Neural Networks 63 (2015) 31–47

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Robust sequential learning of feedforward neural networks in the presence of heavy-tailed noise

Najdan Vuković [a,*], Zoran Miljković [b,1]

[a] University of Belgrade - Faculty of Mechanical Engineering, Innovation Center, Kraljice Marije 16 ; 11120 Belgrade 35, Serbia

[b] University of Belgrade - Faculty of Mechanical Engineering, Production Engineering Department, Kraljice Marije 16 ; 11120 Belgrade 35, Serbia

ARTICLE INFO

Article history:
Received 18 February 2014
Received in revised form 18 September 2014
Accepted 4 November 2014
Available online 15 November 2014

Keywords:
Feedforward neural networks
Sequential learning
Robust extended Kalman filter
Structured variational approximation
Heavy-tailed noise
Inverse Wishart distribution

ABSTRACT

Feedforward neural networks (FFNN) are among the most used neural networks for modeling of various nonlinear problems in engineering. In sequential and especially real time processing all neural networks models fail when faced with outliers. Outliers are found across a wide range of engineering problems. Recent research results in the field have shown that to avoid overfitting or divergence of the model, new approach is needed especially if FFNN is to run sequentially or in real time. To accommodate limitations of FFNN when training data contains a certain number of outliers, this paper presents new learning algorithm based on improvement of conventional extended Kalman filter (EKF). Extended Kalman filter robust to outliers (EKF-OR) is probabilistic generative model in which measurement noise covariance is not constant; the sequence of noise measurement covariance is modeled as stochastic process over the set of symmetric positive-definite matrices in which prior is modeled as inverse Wishart distribution. In each iteration EKF-OR simultaneously estimates noise estimates and current best estimate of FFNN parameters. Bayesian framework enables one to mathematically derive expressions, while analytical intractability of the Bayes' update step is solved by using structured variational approximation. All mathematical expressions in the paper are derived using the first principles. Extensive experimental study shows that FFNN trained with developed learning algorithm, achieves low prediction error and good generalization quality regardless of outliers' presence in training data.

1. Introduction

Any real world application of neural network based model of the system is subjected to the high/moderate noise and existence of outliers in data. Outliers have enormous practical significance because these data points occur relatively often in engineering. Outlier may be defined as an observation that numerically significantly differs from the rest of the data (Agamennoni, Nieto, & Nebot, 2012) that it raises suspicion in phenomena or mechanism we believe that actually generated all data. Typically, outliers fall outside of an overall pattern of distribution (Agamennoni, Nieto, & Nebot, 2011; Ting, Theodorou, & Schaal, 2007). In engineering, especially in applications with real time data processing ability, outliers are a common phenomenon that needs to be analyzed and their influence has to be integrated into the model analysis and validation. Failing to recognize their influence may significantly jeopardize performance of the model, especially if our model is to perform sequential processing of the data or run in real time (Miljković, Vuković, Mitić, & Babić, 2013; Vuković & Miljković, 2013). Outliers may occur by chance, but more often, they may originate from temporary sensor failures, some unknown system anomalies or unmodeled reactions from the environment or some other disturbances; all of these may cause data points to fall far away from expected pattern of data distribution, and as an overall result they may cause our model of the system to diverge from designed performance.

In this paper, we develop an original approach for neural network sequential learning that does not require preprocessing of the data to model and process outliers. Our model is based on a standard extended Kalman filter (EKF), which is modified to process outliers as if these were a "normal" data points. Performance of EKF is based on an assumption that system and measurement equations are corrupted with additive white

* Corresponding author. Tel.: +381 63 363 858.
E-mail addresses: nvukovic@mas.bg.ac.rs (N. Vuković), zmiljkovic@mas.bg.ac.rs (Z. Miljković).
[1] Tel.: +381 11 3302 468.

http://dx.doi.org/10.1016/j.neunet.2014.11.001
0893-6080/

Gaussian noise. The noise level is constant, defined with covariance matrix. Gaussian assumption is backed up with Central Limit Theorem—which states that as sample goes to infinity, arithmetic mean of a set of random variables with finite mean and variance having arbitrary distribution, in limit tends to the Gaussian distribution; furthermore, Gaussians are popular due to their simple mathematical form which (in most cases) results in straightforward closed mathematical calculations. However, in nature and in engineering, not much of processes obey Gaussian assumption. Similarly, Gaussian has tin tails, which suggests that there is a zero chance for misreading or fake measurements. Failing to recognize and process non-Gaussian noises can seriously damage model's performance and cause divergence. To provide more flexibility with respect to exogenous noise, in our model we assume additive noise as well, but in contrast to standard EKF we do not assume Gaussian probability distribution of noise and allow observation noise covariance matrix to change over time. These two assumptions have following ramifications: firstly, we acknowledge that real world does not obey Gaussian distribution and that is why we introduce probability distribution of the noise with heavier and longer tail. Secondly, we estimate noise covariance in each iteration, which helps us to introduce possibly unmodeled environmental disturbances in the model, where new information is encoded into estimated noise covariance matrix. Flexibility of this approach is obvious when it comes to explaining outliers in data, especially if it is sequentially processed.

The learning algorithm is developed in sequential form (Vuković, 2012), which means that whenever new data is available, the sequential learning continues learning process by updating the existing neural network, instead of going through entire learning process from the beginning (Vuković & Miljković, 2013). This is why sequential algorithms are preferred, especially in engineering and applications where fast development of neural network based models are needed (Miljković & Aleksendrić, 2009). Sequential learning has the following characteristics (Huang, Saratchandran, & Sundararajan, 2005):

(1) Learning system uses one and only one training example in iteration. The examples are presented to the learner sequentially, one following the other.
(2) Training example is erased from memory after learning procedure finishes update of network parameters.
(3) Learning system has no prior knowledge of the total number of examples.

These features of sequential learning are important for modeling of engineering problems, where new data might be available after neural network model was built. Sequential learning enables learning system to continue learning process if new data is received, without need to memorize and use old data (Vuković & Miljković, 2013). Furthermore, when sequential learning is used the need to have learning algorithm robust to outliers able to sequentially process them is even more emphasized.

This paper is organized as follows. The second part of the paper provides analysis of research results and compares features of proposed sequential learning algorithm with ability to treat outliers with the current state in the field. In Section 3 we provide basic intuition, foundations and mathematical derivation of the learning algorithm. Through various experimental studies using real world and synthetic data, in Section 4 we demonstrate and discuss the potential of the developed learning algorithm for training of two types of FFNN when faced with outliers in the data. Eventually, final conclusions and assessments of learning algorithm's performance are given in Section 5.

2. Related work and contributions of the paper

Robust statistics is a broad field of research and in this section of the paper we wish to concentrate solely on soft computing approaches, especially neural networks. For wider prospective the reader is kindly referred to Agamennoni et al. (2011, 2012), Chandola, Banerjee, and Kumar (2009), Đurović and Kovačević (1999), Hodge and Austin (2004), Markou and Singh (2003a, 2003b) Stanković and Kovačević (1986), Schick and Mitter (1994) and references therein.

If model minimizes L_2 norm, than it emphasizes outliers more than it should; this situation leads to over-fitting and poor generalization of the model when outliers are present. On the other hand, when L_1 norm is minimized, model puts emphasis on data points close to the prediction, which is yet another undesired situation which neglects update step; certain data point may not be an outlier but it may generate large error between prediction of the model and the actual value, which will make the learning algorithm to classify it as an outlier. To solve this issue, research community has proposed a great diversity of robust cost functions called M-estimators (Huber, 2011) for developing of robust statistical/neural models. The main attractiveness of M-estimators is their influence function; it is bounded which guarantees bounded response given arbitrary query point, unlike non robust cost function whose influence function is unbounded. This feature makes M-Estimator popular approach for robust estimation/learning.

In this paragraph we provide information related to the usage of robust cost function in neural network/support vector machine community. To achieve robustness of their model, authors in Lee, Chung, Tsai, and Chang (1999) propose Hample M-estimator to accommodate large errors in data. Instead of Gaussian activation function, authors in their radial basis function (RBF) neural network propose composite of sigmoid functions and introduce growing and pruning of neurons. The robustness in Chuang, Su, Jeng, and Hsiao (2002) is introduced in terms of traditional concept of robustness in statistics; authors make use of robust cost function such as hyperbolic tangent estimator and build their Support Vector Regression (SVR) Network in two phases. In the first phase a classical approach towards SVR optimization is taken. In the second phase, weights are being adjusted using robust learning based on robust cost function. Similarly, in Chuang, Su, and Chen (2001) authors develop fuzzy based model and enable robustness by using different cost function (Tukey's biweight cost function). Authors in Lee, Chiang, Shih, and Tsai (2009) build their model based on RBF and use Welsch M-estimator as cost function. The Welsch is chosen as cost function because of its smoothness and reduction of effects of large errors. Furthermore, their robust RBF model allows growing and pruning of the neurons based on the concept of neuron significance (Huang, Saratchandran, & Sundararajan, 2004; Huang et al., 2005; Vuković & Miljković, 2013). In Zhu, Hoi, and Lyu (2008) authors point out that robustness in the model may be introduced with regularization term in cost function or to use robust cost function. To solve these issues and achieve robustness into their Regularized Kernel regression authors use Huber robust function because of its ability to use both L_2 and L_1 norms. Instead of solving optimization problem in dual, their algorithm is run in primal optimization form. A modified Huber robust cost function is applied in Pernía-Espinoza, Ordieres-Meré, Martínez-de-Pisón, and González-Marcos (2005); authors argue that scalability of the Huber function is not appropriate, hence they develop a τ based approach (Yohai & Zamar, 1988) which uses scale estimator. Similarly to previous result (Chuang et al., 2002), in Chuang and Jeng (2007) and Chuang, Jeng, and Lin (2004) one may see the two phases in learning, (i) determination of initial structure and (ii) robust learning. However, the logistic cost function is used in

this paper instead of tanh robust estimator. Robust optimization of parameters in second phase is performed with annealing robust learning algorithm (ARLA). RBF model trained with this procedure generates estimates of good quality. Logistic function is used in Ko (2012) as well, where authors build robust wavelet neural network for system identification. As in Chuang et al. (2002), the authors apply two stage learning, where in the first phase initial structure is determined using wavelet support vector regression while in the second stage the robust learning using logistic cost function is performed via dynamical learning algorithm.

Yet another distinction we may observe between various approaches is in terms of the way outliers are treated. To be more specific, in some algorithms processing of outliers is conducted in the following manner:

(1) Preprocessing of the data: identify and remove outliers from data;
(2) Train model with data free of outliers.

The main representatives are given in Chuang and Jeng (2007) and Chuang et al. (2004, 2002). In Jeng, Chuang, and Tao (2010) authors use hybrid Support Vector Regression and Gaussian processes so as to achieve robustness of the model in the presence of outliers. Firstly, authors remove outliers and in the second stage Gaussian process is trained with data free of outliers. In Chuang and Lee (2011) authors take two-steps approach as well. They do not make use of robust statistics; instead, outliers are firstly filtered using SVR, and afterwards the clean data is used to train non robust least square support vector regression (LS-SVR). However, this two-steps approach is not suitable for sequential learning; identification and removal of outliers from data prior to training is not possible. In sequential learning, the model is to deal with possible outliers as data arrive.

The concepts of least trimmed squares (Rusiecki, 2007) and least trimmed absolute value of residuals (Rusiecki, 2013) show good results in terms of robustness to outliers.

Connor, Martin, and Atlas (1994) show that recurrent neural networks are not able to deal with outliers and use Hampel two part redescending function. Their model is based on simple and effective intuition: points with large errors are treated as outliers and points with small errors are classified as genuine data points; the main idea is to robustify residual. Finally, authors build their model for recurrent neural network using Huber and Tukey robust cost function. The proposed improvements resulted in considerable improvement in terms of accuracy when faced with data contaminated by outliers. Recently, Li, Han, and Wang (2012) demonstrate how to train Echo State network (ESN) in the presence of noise and outliers in a Bayesian framework, while replacing Gaussian likelihood with Laplacian so as to introduce robustness in the model.

Some authors develop hybrid algorithms so as to achieve robust learning. Yang et al. (2013) propose RBF fuzzy hybrid which minimizes Wilcoxon norm using Particle Swarm Optimization. In Fu, Wu, Jeng, and Ko (2010) logistic cost function is used for robustness of model based on SVR, RBF and fuzzy inference. SVR determines number of rules in inference engine and initial weights for the fuzzy neural network. In the final step of their algorithm, ARLA (Chuang & Jeng, 2007) is applied to adjust parameters using logistic cost function. In similar manner, to enable robustness of the model, hybridization in Ko (2012) is achieved by training wavelet neural network with specific algorithm called annealing dynamical learning algorithm (ADLA), where parameters are optimized with PSO. Other hybrid-like solution such as SVR with RBF neural networks (Chuang & Jeng, 2007; Chuang et al., 2004, 2002) trained with ARLA may be seen as well.

Recently, another approach has emerged for treating outliers in data. Bayesian framework (Andrieu, De Freitas, & Doucet, 2001) provides natural treatment of outliers. For example, in Archambeau and Verleysen (2007) the latent model is based on mixture of Student t distributions so as to enable variational Bayesian methodology to be applied. Similarly, to build robust Relevance Vector Machines (RVM) (Nikolaev & de Menezes, 2008) mixture of Gaussians is used to model one dimensional noise. There is distinction between these two results; (Nikolaev & de Menezes, 2008) assumes factorized solution whereas in Archambeau and Verleysen (2007) there is no need to assume factorization. In the similar manner, Li et al. (2012) demonstrate how to train recurrent neural networks in the presence of noise and outliers using Bayesian approach. Echo State network is trained in a Bayesian framework; instead of Gaussian likelihood it is replaced as Laplacian to introduce robustness in the model. Then, in the second step of their robust learning algorithm, authors approximate Laplacian likelihood with Gaussian to solve analytical intractability. Bayesian approach shows effectiveness and accuracy when it comes to treatment of outliers.

