Optimizing Support Vector Machines for Stormwater Prediction

Patrick Koch, Wolfgang Konen, Oliver Flasch, and Thomas Bartz-Beielstein
{patrick.koch, wolfgang.konen, oliver.flasch, thomas.bartz-beielstein}@fh-koeln.de
Department of Computer Science, Cologne University of Applied Sciences,
51643 Gummersbach,Germany

Abstract

In water resource management, efficient controllers of stormwater tanks prevent flooding of sewage systems, which reduces environmental pollution. With accurate predictions of stormwater tank fill levels based on past rainfall, such controlling systems are able to detect state changes as early as possible. Up to now, good results on this problem could only be achieved by applying special-purpose models especially designed for stormwater prediction. The question arises whether it is possible to replace such specialized models with state-of-the-art machine learning methods, such as Support Vector Machines (SVM) in combination with consequent parameter tuning using sequential parameter optimization, to achieve competitive performance. This study shows that even superior results can be obtained if the SVM hyperparameters and the considered preprocessing is tuned. Unfortunately, this tuning might also result in overfitting or oversearching – both effects would lead to declined model generalizability. We analyze our tuned models and present possibilities to circumvent the effects of overfitting and oversearching.

1 Introduction

Environmental engineering offers important concepts to preserve clean water and to protect the environment. Stormwater tanks are installed to stabilize the load on the sewage system by preventing rainwater from flooding the sewage network and by supplying a base load in dry periods. Mostly, heavy rainfalls are the reason for overflows of stormwater tanks, causing environmental pollution from wastewater contaminating the environment. To avoid such situations, the effluent of the stormwater tanks must be controlled effectively and possible future state changes in the inflow should be detected as early as possible. The time series regression problem of predicting a stormwater tank fill level at time t from a fixed window of past rainfall data from time t back to time $t-W$ will be referred to as the *stormwater problem* in the remainder of this paper.

A model that predicts fill levels by means of rainfall data can be an important aid for the controlling system. Special sensors (Fig. 1) record time series data which can be used to train such a model.

Although many methods exist for time series analysis [4], ranging from classical statistical regression to computational statistics, such methods often require time-consuming investigations on the hyperparameter selection and preprocessing of the data. Besides that, the results are often worse than special-purpose models which are designed from scratch for each new problem. This situation is of course very unsatisfying for the

Fig. 1. Left: rain gauge (pluviometer). Right: stormwater tank.

practitioner in environmental engineering, because new models have to be created and parameters have to be tuned manually for each problem.

For this reason, it would be an advantage to have some standard repertoire of methods which can be easily adapted to new problems. In this paper we use support vector machines (SVMs) [6] for Support Vector Regression as a state-of-the-art method from machine learning and apply them to the stormwater problem. SVMs are known to be a strong method for classification and regression. It has to be noted here, that because of the data sets are time series, records are not necessarily independent of each other, as in normal regression. Therefore we investigate generic preprocessing operators to embed time series data and to generate new input features for the SVM model. In addition, we use sequential parameter optimization (SPOT) [2] and a genetic algorithm (GA) to find good hyperparameter settings for both preprocessing and SVM parameters. We analyze the robustness of our method against overfitting and oversearching of hyperparameters. Preliminary work in stormwater prediction has been done by Hilmer [9], Bartz-Beielstein *et al.* [3] and Flasch *et al.* [7]. A conclusion of these previous publications is that good results can be obtained with specialized models (which are 'hand-crafted' and carefully adapted to the stormwater problem). The main hypotheses of this paper are:

H1 It is possible to move away from domain-specific models without loss in accuracy by applying modern machine learning algorithms and modern parameter tuning methods on data augmented through generic time-series preprocessing operators.

H2 Parameter tuning for stormwater prediction leads to oversearching, yielding too optimistic results on the dataset during tuning.

Hypothesis **H2** puts emphasis on the fact that a distinction between validation set (used during tuning) and test set (used for evaluation) is essential to correctly quantify the benefits from parameter tuning in data mining. The oversearching issue is prevalent in data mining since the output function to tune shows often a high variance when the data used for training or tuning are selected differently.

