and economic tests for comparing the models.

After the theoretical background is set, we show that neural network can approximate any process on the Mackey-Glass chaotic time series. We use autoregressive model, ARMA (2,2) which fit the data best and the feedforward neural network with one layer and three neurons, logsigmoid function and Levenberg-Marquardt optimization. Neural network performed significantly better than other methods with out-of-sample NMSE 0.01, it explained 99% of the variance also when faced to an prediction task. Autoregression and ARMA (2,2) managed the out-of-sample NMSE at 0.162 and 0.132, while we strongly rejected the Diebold-Mariano's null hypothesis of equal errors when comparing the models. Thus neural networks uncovered the process very well with significantly lower errors, and we moved to the real-world empirical analysis of European Stock market returns.

Firstly we conduct the tests on the daily returns and we find that with use of neural networks we did not manage to get significantly lower prediction errors according to Diebold-Mariano test, but we gained some economic significance on PX50 and BUX markets when the direction predictions were significant at 5% resp. 10% significance levels and we were able to predict the next day direction with 56% and 54% probability of correct prediction respectively.

We left the daily series to conduct the same tests on the weekly ones. The in-sample adjusted R^2 of the neural network was impressive, while it explained 48% of PX50, 15% BUX, 28% WIG and 34%DAX variance using only lagged explanatory variables. When faced to out-of-sample forecasting, we were able to reject the null hypothesis of equal prediction errors between linear and neural models with PX50 and BUX series at 5% significance level. Thus Neural networks had significantly better forecasting error when testing the PX50 and BUX series. We also achieved better economical significance of the models, while being able to forecast the PX50, WIG and DAX with directional accuracy of 60%, 58% and 58% significant at 10% levels. Also implied transaction costs computed were higher than the real-world transaction costs, which tells us that the predictions are economically significant.

In the next part, we use the fact that tested markets are co-moving. We use Principal Component Analysis to find if the lagged returns of surrounding markets have significant influence on the tested market, or not. i.e. we test if the lagged returns of BUX, WIG and DAX can be used to explain the PX50 return. And we find that there are significant lags of surrounding markets for each of the tested markets. Then we use these results to model the stock market's return using the cross-country lags on the daily data. And we get similar results, when we could not reject the hypothesis of equal errors of linear and neural models for all series but DAX. Interestingly, neural networks perform significantly better only on daily DAX returns. We again gain economic significance on the PX50, BUX and DAX daily returns with neural networks. WIG daily returns predictions are again not economically significant.

To sum up the results of an application of neural network on Central European stock market returns, we would say that daily returns does not contain significant patterns, as neural networks could not approximate them. It managed to do significantly better in case of German DAX, which was basically picked as a benchmark of the large liquid European stock market. On the other hand, WIG seems to be completely unpredictable using just lagged historical returns. On the PX50, BUX and DAX markets, neural network predictions were economically significant. We managed to gain more predictive edge from weekly returns, while neural networks performed significantly better than linear modelling on PX50 and BUX prediction, and it provided us with economically significant predictions also with ability to predict direction with 60% probability.

Of course our findings have still very strong implication for the markets and traders, but still, it is of quite speculative usage. Even more, there are many problems of using these models in real trading. Main drawback is for example that

most of the models are behaving in the manner that they tend to predict the movement with some lag. This is fine if the markets are steady and the model captures the short-term trends well. But if there are unexpected exogenous moves or crushes of the stock market, the models very often fail to warn us. Much stronger implications of our findings can be made for another very interesting area – pricing of derivatives. We test the neural network on the European call warrant on CEZ, and we find that neural network is able to learn the Black-Scholes pricing model and can explain 99% of the variance of price. This is no surprise to us as the results are the same as from artificial Makey-Glass chaotic time series. Real test were actual market prices obtained from EUWAX market where the warrant is traded. We try to use neural network to approximate the price of the European call warrant on CEZ using only the price of CEZ and time. Thus we relax the constancy of volatility. We also price the warrant with Black-Scholes using the recomputed volatility on the daily basis, so we are more realistic in our analysis. On the out-of-sample results, we conclude that neural network is able to approximate CEZ call warrant price on 92%, while Black-Scholes only on 65%. Out-of-sample errors of network are also significantly lower then with Black-Scholes. Thus we conclude that neural networks may be used as an alternative for derivative pricing.

