Using a multi-objective genetic algorithm for SVM construction

Orazio Giustolisi

ABSTRACT

Support Vector Machines are kernel machines useful for classification and regression problems. In this paper, they are used for non-linear regression of environmental data. From a structural point of view, Support Vector Machines are particular Artificial Neural Networks and their training paradigm has some positive implications. In fact, the original training approach is useful to overcome the curse of dimensionality and too strict assumptions on statistics of the errors in data. Support Vector Machines and Radial Basis Function Regularised Networks are presented within a common structural framework for non-linear regression in order to emphasise the training strategy for support vector machines and to better explain the multi-objective approach in support vector machines' construction.

A support vector machine's performance depends on the kernel parameter, input selection and ϵ-tube optimal dimension. These will be used as decision variables for the evolutionary strategy based on a Genetic Algorithm, which exhibits the number of support vectors, for the capacity of machine, and the fitness to a validation subset, for the model accuracy in mapping the underlying physical phenomena, as objective functions. The strategy is tested on a case study dealing with groundwater modelling, based on time series (past measured rainfalls and levels) for level predictions at variable time horizons.

Key words | artificial neural networks, multi-objective genetic algorithm, radial basis function regularised networks, support vector machines

Orazio Giustolisi
Engineering Faculty of Taranto,
Technical University of Bari,
via Turismo no 8, Paolo VI,
74100 Taranto,
Italy
E-mail: *o.giustolisi@poliba.it*;
oraziogiustolisi@libero.it

ACRONYMS

ANN	artificial neural network
ARX	autoregressive with exogeneous regressor type (model's input)
CoD	coefficient of determination
EC-SVM	evolutionary chaotic support vector machine
ES	evolutionary strategy
GA	genetic algorithm
LP	linear programming
MO	multi-objective genetic algorithm
MO-SVM	multi-objective support vector machine
NARX	input–output artificial neural network having the ARX regressor
NRMSE	normalised root mean squared error
NSGA	non-dominated sorting in genetic algorithm
OPTIMOGA	optimized multi-objective genetic algorithm
OPTISOGA	optimized single-objective genetic algorithm
PAES	Pareto archived evolutionary strategy
QP	quadratic programming
RBFRN	radial basis function regularised network
SO	single-objective genetic algorithm
SO-SVM	single-objective support vector machine
SPEA	strength Pareto evolutionary algorithm
SSE	sum of squared errors
SV	support vector
SVD	singular value decomposition
SVM	support vector machine
VEGA	vector evaluated genetic algorithm
VC	Vapnik & Chervonenkis dimension

doi: 10.2166/hydro.2006.016

NOTATION

$\text{avg}(H_{\text{exp}})$	average value of measured levels
b	parameter of the kernel
C	constraint for the second layer weights ($\|\mathbf{W2}\|_2 < C$) in QP training
D	Tikhonov regularisation parameter
f	filter factors in regularisation strategy of training
$\mathbf{G}$	matrix of the SVM
H	groundwater level
$\hat{H}$	groundwater level returned by the model
H_{exp}	measured value of groundwater level
$H_{\text{t-i}}$, $P_{t\text{-}i}$	past groundwater level and past total monthly rainfall in the SVM's input (components)
h	number of hidden neurons
h_{VC}	VC dimension
K_j	transfer function of the hidden neurons
k	cutting point in the diagram of singular values in LP training
L	function to measure of the closeness to training data in SVM
N	number of data
na	number of past outputs y or H in the model's ARX input
nb	number of past inputs x or P in the model's ARX input
nk	time unit delay of event's input x with respect to the event's output y
P	total monthly rainfall
u_i, v_i	left and right singular vectors corresponding to each singular value σ_i
W1, W2, W	matrix of the first and second layer weights
Y	vector of the event's output in the evaluation subset
$\hat{y}(t)$	model's prediction at time t
α, α^*	Lagrangian multipliers of the QP strategy of training
ϵ	insensitive level of Vapnik's error function or tolerance in LP training
η	level of confidence in approximating function in SVM
$\varphi_{\text{ARX}}(t)$	model regressor or model's input at time t
Σ	matrix of dilatations in radial construction
σ_i	ith singular value from SVD of $\mathbf{G}$
Ω	capacity of the machine in SVM
$\|\ \|_2$	Euclidean norm