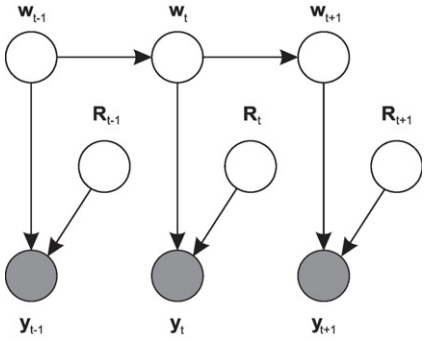


Fig. 1. The extended Kalman filter robust to outliers shown as Dynamic Bayesian network.

This paper differs from aforementioned results along following lines. Firstly, we apply sequential learning which implies simultaneous processing of outliers with other data points; in EKF-OR outliers are processed as any other data point and algorithm naturally down-weights them within Bayesian framework. Other research results (Chuang & Jeng, 2007; Chuang et al., 2004, 2002; Chuang & Lee, 2011) are based on two-steps methodology in which preprocessing of outliers is followed by processing of clear data. Secondly, our model is based on Bayesian approach; the model relies on extended Kalman filter in which robustness to outliers is introduced in observation model using "uncertainty about uncertainty" approach (Agamennoni et al., 2012). To the authors' best knowledge, this is the first paper in which measurement noise covariance is not constant; the sequence of noise measurement covariance is modeled as stochastic process over the set of symmetric positive-definite matrices in which prior is modeled as inverse Wishart distribution. Our learning algorithm is not a hybrid one, instead it follows naturally from first principles. Thirdly, when faced with analytically intractable expression for update step of the filter we apply structured variational approximation (Agamennoni et al., 2012; Barber, 2012; Beal, 2003; Bishop, 2006) which puts tight lower bounds on the marginal likelihood of the data. There is no need to robustify likelihood as in Li et al. (2012), because robustness is introduced in observation model of the filter with uncertain measurement noise covariance.

3. Extended Kalman filter robust to outliers

Extended Kalman filter robust to outliers (EKF-OR) describes mechanism that led to data generation, hence it is a probabilistic

generative model. Similarly to standard EKF, EKF-OR may be represented as Dynamic Bayesian network (DBN) given in Fig. 1, where each white circle represents unobserved (hidden) system state, shaded circles represent observations of system, and directed edges explain stochastic relationships between observed and unobserved variables in form of conditional probability distributions. As it may be seen from Fig. 1, in contrast to standard EKF, in this model measurement noise covariance matrix is treated as unobserved stochastic variable.

Now, let us focus on a particular EKF as it is used for neural network training (Andrieu et al., 2001; Simon, 2002; Vuković, 2012; Vuković & Miljković, 2013). In this setup, state and measurement equation are given as:

$$p\left(\mathbf{w}_t|\mathbf{w}_{t-1}\right) \sim N\left(\mathbf{w}_{t-1}, \mathbf{Q}\right) \tag{1}$$

$$p\left(\mathbf{y}_t|\mathbf{w}_t\right) \sim N\left(\mathbf{H}_t\mathbf{w}_t, \mathbf{R}_t\right) \tag{2}$$

where, in general terms, $N\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ denotes multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$; $\mathbf{w}$ denotes vector of all network parameters that are to be optimized, $\mathbf{Q}$ is a process covariance and $\mathbf{R}_t$ is observation noise covariance matrix that evolves over time. The reader may notice that both distributions are conditionally Gaussian, and that measurement covariance is no longer fixed, i.e. $\mathbf{R}_t$ is being estimated at each filter iteration. This is the first distinction that sets apart our algorithm from its predecessor Kalman filter; it enables us to introduce robustness into the Kalman filter, where robustness to the presence of outliers in the data comes from uncertain measurement noise covariance $\mathbf{R}_t$. As stated in Agamennoni et al. (2012), "uncertainty about uncertainty" enables us to build the model with robust noise properties.

The sequence $\{\mathbf{R}_t\}$ is modeled as stochastic process over the set of symmetric positive-definite matrices (Agamennoni et al., 2012). To continue with analysis, we need to introduce a prior distribution over $\mathbf{R}_t$ at each time step. In Bayesian statistical modeling, a conjugate distribution is distribution that generates the same functional form of posterior as prior (Barber, 2012; Beal, 2003; Bishop, 2006). For the covariance matrix there are two choices for a prior distribution: the first one directly models covariance matrix and it uses inverse Wishart distribution for sampling, while the other approach makes use of Wishart distribution and it samples precision matrices. In this paper, we focus on the first approach, and assume prior distribution over $\mathbf{R}_t$ as inverse Wishart, i.e.

$$\mathbf{R} \sim W^{-1}\left(\nu\boldsymbol{\Omega}, \nu\right) \tag{3}$$

where $\boldsymbol{\Omega}$ and ν denote harmonic mean and degrees of freedom, respectively. Unlike other probability distributions, inverse Wishart distribution can be characterized in different ways depending on the formulation of precision matrix and degrees of freedom. In this paper, we use formulation given in Agamennoni et al. (2012) and define inverse Wishart distribution as a probability distribution over convex cone of $d \times d$ symmetric positive-definite matrices, parameterized with harmonic mean $\boldsymbol{\Omega}$ and degrees of freedom ν.

Now, suppose that ate each time step the noise covariance matrix $\mathbf{R}_t$ is inverse Wishart, i.e.

$$\mathbf{R}_t \sim W^{-1}\left(\nu_t\boldsymbol{\Lambda}_t, \nu_t\right). \tag{4}$$

If we multiply measurement Eq. (2) with Eq. (4), and marginalize out $\mathbf{R}_t$ (integrate over it), we may see that observations $\mathbf{y}_t$ are Student t-distributed (Box & Tiao, 1973):

$$\begin{aligned} p\left(\mathbf{y}_t|\mathbf{w}_t\right) &= \int_{\mathbf{R}_t>\mathbf{0}} p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right) p\left(\mathbf{R}_t\right) d\mathbf{R}_t \\ &\propto \left(1 + \frac{\left(\mathbf{y}_t - \mathbf{H}_t\mathbf{w}_t\right)^T \boldsymbol{\Lambda}^{-1}\left(\mathbf{y}_t - \mathbf{H}_t\mathbf{w}_t\right)}{\nu_t}\right)^{-\frac{\nu_t+1}{2}}. \end{aligned} \tag{5}$$

The Student t-distribution has attractive properties when it comes to modeling of outliers. Fig. 2(a) shows characteristics of Student t distribution and Gaussian distribution. As it can be seen, Student t distribution has heavier and longer tail than Gaussian; it spreads probability mass evenly to the regions far away from its mode, where Gaussian assigns zero probability mass. Tails of Student t distribution decays to zero at less than exponential rate, and in the limit when $\nu_t \to \infty$, the Student t distribution reduces to the Gaussian distribution. By allowing our measurement model to be Student t-distributed, outliers are integrated into EKF framework and treated as any other data point, without necessity to additionally preprocess the data.

Another useful property of Student t distribution is its influence function, shown in Fig. 2(b) for different setting of degrees of freedom ν. The influence function of the distribution describes its sensitivity with respect to infinitesimal changes in the data. Influence function is defined as $-\partial \ln p\left(x\right)/\partial x$, where $p\left(\cdot\right)$ is probability density function and x is random variable.

Influence function of Student t distribution is more robust to data points that greatly differ from the distribution's mean, which is important when it comes to explain outliers in data. In contrast, influence function of Gaussian distribution is more sensitive and more responsive, which is not desired in real world modeling which frequently generates data containing outliers.

Therefore, Student t distribution enables us to: (i) "tell" the learner (network) that there is a probability for malfunction or fake measurements; (ii) have the learner which is less sensitive to sudden changes in data due to influence function.

Having introduced the main intuition we may proceed with learning algorithm. In the following section of the paper, we introduce Bayesian formulation of neural network learning problem, show how to perform update despite having analytically intractable update step based on variational approximation we have used, and derive EKF-OR as an algorithm for network training.

3.1. Bayesian learning of network parameters

Bayesian statistics has gain much research attention in last decades due to its ability to provide reliable statistical models from uncertain data (Barber, 2012; Beal, 2003; Bishop, 2006). Usage of Bayesian methods spreads from image processing, robotics, signal processing etc. Compared to some other methodologies, Bayesian statistics is based on simple (Bayes') rule which describes how to relate model with new evidence. However, although the main idea is simple, Bayesian methods involve complicated mathematics to perform inference, but on most occasions computation of integrals/sums is complex or even intractable. Throughout years, the research community has developed two possible ways to solve the problem: (i) stochastic methods and (ii) deterministic approximation methods. In the first group one typically performs sampling from the distribution so as to estimate unknown parameters/hidden variables; the main representative are Monte Carlo methods (Doucet, De Freitas, & Gordon, 2001) with importance sampling, sequential importance sampling, Markov Chain Monte Carlo, Gibbs sampling etc. Stochastic methods are computationally demanding, especially for sequential data processing. On the other hand, deterministic approximation schemes rely on analytical approximations of posterior distribution; the complicated posterior is approximated with simpler distribution(s) so as to enable inference. The representatives are: Laplace approximation, variational inference (VI) (Tzikas, Likas, & Galatsanos, 2008) and expectation propagation (EP) (Minka, 2001). Being analytical, they provide us with equations that we can use to predict and perform inference, however, due to their approximate nature they can never generate exact results. In this paper, we employ deterministic approximation method using VI (i.e. variational Bayes). We look for the

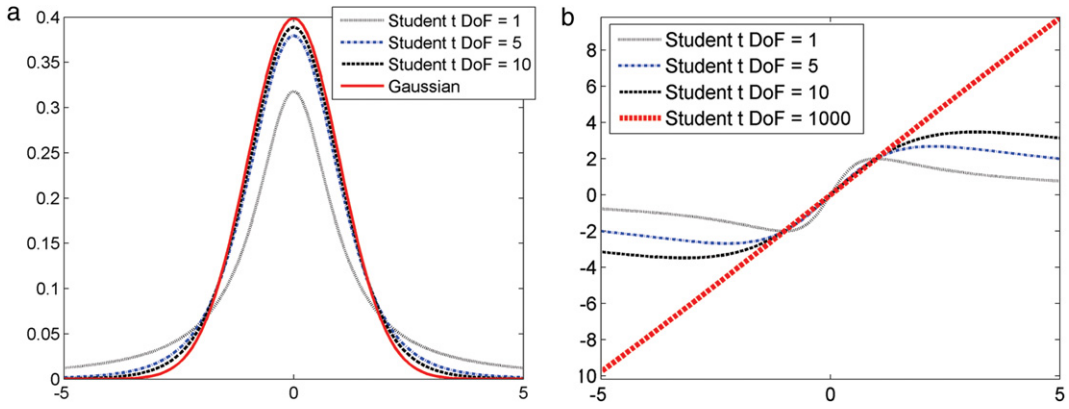


Fig. 2. (a) Student t distribution with different degrees of freedom (DoF) versus Gaussian distribution. (b) As a measure of sensitivity influence function is less responsive to data points that greatly differ from the mean.

solution of the problem in class of probability distributions that factors in a desired manner. VI is chosen over other deterministic approximation schemes because: (i) when compared to Laplace approximation VI provides more accurate results; (ii) when compared to EP, VI scheme provides solution with better predictive distribution.

Let us rewrite Eqs. (1) and (2) in the following form

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \boldsymbol{\varepsilon}_t \tag{6}$$

$$\mathbf{y}_t = \mathbf{g}_t\left(\mathbf{w}_t, \mathbf{x}_t\right) + \boldsymbol{\omega}_t \tag{7}$$

where $\mathbf{g}_t(\cdot,\cdot)$ represents neural network, $\mathbf{w}_t$ is the vector of all network parameters (the state vector), $\boldsymbol{\varepsilon}_t \sim N(\boldsymbol{\varepsilon}_t, \mathbf{Q})$ is the process noise, and $\boldsymbol{\omega}_t \sim \text{Student}(\boldsymbol{\omega}_t, \nu)$ is the measurement noise vector. Measurement noise covariance matrix $\mathbf{R}_t$ is defined with Eq. (3) and it obeys inverse Wishart distribution. Suppose that data is given with the sequence $\mathbf{Y}_T = \{\mathbf{y}_t\}_{t=1}^T$. Let us define posterior of network parameters $\mathbf{w}_t$ given all data as:

$$\alpha(\mathbf{w}_t) \triangleq p\left(\mathbf{w}_t|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T\right) = p\left(\mathbf{w}_t|\mathbf{Y}_T\right). \tag{8}$$

Unfortunately, unlike the standard EKF implementation in which $\alpha(\mathbf{w}_t) \sim N(\mathbf{w}_t, \mathbf{P}_t)$, in the robust setup the Student t distributed measurement vector results in a non Gaussian state vector. In Bayesian setup, parameter estimation (network learning) is performed via Bayes' rule, iteratively multiplying likelihood and prior, and dividing it with evidence so as to generate posterior distribution, i.e.

$$\underbrace{p\left(\mathbf{w}_t|\mathbf{y}_t\right)}_{\text{Posterior}} = \frac{\overbrace{p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right)}^{\text{Likelihood}}\overbrace{p\left(\mathbf{w}_t\right)}^{\text{Prior}}}{\underbrace{p\left(\mathbf{y}_t\right)}_{\text{Evidence}}}$$

$$= \frac{p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right)\overbrace{\int p\left(\mathbf{w}_t|\mathbf{w}_{t-1}\right)p\left(\mathbf{w}_{t-1}\right)d\mathbf{w}_{t-1}}^{p\left(\mathbf{w}_t|\mathbf{y}_1,\mathbf{y}_2,\ldots,\mathbf{y}_{t-1}\right)}}{p\left(\mathbf{y}_t\right)}. \tag{9}$$

One may notice that we are performing sequential update of the neural network parameters $\mathbf{w}_t$ given data vector $\mathbf{y}_t$ at time instant t. Learning problem is decomposed into two steps. In the first step, we propagate the state vector $\mathbf{w}_t$, i.e. perform convolution of state transition $p\left(\mathbf{w}_t|\mathbf{w}_{t-1}\right)$ and prior belief $p\left(\mathbf{w}_{t-1}\right)$ so that we can determine $p\left(\mathbf{w}_t|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{t-1}\right) \sim N\left(\boldsymbol{\mu}_{t|t-1}, \mathbf{P}_{t|t-1}\right)$. If we assume $p\left(\mathbf{w}_{t-1}\right) \sim N\left(\mathbf{w}_{t-1}, \boldsymbol{\Sigma}_{t-1}\right)$, than we can easily solve the integral in the numerator of Eq. (9), whose solution is given as:

$$\boldsymbol{\mu}_{t|t-1} = \boldsymbol{\mu}_{t-1} \tag{10}$$

$$\mathbf{P}_{t|t-1} = \mathbf{P}_{t-1} + \mathbf{Q}. \tag{11}$$

So far, the calculations have been straightforward; the problem arises when likelihood $p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right)$, given as conditional distribution, is to be multiplied with $p\left(\mathbf{w}_t|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{t-1}\right)$. Namely, the non Gaussian noise model makes calculation of this analytically intractable, hence some other approach has to be taken to estimate $p\left(\mathbf{w}_t|\mathbf{y}_t\right)$. More precisely, non Gaussian measurement model makes our prediction non Gaussian as well. To solve this issue we have to apply some form of statistical inference. For example, we may use Laplace transformation and approximate joint posterior as a Gaussian (Barber, 2012; Bishop, 2006); however, the exact joint posterior cannot be accurately approximated with Gaussian which makes former steps of Bayesian learning inaccurate (one cannot propagate moments of the distribution because these do not represent the real distribution).