In these test we also compare conjugate gradient and Levenberg-Marquardt optimization methods. We reject the null hypothesis of equal prediction errors while Levenberg-Marquardt method produced significantly lower errors. This is also reason why we used it in all tests.

In this research we presented the models using only lagged historical data, and even if we could gain some predictive edge using neural network models, it is clear that further analysis needs to be done. Mainly usage of other variables affecting the price of stocks should be considered, as we see that the data does not explain themselves well.

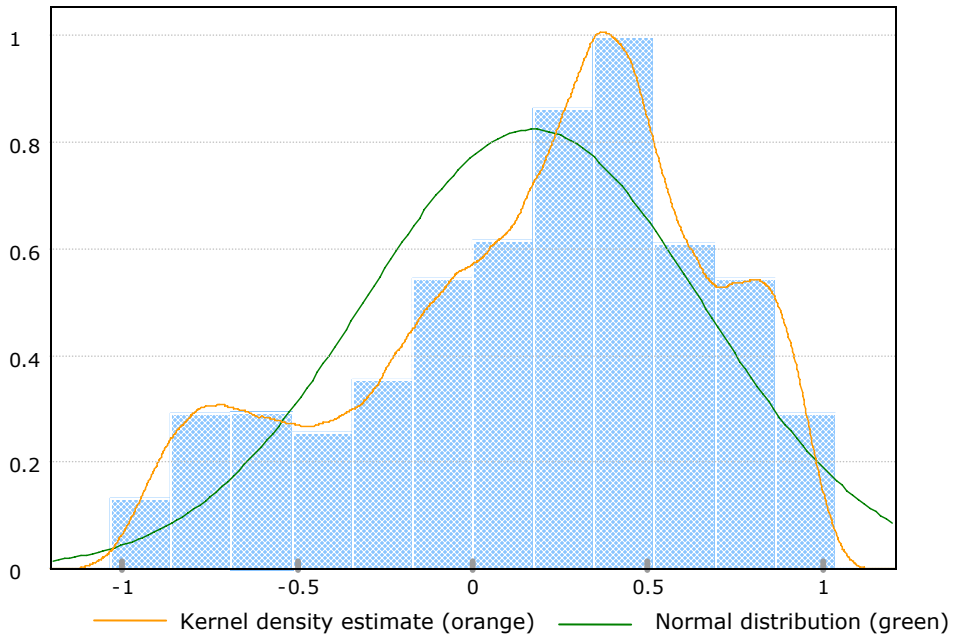
Appendix A: distribution of Mackey-Glass series

Table 15: Mackey-Glass chaotic time-series distribution:

Mean	Median	Maximum	Minimum	Std. Dev.	Skewness	Kurtosis
0.171735	0.263853	1.000000	-1.000000	0.483982	-0.523041	2.458327

Significant at the 1% level.

FIGURE A.1.: Histogram of a Mackey-Glass chaotic time-series distribution.



Appendix B: OLS Estimation results

for PX50, BUX, WIG and DAX models

Table 16: PX50 model

Variable	Coefficient	Std. Error	t-Statistic	Prob.
β_0	0.000834	0.000331	2.518340	0.0119
β_1	-0.066742	0.031630	-2.110083	0.0350
β_2	0.063869	0.027743	2.302173	0.0215
β_3	0.064417	0.027921	2.307141	0.0212
β_4	0.077691	0.024692	3.146457	0.0017
β_5	0.058844	0.018724	3.142688	0.0017

Table 17: BUX model

Variable	Coefficient	Std. Error	t-Statistic	Prob.
β_0	0.000766	0.000376	2.037470	0.0418
β_1	0.047583	0.028101	1.693296	0.0906
β_2	0.068265	0.027978	2.439997	0.0148
β_3	0.060791	0.021326	2.850566	0.0044
β_4	0.033276	0.021337	1.559500	0.1191

