Power Prediction in Smart Grids with Evolutionary Local Kernel Regression

Oliver Kramer, Benjamin Satzger, and Jörg Lässig

International Computer Science Institute, Berkeley CA 94704, USA,
{okramer, satzger, jla}@icsi.berkeley.edu,
http://www.icsi.berkeley.edu/

Abstract. Electric grids are moving from a centralized single supply chain towards a decentralized bidirectional grid of suppliers and consumers in an uncertain and dynamic scenario. Soon, the growing smart meter infrastructure will allow the collection of terabytes of detailed data about the grid condition, e.g., the state of renewable electric energy producers or the power consumption of millions of private customers, in very short time steps. For reliable prediction strong and fast regression methods are necessary that are able to cope with these challenges. In this paper we introduce a novel regression technique, i.e., evolutionary local kernel regression, a kernel regression variant based on local Nadaraya-Watson estimators with independent bandwidths distributed in data space. The model is regularized with the CMA-ES, a stochastic non-convex optimization method. We experimentally analyze the load forecast behavior on real power consumption data. The proposed method is easily parallelizable, and therefore well appropriate for large-scale scenarios in smart grids.

1 Introduction

If we want to design smarter electric grids that are more adaptive or even "intelligent", the large amount of information about the grid status, e.g., current wind or solar energy production, or actual power demands of customers must be considered. Prediction is important for energy saving and cost efficient real-time decisions. An increasing infrastructure of smart meters at the level of consumers is supported by governmental laws in many countries, and is currently leading to an explosion of data about electric grids. Already today, smart meters are able to yield the consumption status of each customer every second. To be able to analyze this large amount of data, strong large-scale techniques have to be developed that allow precise predictions of power supply and power consumption behaviors, e.g., to avoid voltage band violations. For each customer, a short-, mid-, and long-term profile can be computed, leading to a precise estimation of future energy consumption habits. The development of large-scale data mining techniques in a distributed computing scenario becomes an essential challenge in smart energy grids.

A survey of methods for the prediction of power supply and demand give Alfares and Nazeeruddin [3]. They classify related methods into nine classes, from

multiple regression to expert systems. But most methods are not parallelizable or demand a new and expensive training process if the data archive is changed. Short-term load forecasting with kernel regression has been proposed by Agarwal *et al.* [2]. They compared kernel regression to artificial neural networks, ordinary least squares and ridge regression. We will propose a more flexible kernel regression hybrid in this paper. Prediction in energy systems is not only restricted to load forecasting, but also concentrates on other properties. As an example, Nogales *et al.* [9] analyze dynamic regression and transfer function models to forecast next-day electricity prices. Lora *et al.* [7] also predict next-days prizes using a weighted nearest neighbors approach, concentrating on parameter determination, e.g., size of time series windows and number K of neighbors.

In this paper we introduce a novel hybrid regression method that we think is well appropriate to solve these tasks. It unifies kernel regression and evolutionary computation. In Section 2 we introduce evolutionary local kernel regression (ELKR), and discuss, how ELKR can be parallelized for large-scale decentralized smart grid scenarios. Section 3 shows an experimental analysis on real data of power demand in electricity grids. Finally, Section 4 summarizes the results and provides an outlook to future work.

2 Evolutionary Local Kernel Regression

2.1 The Nadaraya-Watson Estimator

Kernel regression is a non-parametric approach and allows fast estimations for reasonable bandwidth choices. It is based on the Nadaraya-Watson estimator that has been introduced by Nadaraya and Watson [8, 12]. We assume that $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ is a set of N recorded input output values, e.g., N recorded power consumption values within a particular time period. For an unknown $\mathbf{x}'$ we like to forecast $\mathbf{f}(\mathbf{x}')$. Kernel regression is based on a density estimate of data samples with a kernel function K. A typical kernel function is the Gaussian kernel:

$$K_{\mathbf{H}}(\mathbf{z}) = \frac{1}{(2\pi)^{q/2}\det(\mathbf{H})} \exp\left(-\frac{1}{2}\left|\mathbf{H}^{-1}\mathbf{z}\right|^2\right). \tag{1}$$