To overcome this problem, in this research we apply variational inference approach (Tzikas et al., 2008) in form of structured variational approximation (Barber, 2012; Beal, 2003; Bishop, 2006). Variational calculus has been in use since 17 century, frequently applied for problems in mathematics, physics, and engineering.

3.2. Structured variational approximation

Let us define new sequence of random variables as $\mathbf{z}_t = \{\mathbf{w}_t, \mathbf{R}_t\}$, and let $q(\cdot)$ be an approximate posterior distribution over $\mathbf{z}_t$ given $\mathbf{y}_t$. Now, we can express the marginal log-likelihood of the data in the following form:

$$\ln p\left(\{\mathbf{y}_t\}\right) = L[q] + KL[q \parallel p] \tag{12}$$

where $L[q]$ denotes lower bound on the data marginal log-likelihood, defined as

$$L[q] = \iint \ldots \int q\left(\{\mathbf{z}_t\}\right) \ln \frac{p\left(\{\mathbf{y}_t, \mathbf{z}_t\}\right)}{q\left(\{\mathbf{z}_t\}\right)} \prod_{t=1}^{T} d\mathbf{y}_t \tag{13}$$

and $KL[q \parallel p]$ is Kullback–Leibler (KL) divergence, also known as relative entropy:

$$KL[q \parallel p] = \iint \ldots \int q\left(\{\mathbf{z}_t\}\right) \ln \frac{p\left(\{\mathbf{z}_t\} \mid \{\mathbf{y}_t\}\right)}{q\left(\{\mathbf{z}_t\}\right)} \prod_{t=1}^{T} d\mathbf{y}_t. \tag{14}$$

KL cannot be called a distance (metric) measure because it is not symmetric; however, it is used to quantify similarity between two distributions.

Eq. (12) can be derived directly from definition of KL divergence Eq. (14) by decomposing conditional probability $p(\{\mathbf{z}_t\}|\{\mathbf{y}_t\}) = p(\{\mathbf{y}_t, \mathbf{z}_t\})/p(\{\mathbf{y}_t\})$ and exploiting the fact that $p(\{\mathbf{y}_t\})$ does not depend on approximate distribution $q(\cdot)$. Analyzing Eq. (12), one can deduce that a perfect fit $\ln p(\{\mathbf{y}_t\}) = L[q]$ implies KL divergence to be zero, i.e. $KL[q \parallel p] = 0$. However, this is hardly achievable, and knowing that $KL[q \parallel p] \geq 0$, one may conclude that $\ln p(\{\mathbf{y}_t\}) > L[q]$, which is why $L[q]$ is known as lower bound on data log-likelihood. Now, to proceed with approximation, from Eq. (12), definition of lower bound Eq. (13) and KL divergence Eq. (14) one may easily notice that minimization of $KL[q \parallel p]$, which has to be performed to achieve good approximation of joint posterior with $q(\cdot)$, inevitable leads to maximization of $L[q]$. These two terms are interconnected and maximization of one implies minimization of other. However, the key point is that $L[q]$ operates on complete data log-likelihood which is more convenient and does not involve operations on the true posterior, because it cannot be evaluated.

Structured variational inference enables one to search for the solution among family of distributions $q(\cdot)$ satisfying some previously defined conditions. For example, if one proposes family of distributions that factor as:

$$q(\{\mathbf{w}_t, \mathbf{R}_t\}) = q(\{\mathbf{w}_t\})\, q(\{\mathbf{R}_t\}) \tag{15}$$

where inner statistical dependencies between state and noises are preserved but dependencies between them are omitted, the standard variational theory provides solution to the problem (Barber, 2012; Beal, 2003; Bishop, 2006). In physics, this is a common approach and it is called the mean field theory assumption. Statistical inference algorithms enables drawing of conclusions from uncertain data. In general, exact inference in robust state-space models is not possible due to dependencies between $\mathbf{w}_t$ and $\mathbf{R}_t$; as soon as new observation is received, $\mathbf{w}_t$ and $\mathbf{R}_t$ become conditionally dependent due to Eq. (2). Thus, approximate inference has to be employed. In order to build any model, we as engineers need to make some assumptions and to slowly take it from there. We need to admit that some aspects of the problem need to be approximated so as to enable any inference. With this in mind, we assumed Eq. (15); the inner statistical dependencies of $\mathbf{w}_t$ and $\mathbf{R}_t$ are kept but dependencies between them are omitted. That is, we are looking for solution of our initial problem among distributions that factor as given in Eq. (15) so as to minimize $KL[q \parallel p]$ given by Eq. (14). This assumption produces highly accurate results provided that individual terms are peaked. Robust state space models driven by state dependent noise is an active field of research and one may find interesting results in Agamennoni and Nebot (2014) but this direction is out of scope of this paper.

We make use of this setup, and based on Eq. (15) maximize Eq. (13). In Beal (2003) and Bishop (2006), it is shown that approximate posteriors of noise and state vector that maximize Eq. (13) are given with the following expressions:

$$\ln q(\{\mathbf{w}_t\}) = E_{q(\{\mathbf{R}_t\})}[\ln p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\})] + \cdots \tag{16}$$

$$\ln q(\{\mathbf{R}_t\}) = E_{q(\{\mathbf{w}_t\})}[\ln p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\})] + \cdots \tag{17}$$

where $E_{q(\cdot)}[\ln p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\})]$ stands for expectation of $\ln p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\})$ calculated with respect to the $q(\cdot)$. It should be stressed out that due to interdependences between $q(\{\mathbf{w}_t\})$ and $q(\{\mathbf{R}_t\})$, Eqs. (16) and (17) do not provide closed form analytically tractable solutions, but rather we must iterate until optimal values are reached. Important observation is related to the convergence properties of this algorithm; it is satisfied because lower bound $L[q]$ is concave.

However, it is well known fact that factorized solutions systematically underestimate posterior variance (Bishop, 2006). They tend to produce solutions that are compact, capturing the posterior mean but underestimating the posterior variance. In contrast, EP captures the posterior mean but it systematically overestimates posterior variance; EP uses different way to compute KL divergence, i.e. $KL[p \parallel q]$. This is the main advantage of VI when compared to EP: EP puts mass in regions of space where there are no data which results in poor predictive distribution. If one pictures multimodal posterior distribution, then VI tries to find one of these modes by minimizing Eq. (14), i.e. $KL[q \parallel p]$; in contrast, EP will average over all modes resulting in poor predictive distribution, which is the main reason why we have chosen VI instead of EP. In literature one may find a number of ways for improvement of EP but these methods are out of scope of this paper.

3.3. Derivation of the EKF-OR learning algorithm

Having defined the Bayesian learning formulation, introduced lower bound on the data log-likelihood, assumed family of approximate distributions that factors according to Eq. (15), and pointed out the approximate expressions for $q(\{\mathbf{w}_t\})$ and $q(\{\mathbf{R}_t\})$, we may proceed and derive approximate state posterior.

In the first step, we have to specify the complete data likelihood given as:

$$p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\}) = p(\{\mathbf{w}_1\}) \prod_{t=2}^{T} p(\mathbf{w}_t|\mathbf{w}_{t-1}) \times \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t)\, p(\{\mathbf{R}_t\}). \tag{18}$$

Taking the log of Eq. (18) gives us:

$$\ln p(\{\mathbf{w}_t, \mathbf{R}_t, \mathbf{y}_t\}) = \ln p(\{\mathbf{w}_1\}) + \sum_{t=2}^{T} \ln p(\mathbf{w}_t|\mathbf{w}_{t-1}) + \sum_{t=1}^{T} \ln p(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t) + \ln p(\{\mathbf{R}_t\}). \tag{19}$$

Taking the expectation of Eq. (19) with respect to $q(\{\mathbf{R}_t\})$ gives us Eq. (16), i.e.

$$\ln q(\{\mathbf{w}_t\}) = \ln p(\{\mathbf{w}_1\}) + \sum_{t=2}^{T} \ln p(\mathbf{w}_t|\mathbf{w}_{t-1}) + \sum_{t=1}^{T} E_{q(\{\mathbf{R}_t\})}[\ln p(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t)] + \cdots. \tag{20}$$

One may notice that the last term of Eq. (20) is quadratic with respect to the state vector $\mathbf{w}_t$. Additionally, Eq. (20) has the same form as standard EKF. Furthermore, if initial state $\mathbf{w}_1$ is Gaussian, than marginal posterior can be coded with its mean $\{\boldsymbol{\mu}_1\}$ and covariance $\{\mathbf{P}_1\}$.

Eq. (20) is similar to standard EKF recursion (Andrieu et al., 2001; Simon, 2002; Vuković, 2012; Vuković & Miljković, 2013), where one starts with initial estimate of the state $\mathbf{w}_1$ and iteratively propagates it via state transition model Eq. (6) and updates it using newest observation $\mathbf{y}_t$ via measurement update Eq. (7). The main difference is in the formulation of measurement covariance: in standard EKF, measurement covariance is treated as a constant,

whereas in EKF-OR we use expected value of the measurement covariance instead $E\left[\mathbf{R}_t^{-1}\right] = \boldsymbol{\Omega}_t^{-1}$, i.e.

$$\begin{aligned} &E_{q(\{\mathbf{R}_t\})}\left[\ln p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right)\right] \\ &= E_{q(\{\mathbf{R}_t\})}\left[\left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)^T \mathbf{R}_t^{-1}\left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)\right] \\ &= \left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)^T E_{q(\{\mathbf{R}_t\})}\left[\mathbf{R}_t^{-1}\right]\left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right) \\ &= \left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)^T \boldsymbol{\Omega}_t^{-1}\left(\mathbf{y}_t - \mathbf{g}\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right). \end{aligned} \tag{21}$$

On the other hand, to derive expression for Eq. (17), we need to specify a model for measurement noise. We shall focus on the case of independent identically distributed (IID) noise, characterized by the following prior at each time stamp

$$\mathbf{R}_t \sim W^{-1}\left(s\boldsymbol{\Omega}, s\right). \tag{22}$$

Having defined IID noise, we may proceed and provide algebraic expression for Eq. (17). Taking expectation of Eq. (19) with respect to $q\left(\{\mathbf{w}_t\}\right)$ results in:

$$\ln q\left(\{\mathbf{R}_t\}\right) = \sum_{t=1}^{T} E_{q(\{\mathbf{w}_t\})}\left[\ln p\left(\mathbf{y}_t|\mathbf{w}_t, \mathbf{R}_t\right)\right] + \sum_{t=1}^{T} \ln p\left(\mathbf{R}_t\right). \tag{23}$$

The last term in Eq. (23) implies that noise $q\left(\{\mathbf{R}_t\}\right)$ further factors as

$$q\left(\{\mathbf{R}_t\}\right) = \prod_{t=1}^{T} q\left(\mathbf{R}_t\right). \tag{24}$$

This result comes as consequence of Eq. (15) and IID assumption.

Now, Eq. (20) tells us that marginal of state vector given observations is Gaussian $\mathbf{w}_t|\{\mathbf{y}_t\} \sim N\left(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right)$. It remains to define distribution of $\mathbf{R}_t|\{\mathbf{y}_t\}$. Knowing that Eq. (22) is the conjugate prior for Eq. (2), it is easy to show the following proportionality

$$\mathbf{R}_t|\{\mathbf{y}_t\} \sim W^{-1}\left(\nu_t\boldsymbol{\Omega}_t, \nu_t\right). \tag{25}$$

That is, given observations $\{\mathbf{y}_t\}$, measurement covariance $\mathbf{R}_t$ is distributed according to inverse Wishart. Harmonic mean $\boldsymbol{\Omega}_t$ is given as:

$$\boldsymbol{\Omega}_t = \frac{s\mathbf{R} + \mathbf{S}_t}{s+1} \tag{26}$$

with $\nu_t = s+1$ degrees of freedom. Harmonic mean $\boldsymbol{\Omega}_t$ is a convex combination of the nominal noise $\mathbf{R}$ and the expected sufficient statistics matrix $\mathbf{S}_t$, calculated as:

$$\mathbf{S}_t = \left(\mathbf{y}_t - \mathbf{g}_t\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)\left(\mathbf{y}_t - \mathbf{g}_t\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)^T + \mathbf{H}\boldsymbol{\Sigma}_t\mathbf{H}^T \tag{27}$$

where $\boldsymbol{\mu}_t = \hat{\mathbf{w}}_t$ is expected value of the network parameters—the state vector $\mathbf{w}_t$. Eq. (26) implies that in the limit, when $s \to \infty$, harmonic mean $\boldsymbol{\Omega}_t$ reduces to nominal noise covariance $\mathbf{R}$.