Table 18: WIG model

Variable	Coefficient	Std. Error	t-Statistic	Prob.
β_0	0.000554	0.000324	1.710250	0.0875
β_1	0.075760	0.032033	2.365101	0.0182
β_2	0.061295	0.027892	2.197569	0.0282
β_3	-0.063423	0.030753	-2.062354	0.0394
β_4	0.053653	0.027227	1.970563	0.0490
β_5	0.044133	0.019863	2.221846	0.0265
β_6	0.044053	0.018288	2.408821	0.0161

Table 19: DAX model

Variable	Coefficient	Std. Error	t-Statistic	Prob.
β_0	0.052203	0.028030	1.862370	0.0628
β_1	-0.110060	0.029929	-3.677415	0.0002
β_2	0.067533	0.041382	1.631945	0.1029
β_3	0.085239	0.045611	1.868815	0.0619
β_4	0.077619	0.036835	2.107219	0.0353

References

Aiken, M. and M. Bsat. (1999): Forecasting Market Trends with Neural Networks. Information Systems Management 16 (4), 42-48.

Anthony, M., Biggs, N.L. (1995): A computational learning theory view of economic forecasting with neural nets. In a References, editor, *Neural Networks in the Capital Markets.* John Wiley (1995)

Baltagi, Badi H. (2002): Econometrics. 3rd ed., Springer 2002, 401p, ISBN: 3-540-43501-8

Barkoulas, J. and Travlos, N. (1998): Chaos in an emerging capital market? The case of the Athens Stock Exchange, *Applied Financial Economics*, Vol.8, 231-243

Barro, R.J, (1990): The Stock Market and Investment, *Review of Financial Studies*, 3, 115-131.

Bellman, R. (1961): Adaptive Control Processes: *A Guided Tour*. Princeton, NJ: Princeton University Press.

Bishop, C. (1996): *Neural Networks for Pattern recognition*. Oxford University Press,1

Black, F and Scholes (1973): The Pricing of Options and Corporate Liabilities, *Journal of Political Economy,* 81, pp/ 637 – 659.

Bollerslev, T. (1986): Generalized Autoregressive Conditional Heteroskedasticity, *Journal of Econometrics* 31, pp.307-327

Bollerslev, T., Wooldridge, J.M. (1988): Quasi-Maximum Likelihood Estimation of Dynamic Models with Time-Varying Covariances. Working papers 505, *Massachusetts Institute of Technology (MIT)*, department of Economics.

Box, G. E. P., and Jenkins, G. (1976), *Time Series Analysis: Forecasting and Control*, Holden-Day.

Brent (1973): Algorithms for Minimization without Derivatives, Chapter 4. Prentice-Hall, Englewood Cliffs, HJ

Cambazoglu, B.B. (2003): Predicting the IMKB 30 Index, *Dept.of computer Engineering Bilkent University*, Ankara, working paper. http://www.smartquant.com/references/NeuralNetworks/neural4.ps

Campbell, J., Lo A.W. and A.C. MacKinlay (1997): *The Econometrics of Financial Markets*, Princeton University Press, Princeton, ISBN – 0-691-04301-9

Chen, H.F., Roll.R. and Ross, S.A. (1986): Economic Forces and the Stock Market, *Journal of Business*, 56, 383-403.

Dai, H. and Juan, Y. (1996): Convergence properties of the Fletcher-Reeves method, IMA J. Numer. Anal. 16:155--164.

Dayhoff, Judith E., and James M. DeLeo (2001): Artificial Neural Networks: Opening the Black Box. *Cancer* 91 : 1615-1635

Dickey, D.A., and W.A. Fuller (1979): Distribution of the Estimators for Autoregressive Time series With a Unit Root. *Journal of the American statistical association* 74: 427-431.

