JIRSS (2013)
Vol. 12, No. 2, pp 293-319

Simulation and Prediction of Wind Speeds: A Neural Network for Weibull

Stefan Markus Giebel[1], Martin Rainer[2], Nadi Serhan Aydın[3]

[1]University of Luxembourg, Luxembourg, and Johannes Kepler University, Linz, Austria.
[2]ENAMEC Institut and Risk Management Research Center, Univ. Würzburg, Germany, and Institute of Applied Mathematics, METU, Ankara, Turkey.
[3]Institute of Applied Mathematics, METU, Ankara, Turkey, and SESRIC, OIC Center, Ankara, Turkey.

Abstract. Wind as a resource of renewable energy has obtained an important share of the energy market already. Therefore simulation and prediction of wind speeds is essential for both, for engineers and energy traders.

In this paper we analyze the surface wind speed data from three prototypic locations: coastal region (Rotterdam), undulating forest landscape few 100 m above sea level(Kassel), and alpine mountains about 3000 m above sea level (Zugspitze).

Rather than matching the conventional Weibull distribution to the wind speed data, we investigate two alternative models for wind speed prediction, both being refinements of a log-normal model, but with very different approaches and capability for capturing the extremal events.

In both models deterministic effects such as trend and seasonality are separated. The first (structural stochastic) model predicts wind speeds exponentially from a linear combination of separate mean-reverting jump processes for the high and low wind speed regimes, and the regular (diffusive) wind speed regime. The second (neuro-stochastic) model is a prediction with volatility-enhanced trend, with parameters dynamically

Stefan Markus Giebel(✉)(stefan.giebel@enamec.de), Martin Rainer(martin.rainer @enamec.de), Nadi Serhan Aydın(serhan.aydin@metu.edu.tr)
Received: September 2012; Accepted: June 2013

learned by the middle-layer neurons of an MLP-type neural network operating on dynamically updated and re-weighted history.

The numerical results suggest that, for a coastal region (e.g. Rotterdam) the R^2-determination is higher, while for the undulating forest regions (e.g. Kassel) and even more the higher mountain regions (e.g. Zugspitze) the structural stochastic model yields higher determination.

The neuro-stochastic algorithm opens a new path within statistical learning: feature space and kernel functions are completely defined by the parameters of the stochastic process.

Keywords. Alternative energy, jump-diffusion processes, neural network, Weibul distribution.

MSC: 60J75, 68T01, 92B20, 93E35.

1 Introduction

For many natural energy resources, particularly renewable ones such as wind or solar radiation, their availability is subject to stochastic fluctuations. The reason behind, in this case usually are physical processes which are themselves best described only by stochastic models. E.g. the stochastic fluctuation of wind speed is important, both, as a challenge for engineering and operation of wind turbines and as an economic risk factor for the operator which has direct impact on the cost of production.

How much of each stochastic component can be viewed as a kind of macro-description of some more underlying non-linear chaotic dynamics, is an open question beyond our current investigation.

It is common practice to fit (surface) wind speeds by a particular extremal value distribution, namely a Weibull distribution with shape not too far away from a Rayleigh distribution (shape parameter $b = 2$). Notably, the core of such a distribution is similar to a log-normal distribution. Therefore we investigate two alternative models for wind speed prediction, both being refinements of a log-normal model.

In both models deterministic effects such as trend and seasonality are separated in advance.

The first model is purely structural and based on stochastic jump and diffusion processes. It predicts wind speeds exponentially from a linear combination of separate Ornstein Uhlenbeck (OU) processes, namely mean-reverting jump processes for the high and low wind speed regimes, and for the regular (diffusive) wind speed regime. All parameters of all the processes are calibrated by convention nonlinear regression methods.

The second model is based on a stochastic model with a volatility-enhanced trend (corresponding to a transformed normal prediction) combined with a supervised neural net predicting its parameters dynamically. Considering in our case a target, unsupervised neural nets like in [2] are not of interest. We consider a multi-layer perceptron (MLP) network with just one intermediate layer. The neurons of latter are learning the parameters of the stochastic process from a flexible weighted history of historical wind speeds. The update rule applied is daily on-line.

Extremely high or low wind speeds are modeled geometrically as exponential transforms of jump processes. The latter are like the spikes in the additive model of [3]. Hence, even more than the diffusive core process, the jump processes are subject to mean-reversion.

In the first model the parameter calibration for the stochastic processes employs usually a version of maximum likelihood estimation or least square nonlinear regression methods. These methods are neglecting the fact, that not all historical data points should be weighted equally, but rather different weights should be given to different points of the history, due to their different importance, due to structures (e.g. Elliott waves) of the time series, and even more due the different strength of memory impact, roughly decaying with the size of the time lag.

For the standard nonlinear regression methods based on the Levenberg-Marquardt method [24], [26] there exists a well-elaborated theory, including proofs of convergence in sufficiently well-behaved local regions of the parameter space. Nevertheless, in larger regions of the parameter they usually fail to detect the correct minimum among several local ones. Therefore, simulated annealing [7] or adaptive lattice algorithms [29] have been proposed as alternatives for global calibration of nonlinear functions.

As we will see below, learning neural perceptron layers may provide a method to calibrate stochastic asset models with more realistic and dynamical weights on the historical input data.

Neural networks as mathematical methods have been developed in particular in the context of pattern recognition (see e.g. [5]). their applications in this context have a wide range including such different topics like corporate decision planning and business processes [22], recruiting [23], criminal profiling [31], [10], energy consumption [4], electronic noses for odour detection and classification [19][11], oncology [1], tumor shape analysis [12], [14], weather [25] and many more. For application of mathematical models on real data, the general remarks in [8] should be considered. Also in mathematical finance, neural networks of multilayer perceptron (MLP) type have been investigated as a serious alternative to conventional statistical estimation (see e.g. [28]).

In [13] a new synthesis of neural networks and stochastic processes for the purpose of forecasting was proposed. In [16] a method was introduced to find (by a stepwise procedure) an optimal length of memory. The new neural network calibration was demonstrated to provide a serious alternative to conventional nonlinear calibration of stochastic processes.

Below we apply a neural network calibrated stochastic model in the spirit of [13] to simulate and predict daily wind speeds.

We proceed as follows. First, we review the stochastic processes and calibration methods, applied for our first stochastic wind speed model.

Then we review the MLP neural network methodology.

Following the approach of [13] we then combine stochastic process and neural network, calibrating the parameters of our second stochastic wind speed prediction. This approach can be an advancement to a pure neural network approach like in [32], [18] or in [17].

2 Framework for the Stochastic Model

In the following we present a general modeling framework for stochastic modeling of time series data from regular observations of some process in environment, climate and energy.

2.1 Independent Increment RCLL Semimartingale Drivers

In this section we review the general setting for the underlying stochastic processes.