Additionally, there is another interpretation of Eq. (27). Namely, when predicted output of the network $\mathbf{g}_t\left(\hat{\mathbf{w}}_t, \mathbf{x}_t\right)$ is close to the current measurement (data point) $\mathbf{y}_t$, approximate noise remains the same as the nominal $\mathbf{R}$; on the other hand, when there is big difference between predicted output $\mathbf{g}_t\left(\hat{\mathbf{w}}_t, \mathbf{x}_t\right)$ and current measurement $\mathbf{y}_t$, matrix $\mathbf{S}_t$ will be larger than $\mathbf{R}$, and hence it will dominate in Eq. (26). As a consequence, harmonic mean $\boldsymbol{\Omega}_t$ will be larger than $\mathbf{R}$, which in turn will result in down-weighting of the measurement $\mathbf{y}_t$ since it is being treated as an outlier (it greatly differs from prediction). This means that, there is no need for preprocessing of the data, labeling of outliers, or implementation of similar approaches for separation of outliers from the rest of data in this setup; processing of outliers is carried out sequentially within learning algorithm.

Having introduced basics of Bayesian learning, how robustness is included, and how the variational inference is iteratively used to update EKF-OR parameters, the learning algorithm is given as follows and it may be seen in Algorithm 1:

Algorithm 1
Learning algorithm based on Extended Kalman Filter Robust to Outliers EKF-OR (IID noise case)
input$(\mathbf{g}(\cdot,\cdot), J, p_0, q_0, r_0)$

1. **For each observation**$(\mathbf{x}_t, \mathbf{y}_t)$, $\mathbf{t} = \mathbf{1}, \ldots, \mathbf{n}$ **do**
1.1 Predict state vector and covariance
$\boldsymbol{\mu}_{t|t-1} = \mathbf{w}_{t-1}$; $\mathbf{P}_{t|t-1} = \mathbf{P}_{t-1} + \mathbf{Q}$
1.2 Set
$\mathbf{m}_t \leftarrow \mu_{t|t-1}$; $\mathbf{M}_t \leftarrow \mathbf{P}_{t|t-1}$
2. **While**
2.1 Update noise covariance given state
2.2 $\mathbf{S}_t \leftarrow \left(\mathbf{y}_t - \mathbf{g}_t\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)\left(\mathbf{y}_t - \mathbf{g}_t\left(\boldsymbol{\mu}_t, \mathbf{x}_t\right)\right)^T + \mathbf{H}\mathbf{P}_t\mathbf{H}^T$
2.3 $\boldsymbol{\Omega}_t \leftarrow \frac{s\mathbf{R}+\mathbf{S}_t}{s+1}$
2.4 Update state given noise
2.5 $\mathbf{K}_t \leftarrow \mathbf{M}_t\mathbf{H}^T\left(\mathbf{H}\mathbf{M}_t\mathbf{H}^T + \boldsymbol{\Omega}_t\right)^{-1}$
2.6 $\boldsymbol{\mu}_t \leftarrow \mathbf{m}_t + \mathbf{K}_t\left(\mathbf{y}_t - \mathbf{g}\left(\mathbf{m}_t, \mathbf{x}_t\right)\right)$
2.7 $\mathbf{P}_t \leftarrow \left(\mathbf{I} - \mathbf{K}_t\mathbf{H}\right)\mathbf{M}_t\mathbf{H}$
EndWhile
3. **EndFor**
output

In the first step, we need to define basic neural network parameters such as: (i) type of FFNN (Multilayer perceptron—MLP, radial basis function—RBF, functional link—FL network), type of neuronal activation function (hyperbolic tangent, sigmoid, Gaussian, Hyper basis Gaussian, inverse quadratic, Chebyshev, Legendre, etc.) and number of neurons J, (ii) EKF parameters: initial state uncertainty $\mathbf{P}_0 = p_0\mathbf{I}_\mathbf{P}$, state transition uncertainty $\mathbf{Q} = q_0\mathbf{I}_\mathbf{Q}$, and measurement uncertainty $\mathbf{R} = r_0\mathbf{I}_\mathbf{R}$, where $\mathbf{I}_\mathbf{P}$, $\mathbf{I}_\mathbf{Q}$, and $\mathbf{I}_\mathbf{R}$ are the identity matrices of appropriate dimensions.

In the first step of the learning algorithm, the state vector is propagated via process Eq. (6); only the covariance changes in this process, due to addition with estimated error covariance of the state vector. The second step implies introduction of new variables needed for iteratively solving Eqs. (20) and (23); the parameters take values of the predicted state vector and its associated covariance.

In the second step of the EKF-OR learning algorithm, we have to take new measurement $\mathbf{y}_t$ and generate estimate of the state vector $\mathbf{w}_t$. To do this, an iterative procedure is applied. Namely, as it has been stated, to solve Eqs. (16) and (17), i.e. Eqs. (20) and (23), for approximate distributions $q\left(\{\mathbf{w}_t\}\right)$ and $q\left(\{\mathbf{R}_t\}\right)$ we have to apply iterative procedure because no closed form solution exist. Firstly, we estimate $q\left(\{\mathbf{R}_t\}\right)$ given our best current estimate of the state vector's parameters: in 2.2 expected sufficient statistics matrix $\mathbf{S}_t$ is calculated. In 2.3, we update noise with newly estimated $\mathbf{S}_t$. The second step 2.4 implies update of our current best estimate of the state vector parameters given new estimate of the noise covariance $\boldsymbol{\Omega}_t$. In 2.5 we calculate the Kalman gain, which is similar to the gain of the standard EKF, where the main difference between them is in the formulation of the noise covariance. Lines 2.6 and 2.7 perform update of $q\left(\{\mathbf{w}_t\}\right)$ parameters. To control the convergence of the algorithm, we may use innovation likelihood declaring that change in two consecutive iterations has to be greater than previously defined tolerance, otherwise the algorithm stops and processes new input pair $(\mathbf{x}_t, \mathbf{y}_t)$.

In Algorithm 1, one may see that other nonlinear Kalman filters are applicable as well. For example, unscented transformation and unscented Kalman Filter (UKF) is known to produce better results in terms of accuracy than EKF (Vuković, 2012). Similarly,

information filter (IF) may be applied as learning algorithm of neural network (Vuković, 2012). Both of these algorithms may be used instead of EKF in our learning algorithm without necessity to change the basic idea of the algorithm.

Time complexity of EKF is $O\left(n^2\right)$ for prediction step and $O\left(n^3\right)$ for update step, where n stands for the dimension of the state vector $\mathbf{w}(\mathbf{w} \in R^n)$ (Vuković & Miljković, 2013). Update step in EKF-OR implies iterative calculation of Eqs. (16) and (17), i.e. Eqs. (20) and (23), which means that update step of the filter is repeated k times until convergence (Algorithm 1, lines 2.1–2.7). Therefore, the computational complexity of EKF-OR is $O\left(kn^3\right)$, which is of the same order of magnitude as the complexity of EKF. However, it still grows linearly with number of iterations k. Coefficient k is determined by monitoring innovation likelihood to see if algorithm has converged. In experiments, EKF-OR typically needed five to six iterations in the update step. From engineering perspective, this increase in complexity of EKF-OR is a price that has to be paid in order to have learning algorithm able to sequentially process outliers and "normal" data points.

4. Experimental results

In this section of the paper we analyze performance of EKF-OR learning algorithm over real world and simulated datasets in terms of accuracy, generalization and robustness to outliers and unknown noise. Performance of the proposed EKF-OR learning algorithm for feedforward neural network training is tested over several benchmark problems for regression, system identification, and time series prediction.

4.1. Experimental setup

The first neural network is the radial basis function neural network with Gaussian processing units (GRBF). The output function of GRBF neural network is given as (Andrieu et al., 2001; Poggio & Girosi, 1989, 1990; Simon, 2002; Vuković, 2012):

$$\mathbf{y}_i = f\left(\mathbf{x}_i\right) = \sum_{j=1}^{J} w_j h_j\left(\mathbf{x}_i, \boldsymbol{\mu}_j, \sigma_j\right) \tag{28}$$

where $h_j(\cdot,\cdot,\cdot)$ stands for the jth basis function; $\mathbf{x}_i \in R^{n_x}$ is the input vector in the GRBF neural network (n_x stands for the number of the input dimensions, i.e. $n_x = \dim(\mathbf{x}_i)$), $\boldsymbol{\mu}_j \in R^{n_x}$ is the center of jth basis function and $\sigma_j \in R^1$ represents the scalar parameter; w_j is connecting weight of the jth basis function, and J stands for total number of processing units of the GRBF network. Gaussian activation function is chosen as basis function $h_j(\cdot,\cdot,\cdot)$, defined as:

$$\begin{aligned} h_j\left(\mathbf{x}_i, \boldsymbol{\mu}_j, \sigma_j\right) &= \exp\left(-0.5 \left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|^2 / \sigma_j^2\right) \\ &= \exp\left\{-0.5 \left(\mathbf{x}_i - \boldsymbol{\mu}_j\right)^T \left(\mathbf{x}_i - \boldsymbol{\mu}_j\right) / \sigma_j^2\right\}. \end{aligned} \tag{29}$$

As it can be seen, all neurons have different width parameter: $\sigma_j \in R^1$, $j = 1, \ldots, J$.

In the second and the third experiments we compare EKF-OR's performance for training of hyper basis function (HBF) network (Billings, Wei, & Balikhin, 2007; Mahdi & Rouchka, 2011; Nishida, Yamauchi, & Omori, 2006; Poggio & Girosi, 1989, 1990; Vuković & Miljković, 2013) for system identification and chaotic time series prediction. The results are compared to results presented in Chuang and Lee (2011) and Yang et al. (2013). Main characteristic of the HBF network is the ability to scale local dimensions of the input vector $\mathbf{x}_i \in R^{n_x}$. It is similar to the GRBF network and it can be seen as straight forward generalization of the simple Gaussian function, but where closeness is measured with a Mahalanobis-like distance instead of Euclidian-like distance. However, we emphasize that this is not the real Mahalanobis distance measure but rather Mahalanobis-like distance measure. In HBF network every neuron has a unique diagonal weighting matrix $\boldsymbol{\Xi}_j$, with varying size and restricted orientation: $\boldsymbol{\Xi}_j = \operatorname{diag}\left(1/\sigma_1^2, 1/\sigma_2^2, \ldots, 1/\sigma_{n_x}^2\right)$. Similarly to GRBF network, HBF network is defined with Eq. (28) but its activation function applies the Mahalanobis-like distance and it is computed in the following form:

$$\begin{aligned} h_j\left(\mathbf{x}_i, \boldsymbol{\mu}_j, \boldsymbol{\Xi}_j\right) &= \exp\left(-0.5 \left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|_{\boldsymbol{\Xi}_j}^2\right) \\ &= \exp\left\{-0.5 \left(\mathbf{x}_i - \boldsymbol{\mu}_j\right)^T \boldsymbol{\Xi}_j \left(\mathbf{x}_i - \boldsymbol{\mu}_j\right)\right\} \end{aligned} \tag{30}$$

where $\|\cdot\|_{\boldsymbol{\Xi}_j}^2$ is Mahalanobis-like norm weighted with positive definite square matrix $\boldsymbol{\Xi}_j$. Weighting matrix $\boldsymbol{\Xi}_j$ makes similarity between the input vector $\mathbf{x}_i$ and jth center vector $\boldsymbol{\mu}_j$ invariant to scaling and local orientation of the data (Mahdi & Rouchka, 2011). The Mahalanobis-like distance can be thought of as a generalization of the Euclidian distance for similarity measure.

All codes are written and run in Matlab 7.12 environment; experiments are conducted on laptop computer with Intel(R) Core™ i5-4200U CPU @ 1.6 GHz (2.3 GHz) with 6 GB of RAM, running on 64-bit Windows 7.

Artificial outliers are added to each training set: we simulate various levels of outliers' existence and their offset from nominal values. For all experiments the following procedure is adopted:

(1) Add certain number of outliers generated according to predefined mechanism.
(2) Train feedforward neural network using EKF-OR with training set contaminated by outliers.
(3) Test performance of optimized feedforward neural network using test set free of outliers.

The accuracy is measured with root mean square error (RMSE) defined as:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \left(\mathbf{y}_t - \hat{\mathbf{y}}_t\right)^T \left(\mathbf{y}_t - \hat{\mathbf{y}}_t\right)}. \tag{31}$$

The lower numerical value of RMSE for testing set free of outliers implies better generalization.

4.2. Real world datasets

In the first experiment we directly compare performance of EKF-OR versus standard EKF for training of RBF network with Gaussian processing units. The first dataset is *Auto MPG*; it predicts fuel consumption in miles per gallon of the car using information encoded in seven attributes of input vector. The second dataset *Weather Ankara* predicts mean air temperature in Ankara given measurements. The third dataset *Boston housing* deals with prediction of the mean price of the houses in Boston. The main characteristics of datasets are given in Table 1, while precise information may be found in Bache and Lichman (2013) and Guvenir and Uysal (2000). Prior to training, both input and output variables are normalized so that each dimension has the same impact during learning process.

The experimental procedure is carried out in the following manner. EKF-OR is tested against standard EKF learning algorithm. In the first case, the training set is free of outliers and it is formed of 70% of available data, while testing set consists of the 30% of available data (Table 1—column "Without outliers"). We tested five distinct GRBF network structures with five, ten, 15, 20 and 30 neurons in the processing layer. To obtain statistically significant

Table 1
Specification of benchmark datasets. n_x denotes the number of attributes (dimension of input vector $\mathbf{x}_i$), N—the total number of examples, N_{train}—the number of training examples, and N_{test}—the number of testing examples.