INTRODUCTION

Artificial Neural Network (ANN) construction is the key issue to get a feasible regressive model in a generalisation scenario. It is well known that ANN construction implies two main troubles (Haykin 1999; Kecman 2000; Giustolisi & Laucelli 2005): (1) the *curse of dimensionality* and (2) *overfitting* to training data.

The *curse of dimensionality* is the exponential increasing of the parameter (weight) number with increasing the dimension of the input space in order to perform a function approximation preserving a constant level of accuracy. At the same time, the events of the training set become sparse (statistically speaking), increasing the dimension of the model's input space too.

A further issue relates to the ANNs' flexibility in mapping training events, which causes *overfitting*. This is ANNs' use to fit training events too strictly due to the huge number of weights. This trouble increases when the *curse of dimensionality* occurs (sparseness of the events of the training set) because over-fitted ANNs are candidates to produce poor predictions for those events that are far from the training ones in the model's input space.

There are several techniques to avoid overfitting (Giustolisi & Laucelli 2005), which are generally based on limiting fitness to training data. From the *curse of dimensionality* viewpoint, Support Vector Machines (SVMs), based on the Vapnik & Chervonenkis (1971) learning paradigm (Vapnik 1995; Haykin 1999; Kecman 2000), are a special improvement of ANN modelling. SVMs overcome the *curse of dimensionality* and related overfitting troubles using Vapnik's ε-insensitive error function that allows the selection of the hidden neuron number for a given accuracy of the model. Thus, the capacity of the SVM kernel machine is driven by the complexity of data, unlike in the ANNs where the capacity of the machine is *a priori* assumed throughout the selection of the hidden neuron number.

Therefore, SVM construction requires the selection of: (i) the regressor or model's input as in general regressive

input–output artificial neural networks (Haykin 1999; Giustolisi 2000); (ii) ε of the linear insensitive cost function, which works similarly to the selection of the hidden neuron number in classical regressive ANNs; and (iii) the parameter of the kernel (transfer function of the hidden neurons).

The use of the aforementioned variables as decision variables in a population-based optimisation strategy (here the Genetic Algorithm (GA) paradigm is used) may be a way of constructing an optimal SVM. It is possible to set up the strategy in a single-objective scenario as recently done by Yu *et al.* (2004) or in a multi-objective scenario as reported in this work. In this paper a two-objective strategy is used and compared with a single-objective approach.

The key idea for the multi-objective approach is to minimise the "final" hidden neuron number (the number of support vectors for the SVM) and at the same time maximising the fitness to a validation set as presented in the remainder. In the single-objective approach, the fitness to a validation set is maximised alone.

The selection of a cross-validation strategy (*estimation* and *validation* subsets for construction and a *test* set of "unseen data" for statistically computing generalisation performance) is used to make feasible the strategy. On the one hand, the perfect mapping (maximum fitness to training data) is always obtained when ε is equal to zero. This corresponds to supporting the mapping with the maximum number of vectors (hidden neurons) that equals the number of training data. On the other hand, we are interested in optimising the decision variables, looking at the generalisation performance. Thus, maximising the fitness to a validation set and minimising the capacity of the machine may be considered a cross-validation strategy in a multi-objective scenario based on evolutionary computing.