The kernel density function becomes a measure for the *density* of two points $\mathbf{x}_1, \mathbf{x}_2$, if their distance $\mathbf{z} = |\mathbf{x}_1 - \mathbf{x}_2|$ is fed into the function. An essential part of kernel regression is the bandwidth h that becomes a diagonal matrix $\mathbf{H} = \mathrm{diag}(h_1, h_2, \ldots, h_d)$ in case of the multivariate expression. Parameter q is scaling the Gaussian function in height, and usually set to $q = 1$. The sum of density estimates of an unknown point $\mathbf{x}$ and all data samples, multiplied with the corresponding function values, yield the Nadaraya-Watson estimate for $\mathbf{f}(\mathbf{x}; \mathbf{H})$, i.e.:

$$\mathbf{f}(\mathbf{x}; \mathbf{H}) = \sum_{i=1}^{N} \mathbf{y}_i \frac{K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)}{\sum_{j=1}^{N} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_j)}. \tag{2}$$

Hence, each sample $(\mathbf{x}_i, \mathbf{y}_i)$ contributes to the prediction of the function value of $\mathbf{x}$, weighted by the normalized (denominator) kernel densities in data space.

If we assume that the results of N kernel density computations can be saved, each prediction can be computed in $\mathcal{O}(N)$. If the model is trained with regard to the bandwidth matrix $\mathbf{H}$ on the training set, it may over-adapt to the training examples, and loose the ability to generalize, an effect known as overfitting. Overfitting is likely to happen if the training set's size is small, or if the number of free parameters of a model is comparatively large. Small bandwidth values lead to an overfitted prediction function, while high values generalize too much. To avoid overfitting, Clark [5] proposed to select the bandwidth matrix $\mathbf{H}$ as result of LOO-CV. The idea of determining the optimal bandwidth matrix $\mathbf{H}$ by LOOC-CV is to apply the Nadaraya-Watson estimator, leaving out the data pair $(\mathbf{x}_i, \mathbf{y}_i)$ for each summand. The resulting error function that has to be minimized is:

$$e_{loocv} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{f}_{-i}(\mathbf{x}_i; \mathbf{H})\|_2 \tag{3}$$

$$= \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{y}_i - \sum_{j\neq i} \frac{K_{\mathbf{H}}(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{k\neq i} K_{\mathbf{H}}(\mathbf{x}_j, \mathbf{x}_k)} \mathbf{y}_j\|_2 \tag{4}$$

Here, $\mathbf{f}_{-i}$ denotes the Nadaraya-Watson estimator leaving out the i-th data point. All points, without the data sample $\mathbf{y}_i$ itself, contribute to the estimation of $\mathbf{f}(\mathbf{x}_i)$.

2.2 Extending Kernel Regression to ELKR

This section enhances kernel regression by the concept of locality, the evolution of kernel parameters, and minimization of the prediction error with a stochastic optimization method. The concepts are introduced in detail in the following.

Local Models To handle multiple data space conditions in different areas of the data space, we introduce the concept of locality. Each Nadaraya-Watson model is specified by a codebook vector $\mathbf{c}_i$ from the set of codebook vectors $\mathcal{C} = (\mathbf{c}_1, \cdots, \mathbf{c}_K) \in \mathbb{R}^q$ in latent space. A data element $(\mathbf{x}_i, \mathbf{y}_i)$ is assigned to the Nadaraya-Watson model $\mathbf{f}_{k^*}(\mathbf{x}_i; \mathbf{H}_{k^*})$ with minimal distance to its codebook vector, i.e., $k^* = \arg\min_k \|\mathbf{x}_i - \mathbf{c}_k\|$. Algorithm 1 shows the pseudo-code of the approach with K local regression models, and archive $\mathfrak{A}$ of samples $(\mathbf{x}_i, \mathbf{y}_i)$, $1 \leq i \leq N$. At the beginning, the codebook vectors are randomly distributed in data space. In the training phase, the codebook vectors and the other properties of each local model are optimized, in order to adapt to local data space distributions that may afford separate bandwidths. The optimization goal is to minimize the overall data space reconstruction error of all local models, i.e., with regard to the codebook vector set $\mathcal{C}$, and the local bandwidth matrices $\mathbf{H}_k$:

$$e_{\mathrm{ELKR}} = \frac{1}{N}\sum_{i}^{N} \|\mathbf{f}_{k^*}(\mathbf{x}_i; \mathbf{H}_{k^*}) - f(\mathbf{x}_i)\|_2, \tag{5}$$

with

$$k^* = \arg\min_{k=1,\ldots,K} \|\mathbf{x}_i - \mathbf{c}_k\|. \tag{6}$$

The assignment of data samples to local models results in a significant speedup. If we assume that the data samples are on average uniformly distributed to all K models, the computation time for the prediction of one data element can be reduced to the $1/K$-th.

Algorithm 1 LOCAL KERNEL REGRESSION

Require: Archive $\mathfrak{A}$, $\mathbf{H}$, K, request $\mathbf{x}_1, \ldots, \mathbf{x}_u$
1: Initialize K local Nadaraya-Watson models
2: **for** $i = 1$ to $|\mathfrak{A}|$ **do**
3: Assign data samples $(\mathbf{x}_i, \mathbf{y}_i)$ to closest local model
4: Compute $\mathbf{f}(\mathbf{x}_i,\mathbf{H})$
5: **end for**

Stochastic LOO-CV Taking into account every data sample, the computation of e_{cv} can be a computationally expensive undertaking. To accelerate the bandwidth adaptation mechanism in case of large data sets, we stochastically select at least one data element for each kernel regression model, i.e., we compute $\mathbf{f}_{-i,k}(\mathbf{x}_i; \mathbf{H}_k)$ for a randomly chosen element i. Stochastic LOO-CV may not be as strong as LOO-CV, and it will be subject to future work to analyze, how many stochastic repetitions will be necessary in practical applications to balance between overfitting and generalizations. In our scenario one repetition turned out to be sufficient for satisfying prediction capabilities, see Section 3.

CMA-ES Engine We use the CMA-ES [6, 10] for adaptation of the free parameters of the k-th model, i.e., of the codebook vector $\mathbf{c}_k$ and bandwidth matrix $\mathbf{H}_k$. Historically, covariance matrix adaptation techniques have developed from evolution strategies (ES) [4]. In each generation ES produce λ offspring solutions and select μ solutions as parents for generation $t+1$. Often, intermediate recombination is applied: for two uniformly selected parents $\mathbf{x}_1$ and $\mathbf{x}_2$, the offspring is $\mathbf{x}' := \frac{1}{2}\mathbf{x}_1 + \frac{1}{2}\mathbf{x}_2$. ES became famous for their Gaussian mutation operator with σ-self-adaptation. In case of the CMA-ES the covariance matrix of differences between a successful solution and its parents is basis of the mutation process. The CMA-ES step sizes are controlled using the derandomized cumulative path-length control by Ostermeier *et al.* [10]. Its main idea is to construct a solution path based on the difference between a current solution and a predecessor that is c generations older. This difference is the basis of a so called evolution path. A parameter defines the number of generations taken into account, i.e., defining the exploitation of past information. For a more detailed introduction to the CMA-ES we can recommend the tutorial by Hansen [6]. The CMA-ES has two advantages. First, it belongs to the class of non-convex stochastic optimization methods and allows a flexible LOO-CV minimization with a low chance of getting trapped in local optima. Secondly, like kernel regression it is embarrassingly parallelizable, e.g., on the level of subpopulations or candidate solutions, see Section 2.3.