Diebold, F.X., and Roberto Mariano (1995): Comparing Predictive Accuracy, *Journal of Business and Economic Statistic*, 3: pp. 253-263

Engle, R. (1982): Autoregressive Conditional Heteroskedasticity with estimates of the Variance of United Kingdom Inflation, *Econometrica* 50, pp.987-1007.

Fama, E.F. (1965): The Behavior of Stock Market Prices, *Journal of Business*, 38, pp. 34-105.

Fama, E.F. (1970): Efficient Capital Markets: A Review of Theory and Empirical Work, *Journal of Finance,* XXV, No.2, pp. 383-417.

Fama, E. and French, K. (1988): Dividend Yields and Expected Stock Returns, *Journal of Financial Econometrics*, 19, pp.3-29.

Fama, E. and French, K. (1989): Business Conditions and Expected Returns on Stocks and Bonds, *Journal of Financial Econometrics,* 25, 23-49.

Fama, E. and French, K. (1988): Stock Returns, Expected Returns, and Real Activity, *Journal of Finance*, 45, pp.1089-1108.

Filacek, J.Kaplička, M.Vošvrda (1998), Testování hypotézy efektivního trhu na BCPP (in czech), *Journal of Finance*

Granger, Clive W.J., and Yongil Jeon (2002): Thick Modelling. Unpublished Manuscript, Department of Economics, University of California, San Diego, *Economic Modelling*, forthcoming

Greene, W.H. (1993): Econometric Analysis, Macmillam Press, New York, ISBN 0-131-10849-2

Grossman, S.J., Stiglitz, J.E. (1980): On the Impossibility of Informationally Efficient Markets, *American Economic Review*, Vol.70, No.3, pp.393-408

Hamilton, J.D. (1994): *Time Series Analysis*, Princeton University Press, Princeton, ISBN 0-691-04289-6

Hartl, R.F. (1990): *A Global Convergence Proof for a Class of Genetic Algorithms*, Working paper, Vienna University of Technology, Institute of Econometrics

Hawanini, G. and Keim, D.B. (1993): On the predictability of common stock returns: World-wide evidence, *Handbook of Finance*

Hellstrom, T. and Holmstrom, K. (1998): Predicting Stock Market. Technical Report, *IMa-TOM-1997-09*, Center of Mathematical Modelling, Mälardalen University

Henriksson, R.D. and R.C. Merton (1981): On Market Timing and Investment Performance. II.Statistical Procedures for Evaluating Forecasting Skills, *Journal of Business*, 54, pp. 513-533.

Hertz, Krogh, and Palmer (1991): *Introduction to the Theory of Neural Computation*. Addison-Wesley, ISBN 0-201-51560-1.

Hestenes, Magnus, R and Stiefel, E. (1952): Methods of conjugate gradients for solving linear systems, *J. Research Nat. Bur. Standards* 49, 409–436.

Hornik, K. Stinchcombe, M., White, H.: (1989): Multifactor feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366

Hsieh, D.A. (1991): Chaos and Nonlinear Dynamics: Application to Financial Markets, *journal of Finance,* Vol 46, No.5, pp. 1839-1877

Hutchinson, J.M., A.W. Lo and T. Poggio (1994), "*A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks*", The Journal of Finance, Vol. 49, No. 3, pp851-889.

Jarque, C.M., and A.K.Bera (1980): Efficient Tests for Normality, Homoskedasticity, and Serial Independence of Regression Residuals. *Economics Letters* 6:255-259

Kuan, Chung-Ming, Halbert White (2004): "Artificial Neural Networks: An Econometric Perspective," *Econometric Reviews* 13, pp. 1-91

Leroy, S.F. (1973) Risk aversion and the Martingale property of Stock Prices, *International Economic Review*, Vol 14, No. 2, pp. 436-446

Levenberg, K. (1944): A Method for the Solution of Certain Problems in Least Squares. Quart. Appl. Math. 2, 164-168.