SVM STRUCTURE AND TRAINING

In this paper, we will use radial construction for SVM (Haykin 1999; Giustolisi 2004; Giustolisi & Laucelli 2005). Therefore, the initial mathematical structure of the SVM for regression is (assumed a radial construction):

$$\hat{y}(t,\mathbf{W},\mathrm{K}) = \sum_{\mathrm{j}=1}^{h=N} \left(\mathrm{K_j}\left(\|\varphi_{ARX}(t) - \mathrm{W1_j}\|_2, \sum\right)\cdot \mathrm{W2_j}\right) + \mathrm{W2_0} \quad (1)$$

where $\hat{y}(t)$ is the model's prediction or output at time t, **W1** and **W2** are the first and second layer weights, $\varphi_{\mathrm{ARX}}(t)$ is the model's regressor or model's input at time t whose components are the past event's inputs and outputs (ARX regressor) and $h = N$ is the number of hidden neurons equal to the number of training data. We will use the notation ARX(na, nb, nk) to address a regressor made up by na past outputs y and nb past inputs x that are nk-time-unit-delayed with respect to the output (see Figure 1).

From the kernel point of view, in Equation (1) Σ is a matrix of dilatations (usually a common dilatation parameter among the kernels is used instead of Σ), **W1** are the centres of the kernel function and $\|\ \|_2$ is the Euclidean norm.

Finally, the first-layer bias usually does not exist in the radial construction, being implicit in the kernel function, and the second layer bias is assumed equal to zero ($\mathbf{W2}_0 = 0$). A useful matrix form for Equation (1) is presented:

$$\hat{y}(t,\mathbf{W},\mathrm{K}) = \mathrm{K}\left(\sum, \mathbf{W1}, \varphi_{ARX}\right)\times \mathbf{W2} = \mathbf{G}\times\mathbf{W2} \quad (2)$$

Equation (2) will be useful in the remainder of the paper.

For understanding the structure of the SVMs and their on-line prediction features in the context of ANNs, the differences with input–output ANNs having the ARX regressor as the model's input (NARX) (Haykin 1999; Giustolisi & Laucelli 2005; Giustolisi 2004) are emphasised:

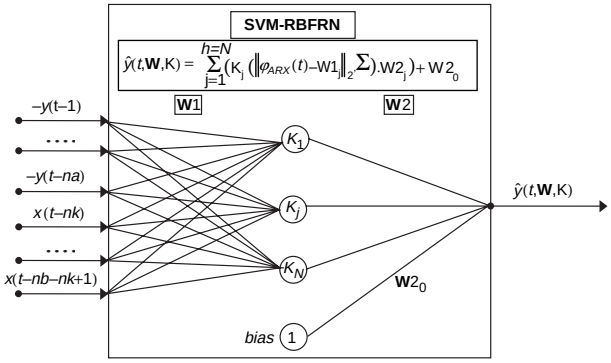

Figure 1 | SVM or RBFRN structure.

1. For SVMs the initial number of hidden neurons h is equal to the number of training data N, as in Equation (1), while for NARXs $h < N$ and it is better if $h \ll N$ (Giustolisi & Laucelli 2005).
2. For SVMs the first layer weights **W1** (centres of the kernel function in radial construction) for each component of the regressor is fixed from the ARX regression matrix of the training data, while for NARXs they are estimated together with **W2** during the training process.
3. For SVMs the second layer weights **W2** are evaluated as a linear optimisation problem assuming the so-called Vapnik ε-insensitive loss (error) function generating a ε-tube the regression function must lie within, after a successful learning. Note for NARXs the training process usually relates to a non-linear least square optimisation (Haykin 1999; Giustolisi & Laucelli 2005).
4. For SVMs the initial and final number of hidden neurons differ because of the training paradigm. Assumed $\varepsilon > 0$, some weights of the second layer are equal to zero (sparse solution) after learning. The non-zero weights correspond one-to-one to data points that are outside or on the bound of the ϵ-tube "supporting" the regression, while the other data points (weights equal to zero) are inside the ϵ-tube and they do not support the regression (error function is equal to zero, see Figure 2). For NARXs the number of neurons does not usually change before training.