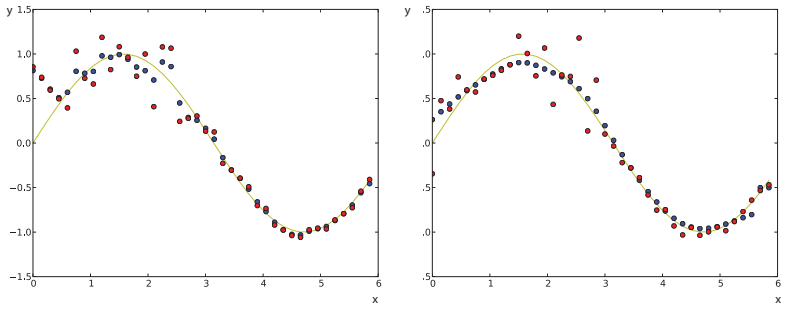

Fig. 1. Illustration of the benefits of more than one kernel regression model. The usual Nadaraya-Watson estimator with only one bandwidth (left part) and two local models (right part). The local model allows an adaptation to local noise conditions.

Illustration Figure 1 illustrates the advantage of local Nadaraya-Watson models for a simple sinus function (indicated by line) in the interval $[0, 6]$ with two different areas of noise (red dots), i.e., $g(x) = \sin(x) + \gamma \cdot \mathcal{N}(0, 1)$, with $\gamma = 0.25$ for $x \in [0, 3[$, and $\gamma = 0.08$ for $x \in [3, 6]$. The different noise values afford different bandwidths in the two areas, i.e., the left and the right part of the function, dark grey dots indicate the prediction. This is not possible with only one single Nadaraya-Watson estimator (left part of Figure 1). To reduce the data space reconstruction error on average, the area with high noise overadapts to the noisy values due to a too small bandwidth. In case of local models, a smoother adaptation to local search characteristics, e.g. different degrees of noise, is possible (right part of Figure 1).

2.3 Parallelization

Evolutionary kernel regression is a method that is parallelizable in two kinds of ways, i.e, on the level of the kernel density computations of kernel regression, and on the level of the evolvement of candidate solutions or subpopulations. First, on the level of the regression method, we assume that the prediction of N_1 values has to be computed. For the sake of simplicity, we assume that each call of a kernel density function K takes one time step. Furthermore, we assume that the archive consists of N_2 data samples. The prediction of the N_1 values takes $\mathcal{O}(N_1 \cdot N_2)$ kernel function computations. For M (uniform) machines, the computation of the kernel density sum, can be distributed and parallelized by assigning the kernel density computations uniformly to M machines. If we assume only constant cost for the distribution of the archive on M machines as well as the aggregation of results of the M machines, i.e. $f(\mathbf{x}) = f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})$, the parallelization results in a total runtime of $\mathcal{O}(N_1 \cdot N_2/M)$. In the distributed computing scenario of a smart grid, the assumed constants and idealizations do not apply, but have to be filled with properties of the real system, e.g., envelope

delay and computational power of the real machines. Second, evolutionary methods are famous for their property to be parallelizable. The application of genetic operators such as recombination and covariance matrix based mutation, as well as the fitness computation of individuals or subpopulations can be distributed on M machines.

3 ELKR in Power Prediction Scenarios

In the following, we will apply ELKR for prediction of power demand based on real power consumption data published by the Irish state-owned company EirGrid, see [1]. The data shows the electricity production required to meet the national electricity consumption, including system losses and generators' requirements. The power consumption is stated in mega-watt (MW) every 15 minutes. In our scenario, we have used the data about the power consumption of three succeeding Sundays in January 2010 and February 2010 to predict the power consumption of the following Sunday.