A semimartingale consists from two terms, one from the finite variation class FV and one from the local martingale class M_{loc}. Similar as proposed in Benth *et al.* (2008), we consider a special type of semimartingale independent increment RCLL (cadlag) process I, with R^n-valued random paths $t \mapsto I(t)$, with Itô representation according to [20]

$$\begin{aligned} I(t) &= \gamma(t) + M(t) + \int_0^t \int_{|z|<1} z\tilde{N}(ds,dz) + \int_0^t \int_{|z|\geq 1} zN(ds,dz) \\ &= \underbrace{\gamma(t) + \sum_{s\leq t} \Delta I(s) 1_{|\Delta I(s)|\geq 1}}_{\in FV} + \underbrace{M(t) + \int_0^t \int_{|z|<1} z\tilde{N}(ds,dz)}_{\in M_{loc}} \qquad (1) \end{aligned}$$

where γ is of finite variation on finite intervals, M is a local continuous martingale with finite quadratic variation C, $N(\cdot)$ is a random jump measure and $\tilde{N}(\cdot) := N(\cdot) - E[N(\cdot)]$ is the compensated random jump measure. In the special case when I is stationary, the compensator $\ell(dt, dz) := E[N(dt, dz)]$ factorizes,

$$\ell(dt, dz) \quad = \quad dt\ell(dz) \tag{2}$$

Here I is a Lévy process with Lévy measure $\ell(\cdot)$ and characteristic triplet $(\gamma, C, \ell(\cdot))$. In order to admit a time-dependent frequency of jumps, one also has to consider the more general case

$$\ell(dt, dz) \quad = \quad dtg(t)\ell(dz) \tag{3}$$

where g is a continuous, strictly positive function. Here $t \mapsto \ell_t(\cdot) := g(t)\ell(\cdot)$ defines a time-dependent, continuous functional into a set of Lévy measures being equivalent to $\ell(\cdot)$ modulo a positive scale-factor. In this case, I is a non-stationary semimartingale.

The class $\mathcal{I}$ of infinitely divisible laws has two important subclasses, the class $\mathcal{SD}$ of self-decomposable laws, and the class of α-stable laws $\mathcal{S}_\alpha$, $\alpha \in]0, 2]$,

$$\mathcal{S}_\alpha \subset \mathcal{SD} \subset \mathcal{I} \tag{4}$$

If increments of I are distributed in a self-decomposable manner, then

$$\ell(dz) \quad = \quad \frac{k(z)}{|z|}dz \tag{5}$$

with a function k increasing on $]-\infty, 0]$ and decreasing on $[0, \infty[$. If the distribution of increments of I is α-stable, then

$$\ell(dz) \quad = \quad \frac{1}{|z|^{1+\alpha}}\left(c_+\mathbf{1}_{z>0} + c_-\mathbf{1}_{z<0}\right)dz \tag{6}$$

i.e. the function k of the self-decomposable measure represents a power law decay,

$$k(z) \quad = \quad |z|^{-\alpha}\left(c_+\mathbf{1}_{z>0} + c_-\mathbf{1}_{z<0}\right) \tag{7}$$

The asymmetry parameter of this stable distribution is

$$\beta \quad = \quad \frac{c_+ - c_-}{c_+ + c_-} \tag{8}$$

Within the full dynamical model, the process I is assigned the role of an ingredient stochastic driver. While I is assumed to have independent increments, the stochastic processes S *may* have correlated increments.

For wind speeds particularly, two particular types of stochastic drivers are investigated, and also applied in practice.

2.1.1 Jump-Diffusion Processes

In this case, we will consider the following stochastic semimartingale drivers I:

Let $I_0(t)$ be a continuous jump-diffusion with increments $dI_0 = \sigma(t)(dW + dL)$, where W be a standard Brownian motion, and L be a compound Poisson process with Lévy measure $\ell(dz) = \lambda_0 F_Z(dz)$ with jump-intensity λ_0, and a jump-size distribution F_Z with non-trivial support on R.

Further, let $I_\pm(t)$ be two (not necessarily time-homogeneous) compound Poisson processes with increments given respectively by $J^+(t)dN_t^+$ for the positive jumps (i.e. $J^+(t)$ having support on R^+ only), and $J^-(t)dN_t^-$ for the negative jumps (i.e. $J^-(t)$ having support on R^- only).

We assume that a common seasonality $\eta(t)$ for positive and negative jump sizes can be factorized as

$$J^\pm(t) = \eta(t)J^\pm \tag{9}$$

For the deseasonalized jump sizes $J^\pm$, let us assume a generalized gamma distribution:

$$F_{J,b,p}{}^\pm := \frac{b^\pm}{\mu_J^\pm \Gamma\left(\frac{b}{p}\right)} \int_0^z \left(\frac{z}{\mu_J^\pm}\right)^{b^\pm - 1} e^{-(\frac{z}{\mu_J^\pm})^{p^\pm}} dz \tag{10}$$

As important special cases, this includes the gamma distribution for $p = 1$, the Weibull distributions for $p = b$, and the log-normal distributions in a limit $\frac{b}{p} \to \infty$. The Weibull distribution reads

$$F_{J,b}{}^\pm := \frac{b^\pm}{\mu_J^\pm} \int_0^z \left(\frac{z}{\mu_J^\pm}\right)^{b^\pm - 1} e^{-(\frac{z}{\mu_J^\pm})^{b^\pm}} dz \tag{11}$$

Moreover, for $b = 1$ it reduces to the common exponential distribution, which is our particular choice for this study. On the other hand, for $b > 1$ the probability density vanishes for vanishing jump sizes. Parameter b is sometimes called the Weibull modul. If small jumps are very rare, a Weibull module of $b > 1$ may be desirable. In this case, a Weibull or a generalized gamma distribution provide an alternative to the log-normal distribution. This is actually the case for wind speeds, where the shape parameter under normal conditions differs only little from $b = 2$ (the Rayleigh distribution).

Seasonality of the jump frequency could, eg, be modeled as

$$\lambda(t) \quad := \quad \lambda_{max} \left(\frac{2}{1 + |\sin(\pi(t-\tau)/k)|} - 1 \right)^{d} \tag{12}$$

for positive/negative jumps each according to [9]. In practice, however, to model seasonality makes only sense if the frequency of jumps suffices to detect the seasonality, if any, from the data.

The compensator measure of the expected jumps reads

$$\ell(dt, dz)^{\pm} \quad = \quad dt\lambda^{\pm}(t) \frac{b^{\pm}}{\mu_J^{\pm}} \left(\frac{z}{\mu_J^{\pm}} \right)^{b^{\pm}-1} e^{-(\frac{z}{\mu_J^{\pm}})} dz \tag{13}$$

Below the continuous diffusion as well as each compound Poisson process will drive its own autoregressive process. The latter will be denoted by $Y^{\pm}$ for the positive and negative jumps, whereas the former by X.

2.1.2 Stable Lévy Processes

In this case, stochastic drivers are given by stable Lévy processes $I := Z_{\alpha}$.

After normalization w.r.t. location μ and scale c, such a process is characterized by its stability index $\alpha \in]0, 2]$. The weighted sum of n combined increments satisfies

$$w_1 dI_1 + \ldots + w_n dI_n \quad = \quad w dI \tag{14}$$

where $w = |\mathbf{w}|_{\alpha}$, $\mathbf{w} = (w_1, \ldots, m_n)^t$. A symmetric stable Lévy process is self-affine, with fractional Hurst scale $H = \alpha^{-1}$. The class of stable probability distributions for increments of stable processes forms a subclass of the self-decomposable probability distributions, which themselves are a subclass of the infinitely divisible distributions.

For $\alpha = 2$, the process is normally distributed, i.e. it is equivalent to a Brownian diffusion with volatility $\sigma = \sqrt{2}c$.