2 Methods

2.1 Stormwater Tank Data

Time series data for this case study are collected from a real stormwater tank in Germany and consists of 30,000 data records, ranging from April to August 2006. Rainfall data are measured in three-minute intervals by a pluviometer as shown in Fig. 1. As training set we always used a 5,000 record time window (Set 2, Fig. 2) in order to predict another

5,000 record time window for testing (Set 4) which is not directly successive with regard to time to the training period (see Tab. 1). Later, we conducted a hyperparameter tuning with SPOT and feature selection where we used all 4 different datasets {Set1, Set2, Set3, Set4}, each containing 5,000 records to analyze the robustness of our approach against oversearching and overfitting.

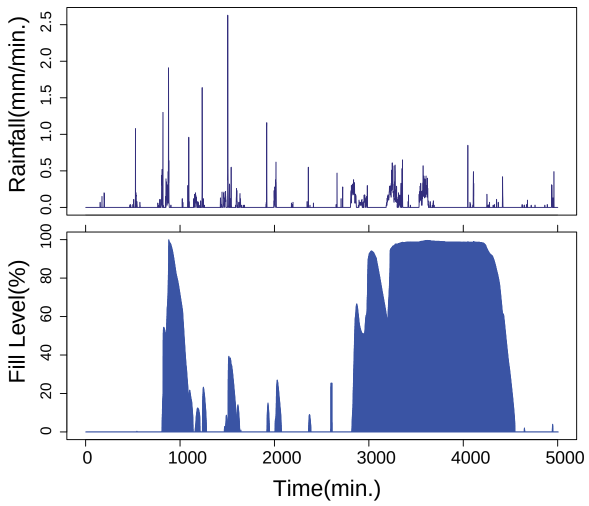

Fig. 2. Time window used for training (Set 2) containing rainfall and fill level of the stormwater tank.

Table 1. Real-world time series data from a stormwater tank in Germany.

Set	Start Date	End Date
Set 1	2006-04-28 01:05:59	2006-05-15 09:40:59
Set 2	2006-05-15 09:40:59	2006-06-01 18:20:59
Set 3	2006-06-19 03:01:00	2006-07-06 11:41:00
Set 4	2006-07-23 20:21:00	2006-08-10 05:01:00

2.2 Evaluation of Models

The prediction error on the datasets is taken as objective function for SPOT and for the GA. For comparing models, we calculate the root mean squared error (RMSE) as a quality measure, where $\langle \cdot \rangle$ denotes averaging over all measurements:

$$RMSE = \sqrt{\langle (Y_{predicted} - Y_{true})^2 \rangle} \qquad (1)$$

We also incorporate a naïve prediction, always predicting the mean value of the training period.

2.3 Sequential Parameter Optimization

The main purpose of SPOT is to determine improved parameter settings for search and optimization algorithms and to analyze and understand their performance.

During the first stage of experimentation, SPOT treats an algorithm A as a black box. A set of input variables $\boldsymbol{x}$, is passed to A. Each run of the algorithm produces some output $\boldsymbol{y}$. SPOT tries to determine a functional relationship F between $\boldsymbol{x}$ and $\boldsymbol{y}$ for a given problem formulated by an objective function $f : \boldsymbol{u} \to \boldsymbol{v}$. Since experiments are run on computers, pseudorandom numbers are taken into consideration if:

– the underlying objective function f is stochastically disturbed, e.g., measurement errors or noise occur, and/or
– the algorithm A uses some stochastic elements, e.g., mutation in evolution strategies.

SPOT employs a sequentially improved model to estimate the relationship between algorithm input variables and its output. This serves two primary goals. One is to enable determining good parameter settings, thus SPOT may be used as a tuner. Secondly, variable interactions can be revealed for helping in understanding how the tested algorithm works when confronted with a specific problem or how changes in the problem influence the algorithm's performance. Concerning the model, SPOT allows for insertion of virtually any available model. However, regression and Kriging models or a combination thereof are most frequently used. The Kriging predictor used in this study uses a regression constant λ which is added to the diagonal of the correlation matrix. Maximum likelihood estimation was used to determine the regression constant λ [8, 1].