Moreover, a comparison between Radial Basis Function Regularised Networks (RBFRNs) and SVM training algorithms is given to better understand the features of SVMs. In fact, RBFRNs and SVMs are both formalised by Equation (1) from a structural point of view. They differ for the training error function: for the RBFRNs the smooth mapping does not produce a sparse solution, while for the SVMs the tolerant mapping generate a sparse solution.

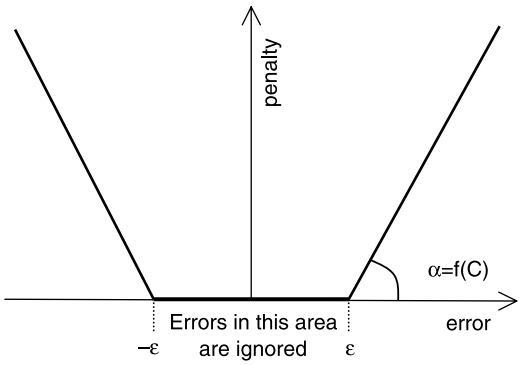

Figure 2 | Vapnik's error function for SVMs.

In fact, the second layer weights for RBFRNs are from

$$\mathbf{W2} = \arg\min_{\mathbf{W2}} \left[\sum_{t=1}^{N} \left(y(t) - \widehat{y}\,(t, \mathbf{W2}, K) \right)^2 + D \cdot \|\mathbf{W2}\|_2 \right] \tag{3}$$

where the first term is a measure of the closeness to the training data as the sum of squared errors (SSE) and the second term is a control for the model complexity related to Tikhonov's (1963) parameter D similar to C in the SVMs (Haykin 1999; Kecman 2000; Giustolisi 2004). The regularisation term smoothes the solution in the sense that similar inputs correspond to similar outputs. In this scenario, the training of RBFRNs is a determined ($N \times N$) linear problem whose solution is regularized by D. Therefore (Giustolisi 2004)

$$\mathbf{W2} = \sum_{i=1}^{N} \left(f_i \cdot \frac{u_i^T \times \mathbf{G}}{\sigma_i} \right) \cdot v_i \quad f_i = \frac{\sigma_i^2}{\sigma_i^2 + D} < 1 \tag{4}$$

where f_i are the filter factors related to the parameter D, σ_i are the singular values of $\mathbf{G}$ from its Singular Value Decomposition (SVD), and u_i and v_i are the columns of the left and right singular vectors corresponding to each singular value.

Note that the regularisation implies the well-conditioning of the mathematical inverse problem of finding **W2**, which may be ill-conditioned, **G** being a square matrix of high order. Note that RBFRNs perform as an interpolation of training data that is controlled, in its roughness/smoothness, by D.

Classical training for SVMs

The SVMs training paradigm uses the Vapnik & Chervonenkis (1971) dimension concept, VC, and the second layer weights are from (Vapnik 1995; Haykin 1999; Kecman 2000)

$$\mathbf{W2} = \arg\min_{\mathbf{W2}} \left[L(y(t) - \widehat{y}\,(t, \mathbf{W2}, K)) + \Omega(N, h_{VC}, \eta) \right] \tag{5}$$

where L is a function measuring the closeness to the training data and Ω is the so-called *capacity of the machine* dependent on the VC dimension h_{VC}, the number of training data N and the level of confidence in the approximating function η. Equation (5) is similar to Equation (3), but the SVM training strategy is to minimise the approximation fidelity to training data and, at the same time, the kernel machine capacity. The machine capacity increases with VC dimension, which depends on the number of support vectors, and decreases with the number of training data. For function L the selection of Vapnik's ϵ-insensitive loss function

$$L(\varepsilon) = |y(t) - \widehat{y}(t, \mathbf{W2}, K)|_{\varepsilon} = \begin{cases} 0 & \text{if } y(t) - \widehat{y}(t, \mathbf{W2}, K) \le \varepsilon \\ |y(t) - \widehat{y}(t, \mathbf{W2}, K)| - \varepsilon & \text{otherwise} \end{cases} \tag{6}$$

is more robust towards outliers and non-Gaussian noise of training data and implies a sparse solution (Haykin 1999; Kecman 2000).

