Nonlinear Forecasting Using a Large Number of Predictors

(Job Market Paper)

Ali Habibnia[*]

ABSTRACT

This paper considers forecasting in a big data environment. We develop a category of nonlinear forecasting based on factor models to benefit from many potential predictors while accounting for any possible nonlinear dynamics within the environment. The problem of forecasting with factor models is a two-step procedure. Proposed model, at the first step, employs an autoassociative neural network to estimate nonlinear factors from a large panel of predictors, and at the second step, applies a nonlinear function on the estimated factors to predict a single time series. Such features can go beyond the covariance structure analysis and enhance the accuracy of forecasting. Applying this approach to forecast equity returns, the proposed model captures the nonlinear dynamic between equities to enhance the performance of the subsequent forecast. This offers a significant improvement to current univariate and multivariate models. We emphasize the fact that linear models can be seen as a special case of the proposed nonlinear model, which basically implies that in the event that nonlinearity is absent between series, the model will subsequently be reduced to a linear model. The empirical results on daily returns of equities on the S&P 500 index from 2005 - 2014 proved the superiority of the out-of-sample forecasting ability of this model vis-à-vis competing approaches.

Key Words: Forecasting; Nonlinear Factor Models; Neural Networks; Nonlinear PCA.
JEL classification: C38, C45, C53, G10, G17.

[*]Department of Statistics, London School of Economics. E-mail: a.habibnia@lse.ac.uk

I. Introduction

World data currently doubles every couple of years with an on-going steady increase in computing power that poses new challenges for economic modelling and forecasting in a big data environment. It challenges state-of-the-art data acquisition, computation and analysis methods. To benefit from many new potential explanatory variables, dimension reduction methods (i.e, Factor models - Stock and Watson (2002a, 2006); Bai and Ng (2002); Deistler and Hamman (2005); Forni, Hallin, Lippi, and Reichlin (2005); Lam and Yao (2012)), shrinkage techniques (Ridge - Hoerl and Kennard (1970); LASSO - Tibshirani (1996); Elastic Net - Zou and Hastie (2005)), and subset selection techniques (Bayesian regression - Mol, Giannone, and Reichlin (2008); Selecting variables - Bai and Ng (2008a)) are used to handle high-dimensional data but they are mostly linear models. Since the linear models cannot explain a number of important features common to much economic and financial data, it is logical to think about a nonlinear statistical model to concurrently handle high-dimensionality and nonlinearity. Only a few attempts considering nonlinear dynamics in high-dimensional setting exist. Bai and Ng (2008b), and Raviv and Dijk (2014), for instance, included the quadratic principal components PCs and the first level cross-products of the original variables to capture nonlinearities. Their model is nonlinear in the variables, but it is linear in the parameters. Meanwhile, Exterkate, Groenen, Heij, and Dijk (2013) applied kernel methods to a ridge regression to introduce a nonlinear ridge regression. Giovannetti (2011) improved the factor model by running a nonlinear regression on linear PCs.

With the rise of big data and the real opportunities that machine learning now brings, there is no better time to find out how novel techniques can be used for economic research. To this end, this paper aims at addressing high-dimensional econometric models and machine learning techniques to analyse and forecast financial big data. Through this study we introduce a nonlinear statistical factor model based on neural networks (NLFM hereafter).

In theory, the optimal forecast of a variable under quadratic loss is its expectation conditional on information available. In practice, the relevant information set might be very large and a factor-based forecast can systematically handle this information while keeping the dimension of the forecasting model small. Forecasting using factor models has recently received much attention, especially in the macroeconomic literature. However the potential of factor-based forecasts can also be investigated in the realm of finance. Several papers have been devoted to forecasting macroeconomic and financial time series with factor models in a data-rich environment where, in general, the idea is to forecast the series of interest by using the common factors estimated from a large panel of predictors, see Stock and Watson (2002b, 2006); Forni et al. (2005); Deistler and Hamman (2005),and Boivin and Ng (2005) for a survey of the 'factor approach methods' to forecasting.

In particular, the problem of forecasting with factor models can be divided into two

steps: (1) The factor estimation step from a large panel of data and the forecasting step that uses the factor estimates to forecast the series of interest. Factors estimation and forecasting equations can be obtained in different ways. For instance, Kalman filter methods and cross-sectional averaging methods as principal components analysis (PCA) are two kind of estimates of the factors. Concerning forecasting with factor models, several approaches have been proposed based on how the common factors are estimated and how the forecast equations are formulated. There are two leading approaches in the literature that differ primarily because of the methodology used to estimate the factors. The static method, suggested by Stock and Watson (2002a) (SW hereafter), and the dynamic method of Forni et al. (2005) (FHLR hereafter). SW performed a forecasting experiment for macroeconomic variables using factors estimated by PCA from a large panel of U.S. monthly predictors. In finance literature, Deistler and Hamman (2005) applied a static factor model (to be more precise, Quasi-static principal component analysis, quasi-static factor models with idiosyncratic errors) when forecasting returns on asset prices. FHLR performed a forecasting experiment based on spectral analysis (a dynamic principal components), and then applied this approach to forecast macroeconomic indicators using a large panel of Euro-area monthly predictors. Moreover, Forni, Hallin, Lippi, and Zaffaroni (2014) recently introduced a method that complements FHLR by assuming an infinite-dimensional factor space. Various papers found substantial forecasting improvements based on the mean squared errors of factor models and those obtained from simple autoregressions and more elaborate structural models, see Stock and Watson (1999); Marcellino, Stock, and Watson (2003), Forni, Hallin, Lippi, and Reichlin (2001), and Forni and Reichlin (2001).

This study introduces a nonlinear financial forecasting methodology based on two concepts: factor models and neural networks. As we believe that understanding the co-movement between financial returns is crucial for forecasting procedure, we use factor models to describe the covariance structure of financial return and produce a parsimonious representation of the market correlation and demonstrate that a small number of common components accounts for a large proportion of the variability of the equities that we consider. It is worth noting that the nonlinear factor-based forecast model that we propose is based on static factor models, however the proposed model can be extended to a dynamic setting when we deal with macroeconomic forecasting.

The rest of the paper is organized as follows. Section II provides the basic framework of factor models, the application of factor models in forecasting financial returns, an overview of nonlinear dimensionality reduction and a review of neural networks from a statistical perspective. In Section III we introduce a nonlinear factor model concerning a nonlinear factor estimation and nonlinear forecast equation. Section IV summarizes our empirical findings, concerning data, a quick review of linearity tests in and between series, competing models, tools for validation of the forecasts and trading (portfolio) simulation. Finally, our conclusion are presented in Section V.

II. Preliminaries

In this paper we answer the question of whether it is possible to forecast with a large panel of predictors, while considering nonlinear dynamics in data and to show how a model with such features can improve financial forecast accuracy. We propose a forecasting model that is able to handle high-dimensionality and nonlinearity by applying a neural network based principal component analysis to estimate common factors from a large panel of data and nonlinearly forecast the series of interest uses the factor estimates with a feedforward neural network. Such features can go beyond the covariance structure analysis. In the following section we initially review the basic framework of factor models and the dimensionality reduction and the neural network model, we then explain our proposed model.

A. *Forecasting financial return series with factor models*

Research into modelling and forecasting a financial time series has a long history. Several models are covered in Tsay (2005) and Campbell, Lo, and MacKinlay (1996) that attempt to explain return time series using linear combinations of one or more financial market factors. The most widely studied single factor models is the capital asset pricing model (CAPM) of Sharpe (1964) and Lintner (1965) that relates the expected return of equities to the expected rate of return on a market index such as the Standard and Poor's 500 Index. Although CAPM is attractive for its simple logic and intuitively pleasing predictions about risk and expected return, the empirical record of the model is poor and it can not explain the behaviour of asset returns, see Fama and French (2004). The empirical failure of the CAPM is perhaps because of its simplicity which implies that a model with multiple factors might be required to fully explain return time series. Arbitrage pricing theory (APT) was one general model developed by Ross (1976) to shore up these deficiencies. APT presents a linear approximate model of expected asset returns using an unknown number of unidentified macroeconomic factors or theoretical market indices. The problem then becomes one of identifying the factors. Factors may be classified in terms of identification into either theoretical or statistical. The theoretical models explain asset returns based on specifying observable macro-economic and financial variables (e.g., Chen, Roll, and Ross (1986)) or observable defined firm-specific variables (e.g., Tsay (2005)) as factors. The relationships between the factors and the return time series are determined using linear regression. The two most frequently used statistical methods, that extract the unobservable or latent factors directly from the portfolios of return time series, are factor analysis and PCA, see Tsay (2005).

Analysis of multiple financial returns often requires high-dimensional statistical models. In practice, the return time series present similar characteristics and share the following stylised facts: comovements, nonlinearity, non-gausianity (skewness and heavy tails), volatility clustering and leverage effect, which makes the modelling of this variable hard, see Hsieh (1991); Bollerslev, Engle, and Nelson (1994); Brooks (1996); Cont (2001).

Hence, factor models have been proposed in the literature to study common patterns of return series. In statistical factor models, variables are represented as the sum of two mutually orthogonal unobservable components: the common components and the idiosyncratic component. This paper focuses on statistical factor models to predict financial return series.

A.1. Linear Factor model general specification

Given a high-dimensional matrix of stationary time series of financial returns, which we denote by x_{it} $(i = 1, ..., m, t = 1, ..., T)$ the factor model enables us to disassemble x_{it} into the few common factors representing the market or the co-movements of the returns and idiosyncratic components representing special features of a given company. x_{it} consists of the m series of financial returns and T time series observations for each company. In terms of matrix notation, let $\boldsymbol{X}$ be the $m \times T$ matrix of observed return series; $\boldsymbol{x}_t$ is a row vector denoting all m observations at time t, while $\boldsymbol{x}_i$ is a column vector denoting all T observations for i^{th} company.

The forecast variable of interest is $\boldsymbol{x}_{iT+h|T}$, the h-step-ahead out-of-sample forecast of a return series in the panel. It is also assumed that the series hase a zero mean and covariance $\boldsymbol{\Gamma}_X(0)$. In a high-dimensional setting m usually is much greater than T and the factor model reduces the dimension to make the estimation feasible. The idea is that a small number q of factors should be able to explain most of the covariance of the data. A factor model also describe the covariance structure of returns and deconstructs risk and return into systemic and systematic components.

A number of different formulations of factor models are imaginable based on how data is explained by common factors and the idiosyncratic component. If the factors (shown by $\boldsymbol{u}_t$ hereafter) has only a contemporaneous effect on $\boldsymbol{x}_t$ and an idiosyncratic component (shown by $\boldsymbol{\xi}_t$ hereafter) has no cross–sectional dependence, the setting is known as the exact static model; see Spearman (1904); Maxwell and Lawley (1971). This is estimated by

$$\boldsymbol{x}_t = \boldsymbol{\lambda}_i' \boldsymbol{u}_t + \boldsymbol{\xi}_t, \tag{II.1}$$

where $\boldsymbol{u}_t$ is a vector of q common factors, $\boldsymbol{\lambda}_i$ is the corresponding vector of loadings for unit i, and ξ_{it} is an idiosyncratic component. In classical factor models, ξ_{it} is serially uncorrelated and iid across i. While in the extension to a dynamic setting, the model allows factors and the idiosyncratic component to be serially correlated. For an exact dynamic model see Sargent and Sims (1977) and Geweke (1977). The dynamic factor model representation is given as:

$$\boldsymbol{x}_t = \boldsymbol{b}_i'(L) \boldsymbol{u}_t + \boldsymbol{\xi}_t, \tag{II.2}$$

where $\boldsymbol{u}_t$ is a vector of dynamic factors. In these two last equations, factors are a q-

dimensional vector ($q \ll m$) and $\boldsymbol{\lambda}$ or $\boldsymbol{B}(L)$ are loadings matrices of size $m \times q$. Hereafter, we will refer to $\boldsymbol{x}_t - \boldsymbol{\xi}_t$ as the common component $\boldsymbol{\chi}_t$ in both static and dynamic cases.

Chamberlain and Rothschild (1983) introduced an approximate (or generalized) static factor model allowing weak cross–sectional dependence on the idiosyncratic component $\boldsymbol{\xi}_t$ and for the generalised dynamic factor models (GDFM) see Forni and Lippi (2001); Forni, Hallin, Lippi, and Reichlin (2000). It is also worth noting that $\boldsymbol{u}_t$ and/or $\boldsymbol{\xi}_t$ must be serially correlated to guarantee the predictability of $\boldsymbol{x}_t$.

This work focuses on a static method of estimating factors as financial data does not present a dynamic relationship between factors and financial series and an unnecessary estimate of the dynamic factor could induce some efficiency loss. The following section starts with an overview of linear and nonlinear dimensionality reduction methods emphasizing PCA and nonlinear PCA as the factor estimation step.

B. *Overview of Nonlinear Dimensionality Reduction*

A variety of dimensionality reduction methods exist, with a variety of motivations in many fields including statistics, machine learning, and applied fields for over a century. In general, dimensionality reduction methods can be used in discovering low-dimensional structure from high-dimensional and noisy observations that extracts some meaningful features of interest in the data. These methods can be interpreted as a simple optimization framework that can be broadly classified into: linear, nonlinear and proximity (graph) based methods.

In the first class we have principal component analysis PCA (Pearson (1901); Eckart and Young (1936)), factor analysis FA (Spearman (1904)), canonical correlations analysis CCA (Hotelling (1936)), multidimensional scaling (Torgerson (1952); Cox and Cox (2001); Borg and Groenen (2005)), distance metric learning (Kulis (2012); Liu and Jin (2006)), linear discriminant analysis (Fisher (1936); Rao (1948)), and several others. Recently, Cunningham and Ghahramani (2015) surveyed the literature on linear dimensionality reduction in their work and gave insights to some rarely discussed shortcomings of traditional approaches. Linear methods are based on a linear projection assuming that the data lives close to a lower dimensional linear subspace. If the data lies on or near a low-dimensional nonlinear manifold, then linear methods, even though computationally efficient, can not model the variability of the data correctly and recover this intrinsic nonlinear structure. However, the basic geometric intuitions behind linear methods play an important role in many algorithms for nonlinear dimensionality reduction.

The algorithms designed to address the problem of nonlinear dimensionality reduction are for instance, Kernel methods (Schölkopf, Smola, and Müller (1997, 1998) ; Jäkel, Schölkopf, and Wichmann (2007)), Neural network methods (Oja (1982); Kramer (1991); Hsieh (2004); Hinton and Salakhutdinov (2006); Kohonen (2001)). Modern methods

for optimization enable a generic nonlinear dimensionality reduction solver, which accepts as input data and an objective that is to be optimised, and returns, as output, an optimal low-dimensional projection of the data. Graph based methods, like the Locally Linear Embedding (Roweis and Saul (2000)), Isomap (Tenenbaum, vin de Silva, and john Langford (2000)), Maximum Variance Unfolding (Weinberger and Saul (2006)), Laplacian Eigenmaps (Belkin and Niyogi (2003)) and several related methods are powerful and nonlinear, but their computational cost scales quadratically with the number of observations, so they generally cannot be applied to very large high-dimensional data sets.