2.4 The INT2 Model for Predictive Control of Stormwater Tanks

In previous works [3, 12], the stormwater tank problem was investigated with different modeling approaches, among them FIR, NARX, ESN, a dynamical system based on ordinary differential equations (ODE) and a dynamical system based on integral equations (INT2). All models in these former works were systematically optimized using SPOT [2]. Among these models the INT2 approach turned out to be the best one [3]. The INT2 model is an analytical regression model based on integral equations. Disadvantages of the INT2 model are that it is a special-purpose model only designed for stormwater prediction and that it is practically expensive to obtain an optimal parameter configuration: the parameterization example presented in [3] contains 9 tunable parameters which must be set. In this paper we compare hand-tuned INT2 parameters with the best parameter configuration found by SPOT in former study [3].

2.5 Support Vector Machines

Support Vector Machines have been successfully applied to regression problems by Drucker *et al.* [6], Müller *et al.* [14], Mattera and Haykin [13], and Chan and Lin [5]. In these studies the method has been shown to be superior to many other methods especially when the dimensionality of the feature space is very large.

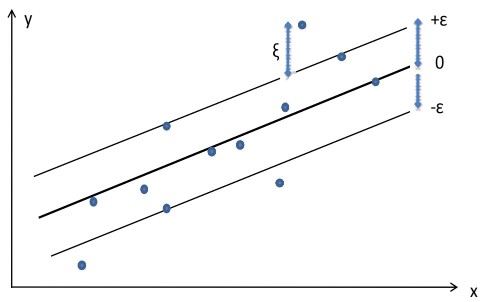

Fig. 3. Example for Support Vector Regression. A tube with radius ϵ is learned to represent the real target function. Possible outliers are being regularized considering a positive slack variable ξ.

Support Vector Machines transform the input space of the training data to a higher-order space using a non-linear mapping, e.g. a radial basis function. In this higher order space a linear function is learned, which has at most ϵ deviation from the real values in general and outliers are regularized by means of the parameter ξ. An example for Support Vector Regression and its parameters ϵ and ξ is depicted in Fig. 3. The transformation to a higher-order space by a non-linear kernel function with parameter γ is not shown here. For a more detailed description of SVM we therefore refer to Smola and Schölkopf [16].

2.6 Preprocessing for Time Series Modeling

In an accompanying work [11] we have analyzed the effects of different preprocessing methods for the stormwater problem. Time series prediction models can benefit from preprocessing operators which generate new features based on the input data. It turns out that best results were obtained with the following preprocessing ingredients:

- Embedding of the input data [10]. Here, the fill level of stormwater overflow tanks $l(t)$ can be represented by a function F on past input features, more precisely by the rainfall $r(t)$ up to $r(t-W)$, where t indicates time and $W \in \mathbb{N}^+$ is the embedding dimension:
$$l(t) := F(r(t), r(t-1), r(t-2), ..., r(t-W)) \tag{2}$$
- Leaky integration of rainfall (termed *leaky rain*) which is computed as follows:
$$L(t) = \sum_{i=0}^{T} \left(\lambda^i \cdot r(t-i)\right) \tag{3}$$
where $\lambda \in [0,1]$ and $T \in \mathbb{N}^+$ is the length of the integration window. Best results with SVM and leaky rain preprocessing can be obtained with two different leaky rain functions using two different values λ_1, λ_2 and T_1, T_2, resp (see Fig. 4).
- More details on the preprocessing are given in [11]. Sequential Parameter Optimization (SPOT) was used to tune the SVM and the preprocessing parameters.

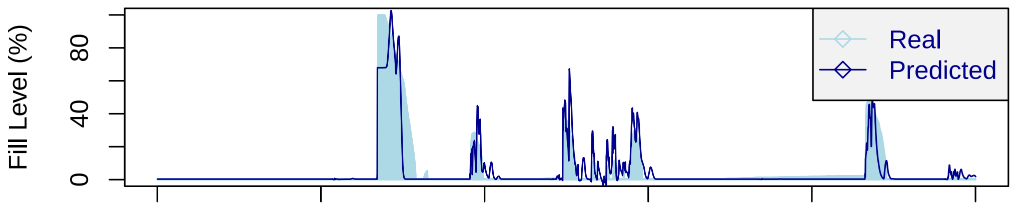

Fig. 4. Prediction for stormwater set 4: the dark line represents the predicted fill level of a SVM model against the shaded area as real fill level. The result was obtained by the best SPOT-tuned SVM model using two leaky rain kernels.