In this scenario, the training of SVMs is a linear problem that becomes a Quadratic Programming (QP) problem (Vapnik 1995; Haykin 1999; Kecman 2000) if the second layer weights are subjected to the inequality $\|\mathbf{W2}\|_2 < C$ as the term for controlling machine capacity. Therefore, the solution of

$$[\alpha, \alpha^*] = \arg\max_{\alpha,\alpha^*} \left(-\frac{1}{2} \begin{bmatrix} \alpha \\ \alpha^* \end{bmatrix}^T \times \begin{bmatrix} \mathbf{G} & -\mathbf{G} \\ -\mathbf{G} & \mathbf{G} \end{bmatrix} \times \begin{bmatrix} \alpha \\ \alpha^* \end{bmatrix} + \begin{bmatrix} \varepsilon - y(t) \\ \varepsilon + y(t) \end{bmatrix} \times \begin{bmatrix} \alpha \\ \alpha^* \end{bmatrix} \right) \tag{7}$$

$$\text{subjected to } 0 \le \alpha, \alpha^* \le C$$

$$\text{subjected to } \sum \alpha = \sum \alpha^* \text{ explicit bias given}$$

where α are the Lagrangian multipliers, provides the second layer weights as

$$\mathbf{W2} = \alpha - \alpha^* \tag{8}$$

A couple of issues are stressed: (i) RBFRNs are trained performing a solution regularisation (Tikhonov 1963), then tuning between smoothing and approximation errors, while the key feature of the SVMs is in performing a subset selection of non-linear-dependent columns of the matrix $\mathbf{G}$ relying on the tolerance ε; and (ii) SVMs work accurately with a small set of training data, but solving QP with increasing N is quite difficult and memory consuming.

For this reason, several authors introduced decomposition-based methods to solve QP or a Linear Programming (LP) approach (Smola *et al.* 1999; Kecman 2000) that should be more robust and faster, in particular increasing the size of the problem. Finally, the way of selecting user-specified parameters ε and C is not supported by a comprehensive theory.

In this work, an alternative strategy to QP, based on 1-norm minimisation (Giustolisi 2004), is used. It is based on a different way of finding the subset of weights **W2** with respect to the tolerance ϵ using the SVD of $\mathbf{G}$, as reported in the next section. The advantages of this alternative are: (i) it requires just one parameter to tune (ε) that is strictly related to the number of support vectors, and (ii) it is fast for small and medium sized problems, a fast SVD algorithm being necessary, like that implemented in the MATLAB environment (Golub & Van Loan 1993), and a fast and efficient solver for a LP problem (Barrodale & Roberts 1973).

Training by 1-norm of weights estimation

To find the second layer weights of the SVM, the problem

$$\min\|\mathbf{W2}\|_1 \quad \text{subject to } \min\|\mathbf{G}_k \times \mathbf{W2} - \mathrm{Y}\|_2 \qquad \mathbf{G}_k = \sum_{i=1}^{k} u_i \cdot \sigma_i \cdot v_i^{\mathrm{T}} \tag{9}$$

is solved (Giustolisi 2004), where $\mathbf{Y}$ is the sequence of the target data and k is the cutting point in singular values σ_i of $\mathbf{G}$ (Figure 3). Solving the problem in Equation (9) returns a sparse solution, as well as in QP training of SVMs. Thus some weights **W2** are set to zero according to the mapping tolerance ε. The parameter which controls the mapping approximation is $\varepsilon = \sigma(k)/\sigma_{\max}$ (see Figure 3). The novel feature is that k corresponds to the hidden neuron number (non-zero **W2**) (Giustolisi 2004).