For $\alpha < 2$, the variance (i.e. the second moment) is infinite. The width of the distribution then is described by the scale c rather than the sample volatility.

For $\alpha \leq 1$, the mean (i.e. the first moment) is infinite. The location μ of the distribution is rather determined by the median than the sample mean.

For $\alpha > 1$, from a Lévy process Z_{α} one could construct stationary processes with memory, as certain integrals of Z_{α}, with H-dependent integrand, where $\alpha > \frac{1}{H} > 1$. Corresponding process increments are no longer independent, and the processes themselves are no longer semimartingales. Therefore, below we will use some semimartingale I with independent increments as an input driver of an autoregressive processes exhibiting memory. This approach has the advantage that semimartingale methods will suffice to handle the stochastic process.

2.2 CAR Processes with Semimartingale Drivers

Given a semimartingale driver I having independent increments, a CAR(n) process $\mathbf{X}$ in R^n is defined as

$$\begin{aligned} dX_q &= X_{q+1}\,dt \quad q = 1,\dots,n-1 \\ dX_n &= -\left(\sum_{q=1}^{n} \alpha_q(t) X_q\right) dt + \sigma(t) dI(t) \end{aligned} \tag{15}$$

which can be written in a compact matrix form as

$$d\mathbf{X} = \mathbf{A}(t)\mathbf{X}dt + \sigma(t)\mathbf{e}_n dI(t) \tag{16}$$

with $n \times n$ matrix

$$A(t) := \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & . & \vdots \\ 0 & 0 & \cdots & \cdots & 1 \\ -\alpha_1(t) & -\alpha_2(t) & \cdots & \cdots & -\alpha_n(t) \end{pmatrix} \tag{17}$$

The CAR(n) case above is still a special case of a more general CARMA (n, m) process, which can be defined generally similar as in [6]. There it was introduced for the case of I being a second-order Lévy processes with $E[L(1)^2] < \infty$.

Nevertheless, here we restrict just to a very simple CAR process. Its component X_1 exhibits all autoregressive features.

Below we will use the component X_1 of the CAR processes, defined as

$$dX_q = X_{q+1}\,dt \quad q = 1,\dots,n-1 \tag{18}$$

$$dX_n = -\left(\sum_{q=1}^{n} \alpha_q(t) X_q\right) dt + \sigma(t) dI(t) \tag{19}$$

with given semimartingale driver I. Discretization of the CAR(n) process X with $dt_i := t_{i+1} - t_i$ yields

$$
\begin{aligned}
X_{q+1}(t_i) &= \frac{X_q(t_{i+1}) - X_q(t_i)}{dt_i} \quad q = 1, \dots, n-1, && (20)\\
X_n(t_{i+1}) &- \left(\left(\frac{1}{dt_i} - \alpha_q(t_i) \right) X_n(t_i) - \sum_{q=1}^{n-1} \alpha_q(t_i) X_q(t_i) \right) dt_i \\
&= \sigma(t_i)\varepsilon_i && (21)
\end{aligned}
$$

where ε_i is a random number distributed according to the pdf of $dI(t_i)$. According to (21) the coefficient functions $\alpha_q(t)$, $q = 1, \dots, n$, can be determined by regression. For time-independent constants α_q, the regression becomes linear. Recursion of (20), inserting into (21), and then resolving for X_1 show that the discretized CAR(n) process is in fact an AR(n) process,

$$
X_1(t_{n+1}) = \sum_{q=1}^{n} \gamma_q \, X_1(t_q) \qquad (22)
$$

with parameters γ_q depending linearly on the original reversion coefficients α_q.

The assumption that there is no autoregression beyond mean reversion leads to our special case where X is CAR(1), a mean-reverting OU process

$$
dX = -\alpha(t) X dt + \sigma(t) dI(t) \quad . \qquad (23)
$$

Such an OU process is known to have the unique strong solution

$$
X(t) = e^{-\int_0^t \alpha(s)ds} \left[X_0 + \int_0^t \sigma(u) e^{\int_0^u \alpha(s)ds} dI(u) \right] \qquad (24)
$$

In the special case where X is CAR(1), it is a mean reverting Ornstein-Uhlenbeck (OU) process

$$
dX = -\alpha(t) X dt + \sigma(t) dI(t) \quad . \qquad (25)
$$

Such an OU process is known to have the unique strong solution

$$
X(t) = e^{-\int_0^t \alpha(s)ds} \left[X_0 + \int_0^t \sigma(u) e^{\int_0^u \alpha ds} dI(u) \right] \quad . \qquad (26)
$$

The general Ito-formula for semimartingales, can be applied to $S = f(X)$, in order to yield an explicit form of the dynamics of $S(t)$.

2.3 Transformation of the Stochastic Process

The process S in R here will be modeled on the basis of the first component X_1 of the CAR process X as stochastic input.

Apart from the trivial transformation, typically, non-linear transformations $X_1 \to S$ are used in order to model the dynamical process S.

We will use the generalized additive representation,

$$S(t, X_1(t)) = g(\Lambda(t) + X_1(t)) \tag{27}$$

where g is assumed to be invertible and at least twice differentiable (i.e. C^2). Let us consider the following special case:

Geometric Model

$g(x) = e^x$. This model admits only positive wind speeds. It introduces additional nonlinearity to the wind speed S.

With any of these models, the wind speed S can be modeled as

$$\begin{aligned} S(t, X(t)) &= g\left(\Lambda(t) + X_1(t)\right) \\ dX_q &= X_{q+1}(t)\ dt \quad q = 1, \ldots, n-1 \\ dX_n &= -\left(\sum_{q=1}^{n} \alpha_q(t) X_q(t)\right) dt + \sigma(t) dI(t) \end{aligned}$$

for given semimartingale driver I.

The general Itô formula for semimartingales can be applied to $S = g(X)$ in order to yield an explicit form of the dynamics of $S(t)$.

3 A Multi-OU Model Describing for Spikes

For the numerical analysis below, we propose a spike-jump-diffusion model which is composed of multiple OU processes with independent semimartingale drivers, namely

$$\begin{aligned} S(t) &= g\left(\Lambda(t) + X(t) + Y^+(t) + Y^-(t)\right) \\ dX &= -\alpha X dt + \sigma(t)(dW + dL) \\ dY^\pm &= -\beta^\pm Y^\pm dt \pm \eta(t) J^\pm dN^\pm(t) \end{aligned} \tag{28}$$

where W is a continuous Brownian motion and L a pure jump process, e.g. a compound Poisson process. $J^\pm$ is the random variable for the heights of positive and negative spiky jumps, $N^{\pm}$ are the Poisson subordinators with spike intensity $\lambda^{\pm}(t)$, which is assumed to be a continuous function of time t.