The results of the SPOT tuning are shown in Tab. 2 and Tab. 3. It might be surprising at first sight that the RMSE for Set 2, the data set on which each SVM was trained, is considerably larger than for all other test sets. This is in contrast to normal overfitting where one would expect the training set error to be lower than all other errors. But here Set 2 (see Fig. 2) is considerably more difficult than all other data sets since it contains long time intervals with 100% fill level which are difficult to reproduce from the observed rainfall alone for any predictive model.

In Tab. 3 the evaluation of the SPOT-tuned SVM model, a naïve predictor (predicting the mean value of the training set) and the special-purpose model INT2 on data sets 1-4 is shown. The worst result is obtained with the naïve prediction, which is no surprise, because no learning is included in this approach. *SVM (default)* gives mediocre results, while *SVM (SPOT-tuned)* is comparable to INT2. Tuning was done using the RMSE on Set 4. As a first conclusion drawn out of these results it seems to be possible to achieve competitive results to special-purpose models using our tuned Support Vector Regression model for the fill level problem.

Table 2. Best SPOT parameter configuration for all tuned parameters evaluated on set 4. The region of interest (ROI) bounds are the final values after preliminary runs.

Type	Parameter	Best Value found	ROI	Remark
Embedding	W_1	39	$[2, 60]$	embed. dimension 1
	W_2	16	$[2, 60]$	embed. dimension 2
	T_1	114	$[50, 120]$	leaky window size 1
	T_2	102	$[50, 120]$	leaky window size 2
	λ_1	0.0250092	$[0.00001, 0.3]$	leaky decay 1
	λ_2	0.225002	$[0.00001, 0.3]$	leaky decay 2
SVM	γ	0.0116667	$[0.005, 0.3]$	RBF kernel width
	ϵ	0.0116667	$[0.005, 0.3]$	ϵ-insensitive loss fct.
	χ	1.25	$[0, 10]$	penalty term

Table 3. Evaluation of all models on time windows 1-4. Shown are the RMSE values, when trained on set 2. *SVM (default)* means results with default SVM kernel parameters, while *SVM (SPOT-tuned)* represents mean values obtained in five runs of SPOT.

Model type	Set 1	Set 2	Set 3	Set 4
Naïve prediction	33.56	42.39	34.63	33.34
INT2 (hand-tuned)	7.74	24.27	12.39	10.61
SVM (default)	11.81	44.82	18.55	14.33
SVM (SPOT-tuned)	10.45	43.92	12.93	7.69

3 Experiments: The Effects of Tuning and Selection

The experimental analysis here and in [11] is based on the radial basis SVM kernel from the *e1071* SVM-implementation in R, since we achieved best results with this kernel choice.

All SVM models obtained in this work are sensitive to the parameters for preprocessing and modeling. It has been shown in [11] that a too large number of irrelevant input features can lead to significantly worse results. Although the input features have been extended first by leaky rain embedding and then tuned by a model-based parameter optimization technique (SPOT), it is not clear if

1. the model is generic enough to perform well on different test data and
2. all input features determined by SPOT are really relevant to achieve a good prediction accuracy.

For clarifying the first point, we investigate if our models lead to misleading fits when we evaluate them on different test sets. A misleading model in machine learning can be characterized by the terms *oversearching* and *overfitting* [15]. Oversearching occurs when the learned concept is misleading due to its tailoring to the training goal caused by a too extensive search. In contrast to the well-known concept of overfitting, an oversearched model must not necessarily be “overcomplex”, but rather misleading due to a too extensive search for the best hyperparameters, i.e. model and preprocessing parameters.

The second point can be clarified by a consequent feature selection mechanism, that creates subsets of input features which are used to build models and then evaluated. Here we perform a model optimization by GA feature selection, where a binary string defines the feature set of the embedded leaky rain input features.