The diversity of dimensionality reduction methods makes a direct intercomparison between different methods very difficult. There are very few studies comparing the performance of different methods on the same problem. See Fodor (2002), Burges (2004) and Cayton (2005) for a review of the most common geometrical and statistical methods of dimensionality reduction. This section treats in detail only Linear and Nonlinear PCA methods for the financial data analysis. We will begin by reviewing the notion of PCA and the extension of ideas from PCA to a nonlinear setting. Then we describe the neural networks and its application in nonlinear PCA.

B.1. Extension of ideas from PCA to a nonlinear setting

Since PCA is perhaps the most popular instance of dimensionality reduction methods and it has been discussed extensively by other authors, only a brief summary is given here. PCA (also known as empirical orthogonal function (EOF), proper orthogonal decomposition (POD) and the Karhunen-Loève decomposition) in a nutshell is a technique for linearly mapping multidimensional data into lower dimensional space while preserving as much information as possible. PCA also transfer a set of correlated variables into a new set of uncorrelated variables.

Given a m-dimensional random vector of the form $\mathbf{x}_t = (x_1, ..., x_m)$, where consists of correlated data points and each variable $\mathbf{x}_i$ $(i = 1, ..., m)$ has T samples labeled by the index t, PCA is an optimal matrix factorization of $\mathbf{x}_t$ into two vectors, $\mathbf{u}_t$ called the scores, a q-dimensional orthonormal principal components $(q \ll m)$, and $\boldsymbol{\lambda}$, called the loading matrix of size $m \times q$, plus a matrix of residuals $\boldsymbol{\xi}_t$ when t is simply the time, and each $\mathbf{x}_i$ is a time series containing T observations.

$$\mathbf{x}_t = \lambda_i' \mathbf{u}_t + \xi_t \tag{II.3}$$

PCA searchs for $\mathbf{u}$, a linear combination of the $\mathbf{x}_i$, and an associated vector $\boldsymbol{\lambda}$. The condition of optimality on the factorization is that the Euc1idean norm of the residual matrix, $\|\xi_t\|^2 = \langle \|\mathbf{x}_t - \lambda_i' \mathbf{u}_t\|^2 \rangle$, must be minimized for the given number of factors. One can easily show that the subspace with minimum reconstruction error is also the subspace with maximum variance. The basis vectors of this subspace are given by the top q eigenvectors of the $m \times m$ covariance matrix.Therefore, to satisfy this criterion, it is known that

the columns of $\boldsymbol{\lambda}$ are the eigenvectors Corresponding to the q largest eigenvalues of the covariance matrix of $\mathbf{x}_t$. It is also useful to view PCA as a linear projection of data from $\mathbb{R}^m$ to $\mathbb{R}^q$. PCA finds the projection such that the best linear reconstruction of the data is as close as possible to the original data. Taking $\boldsymbol{\lambda}'\boldsymbol{\lambda} = \boldsymbol{I}$ without loss of generality, the mapping has the form:

$$\mathbf{u}_t = \mathbf{x}_t \lambda_i \tag{II.4}$$

The loadings $\boldsymbol{\lambda}$ are the coefficients for the linear transformation. The information lost in this mapping can be assessed by reconstruction of the measurement vector by reversing the projection back to $\mathbb{R}^m$:

$$\tilde{x}_t = \mathbf{u}_t \boldsymbol{\lambda}' \tag{II.5}$$

where $\tilde{x}_t = \boldsymbol{x}_t - \boldsymbol{\xi}_t$ is the reconstructed measurement vector. In PCA, input data are projected into the q-dimensional subspace that minimizes the reconstruction error. For background and more details on PCA, see Jolliffe (2016). However, PCA provides optimally parsimonious data compression for any dataset whose distribution lies along orthogonal axes, the common drawback of this methods is that only linear structures can be correctly extracted from the data and nonlinear relations are either missed or misinterpreted by this methods and reduction of dimensions for complex distributions may need non linear processing. In nonlinear PCA, the mapping into feature space is generalized to all arbitrary nonlinear continuous function expressible by a feedforward neural network. By analogy to Eq.II.4, we seek a mapping in the form:

linear relation in PCA is now nonlinearly generalized and can be any nonlinear continuous function representable by a feedforward neural network

$$\mathbf{u}_t = \boldsymbol{\phi}^{(x)}(\mathbf{x}_t) \tag{II.6}$$

where $\boldsymbol{\phi}^{(x)}$ is a nonlinear vector function, composed of q individual nonlinear functions analogous to the columns of λ, such that if u_i represents the ith element of $\boldsymbol{u}_t$,

$$u_i = \phi_i^{(\boldsymbol{x})}(\mathbf{x}_t) \tag{II.7}$$

The inverse transformation, restoring the original dimensionality of the data, analogous to Eq.II.5, is implemented by a second nonlinear vector function $\boldsymbol{\phi}^{(u)}$

$$\boldsymbol{x}_t = \boldsymbol{\phi}^{(u)}(\mathbf{u}_t) \tag{II.8}$$

$\boldsymbol{\phi}^{(u)}$ nonlinearly generates a continuous open curve in the $\boldsymbol{x}_t$ space and a q-dimensional approximation of the original data. The loss of information is again measured by $\boldsymbol{\xi}_t = \boldsymbol{x}_t - \tilde{x}_t$, and analogous to PCA, the functions $\boldsymbol{\phi}^{(x)}$ and $\boldsymbol{\phi}^{(u)}$ are selected to minimise

$||\boldsymbol{\xi}||$. If $\phi^{(x)}$ and $\phi^{(u)}$ are linear, the optimal solution is PCA and if these functions are not linear then we are basically doing nonlinear PCA. In neural network PCA, both mapping (also known as reduction or extraction) and demaping (reconstruction or generation) functions are both approximated by neural networks. PCA is closely related to a particular form of neural network, an autoassociative network method of Kramer (1991) and Scholz, Fraunholz, and Selbig (2008), which is a neural network whose outputs are its own inputs and the goal is to minimise reconstruction error. An autoassociative neural network has a bottleneck network architecture and consists of a "reduction" network, representing the function $\phi^{(x)}$, mapping from m inputs to q outputs, connected directly to a "reconstruction" network, representing the function $\phi^{(u)}$, mapping from q inputs to m outputs.

There is also another class of methods that use artificial neural networks, self organizing maps - SOM (Kohonen (2001)), as a means of high-dimensional data projection to a lower-dimensional discrete representation, preserving the locality between data vectors in the original high-dimensional space. However SOM had empirical success especially in one-dimensional cases (see Fort (2006)), the theoretical justification behind the SOM approach is weak. So this paper focuses on the autoassociative networks. The aim of next subsections is to cover the key concepts of the methods that use multilayer perceptron neural networks as function approximators (particularly feedforward networks for time series analysis), and neural network PCA method.

C. *Neural Networks for Time Series Analysis*

When performing time series analysis we would like to characterise how the value of a target variable changes as some predictors are varied. However linear models are adequate to explain many phenomena in the world, most important economic and financial phenomena are complex and nonlinear in nature. In order to explain nonlinear phenomena, different parametric and nonparametric nonlinear regression models have been developed so far.

Parametric nonlinear regression models attempt to characterise the relationship between predictors and response with parametric nonlinear functions. The parameters can take the form of a polynomial, exponential, trigonometric, power, or any other nonlinear function. In other words, in parametric nonlinear models the shape of the functional relationships between the response and the predictors are predetermined. In many situations, that relationship is unknown and nonparametric nonlinear regression models should be used.

In nonparametric models, the shape of the functional relationships between variables can be adjusted to capture unusual or unexpected features of the data. The main types of nonparametric regression models are kernel-based methods, tree-based regression models and artificial neural networks.

Kernel-based methods can be viewed as a nonlinear mapping from inputs into higher dimensional feature space in the hope that the data will be linearly separable or better structured. It measures distances between observations, then predicts new values based on these distances. Best known example are support vector machines (SVMs), introduced by Vapnik (Vapnik and Chervonenkis (1964, 1974); Vapnik (1982, 1995)), which provide a structured way to use a linear algorithm in a transformed feature space. The key advantage this so-called kernel trick brings is that nonlinear patterns can be found at a reasonable computational cost. Perhaps the biggest limitation of the kernel-based methods lies in choice of the kernel and tuning model parameters.

Tree-based regression models are alternative (nonparametric and nonlinear) approaches to regression that are not based on assumptions of normality and user-specified model statements.These models originated in the 1960s with the development of AID (Automatic Interaction Detection) by Morgan and Sonquist (1963). In the 1980s, statisticians Breiman, Friedman, and Stone (1984) developed CART(Classification And Regression Trees). The fundamental idea is to recursively partition the regressors' space in regions (build a tree) until all the subspaces are sufficiently homogeneous in order to estimate the regression function with the sample average (or the specific local model employed) in each region.

Another class of nonlinear models that we focus on in this paper are neural networks. These are flexible function forms motivated by the way the brain processes information. neural networks consist of a cascade of simple computational units called neurons, which are highly interconnected. Depending on how they are constructed, neural nets can approximate functions that are generally unknown. However Neural networks can use a variety of topologies, based on the universal approximation theorem, a single hidden layer feedforward network architecture with finite number of neurons can approximate arbitrary well any nonlinear function (poofs have been given by Cybenko (1989), Hornik, Stinchcombe, and White (1989), White (1990) and Hornik (1991)). For those inetresred in neural network models, see Bishop and Hinton (1995), Friedman, Hastie, and Tibshirani (2009), Teräsvirta, Tjostheim, and Granger (2010). Also the following papers provide readers with a statistical approach to neural networks (Varian (2014); Teräsvirta, van Dijk, and Medeiros (2005); Kuan and White (1994)).

A neural network is an interesting area of machine learning. It is simultaneously one of the oldest and one of the newest areas. The work on neural networks goes back to the 1940s when researchers tried to build models of the brain. Perceptron, which is an extremely simplified computational model of a biological neuron and a very simple precursor of linear models, goes back to the 50s and people showed amazing performance of the perceptron on a number of problems. Perceptron of course was limited in what it could do, so later on research related to the neural network basically died. It was reborn in the 1980s when researchers figured out how to put multiple perceptrons together into a network and they learnt how network weights with the Backpropagation algorithm worked.

Again there was a great deal of excitements because these models finally seemed to be able to solve all kinds of learning problems. At the same time more powerful regression models came along, support vector regressions, so neural networks again fell out of fashion and were shelved. They returned for a second time recently when people finally figured out how to train them reasonable quickly on a massive scale and a big part of that is due to the changes in hardware that have occurred since the 1980s. It is also worth mentioning that there has been a resurgence in the field of artificial neural networks in recent years, known as "Deep neural networks". Deep neural networks use multiple stages of nonlinear computation and have won numerous contests on an array of complex tasks ranging from pattern recognition and machine learning.

In practice, a neural network is an interlinked collection of neurons that the output of some neurons can become inputs to other neurons. A neural network implements a function $y = \Phi(x; \theta)$; the 'output' of the network, y, is a nonlinear function of the 'inputs' x; this function is parametrised by weights and the bias is $\theta = \{w, \beta\}$.

To describe neural networks, we will begin by describing the simplest possible neural network, which is a computational model of a single neuron (known as perceptron). A perceptron follows the "feedforward" model, meaning it takes a set of observed inputs $x_{it} = (x_{1t}, x_{2t}, ..., x_{NT})$, multiplies each of them by their own associated weight $w_i = (w_1, w_2, ..., w_N)$, and sums up the weighted values and also adds a bias β (always +1) to form a pre-activation z. The network then transforms the pre-activation using a nonlinear activation function $\phi(z)$ (e.g. logistic sigmoid or tanh) to output a final activation y_t. So, a single neuron can be formulated as follows:

$$z = \beta + \sum_{i}^{N} x_{it} w_i, \tag{II.9a}$$

$$y_t = \phi(z) + \varepsilon_t. \tag{II.9b}$$

Single neuron networks with an identity activation function or even a sigmoid function implement linear models, which does not really help us if we want to model nonlinear phenomena. The sigmoid function is almost linear near the mean and has smooth nonlinearity at both extremes. However, by considering the single neuron network to be a basic building block, we can construct more complicated networks, ones that perform powerful nonlinear computations. Instead of a single neuron, we introduce a set of neuron networks. This set of intermediate networks is often referred to as a "hidden" layer, as it does not directly observe input or directly compute the output. By using a hidden layer, we form a multilayered network. A multilayered network with only one hidden layer has two sets of weights: those connecting the inputs to the hidden layer (w_{ij}), and those connecting the output of the hidden layer to the output layer(w_{jk}).

Multilayer neural networks form compositional functions that map the inputs nonlin-

early to outputs. If we associate index i with the input layer, index j with the hidden layer, and index k with the output layer, then an output unit in the network computes an output value y_t given and input $\boldsymbol{x}_t$ via the following compositional function:

$$y_t = \phi_k\left[\beta_k + \sum_j^L \phi_j\left(\beta_j + \sum_i^N x_{it} w_{ij}\right) w_{jk}\right] + \varepsilon_t \tag{II.10}$$

where x_{it} is the value of the ith input node, which can be a matrix of lagged values of y_t and some exogenous variables. $\phi_j(.)$ and j are activation functions and number of nodes (L neurons) used at the hidden layer. $\phi_k(.)$ function denotes the output transfer function than can be either linear or a Heaviside step function. As with the single neuron networks, the choice of activation function for the output layer will depend on the task that we would like the network to perform (i.e. categorization or regression). β_j and β_k are the biases or the weights for the connections between the constant input and the hidden neurons and between the neurons and the output respectively.

From a statistical point of view, formulation of a multilayer feedforward neural network model with more than one hidden layer (i.e. h hidden layers when $h = 1, ..., M$) can be generalised to

$$y_t = \phi_k\left[\beta_k + \sum_h^M \phi_h\left(...\phi_j\left(\beta_j + \sum_i^N x_{it} w_{ij}\right)\right) w_{hk}\right] + \varepsilon_t, \tag{II.11}$$

Using identity function for the output unit activation function (in conjunction with nonlinear activations amongst the hidden units) allows the network to perform a powerful form of nonlinear regression. So, the network can predict continuous target values using a linear combination of signals that arise from one or more layers of nonlinear transformations of the input. Based on the universal approximation theorem, a single hidden layer feedforward network architecture with a finite number of neurons can approximate arbitrary well any bounded continuous function of N real variables. So, in this study, we have focused on a feedforward network with only one hidden layer. And to show that the neural network models can be seen as a generalisation of linear models, we allowed for direct connections from the input variables to the output layer and we assumed that the output transfer function $\{\phi_k(.)\}$ is linear, then the model becomes

$$y_t = \beta_k + \sum_i^N x_{it} w_{ik} + \sum_j^L \phi_j\left(\beta_j + \sum_i^N x_{it} w_{ij}\right) w_{jk} + \varepsilon_t, \tag{II.12}$$

where the first summation represents a linear regression term with a constant. A linear regression term hints the model in a right direction when we know that the data contains a linear component. Moreover, this is more interpretable from a statistical perspective and unraveling a bit of a structure behind the network, which is usually seen merely as a black box. It also has the advantage that, if the problem is essentially linear, the hidden neu-

rons tend to get pruned and we are left with a linear model. Since the functional form of this model is known and we only need to find the number of neurons in hidden layer and to estimate the biases and weights, then feedforward networks are categorized as semi-parametric functions. Here, in this approach, we let the data speak for itself as much as possible.