The distributions of positive and negative spiky jump heights are modeled commonly by two symmetric exponential distributions, i.e.

$$F_{J^{\pm}} := \pm\frac{1}{\mu_{J^{\pm}}}\int_0^x e^{\mp(\frac{z}{\mu_{J^{\pm}}})} dz = 1 - e^{\mp(\frac{x}{\mu_{J^{\pm}}})} \tag{29}$$

respectively, where $\mu_{J^{\pm}}$ is the expected size of the positive and negative spiky jumps. The corresponding semimartingale (compensator) measure is

$$\ell(dt, dz)^{\pm} = dt\frac{\lambda^{\pm}(t)}{\mu_J^{\pm}}e^{\mp(\frac{z}{\mu_J^{\pm}})}dz \tag{30}$$

4 Model Calibration

4.1 Calibration Methodology

We describe the steps of a realistic calibration procedure applicable to any process S given as an Itô transform g of an input process. For this purpose, we investigate the inverse-transform of the historical data,

$$g^{-1}(S)(t) = \Lambda(t) + X(t) + Y^{+}(t) + Y^{-}(t) \tag{31}$$

and calibrate the parameters of the ingredient processes. Calibration is performed with respect to a given history. Usually standard regression methods are applied. However, alternative approaches, such as calibration via neural networks (see [13] and [15]) will be considered below. In detail, we proceed as follows:

4.1.1 Identifying Spiky and Non-Spiky Jumps

We first filter separately the following 3 possible jump types: positive spikes, negative spikes, and non-spiky jumps. As filtering tool we use a running sample-volatility of n days (eg, $n = 30$), and a detect jumps as outliers out of the 97.5 confidence level.

A jump will be detected as a spike, if within 2 days[1] it is followed by a jump in the opposite direction.

[1]The selection of 2 days is based on the strong evidence that, other than the spike formations of two consecutive jumps in opposite directions, there are few jumps that are followed on the next day by a small drift and then offset by another jump in the opposite direction

Next, the mean reversion $\beta^{\pm}$ and the distribution parameters of the spike OU-processes $Y^{\pm}$ are calibrated by regression over available spikes. Spike processes $Y^{\pm}$ with exactly the same jump times and jump heights, but with the calibrated reversion after each jump are subtracted, and the resulting processes then corresponds to $\Lambda(t) + X(t)$.

Non-spiky jumps are left untouched until the seasonal behavior is removed.

4.1.2 Subtracting Deterministic Patterns

We subtract the deterministic function

$$\Lambda(t) \quad = \quad \Lambda_0 + at + \Lambda_y(t) + \Lambda_q(t) + \Lambda_m(t) + \Lambda_w(t) \tag{32}$$

Here $\Lambda_0 + at$ is the linear part. $\Lambda_w(t)$, $\Lambda_m(t)$, $\Lambda_q(t)$, and $\Lambda_y(t)$ are weekly, monthly, quarterly, and annual periodic functions.

We calibrate Λ by the following steps:

1. The linear part is fitted to the data, it should be subtracted yielding

$$\begin{aligned} X_1(t) \quad := & \quad \Lambda_y(t) + \Lambda_q(t) + \Lambda_m(t) \\ + & \quad \Lambda_w(t) + X(t) + Y^+(t) + Y^-(t) \end{aligned} \tag{33}$$

2. Periodic corrections $\Lambda_p(t)$ for $p = w, m, q$ are calibrated as follows. First the averages over days of equal period phase $t \in [1, \ldots, \phi_p]$, with ϕ_p equal to one week, month, and quarter respectively, are determined as an estimate

$$\hat{\Lambda}_p(t) \quad = \quad \frac{1}{n_p} \sum_{j=0}^{n_p} X_1(t - j\phi_p) \tag{34}$$

over a sufficiently large number $n_p(t)$ of averaging periods. Note that, n_p may depend on t. Considering the weighted period average

$$\overline{\Lambda}_p \quad = \quad \frac{\sum_{t=1}^{\phi_p} n_p(t) \hat{\Lambda}_p(t)}{\sum_{t=1}^{\phi_p} n_p(t)} \tag{35}$$

it should be approximately zero, since the linear trend was removed before. However finite sample effects may result in slight deviations from zero. Hence,

$$\Lambda_p \quad := \quad \hat{\Lambda}_p(t) - \overline{\Lambda}_p(t) \tag{36}$$

3. The yearly cycle $\Lambda_y(t)$ is calibrated to annual and semiannual periodical functions.

The whole deterministic part is then subtracted from the de-spiked process, yielding the spikeless mean-reverting process

$$g^{-1}(S)(t) - (Y^+(t) + Y^-(t)) - \Lambda(t) \quad = \quad X(t) \tag{37}$$

4.1.3 Core Process Parameters

We calibrate the mean reversion of the remaining core process X, i.e. the reversion rate α by linear regression.

We substract the reversion part from the core process, yielding

$$dX + \alpha X dt \quad = \quad \sigma(t)(dW + dL) \tag{38}$$

If there are non-spiky jumps, these will be substracted thereafter, yielding the diffusive part $\sigma(t)dW$. We may calibrate $\sigma(t)$ as a rolling volatility. The jumps rescaled by $1/\sigma(t)$ yield the process L. Assuming it to be a compound Poisson process, its intensity $\lambda_0^{\pm}$ is obtained from the frequency of these jumps. The empirical distribution of jump heights is used to calibrated the shape of the jump size distribution with support on R.

5 Stable Processes with Independent Increments

A special type of stationary processes $I(t)$ are the stable processes, where the probability distribution of the independent increments is given by a characteristic function

$$\begin{aligned}
\varphi(\mu, c, \alpha, \beta) &:= e^{i\mu t - |ct|^\alpha (1 - i \operatorname{sgn}(t) \; \beta \; \phi(\alpha))} \\
\phi(\alpha) &:= \tan(\frac{\alpha\pi}{2}), \quad \alpha \neq 1 \\
\phi(\alpha) &:= -\frac{2}{\pi} \ln |t|, \quad \alpha = 1 \quad ,
\end{aligned}$$

with stability index $\alpha \in]0,2]$, skew $\beta \in [-1,1]$, localization μ, and scale c (proportional to the width of the distribution at half max value).

While the variance is infinite for $\alpha < 2$, the scale c is always finite.

For $\beta = 0$, the symmetric stable distributions are simply given by

$$\varphi(\mu, c, \alpha) \quad := \quad e^{i\mu t - |ct|^\alpha} \quad ,$$

with $\alpha = 2$ corresponding to the normal distribution with volatility $\sigma = \sqrt{2}c$. Note that only in this case the variance can be used a measure of the width of the distribution. In the cases $\alpha < 2$, rather the scale c is the appropriate measure for the width of the distribution.

The index α of a symmetric stable processes may be estimated together with the other more convenient parameters μ and c.

Due to ergodicity, a r.v. of symmetric stable increments X_k satisfies

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} e^{itX_k} \quad \overset{!}{=} \quad E[e^{itY}] = \varphi(\mu, c, \alpha) \quad .$$

Hence, the empirical characteristic function,

$$S_n(t) \quad := \quad \frac{1}{n} \sum_{k=1}^{n} e^{itX_k} \quad , \tag{39}$$

may be used to estimate α for given μ and c, as regression value

$$\alpha \quad \approx \quad \frac{\ln[i\mu t - \ln S_n(t)]}{\ln|ct|} \tag{40}$$

over different values of the Fourier parameter t. In practice, (40) is used most efficiently for $\mu = 0$, i.e. after substraction of the localization.