From a mathematical viewpoint, this is a piecewise-polynomial truncated singular value decomposition technique (Hansen & Mosegaard 1996) performing a subset selection for columns in $\mathbf{G}$ (which singularly correspond to hidden neurons) according to the tolerance ε, which is a linear independence constraint for columns (Giustolisi 2004). It is important that the truncating point k controls

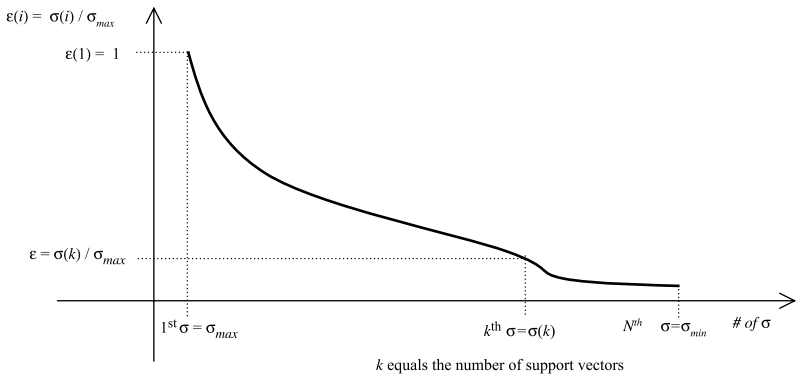

Figure 3 | Diagram of singular values of **G** and truncating point *k*.

the hidden neuron number as well as the tolerance of Vapnik's ϵ-insensitive error function (Vapnik 1995).

USING MULTI-OBJECTIVE GENETIC ALGORITHM FOR SVM

Evolutionary computing for designing SVMs was already used for chaotic SVM (EC-SVM) by Yu *et al.* (2004). They used a single-objective population-based optimisation strategy. The decision variables were: (i) the time delay and the embedded dimension for the input definition and (ii) C, ε and σ (of the Gaussian kernel) for the SVM. The single objective was the fitness to the test set.

In this work, a similar strategy introducing something different is used:

- two radial basis kernels have been tested and compared:

$$\mathrm{K_j}\left\{\|\varphi - \mathrm{W1_j}\|_2, \sum\right\} = e^{-b\cdot\|\varphi-\mathrm{W1_j}\|_2} \quad \text{Gaussian function}$$
$$\mathrm{K_j}\left\{\|\varphi - \mathrm{W1_j}\|_2, \sum\right\} = \frac{1}{\sqrt{b\cdot\|\varphi-\mathrm{W1_j}\|_2+1}} \quad \text{Inverse Multiquadratic function} \tag{10}$$

- Both single-objective and multi-objective strategies have been used and compared.
- The regressor is not from chaos theory, but it is composed by *na* candidate past outputs and *nb* candidate past inputs. The coding of the related decision variable is a binary string (0 or 1 specifies the existence of each component in the regressor).
- The SVM parameters are ϵ and b of the two radial basis kernels in Equation (10).
- For the multi-objective strategy, the maximisation of the fitness to a validation subset and the minimisation of the VC dimension (by means of the number of support vectors) has been performed.
- For the single-objective strategy, the maximisation of the fitness to a validation subset has been made.
- A multi-objective or single-objective genetic algorithm strategy has been used as a population-based optimizer.
- Training of the SVM is performed by a novel 1-norm based strategy claiming just for one parameter to be tuned.

This paper will demonstrate that there are some advantages in using multiple objectives in spite of a single objective. The so-called Pareto front of non-dominated SVMs allows the user to visualise the bests models both from structural parsimony (minimisation of the VC dimension) and fitness from a validation subset viewpoint (i.e. no one of the non-dominated SVMs is, at the same time, better than the others both from parsimony and fitness viewpoints). The Pareto front of the best SVMs improves the testing phase (the model selection phase), providing a more robust user confidence about the generalisation performance of the model that otherwise is only statistically based (see the rest of the paper).