The following list of experiments describes our route for a critical investigation of issues related to overfitting and oversearching caused by parameter tuning and feature selection:

T1 Parameter tuning using SPOT for SVM parameters and preprocessing parameters, where the objective function for SPOT is evaluated on different validation sets;

T2 Feature Selection by Genetic Algorithms applied to the optimal parameter configurations of **T1**;

T3 Final parameter tuning using SPOT for parameter configuration obtained on the reduced feature set from step **T2**;

3.1 T1: Oversearching by SPOT

In our last models we used the prediction error gathered on the test data set as the objective function value for the hyperparameter tuning with SPOT. In the real world this value is unknown and when available during optimization it adds unrealistic benefits to the tuned model. In order to perform a fair comparison and to show the benefits of parameter tuning in a more realistic setting, we should use a different objective function. Otherwise the test set error might be too optimistic due the model being tuned and tested on the same set. In Tab. 4, we present the mean results of five SPOT runs for the SVR model to determine optimal parameter settings which are then alternately evaluated on the sets 1,3,4. Again, data set 2 has been used for training. The best configuration found by SPOT is then applied in turn to the other sets (columns) resulting in 3 RMSE values for each parameter configuration. We used the Matlab version of SPOT allowing a total budget of 200 SVM models to be built and a maximum number of 500 samples in the metamodel.

Table 4. Results of SPOT tuning on the stormwater problem. In each row 1-3 of the table, SPOT tunes the RMSE on validation set 1,3,4 leading to different SPOT-tuned parameter configurations. These configurations were applied to the test sets (columns) to make the results comparable. Each experiment was repeated five times with different seeds and we show the mean RMSE; the numbers in brackets indicate the standard deviations.

		Test		
		Set 1	Set 3	Set 4
Validation	Set 1	**9.11** (0.56)	16.40 (6.42)	12.88 (5.50)
	Set 3	10.82 (1.55)	**12.78**(0.34)	12.36 (3.46)
	Set 4	10.45 (0.28)	12.93 (0.35)	**7.69** (0.48)
	S_t	10.64	14.67	12.62
	V_t	16.7%	14.7%	64.1%

A good indicator for oversearching is when best values are often present in the diagonal of the table. It can be seen that this is the case for all validation sets of Tab. 4. Besides this, standard deviations of the offdiagonal values are also larger than the values on the diagonal.

We quantify the oversearching effect by evaluating the following formula: let R_{vt} denote the RMSE for row v and column t of Tab. 4. We define

$$V_t = \frac{S_t - R_{tt}}{R_{tt}} \quad \text{with} \quad S_t = \frac{1}{3}\left(\sum_{v=1}^{4} R_{vt} - R_{tt}\right) \tag{4}$$

With S_t we evaluate the mean off-diagonal RMSE for the columns $t = \{1, 2, 3\}$ which is an indicator of the true strength of the tuned model on independent test data. The diagonal elements R_{tt} are considerably lower in each column of Tab. 4. In case of no oversearching, a value of V_t close to zero would be expected, whereas values larger than zero indicate oversearching.

In summary, a consequent tuning is beneficial but the tuned RMSE is often subject to oversearching effects. E.g. in our case the RMSE on a certain test set was on average 32% higher[1] when the tuned model had not seen the test data before (the realistic case) as compared to the lower value when the test data were used during tuning (T=V).

3.2 T2: Feature Selection

A Genetic Algorithm (GA) is used to determine good feature subsets for the SVM regressor. We rely here on the GA approach because it has some advantages compared to other feature selection methods: Iterative search algorithms can be used to determine feature subsets, where more features are added or eliminated to build the final feature set (Feature Forward Selection and Feature Backward Elimination). Unfortunately these methods often get stuck in locally optimal feature subsets where they finally converge. GAs offer the possibility to flee out of such local optima and find the global optimum given enough iterations.

Experimental Setup In our experimental analysis we started five GA runs, each with a population size of 100, elitist selection strategy (e.g. the best 20% of total population were definitely survivors) and termination after 100 generations. GA parameters where chosen by means of preliminary runs. Each GA individual has N genes, each of which representing whether a certain feature should be included in the model or not. The basis input feature set consisted of all features drawn from a sample SPOT-tuned configuration set as described in Sec. 3.1. Here, the gene length N equals the sum of the embedding dimensions for the two leaky rain functions, ranging from 55 to 92. The candidate solution is mapped to a feature vector which is passed to the feature selection preprocessing script before the SVM model is built. This process has an overall runtime of about 17 hours on a 2.4 GHz Intel Xeon CPU.