6 Time-Series with Weighted Observations

Below we will consider a neural network operating on the weights given to the data points of a time-series. Hence, for observation at time t with value $X(t)$ a weight $w(t)$, will be introduced. Practically we will use the discrete version $w_i := w(t_i)$ on a discrete time series. Using the average weight

$$\bar{w}_i \quad := \tfrac{1}{n} \textstyle\sum_{i=1}^{n} w_i \quad ,$$

the weights have are normalized according to

$$\hat{w}_i \quad := \tfrac{w_i}{\bar{w}_i} \quad .$$

Weighting a series of observation values X_i with weights $\hat{w}_i$ implies that, the stochastic assumptions will not be made for the process of $X(t)$, but for the modified process $\hat{X}(t) := \hat{w}(t)X(t)$. A normal assumption for the distribution of random increments $d\hat{X}$ of the stochastic process $\hat{X}$, implies

$$d\hat{X}(t) \sim \text{Normal}(\mu, \sigma) \quad \Leftrightarrow \quad d\hat{Y}(t) \sim \text{Levy}(0, \frac{1}{\sigma^2}) \quad ,$$

where the processes $\hat{Y}$ is inferred from a corresponding time-series of random increments

$$d\hat{Y}_i \quad := \quad \frac{1}{(d\hat{X}_i - \mu)^2} \quad .$$

The latter has to be used then, in order to infer the empirical characteristic function, according to (39) with Y replaced by $\hat{Y}$. In the following we again omit the $\hat{\cdot}$-symbol over the random variables.

7 Multi-Layer Perceptrons

In general a given target may be reached only up to a certain error. Given a certain measure $E(\widetilde{y}, y)$ for the distance between the given target state y and the state $\tilde{y}$ computed by the neural network, learning of the neural network corresponds to minimization of $E(\tilde{y}, y)$.

The following training algorithm is inspired by Rumelhart, Hinton and Williams [30]. The total error measure over all states of a given layer is defined as

$$E_{total}(\tilde{y}, y) := \frac{1}{2} \sum_{k=1}^{N} (\tilde{y}_k - y_k)^2 \quad . \tag{41}$$

It will be used below to reset the weights in each layer of the neural network.

For simplicity, we consider now a 2-layer perceptron network, which also will be sufficient for our purpose of calibrating the stochastic process parameters.

The processed state $\tilde{y}$ of the neural network is computed by the following steps.

First the critical parameter for the first layer is computed from n weighted input values as $\sum_{i=1}^{n} w_i \cdot x_i$. We consider a hidden output layer with m neurons. For $j = 1, \ldots, m$, let g_j be the activation function of the j-th neuron of the hidden layer, with an activation value of h_j, given as

$$h_j = g_j(\sum_{i=1}^{n} w_i \cdot x_i) \quad . \tag{42}$$

Usually for all neurons of a given layer a common activation function $g = g_1, \ldots, g_m$, e.g. a sigmoid function, is used.

Next, the output of the previous (hidden) layer becomes the input of the next layer, and the activation proceeds analogously to the previous layer.

Let f be the activation function of the pre-final (here the second) output layer. Then the pre-final critical value is

$$q = f(\sum_{j=1}^{m} u_j \cdot h_j) \quad . \tag{43}$$

Finally, the pre-final critical value q is interpreted by a final activation function F yielding

$$\tilde{y} = F(q) \tag{44}$$

as a final state value computed from the neural network with the given weights of the input variables from input and hidden layers.

Now the neural network performs a training step by modifying the weights of all input layers. The learning mechanism the weights is determined by the target distance measure

$$E = \frac{1}{2}\sum_{i=1}^{n}(y^i - \tilde{y}^i)^2 \quad .$$

The weights of both layers are changed according to the steepest descent, i.e.

$$\nabla_{w_i} E := \frac{\partial E}{\partial w_i} \tag{45}$$

$$\nabla_{u_j} E := \frac{\partial E}{\partial u_j} \tag{46}$$

With a learning rate α, which should be adapted to the data, the weights are changed as follows:

$$w_i^{new} = w_i^{old} - \alpha \cdot \nabla_{w_i} E \tag{47}$$

$$u_j^{new} = u_j^{old} - \alpha \cdot \nabla_{u_j} E \tag{48}$$

The necessary number of iterations depends on the requirements posed by the data, the user, and the discipline. Furthermore the starting weights could include theoretical assumption about relevance, interaction and dependencies. Unlike with standard neural networks procedures, in our approach the meaning of the information contained in the intermediate layers is made transparent and usable, rather than operating with a "black box".

8 Neural Network & Stochastic Model

In this section we apply neural networks to the estimation of the stochastic process.

First we demonstrate a simple combination using a 1-layer perceptron network. In this one layer neural network the variance σ^2 and the mean μ are weighted to predict the target value.

$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{49}$$

With $\tilde{y} = \mu_0 u_1 + \sigma_0 u_2$ we obtain

$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - (\mu_0 u_1 + \sigma_0 u_2))^2 \tag{50}$$

$$\begin{aligned} \nabla_{u_1} E &= -\mu_0 \sum_{i=1}^{n}(y_i - (\mu_0 u_1 + \sigma_0 u_2)) && (51) \\ \nabla_{u_2} E &= -\sigma_0 \sum_{i=1}^{n}(y_i - (\mu_0 u_1 + \sigma_0 u_2)) && (52) \end{aligned}$$

A more complex combination uses a 2-layer neural network, where we compute variance and mean via weighted input variables. The first estimate of the mean is

$$\mu_0 = \frac{1}{n}\sum_{i=1}^{n} w_i x_i \quad ,$$

and for the variance it is

$$\sigma_0^2 = \frac{1}{n-1}\sum_{i=1}^{n}(w_i x_i - \mu_0)^2 \quad .$$

The neural network is then trained by adjusting the weights w_i of the first layer, and the weights u_j of the second layer, according to the respective sensitivities (45) and (46) of the error function (49).

9 Neural Parameter Calibration

In this section we demonstrate the dynamical calibration of process parameters for the stochastic process.

Here, different points of the historical time series, receive different weights, which are learned dynamically when propagating through the historical training set.

In the first layer μ and σ are determined on the basis of weighted values $w_i y_i$ of the time series of y-values.

$$\hat{\mu}_y := \frac{1}{\sum_{i=1}^n dt_i} \sum_{i=1}^n w_i dy_i \tag{53}$$

$$\hat{\sigma}^2 := \frac{1}{\sum_{i=1}^n dt_i} \sum_{i=1}^n w_i dt_i \left[\frac{1}{dt_i} \left(\frac{dS_i}{S_{i-1}} \right)^2 \right] \tag{54}$$

In the second layer a new value is determined with

$$\hat{y}_{i+1} = y_i + (u_1 \hat{\mu}_y + u_2 \hat{\sigma}) dt_i \quad . \tag{55}$$

Here, the u_2-term acts as an volatility enhancement of wind speed trend, and the ratio

$$\lambda_i := \frac{u_2}{u_1} \tag{56}$$

can be interpreted as a time-dependent trend-enhancement factor.

Including this value in the regression the process for determining μ and σ in the first layer is repeated. In contrast to normal neural networks the first weights are used according to the formula for μ and σ^2. In every iteration the weights in the first and second layers are changed according to the steepest descent.

10 Numerical Results for Selected Examples

As an example, we test our structural stochastic and neuro-stochastic models with the daily mean wind speed in 0.1 m/s measured at Rotterdam (Fig. 1 resp. 2), at Kassel (Fig. 3 resp. 4), and at Zugspitze (Fig. 5 resp. 6).