BACKGROUND OF MULTI-OBJECTIVE STATEGY IN GA

GAs have been initially used in performing single-objective optimisation (SOGA) by means of one objective function

to feature a particular property of each individual. However, many real-world problems involve simultaneous optimisation of multiple objectives (Savic 2002). Multi-objective optimisation is very different from single-objective optimisation. In single-objective optimisation one attempts to obtain the best design or decision which is usually the global minimum or the global maximum, depending on whether the optimisation problem is that of minimisation or maximisation. In the case of multiple objectives there may not exist one solution which is the best global minimum or maximum with respect to all objectives. Therefore, since none of the solutions in the non-dominated set is absolutely better than any other, any one of them is an acceptable solution. The choice of one solution over another requires a knowledge of the problem and a number of problem-related factors. Thus, one solution chosen by a designer may not be acceptable to another designer or in a changed environment. Therefore, in multi-objective optimisation problems, it may be useful to have knowledge about alternative Pareto optimal solutions (Savic 2002).

One way to solve multi-objective problems is to scale the vector of objectives into one objective by averaging the objectives with a weight vector. This process allows a simpler optimisation algorithm to be used, but the obtained solution largely depends on the weight vector used in the scaling process. An early GA application of multi-objective optimisation by Schaffer (1984) opened a new avenue of research in this field. His algorithm, the Vector Evaluated Genetic Algorithm (VEGA), gave encouraging results. In this approach appropriate fractions of the next generation, or subpopulation, are selected separately according to each of the objectives. Usual crossover and mutation are applied after shuffling all subpopulations together. Non-dominated individuals are identified by monitoring the population as it evolves. It was later noted by some authors (Fonseca & Fleming 1998) that shuffling subpopulations together corresponds to averaging the fitness components associated with each of the objectives. Since Shaffer (1984) used a proportional fitness assignment, the resulting expected fitness corresponds to a variability weighted linear combination of the objectives analysed.

Later on, a method of ranking and fitness assignment, based on the Pareto optimality concept, was proposed by Goldberg (1989). In this method the population is divided into several subpopulations according to the assignment of a rank number, i.e. in each generation each subpopulation is assigned a rank number, which represent a fitness indicator.

Another multi-objective strategy is introduced by Fonseca & Fleming (1993) and is called MOGA. In this approach, there is an individual's rank corresponding to the number of individuals in the current population by which it is dominated. In this scheme, the non-dominated individuals are assigned the same rank, while the dominated individuals are penalised according to the population density in the corresponding region of the trade-off surface (Savic 2002).

Zitzler & Thiele (1998) proposed an elitist multi-criterion evolutionary algorithm, called the Strength Pareto Evolutionary Algorithm (SPEA). Starting from the initial generation, they create an archive population containing the non-dominated solutions discovered. At each generation, the archive population and the current population are combined and ranked according to a fitness criterion based on the number of solutions they dominate; the dominated solutions are assigned by fitness, which is worse than the worst non-dominated solution one. Moreover, a deterministic clustering is used to ensure diversity among the non-dominated solutions.

Knowles & Corne (1999) proposed a simple multi-objective evolutionary algorithm using a single parent–single child evolutionary algorithm. In this strategy, binary strings and bitwise mutation are used to create children in replacement of real parameters. It is called the Pareto Archived Evolutionary Strategy (PAES).

Deb *et al.* (2002a) introduced a fast and elitist multi-objective genetic algorithm based on evolution of the non-dominated sorting in genetic algorithm (NSGA) presented by Srinivas & Deb (1995) and called NSGA-II. In this algorithm, to sort a population of assigned size according to the level of non-domination, each solution must be compared with every other solution in the population to find if it is dominated. Solutions of the first non-dominated front are stored in the *first Pareto front*, solutions of the second front on the *second Pareto front* and so on. The new population is constituted by solutions on the *first Pareto front*, if they are less than the initial population size: solutions from the next front are taken according to their ranks.