Dataset	Type of data	Type of problem	n_x	N	Without outliers		With artificially added outliers	
					N_{train}	N_{test}	N_{train}	N_{test}
Boston housing	Real	Regression	13	506	354	152	506	506
Weather Ankara	Real	Regression	9	1609	1100	509	1609	1609
Auto MPG	Real	Regression	7	398	320	78	398	398

Table 2
Experimental results obtained with RBF network consisting of J Gaussian processing units for real world datasets free of outliers. Table shows RMSE and standard deviation of the RBF network trained with EKF and EKF-OR averaged over 50 independent runs.

	J	EKF	EKF-OR
Auto MPG	5	0.2146 0.0585	0.1870 0.0375
	10	0.2301 0.0801	0.1998 0.0475
	15	0.2179 0.0590	0.1980 0.0518
	20	0.2208 0.1053	0.1991 0.0551
	30	0.2011 0.0505	0.1984 0.0517
Weather Ankara	5	0.0479 0.0113	0.0477 0.0114
	10	0.0480 0.0115	0.0475 0.0110
	15	0.0467 0.0097	0.0467 0.0094
	20	0.0479 0.0126	0.0485 0.0127
	30	0.0439 0.0072	0.0438 0.0078
Boston housing	5	0.4042 0.1309	0.2585 0.0268
	10	0.3196 0.1303	0.2470 0.0381
	15	0.2972 0.0634	0.2504 0.0459
	20	0.2785 0.0564	0.2370 0.0361
	30	0.2377 0.0462	0.2225 0.0372

results each experiment is repeated 50 times and we report RMSE calculated for testing set.

Table 2 shows mean RMSE and standard deviation calculated over the 50 independent runs, while Fig. 3 depicts visual representation of experimental results. As it may be seen, when training set is free of outliers the performance of EKF-OR is comparable to the performance of EKF. EKF-OR produces slightly lower RMSE but for all three datasets we may conclude that testing RMSE is lower for EKF-OR than EKF. Lower RMSE is result of approximate variational inference carried out in the update step of the EKF-OR learning algorithm; iterative procedure improves estimates of the network parameters given recent measurements.

In the second step of experiment, we artificially introduced outliers in the training set so as to see the EKF-OR's performance when dealing with real data polluted with outliers. Nine different experiments were performed for each of five distinct RBF network topologies (five, ten, 15, 20 and 30 processing units). Firstly, all available data are polluted by three different levels of outliers' contamination: (i) 5%, (ii) 10% and (iii) 20% of data is corrupted by outliers. Outliers are created by replacing some randomly chosen target value of the training instances by random values drawn from $N\left(0, \beta^2\right)$. Secondly, the variance β^2 is varied as: (i) $\beta^2 = 0.01$, (ii) $\beta^2 = 0.05$, and (iii) $\beta^2 = 0.1$. Therefore, each RBF topology is tested for nine distinct settings of training set corrupted by outliers. Each experiment is performed over 50 independent runs; as explained in introductory part of "Experimental results" we report testing RMSE calculated for all available data (training data) free of outliers; the reader is referred to Table 1 for the number of training and testing data points—column "With artificially added outliers". Experimental results are presented in Tables 3–5 while Figs. 4–6 show visual representation of mean RMSE. Each row in Figs. 4–6 shows experimental results for different setting of variance β^2, while columns show how the percentage of outliers in dataset changes performance of tested learning algorithms.

Results show that, for each setting of outliers in training set (pct of outliers and various levels of outliers' variance β^2) EKF-OR outperforms EKF in terms of accuracy, calculated as RMSE over testing set free of outliers. Tables 3–5 show that for all three datasets increase of outliers in training set implies better performance of EKF-OR than EKF. Similarly, when variance is increased, the performance of EKF decreases, when compared to EKF-OR; it achieves better generalization for datasets contaminated by outliers and it is able to learn the real relationship between input–output pairs of data. These conclusions are in congruence with initial intuition: the network trained with EKF-OR should be robust to outliers in data, able to learn relationship between variables and generalize, regardless of presence of outliers in data.

4.3. Nonlinear dynamical system

In this simulated example we analyze performance of EKF-OR when faced with system identification of unknown dynamical system using training data contaminated by outliers. Consider nonlinear dynamical system given as:

$$y(k+1) = \frac{y(k)\, y(k-1)\, y(k-2)\, u(k-1)\, (y(k-2)-1) + u(k)}{1 + y^2(k-2) + y^2(k-1)} \quad (32)$$

where $u(k)$ is the control variable (identically independently distributed) in uniform range $[-2, 2]$. 800 data points are generated for training of the model. Data used for training Eq. (32) contains outliers and noise modeled as gross error model:

$$D_\delta = \{D | D = (1-\delta)G + \delta F, 0 \le \delta \le 1\} \quad (33)$$

where δ represents probability of occurrence of outlier, G and F are defined as Gaussians $G \sim N(0, 0.1)$ and $F \sim N(0, 1)$; G and F occur with probabilities $1-\delta$ and δ, respectively. The gross error model does not guarantee the exact number of outliers given with probability of occurrence $\delta = 0.05$; the total number of outliers may be less or more than specific threshold δ. Testing set is defined with following dynamics of the control input $u(k)$:

$$u(k) = \begin{cases} \sin(3\pi k/250), & k \le 500 \\ 0.25 \sin(2\pi k/250) + 0.2 \sin(3\pi k/50), & k > 500. \end{cases} \quad (34)$$

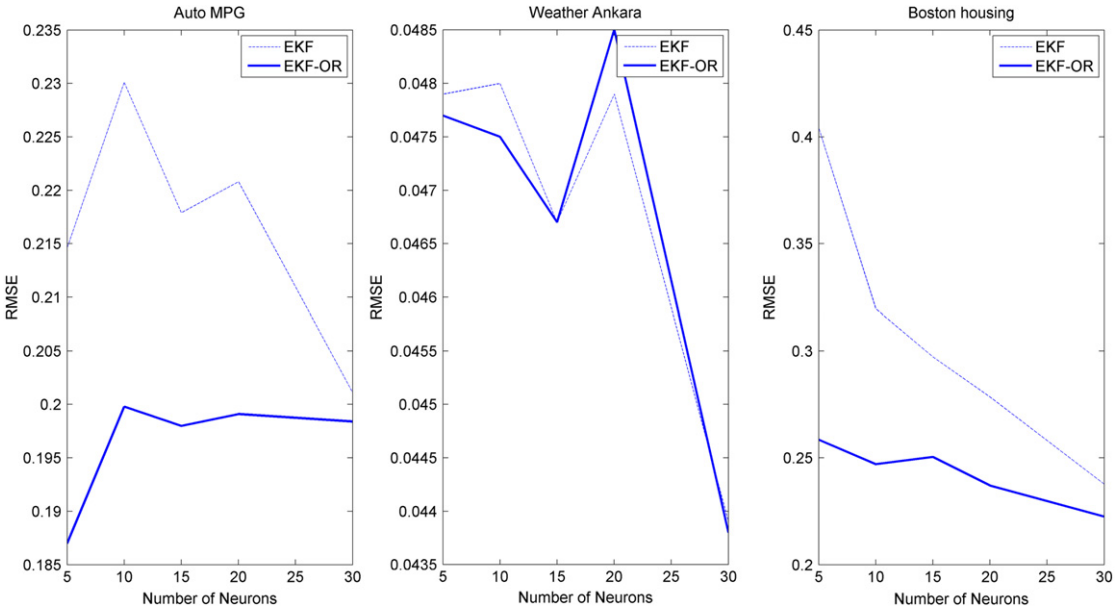


Fig. 3. Graphical representation of mean RMSE averaged over 50 trials for real world datasets.

Table 3
Experimental results obtained with RBF network consisting of J Gaussian processing units for *Auto MPG* real world dataset contaminated by outliers (5%, 10% and 20%). Additionally, variance of mechanism that generates outliers is varied as well. Table shows RMSE and standard deviation for testing set free of outliers averaged over 50 independent runs for EKF and EKF-OR.

	J	β^2	Pct. of outliers in training set					
			5%		10%		20%	
			EKF	EKF-OR	EKF	EKF-OR	EKF	EKF-OR
Auto MPG	5	0.01	0.2473 0.0716	0.2041 0.0496	0.2948 0.1047	0.2191 0.0446	0.3413 0.1724	0.2306 0.0498
	10		0.2354 0.0683	0.1991 0.0420	0.2667 0.0890	0.2207 0.0500	0.2670 0.1012	0.2229 0.0609
	15		0.2378 0.0575	0.2094 0.0557	0.2735 0.0957	0.2177 0.0462	0.3053 0.1573	0.2292 0.0600
	20		0.2166 0.0626	0.1932 0.0447	0.2532 0.0902	0.2069 0.0386	0.3078 0.0930	0.2505 0.0647
	30		0.2276 0.0823	0.2199 0.0692	0.2665 0.1166	0.2471 0.0772	0.2876 0.1254	0.2564 0.0880
	5	0.05	0.3334 0.1251	0.2508 0.0936	0.4542 0.2105	0.2613 0.0695	0.4733 0.2149	0.3258 0.1044
	10		0.2954 0.1016	0.2229 0.0502	0.3707 0.1947	0.2579 0.0649	0.4268 0.2200	0.3095 0.1561
	15		0.2756 0.0976	0.2285 0.0605	0.3914 0.2082	0.2694 0.0867	0.4457 0.2587	0.3078 0.1350
	20		0.2564 0.0732	0.2308 0.0592	0.3732 0.1895	0.2749 0.0939	0.4239 0.1773	0.3217 0.1207
	30		0.2909 0.1024	0.2531 0.0912	0.3461 0.1688	0.2743 0.1243	0.4085 0.2195	0.3423 0.1770
	5	0.1	0.4009 0.2017	0.2544 0.0910	0.4879 0.2088	0.3069 0.1124	0.6343 0.3811	0.3767 0.1671
	10		0.3904 0.2129	0.2386 0.0640	0.4758 0.2103	0.2929 0.1098	0.5121 0.2144	0.3438 0.1335
	15		0.3286 0.1369	0.2661 0.0738	0.4133 0.2236	0.2805 0.1228	0.5871 0.3472	0.3150 0.1213
	20		0.3431 0.1693	0.2416 0.0636	0.4357 0.2121	0.3106 0.1289	0.4815 0.2014	0.3275 0.1248
	30		0.3374 0.1553	0.2721 0.1183	0.4234 0.2027	0.3406 0.1519	0.4755 0.2088	0.3649 0.1352

Specifications of the dataset are given in Table 6. The goal is to build neural model of the system Eq. (32) using HBF neural network (Eq. (28) with activation function given in Eq. (30)) trained with EKF-OR. Input vector is $\mathbf{x} = [y(k), y(k-1), y(k-2), u(k), u(k-1)]$, whereas output is given as scalar $y_p = [y(k+1)]$.

All available data generated according to Eq. (32) is corrupted by outliers given by gross error model Eq. (33); the testing set is free of

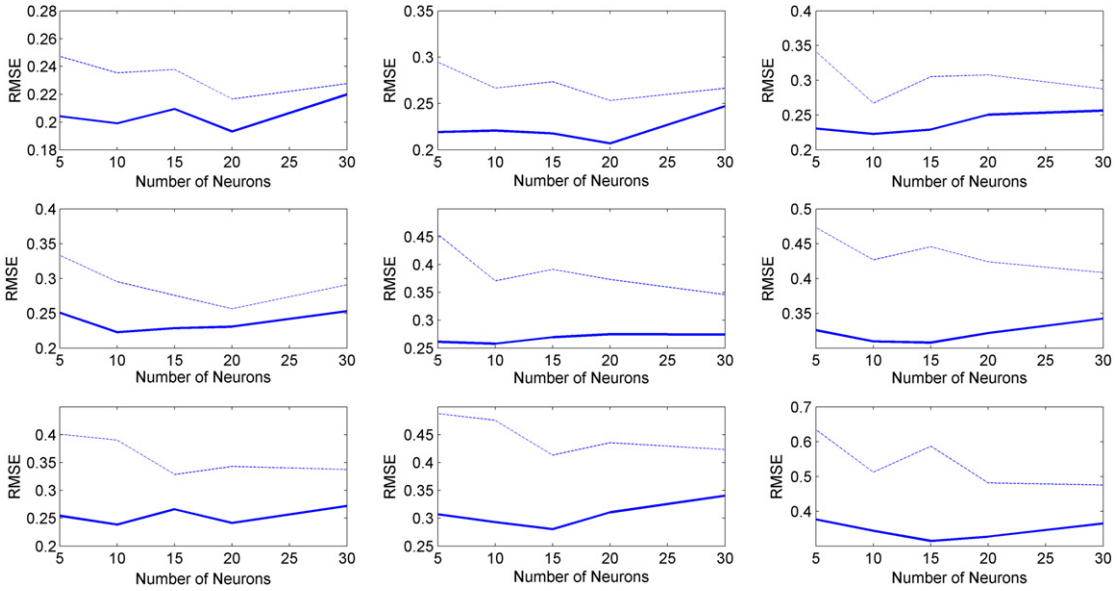


Fig. 4. Mean RMSE of EKF and EKF-OR for *Auto MPG* dataset averaged over 50 trials given in Table 3. From top to the bottom row the variance β^2 is varied (top row $\beta^2 = 0.01$, middle row $\beta^2 = 0.05$, and bottom row $\beta^2 = 0.1$). Percentage of outliers is varied along columns from left to right (first column shows results for 5%, second for 10% and rightmost column for 20%). Solid line represents EKF-OR while dotted line depicts EKF.

Table 4
Experimental results obtained with RBF network consisting of J Gaussian processing units for *Weather Ankara* real world dataset contaminated by outliers (5%, 10% and 20%). Additionally, variance of mechanism that generates outliers is varied as well. Table shows RMSE and standard deviation for testing set free of outliers averaged over 50 independent runs for EKF and EKF-OR.