Choosing the appropriate activation function for hidden and output layers is important and depends on the task we would like the network to perform. Activation functions must be differentiable since the learning algorithms such as backpropagation, which determine parameters in neural networks, require the gradient of the activation functions. Most commonly-used activation functions are the identity/linear function, the logistic sigmoid function, and the hyperbolic tangent function. For instance, identity activation function, $\phi_{linear}(z) = z$ and $y \in (-\infty, \infty)$, is commonly used for the output layer in regression problems. Sigmoidal "S" shape functions like Logistic function, $\phi_{logistic}(z) = \frac{1}{(1+e^{-z})}$ where $y \in (0, 1)$, and hyperbolic tangent function, $\phi_{tanh}(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ where $y \in (-1, 1)$, are other choices for the activation function. What is interesting about these derivatives is that they are either a constant, or are can be defined in terms of the original function. This makes them extremely convenient for efficiently training neural networks, as we can implement the gradient using simple manipulations of the feedforward states of the network. There are several ways that we can think about a derivative of a function. The one that sort of immediately comes to mind is the slope which means how quickly the function is changing around a particular point. Another way to interpret derivative is in relation to the optimum. Suppose that the function is an error function and we are trying to minimise it in some way. What a derivative tells us is whether our current value of parameters is higher or lower than where it should be. In general, estimating the set of network parameters $\theta = \{\boldsymbol{W}, \beta\}$ in a way that minimise the errors that the network makes is known as training/learning neural network. It is equivalent to finding a point in parameter space that makes the height of the error surface small. The error surface gets more and more complicated as we increase the number of layers in the network and the number of units in each hidden layer. In principle, four classes of supervised learning algorithm (desired outputs are available) has been discussed in literature: steepest descent algorithm (also known as backpropagation), Newton's method, Gauss-Newton's algorithm and Levenberg-Marquardt algorithm. It is worth noting that studies in time series and forecasting widely used the conventional feedforward neural network trained with the backpropagation algorithm, however, we explain why the backpropagation is not an efficient algorithm and it converges slowly. Therfore, Levenberg-Marquardt algorithm has been implemented in this study which are fast and have stable convergence. The backpropagation algorithm first introduced by Bryson, Denham, and Dreyfus (1963) and popularised in the field of artificial neural network research by Werbos (1988) and Rumelhart, Hinton, and Williams (1986). The goal of backpropagation learning algorithm is to adjust the weights and biases in a way that minimises the network prediction error function.

To solve this problem, the error function's sensitivity to network weights and biases must be quantified based on a gradient descent optimization. Gradient is normally defined as the first order derivative / gradient of the error function with respect to each of the model parameters. This gradient information will give us the direction in parameter space that decreases the height of the error surface. We then take a step in that direction and repeat, iteratively calculating the gradient and taking steps in parameter space.

To explain the math behind backpropagation algorithm, we need to rewrite the neural network formula adding pre-activation signal,z_j, for hidden layer and pre-activation signal,z_k, for output layer and their corresponding outputs y_j and y_k as these will be used for calculating backpropagated errors and error function gradients.

$$y_t = \overbrace{\phi_k\Bigg[\overbrace{\beta_k + \sum_j^L \underbrace{\phi_j\Big(\underbrace{\beta_j + \sum_i^N x_{it}w_{ij}}_{z_j}\Big)}_{y_j} w_{jk}}^{z_k}\Bigg]}^{y_k} + \varepsilon_t \tag{II.13}$$

Backpropagation algorithm is based on Widrow-Hoff learning rule (Delta rule) and works like this:

1. Propagat the observed input forward through the network layers toward the outputs. Initial network output/prediction, y_k, can be anything, as the initial weights are small random numbers, typically between -1 and 1.
2. Calculate network errors E, with respect to a desired target and backpropagate error signal.

   The prediction sum of the squared is a standard way of quantifying error. Given target values and network outputs we can calculate the value of the error function for each setting of weights.

$$E = \frac{1}{2}\sum_k (y_k - t_k)^2 \tag{II.14}$$

3. Propagate the errors backward through the network towards the inputs by weighting it by the weights in previous layers and the gradients of the associated activation functions. Unlike the output layer we cannot calculate the error for the neurons in the hidden layer directly (because we do not have target variables), so we backpropagate them from the output layer.

   Output layer parameters directly affect the error function, so the gradients for those parameters can be calculated as follows

$$\frac{\partial E}{\partial w_{jk}} = (y_k - t_k)\frac{\partial}{\partial w_{jk}}(y_k - t_k) \tag{II.15}$$

Here, based on Chain Rule and having $y_k = \phi(z_k)$, Thus

$$\frac{\partial E}{\partial w_{jk}} = (y_k - t_k)\frac{\partial}{\partial w_{jk}} y_k \tag{II.16a}$$

$$= (y_k - t_k)\frac{\partial}{\partial w_{jk}} \phi_k(z_k) \tag{II.16b}$$

$$= (y_k - t_k)\phi_k'(z_k)\frac{\partial}{\partial w_{jk}} z_k \tag{II.16c}$$

Recall that $z_k = \beta_j + \sum_j \phi_j(z_j) w_{jk}$ and $\frac{\partial z_k}{\partial w_{jk}} = \phi_j(z_j) = y_j$ and again using Chain Rule we have

$$\frac{\partial E}{\partial w_{jk}} = (y_k - t_k)\phi_k'(z_k) y_j \tag{II.17}$$

For notation purposes, we define δ_k to be all the terms that involve index k, and can be interpreted as the network output error after being back-propagated through the output activation function, so we can rewrite the equation above as

$$\frac{\partial E}{\partial w_{jk}} = \delta_k y_j \tag{II.18}$$

Where this equation shows how much each output layer parameter contributes to the error signal.

We follow the same routine for output layer biases, thus the gradient for the biases is

$$\frac{\partial E}{\partial \beta_k} = (y_k - t_k)\phi_k'(z_k)(1) \;=\; \delta_k \tag{II.19}$$

4. Update all the weights and biases in the output layer using the calculated gradients for the parameters. The derivative of the activation function is used to find the rate of change. The weight adjustment is given by

$$w_{jk}^{new} = w_{jk}^{old} - \eta \frac{\partial E}{\partial w_{jk}} \tag{II.20}$$

Where the constant η is the learning rate (step size) and its value falls between zero and one. The direction of search in weight space for the new value of the weights is elected by $\frac{\partial E}{\partial W}$, that shows the sensitivity of the error function to the weights.

5. Having obtained the error for the hidden layer neurons, now using the calculated gradients for the parameters in the hidden layer to update all the weights in the hidden layer.

The process of calculating the gradients for the hidden layer weights starts just the same but, due to the indirect effect on the output error, the forward and backward signals are used to determine the direction in the parameter space to a move that

lowers the network error.

$$\frac{\partial E}{\partial w_{ij}} = \sum_k (y_k - t_k) \frac{\partial}{\partial w_{ij}} y_k \tag{II.21}$$

The sum does not disappear in this case due to the fact that each of the hidden unit outputs affects the state of each output unit. Thus

$$\frac{\partial E}{\partial w_{ij}} = \sum_k (y_k - t_k) \frac{\partial}{\partial w_{ij}} \phi_k(z_k) \tag{II.22a}$$

$$= \sum_k (y_k - t_k) \phi_k'(z_k) \frac{\partial}{\partial w_{ij}} z_k \tag{II.22b}$$

Where z_k is indirectly dependent on w_{ij} and by using the Chain Rule we have

$$\frac{\partial z_k}{\partial w_{ij}} = \frac{\partial z_k}{\partial y_j} \frac{\partial y_j}{\partial w_{ij}} \tag{II.23a}$$

$$= \frac{\partial}{\partial y_j} y_j w_{jk} \frac{\partial y_j}{\partial w_{ij}} \tag{II.23b}$$

$$= w_{jk} \frac{\partial y_j}{\partial w_{ij}} \tag{II.23c}$$

$$= w_{jk} \frac{\partial \phi_j(z_j)}{\partial w_{ij}} \tag{II.23d}$$

$$= w_{jk} \phi_j'(z_j) \frac{\partial z_j}{\partial w_{ij}} \tag{II.23e}$$

$$= w_{jk} \phi_j'(z_j) \frac{\partial}{\partial w_{ij}} (\beta_i +_i x_i w_{ij}) \tag{II.23f}$$

$$= w_{jk} \phi_j'(z_j) x_i \tag{II.23g}$$

Plugging the last two Equations gives

$$\frac{\partial E}{\partial w_{ij}} = \sum_k (y_k - t_k) \phi_k'(z_k) w_{jk} \phi_j'(z_j) x_i \tag{II.24a}$$

$$= \phi_j'(z_j) x_i \sum_k (y_k - t_k) \phi_k'(z_k) w_{jk} \tag{II.24b}$$

$$= x_i \phi_j'(z_j) \sum_k \delta_k w_{jk} \tag{II.24c}$$

$$= \delta_j x_i \tag{II.24d}$$

Thus, the gradient for the hidden layer weights is simply the output error signal backpropagated to the hidden layer, then weighted by the input to the hidden layer and this can be interpreted as a proxy for the contribution of the weights to the output error signal. Similar to what we have seen in the output layer, the gradient

for the biases is

$$\frac{\partial E}{\partial \beta_i} = \delta_j \tag{II.25}$$

The weight adjustment for hidden layer is given by

$$w_{ij}^{new} = w_{ij}^{old} - \eta \frac{\partial E}{\partial w_{ij}} \tag{II.26}$$

6. By repeating iteratively the steps network can be trained in a way that converges to optima. The set of new weights are repeatedly presented to the network until the error value is minimised. Around the optimum point, all the elements of the gradient would be very small, which leads to tiny changes in new weights.

For Newton's method the second-order derivatives of the error function need to be calculated for each elements of gradient vector and the weights update rule is calculated by the following formula:

$$W^{new} = W^{old} - H^{-1} \frac{\partial E}{\partial W} \tag{II.27}$$

Where H^{-1} denotes the inverse of Hessian matrix and is calculated by getting the second-order derivatives of error function. The inverse of Hessian matrix can be too complicated and may be singular.

In Gauss-Newton algorithm, in order to simplify the calculation process, Hessian matrix is approximated by Jacobian matrix and the update rule is defined as:

$$W^{new} = W^{old} - \left(J'J\right)^{-1} \frac{\partial E}{\partial W} \tag{II.28}$$

In this case, it does not require the calculation of second-order derivatives of the error function, but again this approximation of Hessian matrix $J'J$ may not be invertible.

The Levenberg-Madquardt (LM) algorithm algorithm has been proposed by Levenberg (1944) and Marquardt (1963) independently. This learning algorithm is based on a modification of the Gauss-Newton method and provides a numerical solution to iteratively minimise a least square error function without having to compute the Hessian matrix. This algorithm approximates the Hessian matrix and the gradient by calculation of the Jacobian matrix (J), which contains the first derivatives of the model output errors (e) with respect to the network weights and biases. Weight updates rule of the LM algorithm can be formulated as follow:

$$W^{new} = W^{old} - \left(J'J + \mu I\right)^{-1} J'e \tag{II.29}$$

When μ (combination coefficient) is positive, elements on the diagonal of the approx-

imated Hessian matrix will be larger than zero and guarantee that $(J'J + \mu I)$ is always invertible. It also converges to the Gauss-Newton algorithm when μ is zero and converges to a gradient descent with a small learning rate when μ is large enough. The LM algorithm is much faster and more stable than previous learning algorithms since it can switches efficiently between those algorithm during the learning process.

Developments in neural network modelling have further led to the nonlinear generalization of PCA. The next subsection explains how the feed forward neural network can be extended to perform nonlinear PCA (NLPCA).

III. Nonlinear Factor Model

A. *Factor estimation using neural network PCA*

The NLPCA basically employs a standard feedforward neural network with four layers of transfer functions mapping from the inputs to the outputs or as explained before one can think of an overall network consisting of two feedforward neural networks (reduction and reconstruction network), which are connected directly and the output of first network is the input for the second one. The layer in the combined network that joins the reduction and reconstruction networks is called the bottleneck layer and contains q neurons, where $(q \ll m)$, giving a reduced dimensionality representation of the input data. By replacing all the layers of transfer functions with the identity function, neural network PCA returns the same result as classical PCA.

Figure.III.1 illustrates such a network with three layers of hidden neurons sandwiched between the input layer and the output layer. m dimensional data are compressed to q nonlinear component in the bottleneck reduced layer. This architecture, called an autoassociative network. In this architecture both weight (and bias) parameters of the network and nonlinear component are optimised simultaneously through an unsupervised learning procedure to minimise the reconstruction error. Note when the bottleneck layer has more than one neuron to accomplish the dimension reduction, the residual of the first nonlinear component can be input into the same network to extract the second nonlinear component and so on. Although a better strategy is to penalize the reconstruction error term by adding $(u_1 \cdot u_2)^2$ penalty term to extract a curve surface force the nonlinear components to be uncorrelated with each other (Hsieh (2004)).