We have tested various settings for ELKR. In our experiments we use $K = 5$ local kernel regression models with Gaussian density kernel, and allow the evolvement of bandwidth matrices $\mathbf{H}_k$ based on stochastic LOO-CV. For this sake, we use a $(5, 10)$-CMA-ES, i.e, a parental population of size $\mu = 5$, and an offspring population of size $\lambda = 10$. The CMA-ES terminates after 200 fitness function evaluations, while the bandwidth is initialized with $h = 5.0$. Figure 2 shows typical runs of ELKR. The upper left part of the figure shows the power prediction of the consumption on Feb 7 based on the consumption on Sundays Jan 17, Jan 24, and Jan 31 (scenario A). The upper right part shows the prediction of Feb 14 based on Jan 24, Jan 31, and Feb 7 (scenario B). The lower left part of Figure 2 shows the prediction of the power consumption on Feb 21 based on Feb 14, Feb 7, and Jan 31 (scenario C). The colored lines show the three observed power consumption developments. In each figure, the bright grey dots show the prediction while the dark grey dots show the actual power consumption that has to be predicted. In all experiments we can observe that ELKR is able to make satisfying predictions. For comparison, the lower left part shows the same prediction with a fixed bandwidth of $h = 5.0$. The regression model does not adapt to the data, but generalizes too much.

Table 1. Comparison of common kernel regression with a constant bandwidth $h = 5.0$, ELKR with $K = 5$ and CMA-ES based stochastic LOO-CV, least median square regression and backpropagation, in terms of training error e_{TR} on the archive, and error e_{TE} on the test set. For ELKR best and median values of 15 runs are shown.

	$K = 1$, $h = 5.0$		$K = 5$, LOO-CV				LMS		NN	
	e_{TR}	e_{TE}	best e_{TR}	med e_{TR}	best e_{TE}	med e_{TE}	e_{TR}	e_{TE}	e_{TR}	e_{TE}
A	93.89	86.88	66.91	67.39	63.88	64.44	74.08	64.78	134.67	124.51
B	91.06	118.66	58.06	60.79	104.33	104.72	57.79	105.58	104.64	154.01
C	97.44	86.00	72.43	79.58	56.04	64.00	71.95	58.79	142.95	159.31

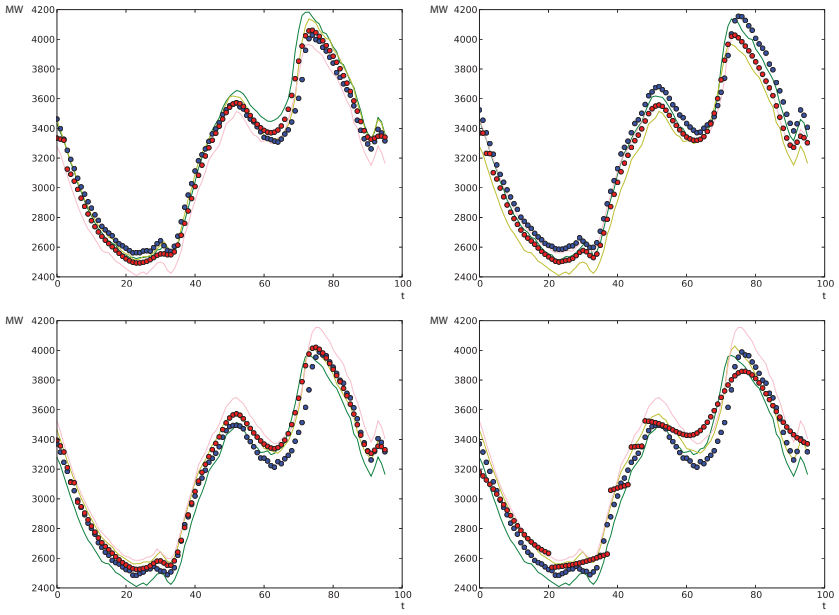

Fig. 2. Prediction of power demands (in MW) with ELKR for three Sundays in January and February 2010 ($t = 1, \ldots, 96$), each based on the previous three Sundays (colored lines). Bright grey dots show the power prediction, dark grey dots show the amount of power that has been consumed. ELKR shows satisfying forecast capabilities. For comparison, the lower right part shows a run with local models, but constant bandwidth of $h_k = 5.0$.