In every prediction step, previous values are used to predict the next one. The models were implemented in MATLAB and Mathematica,

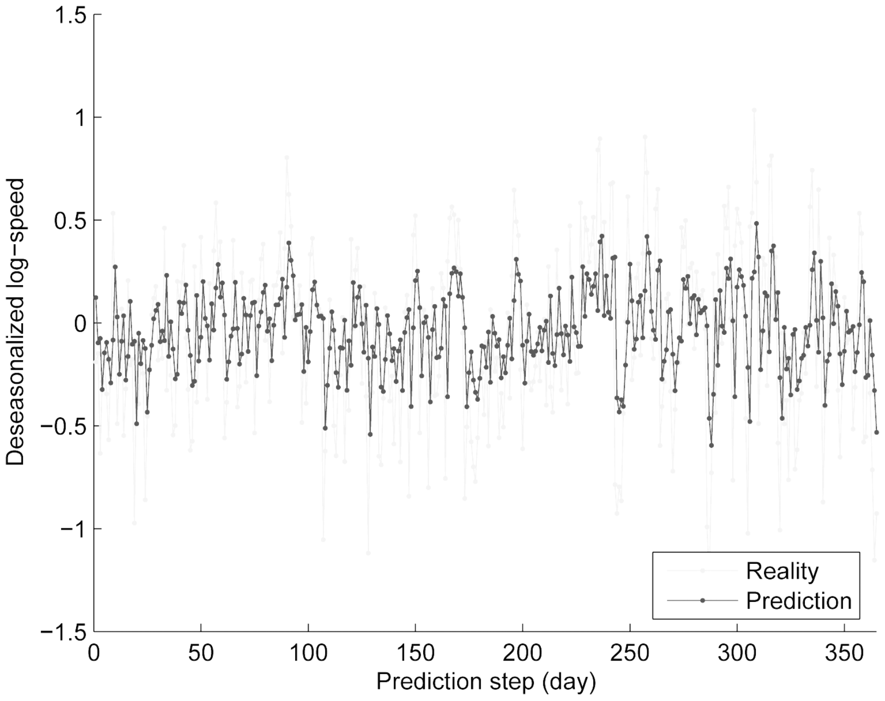

Figure 1: Wind speed log-residuals at Rotterdam: daily predictions after detrending by structural stochastic model (bold black line) between 1.1.2010-31.12.2010 with 2 years of previous history

respectively. In all figures the predicted values are compared to the real values (faint lines). The models can be compared on the basis of their coefficients of determination, R^2.

Using the stochastic model and multi-layer neural networks, respectively, we find for Rotterdam $R^2_{\text{sm}} = 0.17$ and $R^2_{\text{nn}} = 0.19$, for Kassel $R^2_{\text{sm}} = 0.18$ and $R^2_{\text{nn}} = 0.12$, and for Zugspitze $R^2_{\text{sm}} = 0.25$ and $R^2_{\text{nn}} = 0.12$. Hence, around 12 to 25 per cent of the fluctuations could be explained by at least one of the compared models. These results suggest that, for a coastal region (e.g. Rotterdam) the R^2-determination is higher, while for the undulating forest regions (e.g. Kassel) and even more the higher mountain regions (e.g. Zugspitze) the structural stochastic model yields higher determination. This may be due to the fact that, on the one hand, the structural stochastic model is capturing

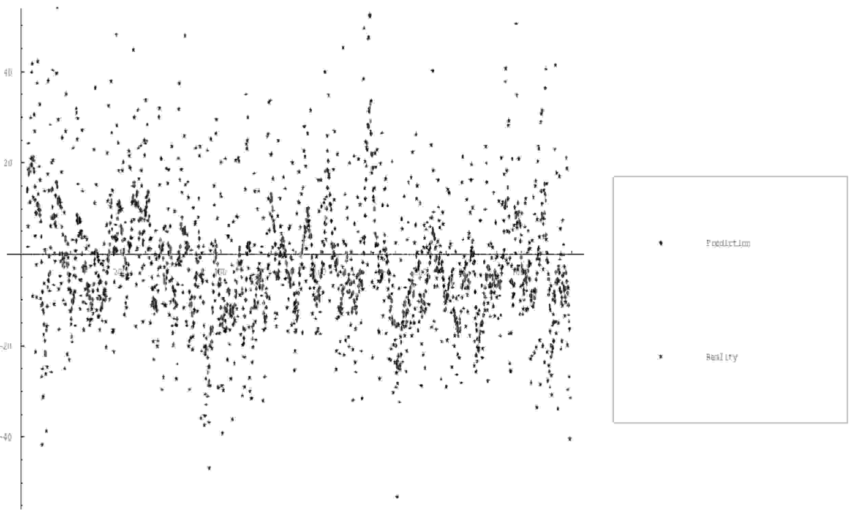

Figure 2: Wind speed (in units of $0.1\frac{m}{s}$) at Rotterdam: daily predictions of residuals after detrending by neuro-stochastic model (bold black spots) between 1.1.2010-31.12.2010 with 2 years of previous history

of extremal wind speed distribution more explicitly and in more detail, while on the other hand extreme fluctuations in wind speed are indeed more likely in high altitude, and wind speed is close to zero more likely where the surface of the landscape is rough.

In any case a considerable fraction of wind speed fluctuations is still explained by the neural network, although the residuals are supposed to contain information only about spikes and autoregression, features which are not modeled explicitly by the neuro-stochastic model.

The p-value in all examples is $p < 0.001$is considerably smaller than the significance level of 0.05, hence the correlation between reality and prediction is highly significant. As expected, only a low level of positive correlation ranging between 0.346 and 0.436 is achieved in the case of neural networks. This range improves as much as 0.42-0.50 in stochastic approach. Considering a target time horizon of 2 years, good agreement of the predicted values with the reality is achieved. Calibrating the model-parameters with neural weights adapting continuously to the history, the model learns to capture the optimal trend according to movements of the past 2 years.

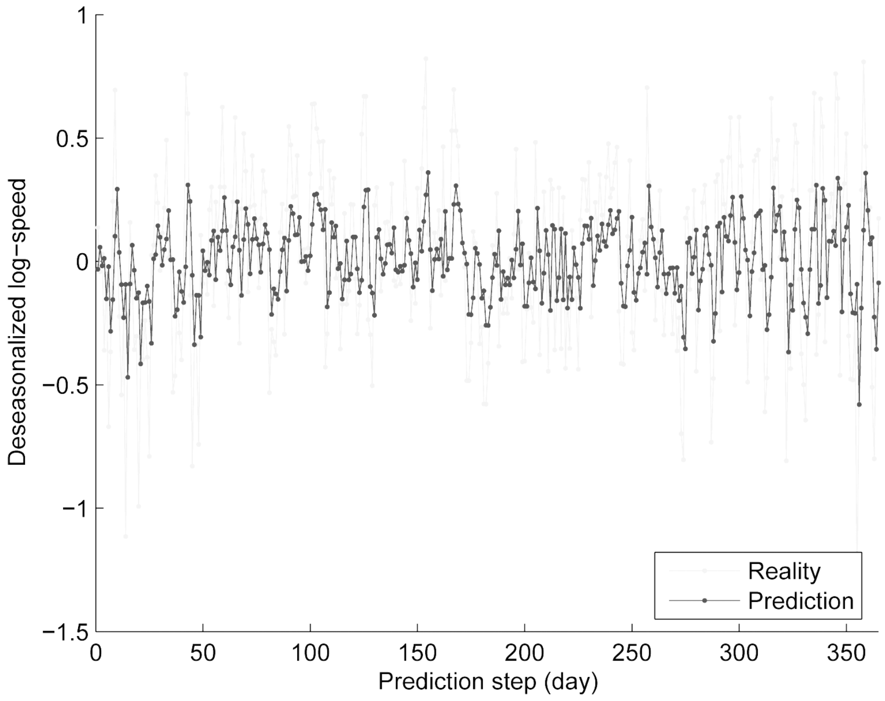

Figure 3: Wind speed log-residuals at Kassel: daily predictions after detrending by structural stochastic model (bold black line) between 1.1.2010-31.12.2010 with 2 years of previous history