Results In each objective function, the RMSE was calculated on the validation sets as defined in Sec. 2.1. This resulted in different feature vectors, which were evaluated again on each validation set. The number of selected features ranges from a minimal feature set of 5 (mean value of GA runs when set 1 was used for evaluation) up to a maximum feature set of 50 (mean value of 5 GA runs). The number of features only vary slightly for runs of the same configuration, but usually differ for different configurations.

[1] average of all V_t in Tab. 4

The evaluation is presented in Tab. 5. Again it has to be noted, that all configurations seem to suffer from oversearching, when the validation set V (the set on which the GA was performed) is equal to the test set T: the diagonal in Tab. 5 shows always the seemingly best values. Compared with the results gathered by the SPOT tuning (Tab. 4), GA feature selection leads to a slightly better predictive performance if we look at S_t, the mean off-diagonal RMSE.

Table 5. Mean results of five runs using feature selection by genetic algorithms. SVM and preprocessing parameters were obtained using the SPOT configurations 1,3,4 (see Sec. 3.1). The table shows the RMSE values for feature subsets on the validation sets leading to different feature configurations (rows). These configurations were evaluated on the test sets (columns); values in brackets indicate the standard deviation over five runs.

		Test		
		Set 1	Set 3	Set 4
	Set 1	**9.36 (0.11)**	15.48 (0.90)	11.44 (0.91)
Validation	Set 3	10.80 (0.19)	**12.11(0.59)**	7.78 (0.30)
	Set 4	10.99 (0.07)	13.04 (0.04)	**7.36 (0.03)**
S_t		10.90	14.26	9.61
V_t		16.40%	17.75%	30.57%

Even when feature selection does not produce much better results than SPOT tuning alone, it has an obvious positive effect on the RMSE ranges: the standard deviations of the configurations are considerably smaller with feature selection than without, leading to better generalizing models. Also the variance between the three off-diagonal RMSE's is lower than the high off-diagonal variance observed in experiment **T1**. A reason for this might be the complexity decrease of the models due to the lower number of input features. In addition to this, the runtime for model-building is also reduced.

3.3 T3: Final Optimization Using SPOT

One might expect that the best parameter configuration of SVM and embedding parameters has changed with the reduced feature subset found by GA. To check this hypothesis we have conducted SPOT again for two runs of the GA feature configurations, leading to tuned parameter configurations for the reduced feature sets. Although we observed slight improvements on the validation data set used for tuning, the results got worse on the test sets which indicates overtuning caused by this optimization. As a consequence we claim that the parameter configurations of **T1** are still valid after GA optimization. Nevertheless we cannot exclude that this final step can be important in other applications. We suggest to perform a parameter tuning, as soon as there is a change in the input to the regression model, at least for a few runs.

4 Conclusion and Outlook

We analyzed different predictive models based on Support Vector Machines (SVM) for a practical application named stormwater prediction. The results gathered by the general SVM method are in most cases better than the best-known special-purpose model (INT2). This can be seen as a confirmation of our hypothesis **H1**. It might have a great impact for applications which need a lot of similar models to be built since with our approach most of the time-consuming work of defining and tuning domain-specific models can be replaced by automatic processes.

Our results have also shown that one has to be careful when optimizing data-mining models by means of parameter tuning, e.g. with SPOT and GA: parameter tuning will often lead to oversearching and to too optimistic error estimates on the data sets used for tuning (as measured by V_t in Tab. 4 and Tab. 5), which was the statement of our hypothesis **H2**. Therefore the distinction between validation data sets (used for tuning) and independent test sets is essential to obtain a realistic estimate on the improvement reached by tuning. Nevertheless, our results have shown that tuning leads to better models as measured by independent test set RMSE. Also feature selection led to more stable results in our case study, which indicates better-generalizing models. In a nutshell, feature selection and SPOT tuning can help to improve results, but must always be validated on different test sets to recognize possible overfitting and oversearching effects.