Writing $\boldsymbol{x}_t, \tilde{\boldsymbol{x}}_t \in \mathbb{R}^m$ for the input and output data of the overall network, $\boldsymbol{y}_t^{(x)}, \boldsymbol{y}_t^{(u)} \in \mathbb{R}^j$ for the values of the neurons in the input and output hidden layers, $\boldsymbol{z}^{(x)}, \boldsymbol{z}^{(u)}$ for the values of the pre-activation (sum of each input variable multiply by the corresponding weight plus the bias) in the input and output hidden layers and $\boldsymbol{u} \in \mathbb{R}^q$ for the values of

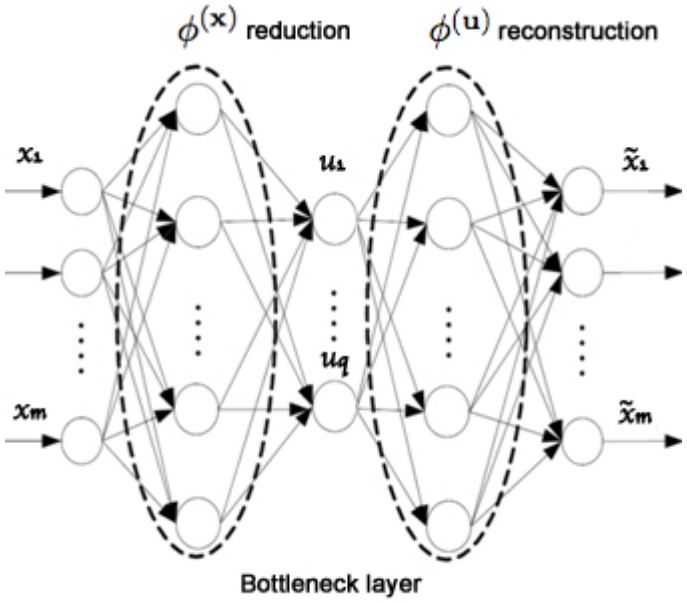


Figure III.1. Schematic diagram of the standard autoassociative neural network architecture for calculating the nonlinear principal component analysis (NLPCA).

the bottleneck neurons, the network transfer functions can be formulated as follow

$$\boldsymbol{y}_t^{(x)} = \boldsymbol{\phi}_j^{(x)}(\boldsymbol{z}^{(x)}) \tag{III.1a}$$

$$\boldsymbol{u} = \boldsymbol{\phi}_k^{(x)}(\boldsymbol{y}_t^{(x)}) \tag{III.1b}$$

$$\boldsymbol{y}_t^{(u)} = \boldsymbol{\phi}_j^{(u)}(\boldsymbol{z}^{(u)}) \tag{III.1c}$$

$$\tilde{\boldsymbol{x}}_t = \boldsymbol{\phi}_k^{(u)}(\boldsymbol{y}_t^{(u)}) \tag{III.1d}$$

Where the transfer function $\boldsymbol{\phi}_j^{(x)}$ in Eq.III.1a, maps from the input vector of length m, to the first hidden layer (reduction layer). Likewise the second transfer function $\boldsymbol{\phi}_k^{(x)}$ in Eq.III.1b, which is usually an identity function, maps from the reduction layer to the bottleneck layer to extract nonlinear component values. In the following, $\boldsymbol{\phi}_j^{(u)}$ in Eq.III.1c, maps from the nonlinear component vector of length q, to the final hidden layer (reconstruction layer). $\boldsymbol{\phi}_k^{(u)}$ in Eq.III.1b, is usually an identity function and produces an approximation of the original data.

Here we briefly describe the model fitting considerations of NLPCA. First of all, note that in contrast to PCA, NLPCA approximation is not an unique analytic solution and must be found through numerical minimisation. In other words, an iterative minimisation procedure carries. Entire samples will be divided into training and validation, and optimisation terminates when either an error over training data stop changing or an error over validation data start increasing. We do not just pick the approximation with the lowest error, but also ensure the approximation is robust. Robustness happens when validation

and training share same shape and orientation approximation. It is also worth mentioning that the advantages of NLPCA over the PCA is highly dependent on the data. Noisy and short data can effect on the performance of NLPCA. When underlying relation in data is linear and Gaussian, NLPCA should yield PCA solution. To control the model's complexity and subsequently avoid over fitting, the following strategies can be employed. A model validation based on a missing data approach and regularisation with a weight penalty ensures that NLPCA robustly characterises the underlying structure and it does not describe data in a nonlinear way when the inherent data structure is, in fact, linear (For more details see Christiansen (2005)). A number of runs with random initial weights parameters helps optimization procedure to reach a solution as close as possible to the global minimum in error space. Moreover, Monahan (2001) shows in his work how to measure explained variance of the first nonlinear component in NLPCA, similarly to PCA, by defining residual vectors.

B. *Nonlinear forecasting step*

Here we compare different linear and nonlinear ways that the forecast equation for the variable of interest $x_{iT+h|T}$ can be formulated in the factor model setting. There are also two different ways to obtain an h-step-ahead forecast directly from a long-horizon equation or a sequence of 1-step-ahead forecasts. For instance, Marcellino, Stock and Watson (2006) have shown that the sequential approach outperforms the direct approach (current paper focuses on 1-step-ahead forecast of financial return series). Assuming the factors and the loadings are observed, the general formulisation of forecast equation is

$$x_{iT+h|T} = \boldsymbol{\lambda}_i' \boldsymbol{u}_{T+h|T} + \xi_{iT+h|T} = \chi_{iT+h|T} + \xi_{iT+h|T} \tag{III.2}$$

But in practice, the parameters and the factors are unknown and have to be estimated and the forecast of factors $\hat{\boldsymbol{u}}_{T+h|T}$, idiosyncratic component $\hat{\xi}_{iT+h|T}$, and/or the common component $\hat{\chi}_{iT+h|T}$ are needed to make the forecast feasible. This can be done by an univariate autoregressive model when there is only one factor and a vector-autoregressive when there is more than one factor. A forecast of common components can be obtained by projecting each common component on factors.

Empirical studies, for instance Stock and Watson (2002b), Boivin and Ng (2005), Bai and Ng (2006), and Ludvigson and Ng (2009), predict series of interest by using factors ignoring the idiosyncratic component. If this is the case, we only use the information that is common to all predictors, and not idiosyncratic information that might also be relevant for prediction. Here three different types of 1-step-ahead forecasts are considered in order to forecast a series using factors and an idiosyncratic component (given a static approach for factor estimation),

$$LinearFactorModel \begin{cases} \hat{x}_{iT+1|T} = \hat{\beta}_i' \hat{\boldsymbol{u}}_T & \text{(III.3a)} \\ \hat{x}_{iT+1|T} = \hat{\boldsymbol{\lambda}}_i' \boldsymbol{u}_{T+1|T} & \text{(III.3b)} \\ \hat{x}_{iT+1|T} = \hat{\boldsymbol{\lambda}}_i' \boldsymbol{u}_{T+1|T} + \hat{\xi}_{iT+1|T} & \text{(III.3c)} \end{cases}$$

Where $\hat{\beta}_i$ estimated by a linear regression between variable of interest and the lags of factors. One can also add the lags of the variable of interest to the model.

Based on autoassociative neural networks and feedforward neural networks models as explained in the previous section, now we can define the nonlinear extension of factor models. First of all, a factor estimation step in nonlinear setting is done by NLPCA producing $\boldsymbol{u}_t^{(NL)}$.

$$NonlinearFactorModel \begin{cases} \hat{x}_{iT+1|T} = \Phi(\hat{\boldsymbol{u}}_T^{(NL)}) & \text{(III.4a)} \\ \hat{x}_{iT+1|T} = \boldsymbol{\phi}_k^{(u^{NL})}(\boldsymbol{\phi}_j^{(u^{NL})}(\hat{\boldsymbol{u}}_{T+1|T}^{(NL)})) & \text{(III.4b)} \\ \hat{x}_{iT+1|T} = \boldsymbol{\phi}_k^{(u^{NL})}(\boldsymbol{\phi}_j^{(u^{NL})}(\hat{\boldsymbol{u}}_{T+1|T}^{(NL)})) + \hat{\xi}_{iT+1|T}^{(NL)} & \text{(III.4c)} \end{cases}$$

In III.4a first we estimate the common factor using NLPCA, then we use them as the input for a feedforward neural network where the targeta variable is the 1-step-ahead forecast of variable of interest. In III.4b after estimating the factor, we forecast the factor using a feedforward neural network then we propagate a predicted factor through reconstruction layer of NLPA. In III.4c ,similar to model explain in III.4b, we propagate the forecast value of the factor through a reconstruction layer of NLPA and we also add the forecast value of the corresponding idiosyncratic component obtained from a feedforward neural network. In the next section we illustrate our modelling strategy on a financial empirical example.

IV. Empirical analysis

A. *Data*

The data are daily returns of $m = 418$ equities on the S&P 500 index from 04.01.2005 through 31.12.2014, for a total of 2517 observations. We will use the 03.01.2005 - 31.12.2013 period ($T = 2265$ in-sample size) to estimate our forecasting models, and we will use the holdout sample period 02.01.2014 - 31.12.2014 (252 observations) to examine models' out-of-sample forecasting performance. We calculate 1-step (here one day) ahead forecasts of targets ($\hat{x}_{it+1|t}$ return series to be forecast) based on a rolling (moving) estimation window. The estimators of the real value parameters are updated at every time instance. Choice of window size T is a compromise between overly noisy and overly smoothed estimate of parameters and is usually chosen such that $T/N \geq 1$. Our empirical experiments

show that different values of T can have an effect on a models' forecasting performance (notably the Random Walk model).

We show the entire series in Figure IV.1 , which display clear comovements between the return time series. Heavy tails, leverage effects, nonlinear dependencies and volatility clustering are some other stylized facts that are common to a wide set of high-frequency (e.g., daily) financial return series. Figure IV.1 also demonstrate that market become more tightly coupled in volatile periods.

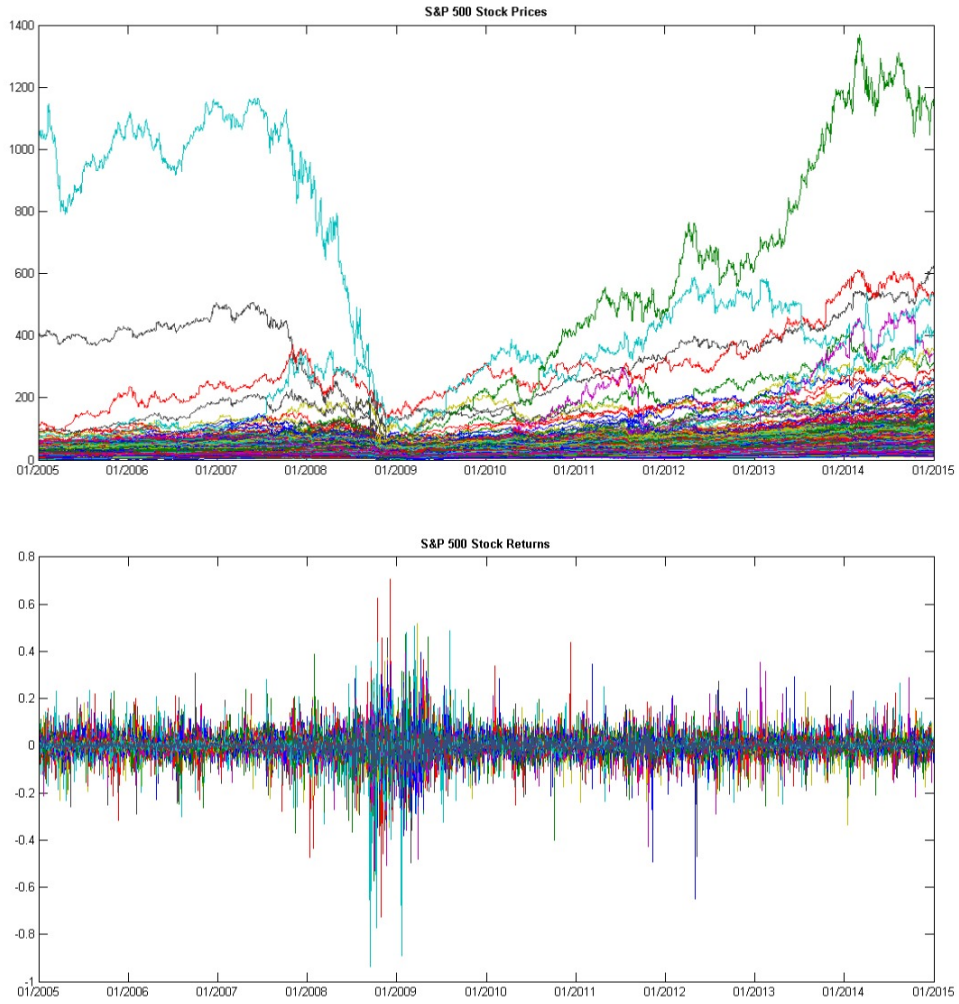


Figure IV.1. Target time series: prices and returns

We also zoom in on the 02.01.2014 - 31.12.2014 out-of-sample period in Figure IV.2.a in order to reveal the comovement pattern in better detail. Understanding of the comovement between financial equities is crucial for forecasting procedure, therefore we use PCA and NLPCA to produce a parsimonious representation of market correlation (NLPCA is beyond correlation) and demonstrate that a small number of components account for a large proportion of the variability of equities that we consider. In general, we are interested in understanding what factors, lead to movements in an asset's return. One way of identifying these factors, which presumes no knowledge of any factors, and hence is entirely

statistical in nature, is via principal component analysis. PCA can be used to identify the underlying statistical factors that explain comovement in asset returns (See Fenn, Porter, Williams, McDonald, Johnson, and Jones (2011) for a use of principal component analysis and random matrix theory to investigate financial market correlations).

Figure IV.2.b visualises the returns' correlation matrix using a heatmap, ranging from black (zero) to white (one). PCA and NLPCA are principally useful when the data under consideration is correlated. In this study PCA and NLPCA are used to analyse the correlated returns of 418 equities.

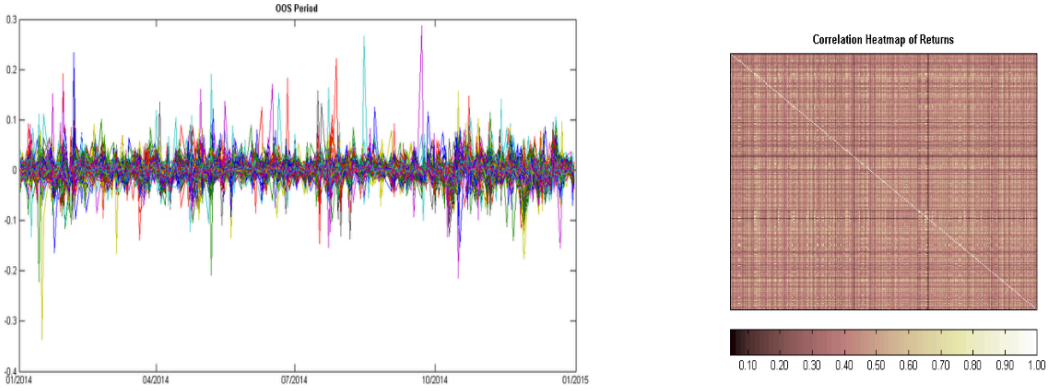


Figure IV.2. Visualization of comovement between return series; Right: Correlation matrix heatmap, Left: Return series in out-of-sampleperiod

The first principal component is typically assumed to represent the broad market. The next few are assumed to be the sector/style related factors. The remaining components represent the idiosyncratic properties of stocks. For the given set of equities we can conclude that about 35% of all the variance is because of the broad market factor (systematic risk).

It is worth mentioning that the fraction of the variance in return series explained by the first few linear and nonlinear PCs changes overtime. In Figure IV.3, we show how the fraction of the variance explained by first three PCs changes as a function of time for time windows of length 200. Both linear and nonlinear PCs follow the same pattern and in all windows, explained the variance by nonlinear PCA are somewhat higher than the explained variance by linear PCA.

Figure IV.3 suggests that market returns have become more correlated during the period of crisis and a sharp increase in the fraction of the variance explained by the first component coincided with major shocks to the financial system (i.e. the collapse of Lehman Brothers in September 2008). It is very instructive for building forecast models and input selection to consider the changes in the variance explained by the first few PCs. We may include the second component (or macro factors) as model predictor when the first component could not account for a large proportion of the variability of data. The square of the first component (a proxy for volatility of the first component) may also improve the forecast performance of the mean returns.