Table 1 compares kernel regression with only one model and constant bandwidth $h = 5.0$ to ELKR with stochastic LOO-CV and CMA-ES based bandwidth optimization (200 generations). The results are measured in terms of *training* error e_{TR}, i.e., average absolute deviation from all three training data sets, and test error e_{TE} on the fourth Sunday, i.e., average absolute deviation from the test values of a fourth data set, see scenarios A-C above. In case of ELKR, the best and median results of 15 runs are shown. The experimental results show that ELKR achieves significantly better prediction accuracies on the training archive data and the test data set. Interestingly, the accuracy on the forecast data set is higher than the accuracy on the archive data in case of scenario A and C. This is probably caused by higher deviations of particular power demand curves, i.e., outliers in the archive. Furthermore, Table 1 shows a comparison to least median square regression (LMS) and a backpropagation neural network [11] (96 neurons on input, 48 neurons on hidden layer). ELKR shows competitive results to LMR, and even slightly better results on the test set, while it outperforms the backpropagation network.

4 Conclusion

In this paper we have introduced an extension of kernel regression that is based on local models, and independent stochastic optimization of the bandwidth matrices $\mathbf{H}_k$. LOO-CV can be performed without additional cost. The assignment to local models saves computations, as only the kernel regression model with the closest codebook vector is taken into account for prediction. We have shown experimentally how the model behaves on real data in a power prediction scenario. ELKR has shown significantly higher accuracies than common kernel regression or backpropagation, and competitive results to LMR. The regression as well as the evolutionary part of ELKR are easily parallelizable, and therefore well appropriate for large-scale data mining scenarios in smart grids. We plan to perform parameter studies of ELKR on real power data and also conduct experiments on other regression problems, e.g., on UCI machine learning library data sets. In this context we will concentrate on prediction problems that afford different regularizations in different parts of the data space.

Acknowledgement

The authors thank the German Academic Exchange Service (DAAD) for funding their research, and EirGrid for the permission to use their energy data.

References

1. Download centre Eirgrid. http://www.eirgrid.com/.
2. V. Agarwal, A. Bougaev, and L. H. Tsoukalas. Kernel regression based short-term load forecasting. In *ICANN (2)*, pages 701–708, 2006.
3. H. Alfares and M. Nazeeruddin. Electric load forecasting: literature survey and classifcation of methods. *International Journal of Systems Science*, 33(1):23–34, 2002.
4. H.-G. Beyer and H.-P. Schwefel. Evolution strategies - A comprehensive introduction. *Natural Computing*, 1:3–52, 2002.
5. R. Clark. A calibration curve for radiocarbon dates. *Antiquity*, 46(196):251–266, 1975.
6. N. Hansen. The CMA evolution strategy: A tutorial. Technical report, TU Berlin, ETH Zürich, 2005.
7. A. T. Lora, J. M. R. Santos, A. G. Expósito, J. L. M. Ramos, and J. C. R. Santos. Electricity market price forecasting based on weighted nearest neighbors techniques. *IEEE Transactions on Power Systems*, 22(3):1294–1301, 2007.
8. E. Nadaraya. On estimating regression. *Theory of Probability and Its Application*, 10:186–190, 1964.
9. F. J. Nogales, J. Contreras, A. J. Conejo, and R. Espinola. Forecasting next-day electricity prices by time series models. *IEEE Transactions on Power Systems*, 17:342–348, 2002.
10. A. Ostermeier, A. Gawelczyk, and N. Hansen. A derandomized approach to self-adaptation of evolution strategies. *Evol. Comput.*, 2(4):369–380, 1994.
11. D. Rumelhart, G. Hintont, and R. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.
12. G. Watson. Smooth regression analysis. *Sankhya Series A*, 26:359–372, 1964.