Although the user is satisfied with the current results, we plan to extend and validate our study on other datasets, first by applying our methodology to different stormwater tanks and more comprehensive data (time periods stretching over several years). Also we want to increase the prediction horizon to larger values, enabling longer response times for the effluent control. Furthermore, other time series problems will be investigated, especially problems where no good preprocessing is known yet.

5 Acknowledgements

This work has been supported by the Bundesministerium für Bildung und Forschung (BMBF) under the grants FIWA and SOMA (AiF FKZ 17N2309 and 17N1009, "Ingenieurnachwuchs") and by the Cologne University of Applied Sciences under the research focus grant COSA. We are grateful to Prof. Dr. Michael Bongards, his research group and to the Aggerverband Gummersbach for discussions and for the stormwater tank data.

References

1. Bartz-Beielstein, T.: SPOT: An R package for automatic and interactive tuning of optimization algorithms by sequential parameter optimization. Tech. Rep. arXiv:1006.4645. CIOP TECHNICAL REPORT 05-10. COLOGNE UNIVERSITY OF APPLIED SCIENCES (Jun 2010), http://arxiv.org/abs/1006.4645, comments: Article can be downloaded from: http://arxiv.org/abs/1006.4645. Related software can be downloaded from http://cran.r-project.org/web/packages/SPOT/index.html
2. Bartz-Beielstein, T.: Experimental Research in Evolutionary Computation—The New Experimentalism. Natural Computing Series, Springer, Berlin, Heidelberg, New York (2006)
3. Bartz-Beielstein, T., Zimmer, T., Konen, W.: Parameterselektion für komplexe modellierungsaufgaben der wasserwirtschaft – moderne CI-verfahren zur zeitreihenanalyse. In: Mikut, R., Reischl, M. (eds.) Proc. 18th Workshop Computational Intelligence. pp. 136–150. Universitätsverlag, Karlsruhe (2008)
4. Brockwell, P.J., Davis, R.A.: Time series: theory and methods. Springer Verlag (2009)
5. Chang, C., Lin, C.: IJCNN 2001 challenge: Generalization ability and text decoding. In: Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on Neural Networks. vol. 2 (2001)
6. Drucker, H., Burges, C., Kaufman, L., Smola, A., Vapnik, V.: Support vector regression machines. Advances in neural information processing systems pp. 155–161 (1997)
7. Flasch, O., Bartz-Beielstein, T., Koch, P., Konen, W.: Genetic programming applied to predictive control in environmental engineering. In: Hoffmann, F., Hüllermeier, E. (eds.) Proceedings 19. Workshop Computational Intelligence. pp. 101–113. KIT Scientific Publishing, Karlsruhe (2009)
8. Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modelling. Wiley (2008)
9. Hilmer, T.: Water in Society – Integrated Optimisation of Sewerage Systems and Wastewater Treatment Plants with Computational Intelligence Tools. Ph.D. thesis, Open Universiteit Nederland, Heerlen (2008)
10. Kantz, H., Schreiber, T.: Nonlinear time series analysis. Cambridge Univ. Press (2004)

11. Koch, P., Flasch, O., Konen, W., Batz-Beielstein, T.: Predicting fill levels: Generic preprocessing and tuning of support vector regression models. in preparation for http://arxiv.org (2010)
12. Konen, W., Zimmer, T., Bartz-Beielstein, T.: Optimierte Modellierung von Füllständen in Regenüberlaufbecken mittels CI-basierter Parameterselektion. at – Automatisierungstechnik 57(3), 155–166 (2009)
13. Mattera, D., Haykin, S.: Support vector machines for dynamic reconstruction of a chaotic system. In: Advances in kernel methods. pp. 211–241. MIT Press (1999)
14. Müller, K., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.: Predicting time series with support vector machines. Artificial Neural Networks–ICANN'97 pp. 999–1004 (1997)
15. Quinlan, J.R., Cameron-jones, R.M.: Oversearching and layered search in empirical learning. In: In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. pp. 1019–1024. Morgan Kaufmann (1995)
16. Smola, A., Schölkopf, B.: A tutorial on support vector regression. Statistics and Computing 14(3), 199–222 (2004)