	J	β^2	Pct. of outliers in training set					
			5%		10%		20%	
			EKF	EKF-OR	EKF	EKF-OR	EKF	EKF-OR
Weather Ankara	5	0.01	0.1095	0.0859	0.1744	0.1012	0.2286	0.1219
			0.0957	0.0432	0.1689	0.0563	0.2045	0.0728
	10		0.1032	0.0865	0.1635	0.1104	0.1802	0.1338
			0.0798	0.0577	0.1419	0.0748	0.1190	0.0820
	15		0.1012	0.0878	0.1637	0.1127	0.1726	0.1335
			0.0581	0.0497	0.1273	0.0631	0.1419	0.1012
	20		0.1003	0.0857	0.1522	0.1051	0.1765	0.1426
			0.0692	0.0461	0.0919	0.0492	0.1222	0.0877
	30		0.1011	0.0900	0.1390	0.1014	0.1614	0.1225
			0.0691	0.0586	0.1162	0.0547	0.1271	0.0690
	5	0.05	0.2511	0.1133	0.3384	0.1553	0.5546	0.2231
			0.2025	0.0572	0.2741	0.0922	0.3582	0.1296
	10		0.2387	0.1254	0.3213	0.1793	0.5406	0.2543
			0.2383	0.0815	0.2398	0.1445	0.4657	0.2221
	15		0.2236	0.1370	0.2498	0.1497	0.4758	0.1865
			0.2099	0.0846	0.1988	0.0967	0.3632	0.1135
	20		0.1791	0.1136	0.2558	0.1581	0.4219	0.2237
			0.1603	0.0735	0.2298	0.1041	0.2622	0.1461
	30		0.1747	0.1334	0.2375	0.1511	0.3789	0.1956
			0.1503	0.1146	0.1869	0.1009	0.3165	0.1423
	5	0.1	0.3670	0.1785	0.5867	0.1952	0.6895	0.3060
			0.2653	0.1307	0.3196	0.1321	0.5373	0.1976
	10		0.2725	0.1429	0.4450	0.1849	0.6448	0.2726
			0.1933	0.1043	0.3042	0.1465	0.4780	0.2174
	15		0.3113	0.1542	0.4502	0.1898	0.6800	0.2398
			0.2325	0.0978	0.3445	0.1205	0.5793	0.1377
	20		0.3245	0.1384	0.4623	0.1956	0.7516	0.2440
			0.3250	0.1083	0.4266	0.1362	0.6798	0.1646
	30		0.3030	0.1512	0.3922	0.1861	0.6539	0.2789
			0.2874	0.1206	0.2951	0.1156	0.5575	0.1985

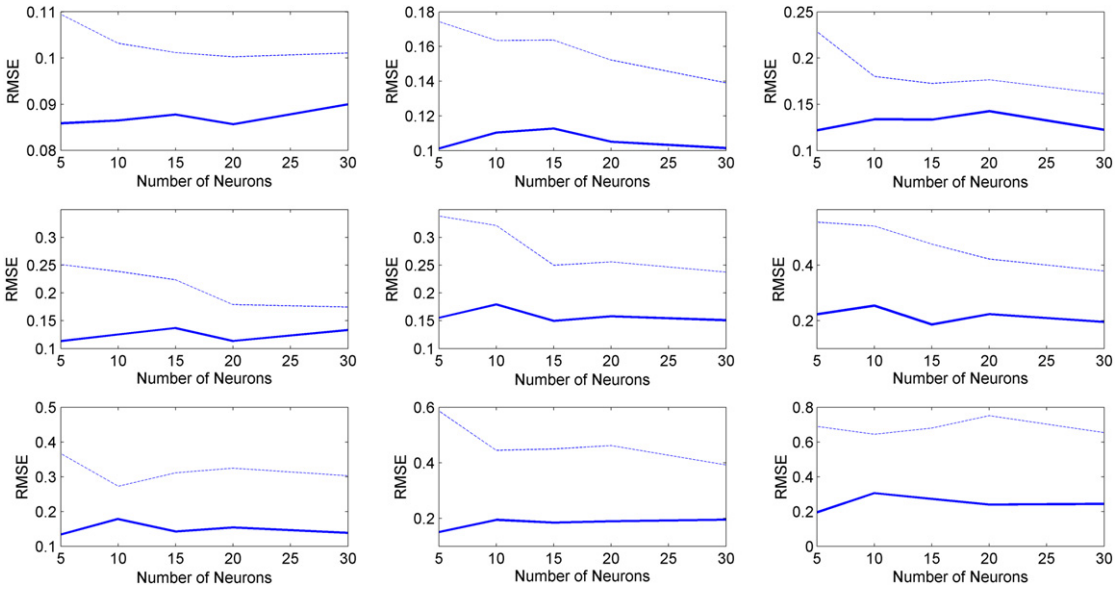


Fig. 5. Mean RMSE of EKF and EKF-OR for *Weather Ankara* dataset averaged over 50 trials given in Table 4. From top to the bottom row the variance β^2 is varied (top row $\beta^2 = 0.01$, middle row $\beta^2 = 0.05$, and bottom row $\beta^2 = 0.1$). Percentage of outliers is varied along columns from left to right (first column shows results for 5%, second for 10% and rightmost column for 20%). Solid line represents EKF-OR while dotted line depicts EKF.

Table 5
Experimental results obtained with RBF network consisting of J Gaussian processing units for *Boston housing* real world dataset contaminated by outliers (5%, 10% and 20%). Additionally, variance of mechanism that generates outliers is varied as well. Table shows RMSE and standard deviation for testing set free of outliers averaged over 50 independent runs for EKF and EKF-OR.

	J	β^2	Pct. of outliers in training set					
			5%		10%		20%	
			EKF	EKF-OR	EKF	EKF-OR	EKF	EKF-OR
Boston housing	5	0.01	0.4430	0.2690	0.4354	0.2753	0.4868	0.3071
			0.1445	0.0468	0.1375	0.0495	0.1647	0.0543
	10		0.3755	0.2648	0.3855	0.2702	0.4298	0.3046
			0.1224	0.0469	0.1291	0.0436	0.1469	0.0712
	15		0.3321	0.2684	0.3426	0.2685	0.4144	0.2927
			0.1182	0.0494	0.1279	0.0439	0.1506	0.0531
	20		0.3139	0.2588	0.3451	0.2781	0.4433	0.3074
			0.1147	0.0373	0.1072	0.0410	0.1593	0.0683
	30		0.3234	0.2701	0.3352	0.2843	0.4074	0.3160
			0.0862	0.0600	0.1177	0.0578	0.1399	0.0839
	5	0.05	0.4868	0.3144	0.4989	0.3725	0.6331	0.4164
			0.1800	0.0707	0.1266	0.1026	0.2630	0.1230
	10		0.4611	0.3026	0.5437	0.3419	0.5605	0.3833
			0.1688	0.0613	0.1560	0.0946	0.2332	0.1403
	15		0.4328	0.2998	0.4560	0.3279	0.5246	0.4091
			0.1924	0.0735	0.1474	0.0640	0.1668	0.1234
	20		0.4344	0.3043	0.4817	0.3291	0.4960	0.3816
			0.1737	0.0687	0.1884	0.0747	0.1691	0.1478
	30		0.4313	0.3129	0.4430	0.3275	0.5055	0.3706
			0.1759	0.0626	0.1526	0.0762	0.1584	0.1012
	5	0.1	0.5423	0.3481	0.5472	0.4055	0.7643	0.5055
			0.1595	0.0882	0.1710	0.1096	0.4746	0.1584
	10		0.4669	0.3410	0.5606	0.3387	0.7643	0.4436
			0.1378	0.0904	0.1894	0.0813	0.4746	0.1182
	15		0.5077	0.3128	0.5536	0.4115	0.6553	0.4223
			0.1933	0.0618	0.1803	0.1542	0.3358	0.1635
	20		0.4447	0.3166	0.5686	0.3504	0.6927	0.4053
			0.1560	0.0756	0.2036	0.0888	0.3244	0.1112
	30		0.4824	0.3171	0.5172	0.3670	0.7729	0.4502
			0.1732	0.0646	0.1777	0.0979	0.4661	0.1976

outliers—Eq. (34). Therefore, there are 800 data points for training (contaminated by outliers) and 800 data points free of outliers—Table 6.

HBF neural network is trained for 50 independent runs and experimental results are given in Table 7. The performance of HBF EKF-OR is compared to results of the least square support vector

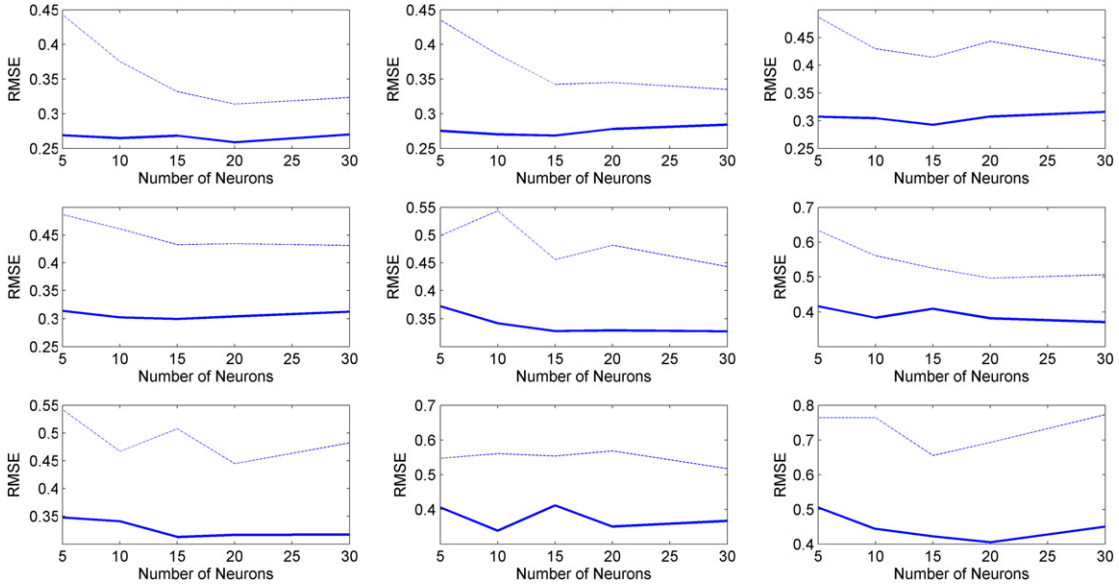


Fig. 6. Mean RMSE of EKF and EKF-OR for *Boston housing* dataset averaged over 50 trials given in Table 5. From top to the bottom row the variance β^2 is varied (top row $\beta^2 = 0.01$, middle row $\beta^2 = 0.05$, and bottom row $\beta^2 = 0.1$). Percentage of outliers is varied along columns from left to right (first column shows results for 5%, second for 10% and rightmost column for 20%). Solid line represents EKF-OR while dotted line depicts EKF.

Table 6
Specification of simulated benchmark problem *nonlinear dynamical system*. The training set is contaminated with artificially added outliers modeled as gross error model with probability of occurrence of an outlier $\delta = 0.05$.

Dataset	Type of data	Type of problem	n_x	N	N_{train}	N_{test}
Nonlinear dynamical system	Simulated	System identification	5	800	800	800

machines for regression (LS-SVMR), non-robust support vector regression network (SVRN) and robust support vector regression network (RSVRN) given in Chuang and Lee (2011). As experimental results given in Table 7 show, HBF EKF-OR outperforms all algorithms whose performance for this training set is tested and given in Chuang and Lee (2011). The root mean square error calculated for testing set free of outliers is smaller for HBF trained with EKF-OR. The lowest value is RMSE = 0.0228 while the highest value is RMSE = 0.0771. For this highest value of RMSE, performance of HBF EKF-OR is at the same level of generalization as SVRN and LS-SVMR. For other tested examples, HBF EKF-OR generates lower testing RMSE.

One may notice that the smallest RMSE in all cases is achieved when HBF network is set to have ten HBF processing units. However, even HBF structures with five neurons generate low value of the testing RMSE.

Parameter s, the degrees of freedom of inverse Wishart distribution, has interesting interpretation: when we increase s we put more emphasis on the measurement covariance matrix $\mathbf{R}_t$ instead on the sufficient statistics matrix $\mathbf{S}_t$ (Eq. (26)). Increase in s results in more accurate HBF network; as experimental results in Table 7 indicate, EKF-OR optimizes HBF network parameters and produces lower RMSE for testing set for lowest setting of $s = 10$. Fig. 7 shows output of HBF network trained with proposed learning algorithm EKF-OR given test signal free of outliers Eq. (34).

Knowing that total number of outliers in training set may be higher than predefined probability level of their occurrence $\delta = 0.05$, taking experimental results given in Table 7 and Fig. 7 into consideration, it is safe to conclude that EKF-OR is able to identify outliers and treat them as any other data point in the training set; there is no need for preprocessing of training data to identify and eliminate outliers. Structured variational approach enables EKF-OR to down-weight outliers during sequential learning. This feature

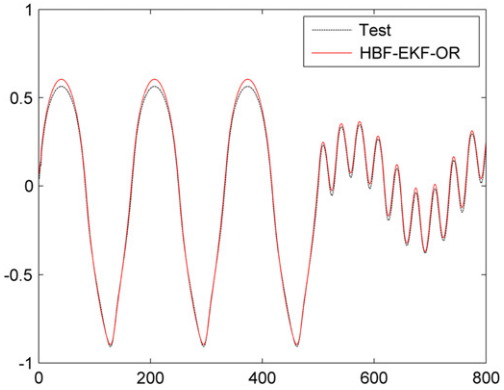


Fig. 7. Approximated *nonlinear dynamical system* with neural network trained with EKF-OR—test output versus generated output of HBF-EKF-OR neural network after learning. Training set contains outliers modeled as gross error model with probability of occurrence of an outlier $\delta = 0.05$. Test set is free of outliers.

of EKF-OR is important, especially for engineering problems when fast neural network models are necessary given data contaminated with outliers.