Lo, A.W. and A.C. MacKinlay (1988): Stock Prices Do Not Follow Random Walk: Evidence From a Simple Specification Test, *Review of Finanacial Studies*, 1, pp.41-66.

Mackey, M. and Glass, L. (1977): Oscillations and chaos in physiological control systems *Science*, pp. 197-287

Malkiel, B.G. (1996): *Efficient Market Hypothesis*, Macmillan, London, 1987.

Marquardt, D. (1963): An Algorithm for Least-Squares Estimation of Nonlinear Parameters. SIAM J. Appl. Math. 11, 431-441.

McCulloch, W.S. and Pitts, W.H. (1943): A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5 pp.115-133

McNelis, P.D. (2005): *Neural Networks in Finance: Gaining predictive edge in the market*, Elsevier Academic Press advanced finance series, ISBN 0-12-485967-4

Merton, R.C. (1973): Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4 (1), 141-183

Mitchell, T.M. (1997): *Machine Learning*, McGraw-Hill, 414p. ISBN 0-07-042807-7

Mohan, N., Jha P., Laha A.K., and Dutta G. (2005): Artificial Neural Network Models for Forecasting Stock Price Index in Bombay Stock Exchange, *IIMA Working Papers 2005-10-01, Indian Institute of Management Ahmedabad*.

Nygren, K. (2004): Stock Prediction – A Neural Network Approach, Master's Thesis, Royal Institute of Technology, KTH, Sweden, supervised by prof. Holmstrom

Patton, A.J., a and A.Timmermann (2004): Properties of Optimal forecasts under Asymetric Loss and Nonlinearity, working paper, Financial Markets Group, London School of Economics.

Patton, A.J., a and A.Timmermann (2006): Testing Forecast Optimality under Unknown Loss, working paper, Financial Markets Group, London School of Economics. http://management.ucsd.edu/pdf/timmermann12.pdf

Pesaran, M.H., and A.Timmermann (1992): A Simple Nonparametric Test of Predictive Performance", *Journal of Business and Economic Statuistics* 10: pp. 461-465.

Peters, E. (1949): *Fractal Market Analysis: Applying Chaos Theory to Investment and Economics*, ISBN: 0-471-58524-6

Poggio, T. and F. Girosi (1990): *Networks for Approximation and Learning*. Proc. of The IEEE, vol.78, No.9, pp. 1481-1497.

Polak, E. (1971): Computational Methods in Optimization, New York, Academic Press

Roberts, H. (1967): Statistical versus Clinical Prediction of the Stock Market, *unpublished manuscript, CRSP*, University of Chicago, May 1967.

Sarle, W.S. (1998), "Prediction with Missing Inputs," in Wang, P.P. (ed.), JCIS '98 Proceedings, Vol II, Research Triangle Park, NC, 399-402, ftp://ftp.sas.com/pub/neural/JCIS98.ps.

Schraudolph, N. and Cummins, F. (2002): Introduction to neural networks. Course notes, IDMSIA, Lugano, Italy, downloaded from www.icos.ethz.ch/teaching/NNcourse/intro.html

Schwarz, G. (1978): Estimating the Dimension of a Model. *Annals of Statistics* 6: 461-646.

Shanno, David F. (1978): Conjugate Gradient Methods with inexact searches, *Mathematics of Operations Research*, Vol.3, no.3

Silverman, B. W. (1986). Density Estimation for Statistics and Data Analysis, Chapman & Hall.

White, H. (1988): Economic prediction using neural networks: The case of IBM daily stock returns, *IEEE International Conference on Neural Networks, San Diego,* pp. 451-459.

Yao, J.T., Tan, C. L. and Poh H.L. (1999): *Neural Networks for Technical Analysis: A Study on KLCI*, International Journal of Theoretical and Applied Finance, Vol. 2, No.2, 1999, pp221-241.

Žikeš F. (2003): *The Predictability of Asset Returns: An empirical Analysis of Central-European Stock Markets*, diploma thesis, Institute of Economic Studies, FSV UK, Prague 2003, supervised by Doc. Ing. M.Vošvrda, CSc.