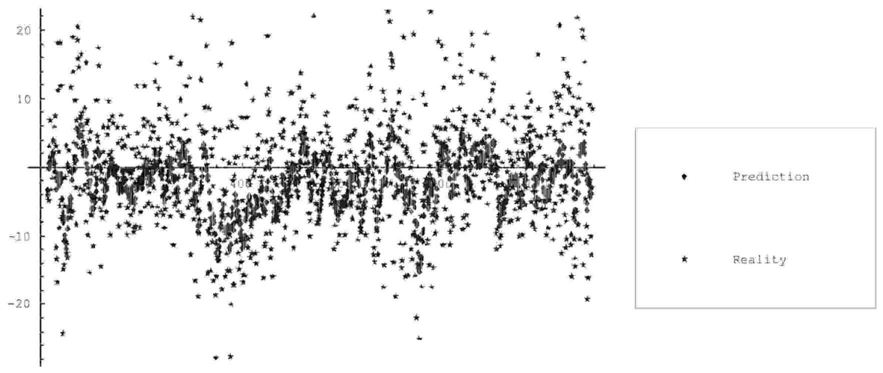

Figure 4: Wind speed (in units of $0.1\frac{m}{s}$) at Kassel: daily predictions of residuals after detrending by neuro-stochastic model (bold black spots) between 1.1.2010-31.12.2010 with 2 years of previous history

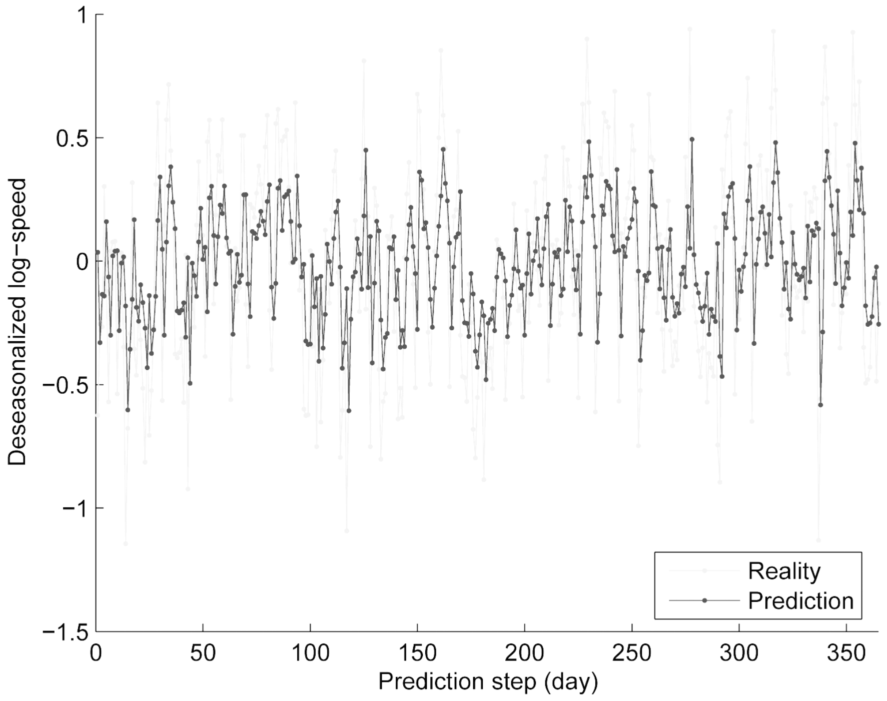

Figure 5: Wind speed log-residuals at Zugspitze: daily predictions after detrending by structural stochastic model (bold black line) between 1.1.2010-31.12.2010 with 2 years of previous history

11 Conclusion and Outlook

In our combined model of neural networks and stochastic processes, the neural transformation processes adapt in such a manner that, from a continuously updated history of fixed length the network is continuously learning the process parameters. In comparison to traditional calibration, our neural methods is taking into account the historical process in a more detailed and dynamical manner, as it is shown in [16]. As a result, the parameters of the stochastic process can be better calibrated than with traditional methods. As our examples demonstrate: Even a relatively simple stochastic process, in combination together with a smart neural network learning continuously the right parameters, the model yields satisfactory predictions. This demonstrates the efficiency of our combined stochastic-neural approach.

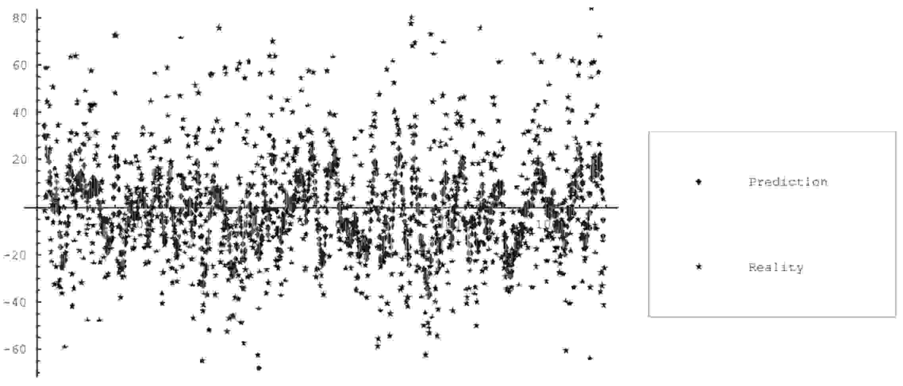

Figure 6: Wind speed (in units of $0.1\frac{m}{s}$) at Zugspitze: daily predictions of residuals after detrending by neuro-stochastic model (bold black spots) between 1.1.2010-31.12.2010 with 2 years of previous history

The numerical results suggest that, for a coastal region (e.g. Rotterdam) the R^2-determination is higher, while for the undulating forest regions (e.g. Kassel) and even more the higher mountain regions (e.g. Zugspitze) the structural stochastic model yields higher determination. This is little surprising, as the latter model puts more detailed effort to capture the separate structure of extremal wind events, with extreme fluctuation in wind speed being more likely in high altitude, and wind speed close to zero being more likely with increasing roughness of surface.

The research is continued to apply our neural network approach also to more demanding stochastic processes, as they may be required for estimations of different climate factors, energy resources or energy demand and prices, such as for gas or electricity.