4.4. Mackey–Glass chaotic time series prediction

The final study is devoted to the Mackey–Glass chaotic time series, which has been recognized as one of the benchmark problems for assessing performance of learning algorithms. The time series is generated with following discrete equation:

$$x(t+1) = ax(t) + bx(t-\tau) / \left(1 + x^{10}(t-\tau)\right). \quad (35)$$

For $a = 0.1$, $b = 0.2$, $\tau = 17$ and choosing $x(0) = 1.2$ the goal is to predict $x(t + \Delta t)$ as a function of previous states given by:

$$x(t+\Delta t) = f\left(x(t), x(t-6), x(t-12), x(t-18)\right). \quad (36)$$

Table 7
Experimental results obtained for *nonlinear dynamical system* with gross error model for outliers and noise. Table shows performance of HBF neural network trained with EKF-OR learning algorithm; performance is measured in terms of RMSE calculated for testing set free of outliers, averaged over 50 independent runs. Number of HBF processing units J and degrees of freedom s are varied so as to assess their influence on EKF-OR's performance. Standard deviations of testing RMSE for SVRN, LS-SVMR, RSVRN and weighted LS-SVMR are not stated in original paper (Chuang & Lee, 2011).

	J	s	RMSE
HBF EKF-OR	1	5	0.0771 ± 0.0028
	5		0.0761 ± 0.0097
	10		$\mathbf{0.0337 \pm 0.0090}$
	1	10	$\mathbf{0.0646 \pm 0.0069}$
	5		$\mathbf{0.0291 \pm 0.0038}$
	10		$\mathbf{0.0228 \pm 0.0055}$
	1	25	$\mathbf{0.0675 \pm 0.0070}$
	5		$\mathbf{0.0687 \pm 0.0021}$
	10		$\mathbf{0.0327 \pm 0.0087}$
SVRN (Chuang & Lee, 2011)			$0.0754 \pm$ NA
LS-SVMR (Chuang & Lee, 2011)			$0.0799 \pm$ NA
RSVRNs (Chuang & Lee, 2011)			$0.0965 \pm$ NA
Weighted LS-SVMR (Chuang & Lee, 2011)			$0.1084 \pm$ NA

500 data points are generated for training ($t = 1, \ldots, 500$) and another 500 data points for testing ($t = 501, \ldots, 1001$). Problem characteristics are given in Table 8. 500 training data points are contaminated by outliers while the testing set, consisting of another 500 data points, is free of outliers— Table 8.

As in previous experiments, outliers have been artificially added to training set; we tested performance of EKF-OR learning algorithm in terms of RMSE calculated for testing set free of outliers. Table 9 shows experimental results averaged over 30 independent trials. Experimental tests were performed with six different setups of training data contaminated by outliers, i.e. 0%, 10%, 20%, 30%, 40% and 50% of outliers.

As it can be seen, EKF-OR optimizes HBF parameters in such manner that it shows stable performance of network regardless of percentage of outliers in the data. For example, it can be seen that for testing case when there are no outliers in the data, the increase in number of neurons results in more accurate network. Quite different can be concluded when number of outliers increase; bigger number of processing units results in less accurate network structures. When outliers are present, the optimal number of HBF neurons to be trained with EKF-OR is in range five to ten.

The results are compared to least square generalized radial basis function network fuzzy system (LS-GRBFNFS) and least Wilcoxon generalized radial basis function networks fuzzy system (LW-GRBFNFS RBFNS) proposed and analyzed in Yang et al. (2013). It should be stressed that LW-GRBFNS is developed to address outliers in the training set. Yang et al. show that LW-GRBFNS is more accurate than conventional approaches in terms of accuracy when training set is contaminated by outliers. To allow fair comparison with results presented in Yang et al. (2013) we adopted the same values for parameters a, b and τ as in Yang et al. (2013).

As experimental results given in Table 9 reveal, HBF EKF-OR easily outperforms LS-GRBFNFS; this is expected result knowing that LS-GRBFNFS is based on conventional least squares approach when it comes to optimization. As discussed in introductory part of the paper, this approach fails when data contains outliers due to unboundedness of the influence function. On the other hand, LW-GRBFNFS generates more accurate estimates than LS-GRBFNS. Furthermore, it outperforms proposed HBF EKF-OR for all setups except for the hardest one (50% of outliers in data). When percentage of outliers increases to 50% HBF EKF-OR generates estimates of the test signal which are by order of magnitude smaller than LW-GRBFNFS. It should be stressed that LW-GRBFNFS is based on integration of radial basis function network and fuzzy system, trained with Particle Swarm Optimization (PSO) algorithm; neural fuzzy integration allows more information to be extracted from training set. On the other hand, the performance of HBF EKF-OR is surprisingly comparable to the performance of LW-GRBFNFS when percentage of outliers in training set increases.

Fig. 8 shows the performance of HBF neural network trained with EKF-OR for Mackey–Glass time series. It may be seen that HBF EKF-OR is able to learn the real mechanism that generates the data given with Eq. (35), regardless of level of training data contamination by outliers.

4.5. Conclusion of experiments

Experimental results obtained using EKF-OR for real data regression with outliers show that EKF-OR outperforms standard EKF in terms of accuracy and generalization (Tables 3–5). Different setups of outliers in terms of their number in training set and offset from training data proved EKF-OR's ability to process outliers and generate good quality estimates of the network parameters. In most extreme cases of outlier presence, EKF-OR outperforms standard EKF by ∼20%–30% where accuracy is measured with RMSE defined with Eq. (31) calculated for testing set of data free of outliers.

Experiments with synthetic data for benchmark problems show how degrees of freedom s Eq. (26) enables user to control EKF-OR; bigger values of s puts more emphasis on measurement noise covariance $\mathbf{R}_t$ and less on sufficient statistics $\mathbf{S}_t$ (Eq. (26)). When compared to experimental results given in Chuang and Lee (2011), Table 7 reveals that EKF-OR is able to generalize better in presence of outliers. Experiment results obtained using Mackey–Glass chaos time series prediction show stable performance of EKF-OR as learning algorithm over different settings of outliers' presence. EKF-OR outperforms LS-GRBFNS for all settings of outliers (Table 9). When percentage of outliers increases from 0% to 50% EKF-OR's increase in testing RMSE ranges from ∼0% to 40%. Compared to results given in Yang et al. (2013) this decrease in generalization ability is far less than LS-GRBFNS and LW-GRBFNS. LW-GRBFNS outperforms EKF-OR for all settings of outlier presence except for 50% of outliers in training set. In this case, EKF-OR produces testing RMSE which is by order of magnitude less than LW-GRBFNS. Advantages of EKF-OR over LS-GRBFNFS and LW-GRBFNFS are twofold: (i) firstly, EKF-OR is sequential learning algorithm and it can run *on-line*; LS-GRBFNFS and LW-GRBFNFS cannot perform learning process *on-line*; (ii) secondly, both LS-GRBFNFS and LW-GRBFNFS require two stage learning process; EKF-OR is a one stage learning process and it does not require additional steps for learning. These advantages of EKF-OR are important for engineering applications of neural networks.

It is known fact that HBF network may cause overfitting to the training data (Mahdi & Rouchka, 2011; Nishida et al., 2006; Vuković & Miljković, 2013) which is not acceptable, especially in engineering applications of neural networks. When data are polluted by outliers than this fact gains on importance. This is the reason why we decided to test performance of the HBF network trained with EKF-OR and see if it can prevent overfitting to training data. As experimental results given in Tables 7 and 9 indicate, HBF network trained with EKF-OR using training data heavily polluted by outliers is able to generalize and sequentially down-weight outliers.

Although experiments have proved EKF-OR's ability to handle outliers in training data some shortcomings need to be stated. In our experiments we noticed that it takes a great number of trial

and error attempts to find optimal values (for example degrees of freedom s). None of the numerical values of parameters is meant to be optimal; we report EKF-OR's performance but for optimal settings another research has to be taken. All parameters are problem specific.

We assumed IID noise case which enables us to derive Eq. (17), but we would like to emphasize that this assumption is not the real situation in engineering applications of neural networks; one may find a number of examples in engineering practice in which noise is not IID. However, our robust sequential learning algorithm for feedforward neural networks enables us to extend our basic setup to more general case. For example, if we assume that noise covariance matrix obeys first order dynamics – slowly drifting noise model as in Agamennoni et al. (2012) or as in Sarkka and Nummenmaa (2009) – our methodology enables us to implement it into inference. On the other hand, in conducted experiments (Sections 4.3 and 4.4) none of the noise mechanism that generates noise and outliers is IID, nevertheless EKF-OR still manages to "recognize" outliers and process them as any other data point, regardless of IID assumption of noise. We believe that this fact is due to: (i) measurements are Student t distributed which is more robust to outliers and less sensitive to infinitesimal changes in input data; (ii) variational inference enables downsizing of outliers using Bayesian framework.

Our learning algorithm can be easily applied for other types of FFNN like multilayer perceptron, functional link neural network etc. Improvements of this concept can be made in terms of enabling ability to grow and prune processing units; for example, RBF and HBF networks in Huang et al. (2005) and Vuković and Miljković (2013) can be trained with EKF-OR.

Recently, an interesting learning algorithm for FFNN is proposed—Extreme Learning Machines (ELM), in which input weights and biases are chosen randomly while output weights are determined using simple matrix inversion; ELM (Huang, Zhu, & Siew, 2006) and its sequential variant (Gu, Liu, Chen, Jiang, & Yu, 2014; Guo, Hao, & Liu, 2014; Huynh & Won, 2011; Liang, Huang, Saratchandran, & Sundararajan, 2006) enables fast and accurate learning. Our approach could be extended to ELM exploiting approaches presented in Soria-Olivas et al. (2011) and Chatzis, Korkinof, and Demiris (2011); for example, in Chatzis et al. (2011) one may see that proposed solution includes approach given in Soria-Olivas et al. (2011) as a special case, and how Bayesian treatment of ELM training leads to superior results. However, the results are based on assumption that observations are Gaussians, which could be changed to include distributions with heavier tails. Variational Bayes algorithm presented in this paper can be used as basis to build robust sequential ELM.

On the other hand, results derived and presented in this paper can be extended for training of recurrent neural networks in presence of outliers as well; for example, given data polluted by outliers EKF-OR can be used to train Echo State Networks (ESN) presented in Li et al. (2012).

5. Conclusion

Outliers in engineering applications are a common reason for failure of model or its poor generalization ability. In real world applications, outliers are often found in data and cause serious issues. Motivated by these facts, in this research we have developed extended Kalman filter robust to outliers (EKF-OR) for training of feedforward neural networks when training data contains outliers. EKF-OR is based on simple intuition of "uncertainty about uncertainty" (Agamennoni et al., 2012); unlike conventional Kalman filter in EKF-OR we allow measurement noise covariance to change over time. There is no need for preprocessing of data or labeling of outliers before training process; EKF-OR processes all data points, regardless if data point is outlier or not. Outliers are "naturally" down-weighted during sequential learning using EKF-OR. Structured variational approximation is applied to solve the problem of analytical intractability of update step in Bayesian framework; structured variational approximation enables our learning algorithm to operate on complete data log-likelihood and to iteratively improve state and noise estimates.

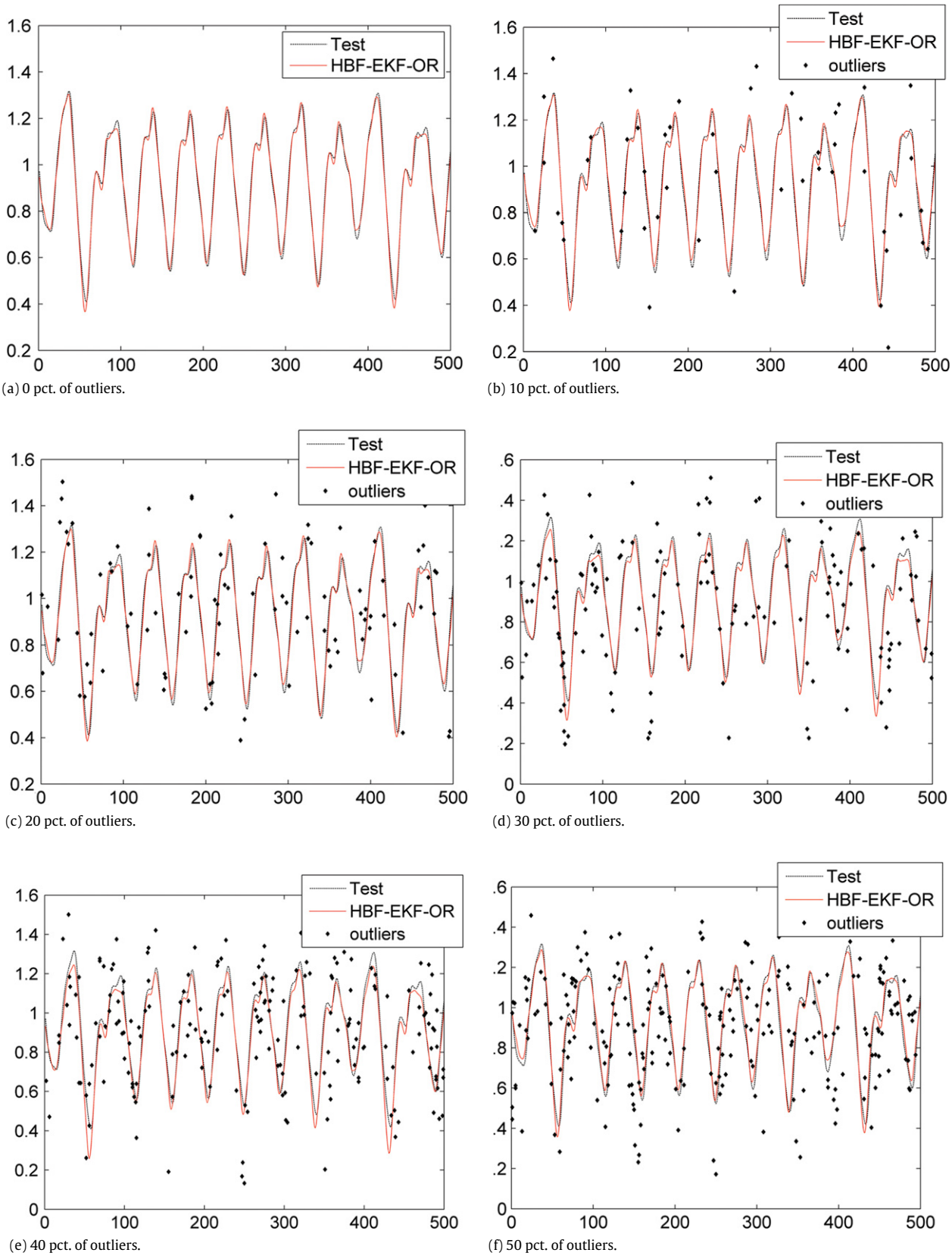
The experiments on real data and simulated engineering problems have shown that EKF-OR generates feedforward neural

Table 8
Characteristics of the *Mackey–Glass* chaotic time series problem.