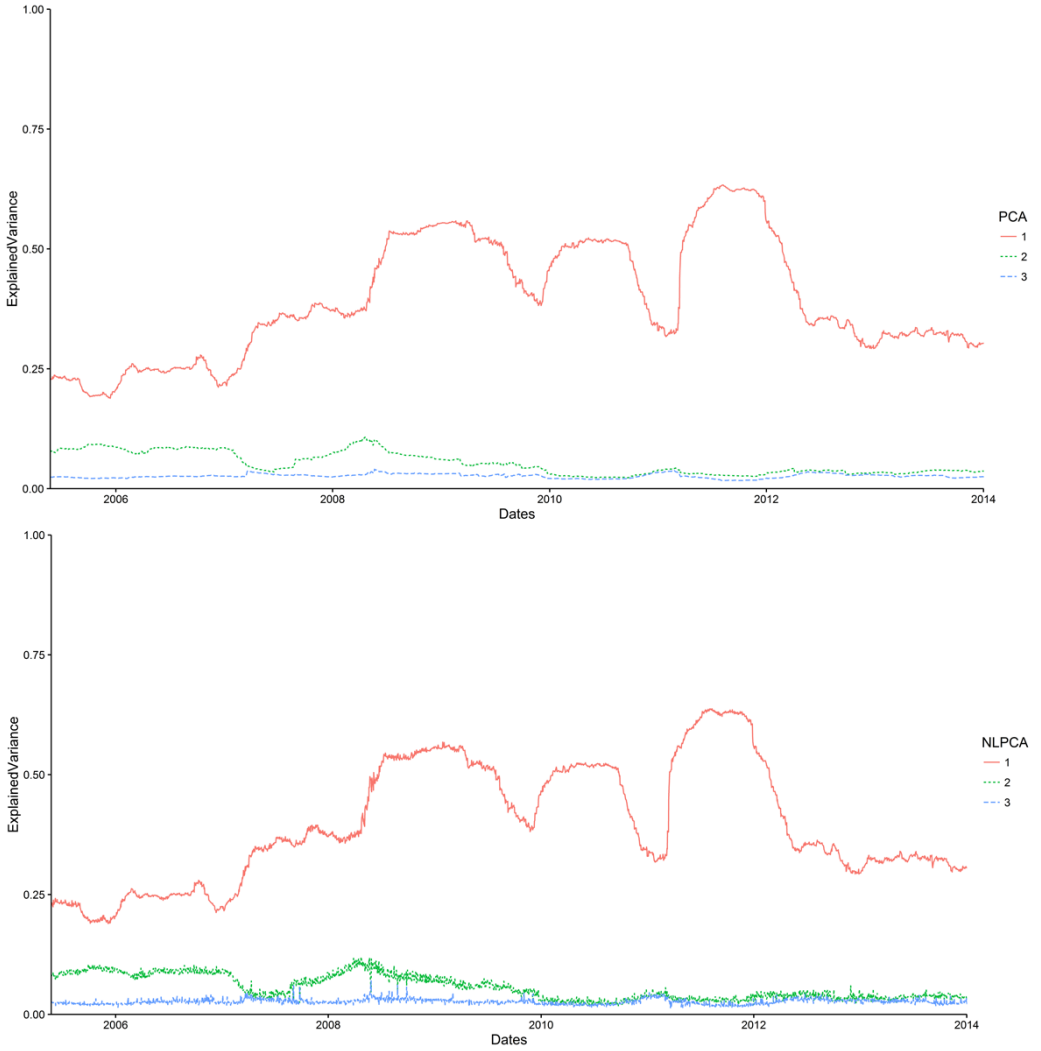


Figure IV.3. Fraction of the variance in return series explained by the first three PCs

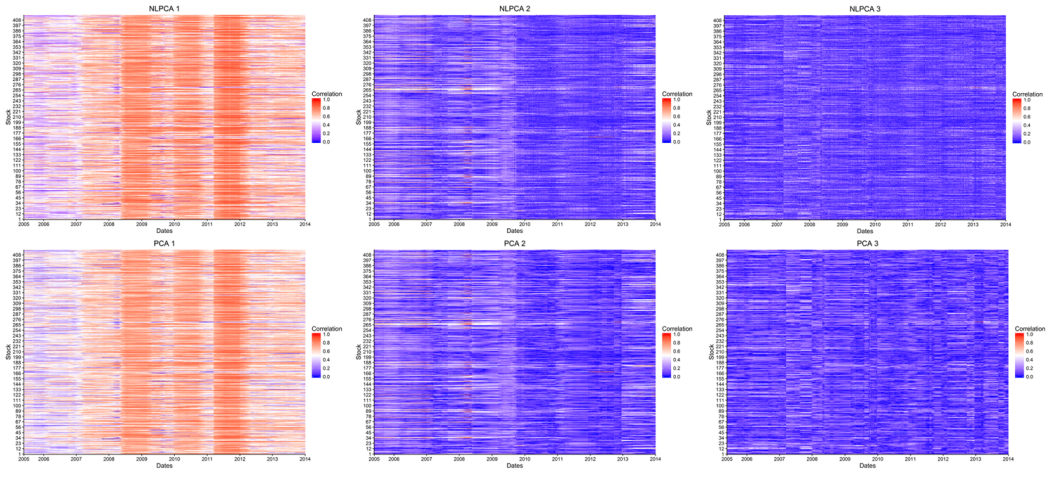


Figure IV.4. Correlation between each equity and the first three PCs/NLPCs as a function of time. equities are on the vertical axis and the horizontal axis represents the time windows.

In Figure IV.4, we characterize the time-evolving relationships between the different equities by investigating the correlations between the return series and the first three

PCs/NLPCs. This figure highlights that there are many time steps at which the correlation between the in the first linear and nonlinear component and return series are significantly large (greater than 0.7). However the correlations decreases between the equity returns and the second and the third components which implies that much of the key structure from the correlation matrices is contained in the first component. Again after Lehman Brothers' collapse, the first component became strongly correlated with nearly all equities implies that many different equities were being driven by the same macroeconomic forces.

B. Testing for nonlinearity in and between financial return series

B.1. Quick review of linearity tests

Another point this paper tries to emphasize is that statistical tests to detect nonlinearity in and between time series can assist in terms of choosing the appropriate factor estimation and forecast equation. Before we apply nonlinear models to financial data, it is logical to first ask if the use of such models is justified by the data. To this purpose, we examine the nonlinear dependencies in and between return series and in first principal component series by applying some of the well-known and novel tests introduced in the literature. For instance, applying a nonlinear principal component analysis (NLPCA) for factor extraction step could be justified when the test shows that there is nonlinear dependencies between return series. On the other hand, testing for nonlinearity on the principal component series helps in terms of forecasting them in an appropriate way. We first discuss the definition and the characteristics of (non)linear processes and then we present briefly review of the linearity tests that have been proposed in the literature. Finally, we investigate nonlinear features of return series and PCs.

In contrast to a linear stochastic process which can be specified by its first and second order statistics (i.e. mean, variance, and autocorrelation) or equivalently Fourier amplitudes, a nonlinear process is generated by a nonlinear dynamic equation of iid random variables consisting of the current and past shocks and can be shown as follows:

$$x_t = \phi(\varepsilon_t, \varepsilon_{t-1}, ...) \tag{IV.1}$$

Where $\phi(.)$ is a nonlinear function. Nonlinearity may arise in different ways. The characteristic of a nonlinear time series such as higher-moment structures, time-varying variance, asymmetric fluctuations, thresholds and breaks can only be modelled by an appropriate nonlinear function like $\phi(.)$ and a linear process is not adequate to model these features.

Classification of different statistical approaches that are testing nonlinearity in a time series is a challenging task as they entail consideration of various types of nonlinear dynamics and come from different disciplines. There are only a few papers available on the literature that tries to establish a classification of linearity tests. Granger and Teräsvirta

(1993), Teräsvirta, Tjøstheim, and Granger (2010), and recently Giannerini (2012) are examples of this.

The main idea behind various linearity tests is a hypothesis testing procedure. Every hypothesis test starts with a null hypothesis (H_0) and an alternative (H_1). In general, the null hypothesis of linearity tests states that observed series are generated by Gaussian linear stochastic processes against an alternative hypothesis that states observed series are rooted in nonlinear dynamics. To be more precise, H_0 tests the hypothesis that the time series is completely specified by its first and second order statistics (i.e. mean, variance, and autocorrelation or its frequency domain counterpart, power spectrum). One can classify linearity tests based on their alternative hypothesis into two broad categories. The tests with a specified model as an alternative and those against a nonspecified alternative. Tests with a specified alternative are also usually called Lagrange Multiplier (LM) tests. LM tests refer to those tests focusing on the coefficients of a nonlinear specified model (i.e. threshold autoregressive model: Tong (1978, 1983) and Tong and Lim (1980); and autoregressive conditional heteroskedasticity: Engle (1982) and are parametric. In this case, Wald and Likelihood Ratio are not applicable directly when a specified nonlinear alternative is only identified under the null hypothesis of linearity; see the chapter 6 of Granger and Teräsvirta (1993).

On the contrary, many of the tests proposed in the literature are against a nonspecified alternative. This group of tests has been more popular in the applications when testing linearity is the main aim. However, they can not assist for model building exercises such as forecast problems. In some of the tests in this group, classified as Diagnostic tests, the null hypothesis states that the series is explained by a white noise process and residuals of a properly specified linear fit should be independent versus the alternative of serial dependence. Therefore different aspects of such time series come under the investigation of different diagnostic tests. For example some tests like RESET (Ramsey (1969)), Keenan (Keenan (1985)) and Tsay's F test (Tsay (1986)) consider an auxiliary regression of residuals on a specific function of X_t, some other ones such as Ljung-Box (Ljung and Box (1978)) and Mcleod-Li (McLeod and Li (1983)) employ the autocorrelation function of the residuals, some like BDS (Brock, Scheinkman, Dechert, and LeBaron (1996)) also measure the density structure of such residuals from a linear fit in an embedded phase space and etc.; see the chapter 5 of Tong (1990) and Li (2003) for an overview of diagnostic tests.

In the rest of the tests against a nonspecified alternative, null hypothesis and alternatives are not based on a linear or nonlinear fit and have the ability to detect the presence of specific nonlinear features as we call them here tests for nonlinearity. For instance, tests based on higher order statistics can detect asymmetry and reversibility in time series utilizing the fact that these statistics do not contain new information for linear series and hence for example the bicorrelation of such data over time (or the bispectrum over different frequencies) is constant; see Mendel (1991); Nikias and Petropulu (1993) and Petrop-

ulu (1999).

The asymptotic null distribution of the classical nonlinearity test statistics mostly depend on rigorous assumptions and are not always accurate, thus to improve the power of tests, randomization, and bootstrap approaches are introduced in the linearity tests literature, widely known as the method surrogate data, the main idea of which is to compare the value of a discriminating nonlinear measure for observed series to that of surrogates series in order to detect a meaningful deviation (Theiler, Eubank, Longtin, Galdrikian, and Farmer (1992) and Schreiber and Schmitz (2000)). A subsection is devoted to different surrogate generating algorithms and also the discussion of appropriate test statistics.

In this paper we use the following tests to detect nonlinearity in a time series: RESET test that tests for functional form, BDS test that tests for the iid series, Teraesvirta NN test and White NN test that test for neglected nonlinearity. For the methods of surrogate data, we apply Amplitude Adapted Fourier Transform (AAFT), and Iterative Amplitude Adapted Fourier Transform (IAAFT) of Schreiber and Schmitz (1996) algorithms for the generation of surrogate data and Bi-Correlation and Time Reversal as discriminating statistic. See Schreiber and Schmitz (1997) for more details. Finally, we apply Entropy testing for nonlinear serial dependence in a time series based upon the combination of the entropy measure together with resampling methods that were recently introduced by Giannerini, Maasoumi, and Dagum (2015). They have proposed a test for identification of nonlinear serial dependence in a time series against the general $H0$ of linearity, in contrast to the more widely examined $H0$ of "independence".

To shows how the underlying structure within the recorded data can be examined to determine whether a conceptually more demanding NLPCA model is required, a test introduced by Kruger, Zhang, and Xie (2008)) has been applied in this paper. The idea behind this test is explained in the following section.

B.2. Linearity test results

The following subsection presents the results of the linearity tests applied to the sample. The nonlinear dependence in and between return series for 418 companies and their first PCs are examined by applying the tests mentioned above. For the BDS test, an AR model and a best ARMA model is used to remove any serial correlation in the data, and the tests apply to the residual series of the model. It is also difficult to determine if complex real world time series like financial returns behave in a linear or nonlinear fashion. The experimental results that we have got from different nonlinearity tests indicated that the financial time series are rarely pure linear and they consists of nonlinear patterns.

Figure IV.5.a, shows the results of the test of nonlinearity based on the Reset test and White NN test. Since Teräsvirta the neural network test for neglected nonlinearity uses a Taylor series expansion of the activation function, the test statistic and the result are quite similar to what we see in the Reset test, therefore we ignore plotting the result of the Teräsvirta test. The evidence against linearity is clear in the series in all lags for Reset test

and in first five lags for White NN test. Figure IV.5.b, shows the results of the Entropy test of linearity based on surrogate data obtained through the sieve bootstrap to PC series. The evidence against linearity is clear in the series in all lags.

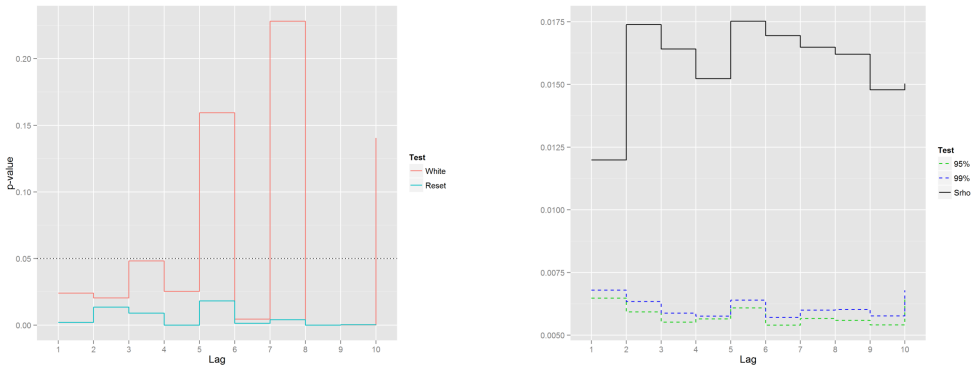


Figure IV.5. Testing for nonlinearity in PC series. Right: Plot of Entropy test statistic of nonlinearity for the PC_1 series. at lags 1:10. The dashed lines indicate the rejection bands at 95% (green/light) and at 99% (blue/dark), Left:Plot of Reset and White test statistics of nonlinearity for the PC_1 series. at lags 1:10. y-axis shows the p-value of the test statistics

Figure IV.6, shows the results of the linearity test based on the Reset test on three stocks. The evidence against linearity is different in different equities.The major limitation of a linear model is the pre-assumed linear form of the model and therefore, no nonlinear patterns can be captured. A pure nonlinear model is not also adequate to handle fairly both linear and nonlinear patterns, especially when the linear component is superior to the nonlinear component. The performance of the linear model and neural networks is not robust when the time series contains complex linear and nonlinear patterns. To overcome this issue, we add a skip-layer to a neural network and then we penalise the loss function to guarantee that model will reduce to a linear model if no nonlinearity exists in the data.