In several such cases, different phases of normal and spiky modes of volatility have to be taken into account. This can be done on the one hand by regime switching models, on the other hand by working with more advanced semi-martingale processes including jumps, also with time-dependent frequency.

Neural networks calibration may improve stochastic models. In the light of their potential to capture all kinds of generalized (linear and non-linear) "trends", neural networks can give us an idea, how much of structural, i.e. "trend" information we loose, if we do not improve our results, especially in the case of renewable energy. Unraveling of hidden structural information could be economically important, e.g. for energy storage and reduction of non-renewable forms of energy. Furthermore it could show up hidden weather trends. In order to apply neural networks, we have to account a priori for a correlation between all measurements. These correlations have to be checked carefully. Furthermore the validity of our model has to be demonstrated, by applying the model on real unknown data, in particular since generally conclusive measures for model validity are not easily available either.

In the broader context of learning algorithms (aka learning machines), the multilayer perceptron (MLP) applied in the second model above is unusual, by the fact that we use tailored activation functions, describing well defined parameters of a stochastic model. Hence every neuron in the intermediate layer has a clear interpretation. This is very different from the common data mining versions of the multilayer perceptron (MLP) and its generalizations, the support vector machines (SVMs). In the standard MLP applied for some activation function, the intermediate layers are usually a hidden black box. Similarly, the feature space of a nonlinear SVM typically remains without interpretation, and the right choice of the kernel function is rarely evident like in [21]. In this aspect our neuro-stochastic model provides an interesting new path for the design of learning algorithms, with a feature space clearly defined by the parameters of the stochastic model, the kernel functions (which here are just the activation functions) designed directly for learning these model parameters. Opening this black box might give a new direction to further research in statistical learning.

References

[1] Aeinfar, V., Mazdarani, H., Deregeh, F., Hayati, M., and Payandeh, M. (2009), Multilayer Perceptron Neural Network with supervised training method for diagnosis and predicting blood disorder and cancer. Industrial Electronics, ISIE 2009, 2075-2080.

[2] Amini, M., Jalili, R., and Shahriari, H. R. (2006), RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. Computer & Security. **25**(6), 459-468

[3] Aydin, N. S. and Rainer, M. (2010), Valuation of power swing options. Preprint, ENAMEC Institut, Würzburg, to appear in Journal of Energy Markets, 2013.

[4] Azadeh, A., Ghaderi, S. F. and Sohrabkhani, S. (2008), A simulated-based neural network algorithm for forecasting electrical energy consumption in Iran. Energy Policy, **36**(7), 2637-2644.

[5] Bishop, C. M. (1995), Neural networks for pattern recognition. Oxford: Clarendon Press.

[6] Brockwell, P. J. and Marquardt, T. (2005), Levy-driven and fractionally integrated ARMA processes with continuous time parameter. Statistica Sinica, **15**, 477-494.

[7] Deutsch, H.-P. (2004), Der Simulated Annealing Algorithmus. Sect. 33.2.3 in: Derivate und Interne Modelle, (3rd edn), Stuttgart: Schäfer-Poeschel.

[8] Frechen, F.-B. and Giebel, S. M. (2012), Odour emission measurement of liquids and the need for electronic noses. Gefahrstoffe-Reinhaltung der Luft - Ausgabe 10.

[9] Geman, H. and Roncoroni, A. (2006), Unterstanding the fine structure of electricity prices. Journal of Business, **79**(3), 1225–1261.

[10] Giebel, S. M. (2010), Zur Anwendung Neuronaler Netze in den Sozialwissenschaften. (Application of neural networks in social science), Dr. Kovac, Hamburg.

[11] Giebel, S. M. (2007), Estimation of odour concentration. VDI-Conference Odour, Bad Kissingen.

[12] Giebel, S. M. (2009), Application of Neural Networks for characteristic shapes in medicine. METU Ankara, available on http://www3.iam.metu.edu.tr

[13] Giebel, S. M. and Rainer, M. (2009), Forecasting Financial Asset Processes: Stochastic Dynamics via Learning Neural Networks. Bull. Soc. Sciences Médicales Luxembourg, **10**(1), 91-107

[14] Giebel, S. M. (2011), Zur Anwendung der statistischen Formanalyse: Einstieg in die drei- und vierdimensionale Formanalyse (Application of Statistical Shape Analysis). AVM München.

[15] Giebel, S. M. and Rainer, M. (2011), Stochastic processes adapted by neural networks with application to climate, energy, and finance. Applied Mathematics and Computation, **218**, 1003-1007.

[16] Giebel, S. M. and Rainer, M. (2013), Neural network calibrated stochastic processes: forecasting financial assets. CJOR, **21**(2), 277-293.

[17] Movagharnejad, K. and Nikzad, M. (2007), Modeling of tomato drying using artificial neural network. **59**(1-2), 78-85.

[18] Kashaninejad, M., Dehgani, A. A., and Kashiri, M. (2009), Modeling of wheat soaking using two artificial neural networks. Journal of Food Engineering, **91**(4), 602-607.

[19] Kermani, B. G., Schiffmann, S. S., and Nagle, H. T., (1999), Using neural networks and genetic algorithms to enhance performance in an electronic nose. Biomedical Engineering, IEEE, **46**(4), 429-439.

[20] Itô, K. (1942), On stochastic processes I: Infinitely divisible laws of probability. Jap. J. Math., **18**, 261–301.

[21] Jahanbakhshi, R., Keshavarzi, R., Intelligent Prediction of Differential Pipe Sticking by Support Vector Machine Compared With Conventional Artificial Neural Networks: An Example of Iranian Offshore Oil Fields. SPE Drilling & Completion, **27**(4), 586-595.

[22] Lackes, R. and Mack, D. (2000), Neuronale Netze in der Unternehmensplanung (Neural Network in corporate planning). München: Verlag Vahlen.

[23] Lackes, R. and Mack D. (1998), Einsatz Neuronaler Netze als Instrument zur Eignungsbeurteilung (Application of Neural Networks in qualification tests) (ZfP 4/98).

[24] Levenberg, K. (1944), A Method for the Solution of Certain Non-Linear Problems in Least Squares, Quart. Appl. Math., **2**, 164-168.

[25] Maqsood, I., Khan, M. R., and Abraham, A. (2004), An ensemble of neural networks for weather forecasting. Neural Computing & Applications, **13**(2), 112-122.

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

[27] McCulloch, W. S. and Pitts, W. (1943), A logical calculus of the idea immanent in neural nets. Bulletin of Mathematical Biophysics, 115-137.

[28] McNelis, P. D. (2005), Neural networks in finance: Gaining predictive edge in the market. London: Elsevier.

[29] Rainer, M. (2009), Calibration of stochastic models for interest rate derivatives. Optimization, **58**, 373-388.

[30] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986), Learning internal representation by error propagation. 318-362 in: Rumelhart, D.E., McClelland, J.L., PDP Research Group, Parallel. Distributes Processing, Cambridge: MIT Press.

[31] Strano, M. (2004), A Neural Network Applied to Criminal Psychological Profiling: An Italian Initiative. International Journal of Offender Therapy and Comparative Criminology, **48**, 495.

[32] Tabari, H., Marofi, S., and Sabziparvar, A. (2010), Estimation of daily pan evaporation using artificial neural network and multivariate non-linear regression. Irrigation Science, **28**(5), 399-406.