Dataset	Type of data	Type of problem	n_x	N_{train}	N_{test}
Mackey–Glass chaotic time series	Simulated	Time series prediction	4	500	500

Table 9
Experimental results obtained using EKF-OR learning algorithm for optimization of HBF network parameters. Table shows RMSE and standard deviation calculated for testing set free of outliers, averaged over 30 independent runs.

Algorithm	J	Pct. of outliers in training set					
		0%	10%	20%	30%	40%	50%
HBF EKF-OR	1	0.0378 $0.7982 \cdot 10^{-4}$	0.0423 $0.3372 \cdot 10^{-3}$	0.0463 0.0116	0.0420 $0.5141 \cdot 10^{-3}$	0.0479 $0.3412 \cdot 10^{-3}$	**0.0380** $\mathbf{0.1138 \cdot 10^{-3}}$
	2	0.0319 0.0042	0.0344 0.0032	0.0394 $0.7148 \cdot 10^{-3}$	0.0376 0.0026	0.0454 0.0037	**0.0424** **0.0045**
	5	0.0272 0.0030	0.0268 0.0018	0.0348 0.0030	0.0362 0.0034	0.0327 0.0025	**0.0425** **0.0017**
	10	0.0261 0.0022	0.0291 0.0036	0.0276 0.0019	0.0337 0.0018	0.0377 0.0021	**0.0385** **0.0025**
	15	0.0267 0.0042	0.0336 0.0027	0.0394 0.0019	0.0340 0.0031	0.0412 0.0011	**0.0654** **0.0019**
	20	0.0286 0.0050	0.0345 0.0023	0.0521 0.0028	0.0435 0.0040	0.0489 0.0035	**0.0540** **0.0026**
LS-GRBFNFS (Yang et al., 2013)		0.1600 0	0.1603 0.0007	0.1744 0.0320	0.1800 0.0040	0.1800 0.0041	0.4135 0.0609
LW-GRBFNFS (Yang et al., 2013)		**0.0107** **0**	**0.0107** **0**	**0.0110** **0.0003**	**0.0110** **0.0007**	**0.0112** **0.0010**	0.2809 0.0179



(a) 0 pct. of outliers.

(b) 10 pct. of outliers.

(c) 20 pct. of outliers.

(d) 30 pct. of outliers.

(e) 40 pct. of outliers.

(f) 50 pct. of outliers.

Fig. 8. Testing data for *Mackey–Glass* chaotic time series prediction and HBF EKF-OR estimate of the series given training set polluted by outliers. Test set is free of outliers.

networks with good generalization ability. It is our belief that mathematical derivation and experimental results presented in this paper provide basis for building more robust learning algorithms.

Acknowledgments

This work is supported by the Serbian Government – the Ministry of Education, Science and Technological Development

– Project title: An innovative, ecologically based approach to the implementation of intelligent manufacturing systems for the production of sheet metal parts (2011–2015) under grant TR35004.

Authors would like to gratefully acknowledge the effort made by action editor and reviewers whose suggestions helped us to improve the manuscript.

References

Agamennoni, G., & Nebot, E. M. (2014). Robust estimation in non-linear state-space models with state-dependent noise. *IEEE Transactions on Signal Processing*, *62*(8), 2165–2175.

Agamennoni, G., Nieto, J., & Nebot, E. (2011). An outlier-robust Kalman filter. In *IEEE international conference on robotics and automation*, ICRA 2011. USA: (IEEE) Institute of Electrical and Electronics Engineers.

Agamennoni, G., Nieto, J., & Nebot, E. (2012). Approximate inference in state-space models with heavy-tailed noise. *IEEE Transactions on Signal Processing*, *60*(10), 5024–5037.

Andrieu, C., De Freitas, N., & Doucet, A. (2001). Robust full Bayesian learning for radial basis networks. *Neural Computation*, *13*(10), 2359–2407.

Archambeau, C., & Verleysen, M. (2007). Robust Bayesian clustering. *Neural Networks*, *20*(1), 129–138.

Bache, K., & Lichman, M. (2013). *UCI machine learning repository*. Irvine, CA: University of California, School of Information and Computer Science., http://archive.ics.uci.edu/ml (last date of access: February 10, 2014).

Barber, D. (2012). *Bayesian reasoning and machine learning*. Cambridge University Press.

Beal, M. J. (2003). *Variational algorithms for approximate Bayesian inference*. (Doctoral dissertation), University of London.

Billings, S. A., Wei, H. L., & Balikhin, M. A. (2007). Generalized multiscale radial basis function networks. *Neural Networks*, *20*(10), 1081–1094.

Bishop, C. (2006). *Pattern recognition and machine learning*. Berlin: Springer.

Box, G., & Tiao, G. (1973). *Bayesian inference in statistical analysis*. John Wiley & Sons.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, *41*(3), 15.

Chatzis, S. P., Korkinof, D., & Demiris, Y. (2011). The one-hidden layer non-parametric Bayesian kernel machine. In *23rd IEEE International conference on tools with artificial intelligence (ICTAI)* (pp. 825–831). IEEE.

Chuang, C. C., & Jeng, J. T. (2007). CPBUM neural networks for modeling with outliers and noise. *Applied Soft Computing*, *7*(3), 957–967.

Chuang, C. C., Jeng, J. T., & Lin, P. T. (2004). Annealing robust radial basis function networks for function approximation with outliers. *Neurocomputing*, *56*, 123–139.

Chuang, C. C., & Lee, Z. J. (2011). Hybrid robust support vector machines for regression with outliers. *Applied Soft Computing*, *11*(1), 64–72.

Chuang, C. C., Su, S. F., & Chen, S. S. (2001). Robust TSK fuzzy modeling for function approximation with outliers. *IEEE Transactions on Fuzzy Systems*, *9*(6), 810–821.

Chuang, C. C., Su, S. F., Jeng, J. T., & Hsiao, C. C. (2002). Robust support vector regression networks for function approximation with outliers. *IEEE Transactions on Neural Networks*, *13*(6), 1322–1330.

Connor, J. T., Martin, R. D., & Atlas, L. E. (1994). Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks*, *5*(2), 240–254.

Doucet, A., De Freitas, N., & Gordon, N. J. (2001). *Sequential Monte Carlo methods in practice*. New York: Springer-Verlag.

Đurović, Ž. M., & Kovačević, B. D. (1999). Robust estimation with unknown noise statistics. *IEEE Transactions on Automatic Control*, *44*(6), 1292–1296.

Fu, Y. Y., Wu, C. J., Jeng, J. T., & Ko, C. N. (2010). ARFNNs with SVR for prediction of chaotic time series with outliers. *Expert Systems with Applications*, *37*(6), 4441–4451.

Gu, Y., Liu, J., Chen, Y., Jiang, X., & Yu, H. (2014). TOSELM: Timeliness online sequential extreme learning machine. *Neurocomputing*, *128*, 119–127.

Guo, L., Hao, J. H., & Liu, M. (2014). An incremental extreme learning machine for online sequential learning problems. *Neurocomputing*, *128*, 50–58.

Guvenir, H.A., & Uysal, I. (2000). Bilkent University Function Approximation Repository. http://funapp.cs.bilkent.edu.tr (last date of access: February 10, 2014).

Hodge, V. J., & Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, *22*(2), 85–126.

Huang, G. B., Saratchandran, P., & Sundararajan, N. (2004). An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *34*(6), 2284–2292.

Huang, G. B., Saratchandran, P., & Sundararajan, N. (2005). A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Transactions on Neural Networks*, *16*(1), 57–67.

Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, *70*(1), 489–501.

Huber, P. J. (2011). *Robust statistics*. Berlin, Heidelberg: Springer.

Huynh, H. T., & Won, Y. (2011). Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks. *Pattern Recognition Letters*, *32*(14), 1930–1935.

Jeng, J. T., Chuang, C. C., & Tao, C. W. (2010). Hybrid SVMR-GPR for modeling of chaotic time series systems with noise and outliers. *Neurocomputing*, *73*(10), 1686–1693.

Ko, C. N. (2012). Identification of nonlinear systems with outliers using wavelet neural networks based on annealing dynamical learning algorithm. *Engineering Applications of Artificial Intelligence*, *25*(3), 533–543.

Lee, C. C., Chiang, Y. C., Shih, C. Y., & Tsai, C. L. (2009). Noisy time series prediction using M-estimator based robust radial basis function neural networks with growing and pruning techniques. *Expert Systems with Applications*, *36*(3), 4717–4724.

Lee, C. C., Chung, P. C., Tsai, J. R., & Chang, C. I. (1999). Robust radial basis function neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *29*(6), 674–685.

Li, D., Han, M., & Wang, J. (2012). Chaotic time series prediction based on a novel robust echo state network. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(5), 787–799.

Liang, N. Y., Huang, G. B., Saratchandran, P., & Sundararajan, N. (2006). A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, *17*(6), 1411–1423.

Mahdi, R. N., & Rouchka, E. C. (2011). Reduced HyperBF networks: Regularization by explicit complexity reduction and scaled Rprop based training. *IEEE Transactions on Neural Networks*, *22*(5), 673–686.

Markou, M., & Singh, S. (2003a). Novelty detection: a review—part 1: statistical approaches. *Signal Processing*, *83*(12), 2481–2497.

Markou, M., & Singh, S. (2003b). Novelty detection: a review—part 2: neural network based approaches. *Signal Processing*, *83*(12), 2499–2521.

Miljković, Z., & Aleksendrić, D. (2009). Artificial neural networks—solved examples with theoretical background. University of Belgrade-Faculty of Mechanical Engineering, Belgrade, (in Serbian).

Miljković, Z., Vuković, N., Mitić, M., & Babić, B. (2013). New hybrid vision-based control approach for automated guided vehicles. *The International Journal of Advanced Manufacturing Technology*, *66*(1–4), 231–249.

Minka, T. P. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the seventeenth conference on uncertainty in artificial intelligence* (pp. 362–369). Morgan Kaufmann Publishers Inc.

Nikolaev, N. Y., & de Menezes, L. M. (2008). Sequential Bayesian kernel modelling with non-Gaussian noise. *Neural Networks*, *21*(1), 36–47.

Nishida, K., Yamauchi, K., & Omori, T. (2006). An online learning algorithm with dimension selection using minimal hyper basis function networks. *Systems and Computers in Japan*, *37*(11), 11–21.

Pernía-Espinoza, A. V., Ordieres-Meré, J. B., Martínez-de-Pisón, F. J., & González-Marcos, A. (2005). TAO-robust backpropagation learning algorithm. *Neural Networks*, *18*(2), 191–204.

Poggio, T., & Girosi, F. (1989). A theory of networks for approximation and learning. A. I. Memo 1140, MIT.

Poggio, T., & Girosi, F. (1990). Networks for approximation and learning. *Proceedings of the IEEE*, 1481–1497.

Rusiecki, A. (2007). Robust LTS backpropagation learning algorithm. In *Computational and ambient intelligence* (pp. 102–109). Berlin, Heidelberg: Springer.

Rusiecki, A. (2013). Robust learning algorithm based on LTA estimator. *Neurocomputing*, 624–632. Special Issue: Image Feature Detection and Description.

Sarkka, S., & Nummenmaa, A. (2009). Recursive noise adaptive Kalman filtering by variational Bayesian approximations. *IEEE Transactions on Automatic Control*, *54*(3), 596–600.

Schick, I. C., & Mitter, S. K. (1994). Robust recursive estimation in the presence of heavy-tailed observation noise. *The Annals of Statistics*, *22*(2), 1045–1080.

Simon, D. (2002). Training radial basis neural networks with the extended Kalman filter. *Neurocomputing*, *48*(1), 455–475.

Soria-Olivas, E., Gomez-Sanchis, J., Jarman, I. H., Vila-Frances, J., Martinez, M., Magdalena, J. R., et al. (2011). BELM: Bayesian extreme learning machine. *IEEE Transactions on Neural Networks*, *22*(3), 505–509.

Stanković, S. S., & Kovačević, B. D. (1986). Analysis of robust stochastic approximation algorithms for process identification. *Automatica*, *22*(4), 483–488.

Ting, J. A., Theodorou, E., & Schaal, S. (2007). A Kalman filter for robust outlier detection. In *IEEE/RSJ international conference on intelligent robots and systems, 2007*, IROS 2007 (pp. 1514–1519). IEEE.

Tzikas, D. G., Likas, C. L., & Galatsanos, N. P. (2008). The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine*, *25*(6), 131–146.

Vuković, N. (2012). *Machine learning of intelligent mobile robot based on artificial neural networks*. (Doctoral dissertation). University of Belgrade—Faculty of Mechanical Engineering (in Serbian), http://dx.doi.org/10.2298/BG20120928VUKOVIC.

Vuković, N., & Miljković, Z. (2013). A growing and pruning sequential learning algorithm of hyper basis function neural network for function approximation. *Neural Networks*, *46*, 210–226.

Yang, Y. K., Sun, T. Y., Huo, C. L., Yu, Y. H., Liu, C. C., & Tsai, C. H. (2013). A novel self-constructing radial basis function neural-fuzzy system. *Applied Soft Computing*, *13*(5), 2390–2404.

Yohai, V. J., & Zamar, R. H. (1988). High breakdown-point estimates of regression by means of the minimization of an efficient scale. *Journal of the American Statistical Association*, *83*(402), 406–413.

Zhu, J., Hoi, S., & Lyu, M. T. (2008). Robust regularized kernel regression. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *38*(6), 1639–1644.