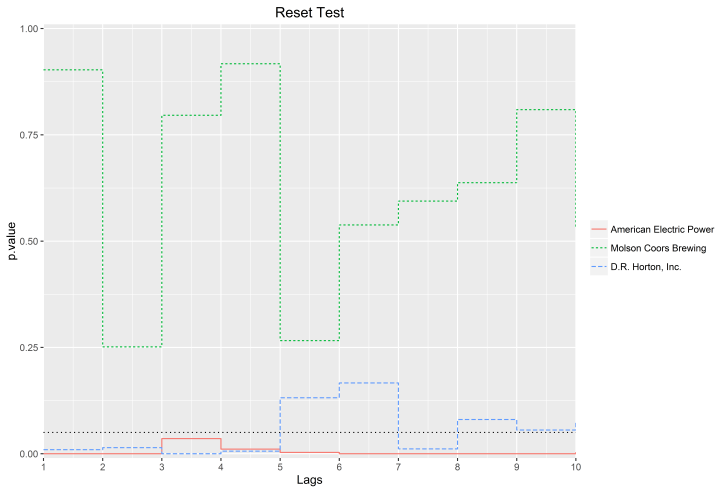


Figure IV.6. Plot of Reset test statistics of nonlinearity for three different equities

B.3. Testing for nonlinearity between time series

In this section, we examine the nonlinear dependencies between return series to show how the underlying structure within the recorded data can be examined to determine whether a conceptually more demanding NLPCA model is required. The test applied in this section introduced by Kruger et al. (2008) and it is designed to detect nonlinearities based on nonlinear principal component analysis and between time series. This test divides the entire data into several disjunct regions through prior knowledge of the process or by direct analysis of the recorded data, and it takes advantage of the residual variance in each of the regions. The test then examines whether the sum of the residual variances or more precisely its PCA equivalent, the sum of the discarded eigenvalues, are significantly different among these regions. For this, the accuracy bounds are calculated for the sum of the discarded eigenvalues of one of the regions taking advantage of the confidence limits for the mean and variances, and hence each value of the correlation matrix. The sum of the residual variances for other regions are then benchmarked against this accuracy bound. If at least one of the sum of the discarded eigenvalues lies outside this bound, the null hypothesis is rejected and NLPCA approaches should be considered. The power of the test is also increased utilizing the principal of cross-validation and thus calculating the accuracy bounds for each region and comparing sum of the discarded eigenvalues of the other regions with it.

In our case, we divided the sample into 4 disjunct regions. The accuracy bounds for each disjuct region and also sum of the discarded eigenvalues were computed. These bounds were based on thresholds for each element of the correlation matrix corresponding to confidence level of 95Note that the processes were normalized with respect to the mean and variance of the regions for which the accuracy bounds were computed. After performing the test, the third region falls out of accuracy bounds of the first region. The third region contains the data starting from 6 July 2009 until 29 September 2011. One can speculate that the nonlinearity could be traced back to the aftermath of the 2008 financial crisis. Fig IV.7 Benchmarking of the residual variances against accuracy bounds of each disjunct region and illustrates that the relationship between recorded financial data is nonlinear.

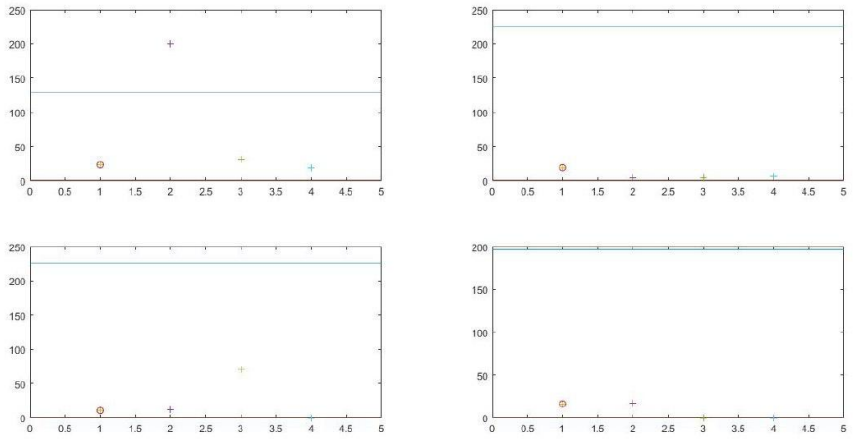


Figure IV.7. Accuracy bounds and residual variances. Sample is divided into smaller disjunct regions; and accuracy bounds are determined for the sum of the discarded eigenvalues of each region. If this sum is within the accuracy bounds for each region, the process is assumed to be linear. Conversely, if at least one of these sums is outside, the process is assumed to be nonlinear. As the figure illustrates, the recorded financial data is nonlinear.

C. Proposed and competing models

In this paper we attempt to answer the question of whether it is possible to forecast with a large panel of predictors, while considering nonlinear dynamics in a high-dimensional dataset and to show how a model with such features can improve financial forecasting accuracy. We proposed a nonlinear high-dimensional forecasting model that is able to handle high-dimensionality and nonlinearity by applying a neural network based principal component analysis to estimate common factors from a large panel of data and nonlinearly forecasts the series of interest uses the factor estimates with a feedforward neural network. We compare our proposed nonlinear factor model NLFM with several competing models and benchmarks. As a benchmark for comparison, we use the sample mean of $\boldsymbol{x}_t$ in the in-sample window as the 1-step ahead forecast. This corresponds to assuming that the log daily price of stocks follows a random walk with drift and it is almost equivalent to the "zero forecast" when the in-sample window is large enough. Furthermore, a buy-and-hold strategy in the market portfolio has been considered as another benchmark. To understand whether considering the comovemen of returns improves forecast accuracy we compare univariate forecasting methods with high-dimensional forecasting methods. The univariate models that we estimate here can be categorised in two groups, linear univariate models: specifically, AR(1) model and best ARMA(p,q) model, and nonlinear univariate model: feedforward neural networks. The high-dimensinal models are also in two groups, linear static factor models and nonlinear neural network factor models. Since the information inside an idiosyncratic component might also be relevant for prediction, we compare the forecast accuracy of linear/nonlinear factor models considering an idiosyncratic component to forecast a return series with models, ignoring the idiosyncratic component.

D. *Forecast evaluation*

Since the predictability of financial returns is used to guide decisions, forecast accuracy and comparing the forecastability of alternative models is of obvious importance to finance managers to discriminate among forecast models and choose an adequate model that is inextricably linked to the predictive performance of the model. Forecast error can be defined simply as $\hat{e}_{it+1} = x_{it+1} - \hat{x}_{it+1}$, when x_{t+1} denotes the observation at time $t+1$ and $\hat{x}_{it+1}$ denotes the forecast of x_{it+1}. Among various models, the one that has the minimal forecast error is often deemed optimal. However, the model with minimum forecast error does not necessarily guarantee profit maximisation, which is the ultimate objective of making predictions of returns in financial markets. There are some forecast accuracy measures from the time series approach, of which the most commonly used are scale-dependent measures based on the absolute errors or squared errors like the Mean Square Error (MSE). These are useful and generally straightforward especially when comparing different models on the same set of data. Root Mean Square Error (RMSE) is often preferred as it is on the same scale as the data. As financial models with minimal direct measures, such as RMSE, do not necessarily guarantee maximised investment profits, an alternative approach, which explicitly addresses these issues, is to evaluate the merits of the alternative financial forecasting models based on indirect measures such as performance of a trading strategy. Armstrong and Collopy (1992), Pesaran and Timmermann (1995, 2000), Granger and Pesaran (2000) and Engle and Colacito (2006) argue that a forecast evaluation criterion should be related to decision making and judge predictability of financial returns in terms of portfolio simulation. More specifically, a trading (portfolio) simulation approach assumes that all competing models are applied with stock market virtual investment decisions, and out-of-sample portfolio performances are used to evaluate the predictability of alternative models. The accuracy measures that this paper employs to discriminate between a competing set of forecasting models are from both time series approach (the out-of-sample RMSE and the out-of-sample coefficient of determination ($R_i^2 = 1 - \frac{(\hat{e}_i - \bar{\hat{e}}_i)'(\hat{e}_i - \bar{\hat{e}}_i)}{(x_i - \bar{x}_i)'(x_i - \bar{x}_i)}$)) and trading simulation approach (the out-of-sample hit rate - also called the probability of detection given by $h_i = \frac{\sum_{t=T_1+1}^{T_2} sign(x_{it}\ \hat{x}_{it|t-1})}{T_2 - T_1}$ - and the portfolio performance during the out-of-sample period). Moreover, to test the equality of forecast accuracy between competing models, we employ the Diebold Mariano (DM) test, which can easily be applied to a wide variety of criteria including RMSE (see Diebold and Mariano (1995) and Harvey, Leybourne, and Newbold (1997)). The portfolio construction is based on passive equally weighted $(1/N)$ portfolios with short sell that are known to be a very stringent benchmarks and that many optimization models fail to outperform (see DeMiguel, Garlappi, and Uppal (2009). We compute the portfolio's out-of-sample return and volatility as well as the Sharpe ratio[1]. It worth mentioning that the hit

[1]In this paper we have also constructed portfolios including only one stock for each competing model. In general, trading simulation results from these portfolios were aligned with the results we get from portfolio

rate shows the proportion of correctly predicted signs of returns and it is sensitive only to missed events rather than false forecasts and should be interpreted with care, but, in contrast to hit rate, portfolio simulation can count in false forecasts.

E. Results

This section compares our proposed nonlinear factor model (NLFM) with several competing models and benchmarks. As mentioned in previous section, three different types of 1-step-ahead forecast are considered in this paper in order to forecast return series using factors and idiosyncratic components. First we compare linear/nonlinear factor models estimating a regression between the return series and the lags of estimated factor[2] ($\hat{x}_{iT+1|T} = \hat{\beta}_i'\hat{u}_T$ shown by FM(u_t) , $\hat{x}_{iT+1|T} = \Phi(\hat{u}_T^{(NL)})$ shown by NLFM(u_t)). Then we compare linear/nonlinear factor models multiplying estimated loadings (and weights in nonlinear setting) by 1-step-ahead forecast of factors($\hat{x}_{iT+1|T} = \hat{\boldsymbol{\lambda}}_i'\hat{u}_{T+1|T}$ shown by FM(u_{t+1}) , $\hat{x}_{iT+1|T} = \boldsymbol{\phi}_k^{(u^{NL})}(\boldsymbol{\phi}_j^{(u^{NL})}(\hat{u}_{T+1|T}^{(NL)}))$ shown by NLFM(u_{t+1})).

Figure IV.8.a illustrates portfolio returns during an out-of-sample period for a linear factor model FM(u_t) and a nonlinear factor model NLFM(u_t) without a forecast of factors. Also Figure IV.8.b illustrates portfolio returns during an out-of-sample period for the linear factor model FM(u_{t+1}) and the nonlinear factor model NLFM(u_{t+1}) with a forecast of factors. In both figures nonlinear factor models significantly outperform linear factor models in terms of portfolio return and Sharpe ratio. Sharpe ratio is a measure for calculating risk-adjusted return and a portfolio with a greater Sharpe ratio gives more returns for the same risk. If a portfolio with lower Sharpe ratio has returned better over a time period than another portfolio with a comparatively higher ratio, it means that the risk of losing by investing in the former fund will be higher. (see table I)

Figure IV.9 illustrates portfolio returns during an out-of-sample period for all four linear/nonlinear factor models with and without a forecast of factors. Between two nonlinear factor models, one without a forecast of factor could outperform the alternative model on the contrary, however, a linear factor model with a forecast of factor outperforms the alternative. So for the rest of the paper we keep FM(u_{t+1}) as representative of linear factor models and NLFM(u_t) as representative of nonlinear factor models. One of the preferential property that a nonlinear factor model has is its low fluctuating portfolio performance during an out-of-sample period, which is appealing for investors.

Based on DM test 55 out of 418 predictions were significantly different between FM(u_t) and FM(u_{t+1}), which in 48 cases FM(u_{t+1}) returned lower forecast RMSE. In 280 stocks FM(u_{t+1}) showed higher R^2 than the alternative model. In a nonlinear factor model 74 out of 418 prediction were significantly different which for NLFM(u_t) 55 times could bit NLFM(u_{t+1}) in terms of RMSE. NLFM(u_t) also showed in 310 stocks a higher R^2 than

simulation, including all 418 stocks and other criterion.

[2]Only the first principal component has been chosen as the predictor in the models.

(a) Linear and nonlinear factor models without a forecast of factors

(b) Linear and nonlinear factor models with a forecast of factors

Figure IV.8. Comparison of linear and nonlinear factor models based on the performance of the portfolio return during an out-of-sample period.

Table I

Portfolio	Return	Sharp Ratio
FM(u_t)	4.35%	9.1770
FM(u_{t+1})	7.51%	17.4927
NLFM(u_t)	7.87%	25.0019
NLFM(u_{t+1})	7.41%	18.8963

Figure IV.9 Comparison of linear and nonlinear factor models based on the performance of the portfolio simulation during an out-of-sample period.

the alternative model. We also used the DM test to compare nonlinear factor models with linear factor models. In 77 out of 418 the predictions were significantly different between

two models, in 61 cases of which NLFM returned a lower forecast RMSE.

Two benchmark forecasts that we compare our models with are random walk with drift (RW) and a buy-and-hold strategy in the market portfolio (S&P 500 Index). We also compare the NLFM model with AR(1) later in this section.

Figure IV.10.a illustrates linear and nonlinear factor models against a buy-and-hold investment strategy on S&P 500 index. The portfolio return for this strategy at the end of the out-of-sample period is equal to 4.68% and the corresponding Sharpe ratio is 15.0060, which means that both linear and nonlinear factor models outperform the market.

Figure IV.10.b shows linear and nonlinear factor models against Random Walk. However RW leads to a higher return (10.42%) at the end of out-of-sample period, the Sharpe ratio corresponding to this model (17.4608) is lower than nonlinear factor models.

(a) Linear and nonlinear factor models against an investment on S&P 500 index

(b) Linear and nonlinear factor model against Random walk

Figure IV.10. Comparison of linear and nonlinear factor models and the benchmark models based on the performance of the portfolio simulation.

The result shows that a nonlinear factor model significantly outperforms the linear factor model, nevertheless the result does not tell us whether the nonlinear factor estimation

caused this improvement or whether it was the nonlinear forecast equation. So we decided to also compare our linear/nonlinear factor models with two competing models; a model with nonlinear factor estimation still with a linear forecast equation, and a model with linear factor estimation benefits with a nonlinear forecast equation.

Figure IV.11 first shows that neither a model with nonlinear factor estimation nor a model with a nonlinear forecast equation could outperform a proposed nonlinear factor model, however they could outperform the linear factor model. On the other hand, the result shows that factor estimation in a nonlinear way is more important than having a nonlinear forecast equation on linear components. (see table II)

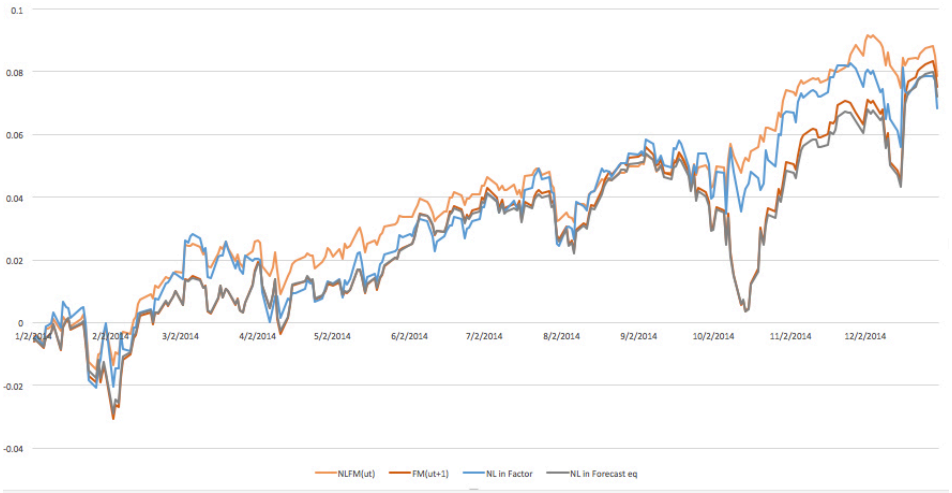


Figure IV.11. Comparison of linear and nonlinear factor models, and the models with only one nonlinear step based on the performance of the portfolio simulation

Table II

Portfolio	Return	Sharp Ratio
Linear FM	7.51%	17.4927
Nonlinear FM	7.87%	25.0019
Nonlinear in factor estimation step	7.61%	18.6259
Nonlinear in forecast equation step	6.83%	17.8577

The fraction of the variance in financial series explained by the first few components are much less than the fraction of the variance in the macroeconomic series explained by the first few linear and nonlinear components. It indicates that the idiosyncratic component is not just a noise and there is information inside idiosyncratic components that might also be relevant for financial forecasting. For this purpose, we compare the forecast accuracy of linear/nonlinear factor models considering idiosyncratic components ($\hat{x}_{iT+1|T} = \hat{\boldsymbol{\lambda}}_i' \boldsymbol{u}_{T+1|T} + \hat{\xi}_{iT+1|T}$, $\hat{x}_{iT+1|T} = \boldsymbol{\phi}_k^{(u^{NL})}(\boldsymbol{\phi}_j^{(u^{NL})}(\hat{\boldsymbol{u}}_{T+1|T}^{(NL)})) + \hat{\xi}_{iT+1|T}^{(NL)}$) to forecast return series with models ignoring the idiosyncratic component[3].

[3]We forecast the idiosyncratic component with an AR(1) process in the linear factor model and a feed-forward neural network in the nonlinear factor model

Figure IV.13 illustrates the effect of adding idiosyncratic components in forecast models. By looking at the individual RMSEs and hit rates, we will see improvements in many stocks, however, in terms of Portfolio simulation, it was more effective to improve a linear factor model. It can be interpreted that NLPCA could extract more information from data and less information remained in the idiosyncratic component. (See table III)

(a) Nonlinear Factor model

(b) Linear Factor model

Figure IV.12. Considering idiosyncratic component to forecast return series and its effect on portfolio return

Table III

Portfolio	Return	Sharp Ratio
Linear FM	7.51%	17.4927
Nonlinear FM	7.87%	25.0019
Linear FM + $\hat{\xi}_{iT+1\|T}$	7.55%	22.4096
Nonlinear FM + $\hat{\xi}_{iT+1\|T}$	7.61%	23.5331

A hybrid forecast model is also proposed in this paper to show how linear factor models can be improved. In this approach, first we estimate the linear model and we collect the residuals obtained from the fitted model. Then we let neural network to model the residuals which contain information about the nonlinearity. In theory, the hybrid model can be an effective tool with a superior forecast when both linear model and neural network model are specified well and are suboptimal. But, in practice we are combining two model specification errors and it is assumed that the time series has only a linear structure in the first step and a nonlinear structure in the second step which can be imprecise.

Figure IV.13 illustrates linear and nonlinear factor models against the hybrid model. The hybrid model outperforms linear factor model however it can not outperform nonlinear factor model in terms of Sharpe ratio. Then again, if a fund with lower Sharpe ratio has returned better over a time period than another fund with a comparatively higher ratio, it means that the risk of losing by investing in the former fund will be higher. (see table IV)

Table IV

Portfolio	Return	Sharp Ratio
Linear FM	7.51%	17.4927
Nonlinear FM	7.87%	25.0019
Hybrid model	9.32%	19.2152

Figure IV.13 Comparison of linear and nonlinear factor models and the Hybrid model based on the performance of the portfolio simulation during out-of-sample period.

To understand whether considering the covariance structure of returns improve forecast accuracy we compare linear and nonlinear univariate forecasting methods with factor models as well.

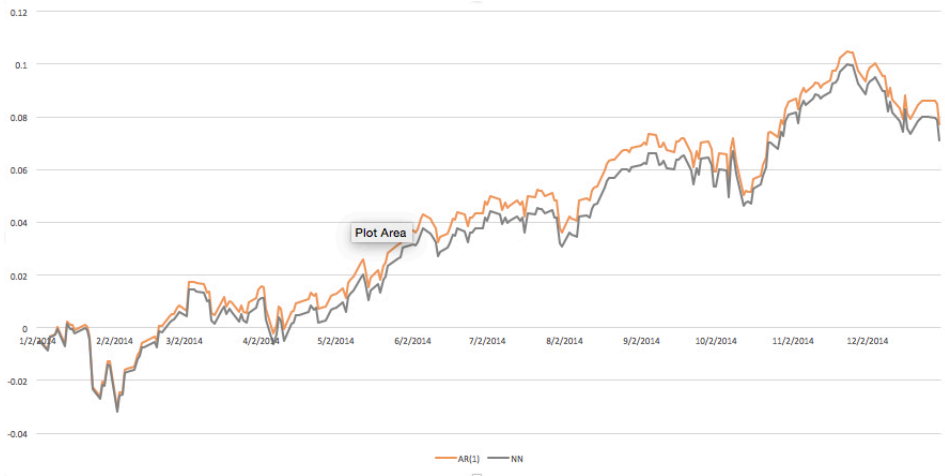
Figure IV.14.a shows how trading strategies based on AR(1) model noticeably outperform best ARMA(p,q) model[4], in terms of portfolio return during out-of-sample period. Portfolio return and Sharpe ratio calculated for the best ARMA model is %2.3 and 10.3660 against %7.7 and 18.3059 for AR(1) model. Therefore an AR(1) model is chosen as representative of linear univariate models. AR(1) model is more successful than a best ARMA model based on all other criterion we used for forecast evaluation.

Figure IV.14.b Complares AR(1) model with a feedforward neural network which we used to forecast return series based on their first lags. Portfolio return follows almost the same pattern during an out-of-sample period. Portfolio return and Sharpe ratio calculated for the neural network model is %7.3 and 18.4610. It is worth mentioning that financial

[4]ARMA model is obtained from a minimization of the penalized AICc and MLE, but it seems overfit.

series follow a different level of nonlinearity and testing for nonlinearity can help us to choose the appropriate forecast model. In our study, in %86[5] when test shows that time series follows a linear/nonlinear pattern, a linear/nonlinear model returns a better result. However figure IV.14.b illustrate portfolios constructed with linear and nonlinear models separately.

(a) AR(1) vs Best ARMA(p,q)



(b) AR(1) vs Neural Network

Figure IV.14. Comparison of linear and nonlinear univariate models based on the performance of the portfolio return during the out-of-sampleperiod.

Figure IV.15.a illustrates the linear univaritae model , AR(1), against the linear factor model. It seems as having the first lag of each stock return as the predictor returns better results in terms of portfolio simulation. However the DM test result for this two competing models show that in only 35 stocks the RMSEs are significantly different and 24 out of 35 stocks have a lower RMSE when a linear factor model is used for forecasting. Furthermore 235 out of 418 stocks have a higher R^2 when a linear factor model is used for forecasting. AR(1) outperforms FM in terms of hit rate.

[5]This number is based on linearity test results on 418 stocks and AR(1) and NN forecasts.

Figure IV.15.b illustrates a neural network as the representative of nonlinear univaritae models , against the nonlinear factor model. Here nonlinear factor models notably beat univariate model. DM test result for these two competing models shows that in only 40 stocks the RMSEs are significantly different and 36 out of 40 stocks has lower RMSE when NLFM is used. Furthermore, 310 out of 418 stocks has higher R^2 when NLFM is used. Another notable issue here is the comparison of NLFM and AR(1) which can be seen as a benchmark in financial forecasting. The NLFM model count also beat the AR(1) model in terms of portfolio simulation criterion. (See Table V)

(a) AR(1) vs Linear Factor Model

(b) Neural Network vs Nonlinear Factor model

Figure IV.15. Comparison of linear and nonlinear univariate and factor models based on the performance of the portfolio return during an out-of-sample period.

Based on the univariate model one can build a forecast model by considering both common factors and the lags of target variable as the predictors. Also macroeconomic variables can be added to the model.

Table V

Portfolio	Return	Sharp Ratio
AR(1)	7.55%	22.4096
Linear FM	7.51%	17.4927
Neural Network	7.12%	17.4610
Nonlinear FM	7.87%	25.0019

V. Conclusion

Forecasting with many predictors has received a good deal of attention in recent years. The most common approach for forecasting with many predictors is to linearly extract common factors, and use them as predictors in a linear forecast equation. Such methods are not efficient because they disregard nonlinear dynamics among predictors and between common factors and the target variable. In this paper, our focus was on answering the question whether "it is possible to forecast with a high-dimensional panel of predictors while considering nonlinear dynamic among variables?"

To answer this question, we proposed a nonlinear generalization of the statistical factor model which at the first step (factor estimation) employs an autoassociative neural network to estimate nonlinear factors from predictors and at the second step (forecasting equation) applies a nonlinear function -feedforward neural network- on estimated factors to predict the dependent variable. This model can go beyond the covariance structure analysis. The method also behave noticeably better in the empirical analysis. The empirical application to forecasting daily returns of equities on the S&P 500 index from 2005 to 2014 provide support for the out-of-sample forecasting ability of this model vis-à-vis exist competing approaches both in terms of the time series approach and the trading simulation approach. We showed how adopting statistical tests to detect nonlinearity in and between time series helps for model selection. Our empirical results encourage further research toward other possible applications of nonlinear factor models.

References

Armstrong, J.Scott, and Fred Collopy, 1992, Error measures for generalizing about forecasting methods: Empirical comparisons, *International Journal of Forecasting* 8, 69–80.

Bai, Jushan, and Serena Ng, 2002, Determining the number of factors in approximate factor models, *Econometrica* 70, 191–221.

Bai, Jushan, and Serena Ng, 2006, Evaluating latent and observed factors in macroeconomics and finance, *Journal of Econometrics* 131, 507–537.

Bai, Jushan, and Serena Ng, 2008a, Forecasting economic time series using targeted predictors, *Journal of Econometrics* 146, 304–317.

Bai, Jushan, and Serena Ng, 2008b, Large dimensional factor analysis, *Foundations and Trends®in Econometrics* 3, 89–163.

Belkin, Mikhail, and Partha Niyogi, 2003, Laplacian eigenmaps for dimensionality reduction and data representation, *Neural Computation* 15, 1373–1396.

Bishop, Christopher M., and Geoffrey Hinton, 1995, *Neural networks for pattern recognition* (Oxford University Press, New York).

Boivin, Jean, and Serena Ng, 2005, Understanding and comparing factor-based forecasts, *International Journal of Central Banking* 1:3, 117–152.

Bollerslev, Tim, Robert F Engle, and Daniel B Nelson, 1994, Arch models, *Handbook of Econometrics* Volume IV, 2961–3031.

Borg, Ingwer, and Patrick J F Groenen, 2005, Modern multidimensional scaling: Theory and applications, *Springer-Verlag, New York* 277–280.

Breiman, Leo, Jerome Friedman, and Charles J. Stone, 1984, *Classification and regression trees* (Chapman and Hall/CRC, New York, NY).

Brock, W. A., J. A. Scheinkman, W. D. Dechert, and B. LeBaron, 1996, A test for independence based on the correlation dimension, *Econometric Reviews* 15, 197–235.

Brooks, Christopher, 1996, Testing for non-linearity in daily sterling exchange rates, *Applied Financial Economics* 6, 307–317.

Bryson, A. E., W. F. Denham, and S. E. Dreyfus, 1963, Optimal programming problems with inequality constraints, *AIAA Journal* 1, 2544–2550.

Burges, Christopher J C, 2004, Geometric methods for feature extraction and dimensional reduction: A guided tour, *Technical Report* MSR-TR-2004-55, Microsoft Research.

Campbell, John Y., Andrew W Lo, and Craig A MacKinlay, 1996, *The econometrics of financial markets*, second edition (Princeton University Press, United States).

Cayton, Lawrence, 2005, Algorithms for manifold learning, *UCSD Technical Report* CS2008-0923.

Chamberlain, Gary, and Michael Rothschild, 1983, Arbitrage, factor structure, and mean-variance analysis on large asset markets, *Econometrica* 51, 1281.

Chen, Nai-Fu, Richard Roll, and Stephen A. Ross, 1986, Economic forces and the stock market, *The Journal of Business* 59, 383.

Christiansen, Bo, 2005, The shortcomings of nonlinear principal component analysis in identifying circulation regimes, *Journal of Climate* 18, 4814–4823.

Cont, Rama, 2001, Empirical properties of asset returns: Stylized facts and statistical issues, *Quantitative Finance* 1, 223–236.

Cox, Michael A. A., and Trevor F. Cox, 2001, Multidimensional scaling, *Springer Handbooks Comp.Statistics* 315–347.

Cunningham, John P, and Zoubin Ghahramani, 2015, Linear dimensionality reduction: Survey, insights, and generalizations, *Journal of Machine Learning Research* 16, 2859–2900.

Cybenko, George, 1989, Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals, and Systems* 2, 303–314.

Deistler, Manfred, and Eva Hamman, 2005, Identification of factor models for forecasting returns, *Journal of Financial Econometrics* 3, 256–281.

DeMiguel, Victor, Lorenzo Garlappi, and Raman Uppal, 2009, Optimal versus naive diversification: How inefficient is the 1/ n portfolio strategy?, *Review of Financial Studies* 22, 1915–1953.

Diebold, Francis X., and Roberto S. Mariano, 1995, Comparing predictive accuracy, *Journal of Business & Economic Statistics* 13, 253.

Eckart, Carl, and Gale Young, 1936, The approximation of one matrix by another of lower rank, *Psychometrika* 1, 211–218.

Engle, Robert, and Riccardo Colacito, 2006, Testing and valuing dynamic correlations for asset allocation, *Journal of Business & Economic Statistics* 24, 238–253.

Engle, Robert F., 1982, Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation, *Econometrica* 50, 987.

Exterkate, Peter, Patrick J. F. Groenen, Christiaan Heij, and Dick J. C. Van Dijk, 2013, Nonlinear forecasting with many predictors using kernel ridge regression, *CREATES Aarhus University and Erasmus University* Working paper.

Fama, Eugene F, and Kenneth R French, 2004, The capital asset pricing model: Theory and evidence, *Journal of Economic Perspectives* 18, 25–46.

Fenn, Daniel J., Mason A. Porter, Stacy Williams, Mark McDonald, Neil F. Johnson, and Nick S. Jones, 2011, Temporal evolution of financial-market correlations, *Physical Review E* 84.

Fisher, Ronald Aylmer, 1936, The use of multiple measurements in taxonomic problems, *Annals of Eugenics* 7, 179–188.

Fodor, Imola K, 2002, A survey of dimension reduction techniques, *Technical Report UCRL-ID-148494, Lawrence Livermore National Laboratory* .

Forni, Mario, Marc Hallin, Marco Lippi, and Lucrezia Reichlin, 2000, The generalized dynamic-factor model: Identification and estimation, *Review of Economics and Statistics* 82, 540–554.

Forni, Mario, Marc Hallin, Marco Lippi, and Lucrezia Reichlin, 2001, Coincident and leading indicators for the euro area, *Economic Journal* 111, 82–5.

Forni, Mario, Marc Hallin, Marco Lippi, and Lucrezia Reichlin, 2005, The generalized dynamic factor model, *Journal of the American Statistical Association* 100, 830–840.

Forni, Mario, Marc Hallin, Marco Lippi, and Paulo Zaffaroni, 2014, The generalized dynamic factor model, *Journal of Econometrics* forthcoming.

Forni, Mario, and Marco Lippi, 2001, The generalized dynamic factor model : representation theory, *Econometric Theory* 17, 1113–1141.

Forni, Mario, and Lucrezia Reichlin, 2001, Federal policies and local economies: Europe and the us, *European Economic Review* 45, 109–134.

Fort, J.C., 2006, Som's mathematics, *Neural Networks* 19, 812–816.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani, 2009, *The elements of statistical learning: Data mining, inference, and prediction, Second edition - 2nd edition*, second edition (Springer-Verlag New York, New York, NY).

Geweke, John, 1977, The dynamic factor analysis of economic time series, *Latent Variables in Socio-Economic Models* Amsterdam: North-Holland.

Giannerini, Simone, 2012, The quest for nonlinearity in time series, *Time Series Analysis: Methods and Applications* 43–63.

Giannerini, Simone, Esfandiar Maasoumi, and Estela Bee Dagum, 2015, Entropy testing for nonlinear serial dependence in time series, *Biometrika* 102, 661–675.

Giovannetti, Bruno Cara, 2011, Nonlinear forecasting using factor-augmented models, *Journal of Forecasting* 32, 32–40.

Granger, Clive W. J., and M. Hashem Pesaran, 2000, Economic and statistical measures of forecast accuracy, *Journal of Forecasting* 19, 537–560.

Granger, Clive W J, and Timo Teräsvirta, 1993, *Modelling non-linear economic relationships* (Oxford University Press, Oxford, United Kingdom).

Harvey, David, Stephen Leybourne, and Paul Newbold, 1997, Testing the equality of prediction mean squared errors, *International Journal of Forecasting* 13, 281–291.

Hinton, Geoffrey, and Ruslan Salakhutdinov, 2006, Reducing the dimensionality of data with neural networks, *Science* 313, 504–507.

Hoerl, Arthur E., and Robert W. Kennard, 1970, Ridge regression: Biased estimation for nonorthogonal problems, *Technometrics* 12, 55–67.

Hornik, Kurt, 1991, Approximation capabilities of multilayer feedforward networks, *Neural Networks* 4, 251–257.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White, 1989, Multilayer feedforward networks are universal approximators, *Neural Networks* 2, 359–366.

Hotelling, Harold, 1936, Relations between two sets of variates, *Biometrika* 28, 321.

Hsieh, David A., 1991, Chaos and nonlinear dynamics: Application to financial markets, *The Journal of Finance* 46, 1839–1877.

Hsieh, William W., 2004, Nonlinear multivariate and time series analysis by neural network methods, *Reviews of Geophysics* 42.

Jäkel, Frank, Bernhard Schölkopf, and Felix A. Wichmann, 2007, A tutorial on kernel methods for categorization, *Journal of Mathematical Psychology* 51, 343–358.

Jolliffe, Ian T, 2016, *Principal component analysis*, second edition (Springer-Verlag, New York).

Keenan, Daniel MacRae, 1985, A tukey nonadditivity-type test for time series nonlinearity, *Biometrika* 72, 39.

Kohonen, Teuvo, 2001, *Self-organizing maps* (Springer-Verlag Berlin Heidelberg).

Kramer, Mark A., 1991, Nonlinear principal component analysis using autoassociative neural networks, *AIChE Journal* 37, 233–243.

Kruger, Uwe, Junping Zhang, and Lei Xie, 2008, Developments and applications of nonlinear principal component analysis – a review, in *Principal manifolds for data visualization and dimension reduction* (Springer-Verlag Berlin and Heidelberg GmbH & Co. K, Berlin).

Kuan, Chung-Ming, and Halbert White, 1994, Reply to comments on "artificial neural networks: An econometric perspective", *Econometric Reviews* 13, 139–143.

Kulis, Brian, 2012, Metric learning: A survey, *Foundations and Trends in Machine Learning* 5, 287–364.

Lam, Clifford, and Qiwei Yao, 2012, Factor modeling for high-dimensional time series: Inference for the number of factors, *The Annals of Statistics* 40, 694–726.

Levenberg, Kenneth, 1944, A method for the solution of certain non-linear problems in least squares, *Quarterly of Applied Mathematics* 2, 164–168.

Li, Wai Keung, 2003, *Diagnostic checks in time series* (Chapman & Hall, London).

Lintner, John, 1965, The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets, *The Review of Economics and Statistics* 47, 13.

Liu, Yang, and Rong Jin, 2006, Distance metric learning: A comprehensive survey, *Technical report, Department of Computer Science and Engineering, Michigan State University,* .

Ljung, G. M., and G. E. P. Box, 1978, On a measure of lack of fit in time series models, *Biometrika* 65, 297–303.

Ludvigson, Sydney C., and Serena Ng, 2009, Macro factors in bond risk premia, *Review of Financial Studies* 22, 5027–5067.

Marcellino, Massimiliano, James H Stock, and Mark W Watson, 2003, Macroeconomic forecasting in the euro area: Country specific versus area-wide information, *European Economic Review* 47, 1–18.

Marcellino, Massimiliano G, James H. Stock, and Mark W. Watson, 2005, A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series, *SSRN Electronic Journal* .

Marquardt, Donald W., 1963, An algorithm for least-squares estimation of nonlinear parameters, *Journal of the Society for Industrial and Applied Mathematics* 11, 431–441.

Maxwell, Albert Ernest, and D.N. Lawley, 1971, *Factor analysis as a statistical method*, second edition (Butterworth & Co Publishers, London).

McLeod, A. I., and W. K. Li, 1983, Diagnostic checking arma time series models using squared-residual autocorrelations, *Journal of Time Series Analysis* 4, 269–273.

Mendel, J.M., 1991, Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications, *Proceedings of the IEEE* 79, 278–305.

Mol, Christine De, Domenico Giannone, and Lucrezia Reichlin, 2008, Forecasting using a large number of predictors: Is bayesian shrinkage a valid alternative to principal components?, *Journal of Econometrics* 146, 318–328.

Monahan, Adam Hugh, 2001, Nonlinear principal component analysis: Tropical indo–pacific sea surface temperature and sea level pressure, *Journal of Climate* 14, 219–233.

Morgan, James N., and John A. Sonquist, 1963, Problems in the analysis of survey data, and a proposal, *Journal of the American Statistical Association* 58, 415.

Nikias, Chrysostomos L, and Athina P Petropulu, 1993, *Higher order spectra analysis: A nonlinear signal processing framework* (PTR Prentice Hall, New York, NY, United States).

Oja, Erkki, 1982, Simplified neuron model as a principal component analyzer, *Journal of Mathematical Biology* 15, 267–273.

Pearson, Karl, 1901, Liii. on lines and planes of closest fit to systems of points in space, *Philosophical Magazine Series 6* 2, 559–572.

Pesaran, M. Hashem, and Allan Timmermann, 1995, Predictability of stock returns: Robustness and economic significance, *The Journal of Finance* 50, 1201.

Pesaran, M. Hashem, and Allan Timmermann, 2000, A recursive modelling approach to predicting uk stock returns, *The Economic Journal* 110, 159–191.

Petropulu, Athina, 1999, Higher-order spectral analysis, *Electrical Engineering Handbook* .

Ramsey, J B, 1969, Tests for specification errors in classical linear least-squares regression analysis, *Journal of the Royal Statistical Society. Series B (Methodological)* 31, 350–371.

Rao, Radhakrishna C, 1948, The utilization of multiple measurements in problems of biological classification, *Journal of the Royal Statistical Society. Series B (Methodological)* 10, 159–203.

Raviv, Eran, and Dick J. C. Van Dijk, 2014, Forecasting with many predictors: Allowing for non-linearity, *Available at SSRN: https://ssrn.com/abstract=2565288* .

Ross, Stephen A, 1976, The arbitrage theory of capital asset pricing, *Journal of Economic Theory* 13, 341–360.

Roweis, Sam T, and Lawrence K Saul, 2000, Nonlinear dimensionality reduction by locally linear embedding, *Science* 290, 2323–2326.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams, 1986, Learning representations by back-propagating errors, *Nature* 323, 533–536.

Sargent, Thomas J, and Cristopher A Sims, 1977, Business cycle modeling without pretending to have too much a-priori economic theory, *New Methods in Business Cycle Research* Federal Reserve Bank of Minneapolis.

Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller, 1997, Kernel principal component analysis, *Artificial Neural Networks — ICANN'97* 1327, 583–588.

Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller, 1998, Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation* 10, 1299–1319.

Scholz, Matthias, Martin Fraunholz, and Joachim Selbig, 2008, Nonlinear principal component analysis: Neural network models and applications, in *Principal Manifolds for Data Visualization and Dimension Reduction*, 44–67 (Springer Science, Berlin Heidelberg).

Schreiber, Thomas, and Andreas Schmitz, 1996, Improved surrogate data for nonlinearity tests, *Physical Review Letters* 77, 635–638.

Schreiber, Thomas, and Andreas Schmitz, 1997, Discrimination power of measures for nonlinearity in a time series, *Physical Review E* 55, 5443–5447.

Schreiber, Thomas, and Andreas Schmitz, 2000, Surrogate time series, *Physica D: Nonlinear Phenomena* 142, 346–382.

Sharpe, William F., 1964, Capital asset prices: A theory of market equilibrium under conditions of risk, *The Journal of Finance* 19, 425.

Spearman, Charles, 1904, General intelligence objectively determined and measured, *The American Journal of Psychology* 15, 201–292.

Stock, James H, and Mark W Watson, 1999, Forecasting inflation, *Journal of Monetary Economics* 44, 293–335.

Stock, James H, and Mark W Watson, 2002a, Forecasting using principal components from a large number of predictors, *Journal of the American Statistical Association* 97, 1167–1179.

Stock, James H, and Mark W Watson, 2002b, Macroeconomic forecasting using diffusion indexes, *Journal of Business & Economic Statistics* 20, 147–162.

Stock, James H., and Mark W. Watson, 2006, Chapter 10 forecasting with many predictors, *Handbook of Economic Forecasting* 515–554.

Tenenbaum, Joshua, vin de Silva, and john Langford, 2000, A global geometric framework for nonlinear dimensionality reduction, *Science* 290, 2319–2323.

Teräsvirta, Timo, Dag Tjostheim, and Clive W J Granger, 2010, *Modelling nonlinear economic time series* (Oxford University Press, Oxford).

Teräsvirta, Timo, Dick van Dijk, and Marcelo C. Medeiros, 2005, Linear models, smooth transition autoregressions, and neural networks for forecasting macroeconomic time series: A re-examination, *International Journal of Forecasting* 21, 755–774.

Teräsvirta, Timo, Dag Tjøstheim, and Clive W. J. Granger, 2010, Modelling nonlinear economic time series .

Theiler, James, Stephen Eubank, André Longtin, Bryan Galdrikian, and Doyne J Farmer, 1992, Testing for nonlinearity in time series: The method of surrogate data, *Physica D: Nonlinear Phenomena* 58, 77–94.

Tibshirani, Robert, 1996, Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 58, 267–288.

Tong, H, and K S Lim, 1980, Threshold autoregression, limit cycles and cyclical data, *Journal of the Royal Statistical Society. Series BMethodological)* 42, 245–292.

Tong, Howell, 1978, On a threshold model, in C Chen, ed., *Pattern recognition and signal processing*, 575–586 (Sijthoff & Noordhoff, Netherlands).

Tong, Howell, 1983, *Threshold models in non-linear time series analysis* (Springer-Verlag New York, New York, Berlin, Heidelberg [usw.]).

Tong, Howell, 1990, *Non-linear time series: A dynamical system approach* (Clarendon Press, Oxford).

Torgerson, Warren S., 1952, Multidimensional scaling: I. theory and method, *Psychometrika* 17, 401–419.

Tsay, Ruey S., 1986, Nonlinearity tests for time series, *Biometrika* 73, 461.

Tsay, Ruey S, 2005, *Analysis of Financial Time Series*, third edition (Wiley).

Vapnik, Vladimir, 1982, *Estimation of Dependences based on empirical data: Springer series in statistics* (Springer-Verlag New York, New York, Heidelberg, Berlin).

Vapnik, Vladimir N, 1995, *The nature of statistical learning theory*, second edition (Springer-Verlag New York, New York).

Vapnik, Vladimir N, and Alexey Ya Chervonenkis, 1964, A note on one class of p erceptrons, *Automation and Remote Control* 25.

Vapnik, Vladimir N, and Alexey Ya Chervonenkis, 1974, Theory of pattern recognition: Statistical problems of learning ⊠⊠in russian⊠⊠⊠, *Nauka*⊠⊠ *Moscow* English translation: Academic, New York.

Varian, Hal R., 2014, Big data: New tricks for econometrics †, *Journal of Economic Perspectives* 28, 3–28.

Weinberger, Kilian Q, and Lawrence K Saul, 2006, An introduction to nonlinear dimensionality reduction by maximum variance unfolding, *National Conference on Artificial Intelligence (AAAI), Nectar paper, Boston MA* .

Werbos, Paul J., 1988, Generalization of backpropagation with application to a recurrent gas market model, *Neural Networks* 1, 339–356.

White, Halbert, 1990, Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings, *Neural Networks* 3, 535–549.

Zou, Hui, and Trevor Hastie, 2005, Regularization and variable selection via the elastic net, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 301–320.