BRIEF DESCRIPTION OF OPTIMOGA

Giustolisi *et al.* (2004) recently proposed an algorithm for multi-objective genetic algorithms (MOGA). It optimises both the exploration of the objective space and the exploitation of the best-computable front of non-dominated solutions. As reported in Giustolisi *et al.* (2004), the algorithm emphasises the explorative and spreading attitudes of an evolutionary strategy, optimising the management of the population size and, in particular, the rank assignment of the solutions in the objective space. Afterwards, once the exploration of the objective space becomes secondary to the convergence towards the trade-off surface of the objective functions, the population of solutions and the genetic operators are managed in order to push a sort of local optimisation. On this premise, this algorithm has been called an optimized multi-objective genetic algorithm (OPIMOGA). Furthermore, the OPTIMOGA can be easily turned into a single-objective GA, which is then referred to as OPTISOGA. The OPTIMOGA/OPTISOGA is able to deal with differently codified numerical problems, which encompass binary, integer and real variables.

As for all evolutionary strategies (ESs), and generate-and-test algorithms in general, the OPTIMOGA draws its strength from two sources: exploration and exploitation. The exploration and exploitation are regarded as opposite contributions to establish the proper level of selective pressure and diversity throughout the search process. The exploration is meant as the phase in which a population of possible solutions is evolved through the objective space in order to create a front of non-dominated solutions. In this phase, the key issue is to prevent solutions from leaving zones of the objective space unexplored, rather than encouraging a wide spreading of the solutions into the objective space. The exploitation is meant as the phase in which the population converges towards the theoretical Pareto front and its fine coverage, corresponding to as close as possible a reproduction of the theoretical front, is sought. Usually, explorative properties are attributed to genetic operators such as recombination and mutation, while exploitative ones to selection. In all ESs, the latter is referred to as the mechanism that considers the quality (fitness) of individuals to make a choice as to who to favour for reproductive purposes. When exploration and exploitation are unbalanced, either the search stalls around unpromising areas of the objective space or it prematurely converges to some local optima. The OPIMOGA algorithm adopts an explorative strategy in the first stage and an exploitative approach once the Pareto front is achieved. At the same time an archive of non-dominated solutions is created. In the second stage an exploitation strategy is pursued: the entire population belonging to the Pareto front is evolved, up to a maximum size. However, when the maximum size is reached, the archive of solutions starts to grow in size, storing the non-dominated solutions. Note that the archive can be involved in the evolution, i.e. solutions from the archive can be used to support the exploitation of the Pareto front. Therefore, the archive is dynamically managed, since it is both upgraded at each evolutionary loop and the solutions that it stores can be used to create the next mating pool. On this premise, we emphasise that the OPTIMOGA is typified by a few parameters; we mean that the user is not required to specifically tune the GAs settings. Furthermore, it is experienced during the development of the OPTIMOGA that the algorithm is not sensible to the variation of its settings. For instance the same initial population size, together with the same crossover and mutation rate, proved to be fitted to different problems. In this scenario, the key issue is avoiding as far as possible the introduction of deterministic procedures in a stochastic search, which is driven by the fitness. The effectiveness of the newly proposed procedure has been assessed on six widely used continuous test problems, namely DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ5 and DTLZ6 (Deb *et al.* 2002b).

THE CASE STUDY

A case study concerned with modelling the relationship between rainfalls P (event's input x) and groundwater levels H (event's output y) by SVMs for an aquifer located in the southeast of Italy is presented (Ricchetti & Polemio 1996).

Background to data

Figure 4 shows the location of the well from which the level data used for the application of multi-objective/single-objective support vector machine (MO/SO-SVM) were sampled. The available data collections are: (1) a rainfall

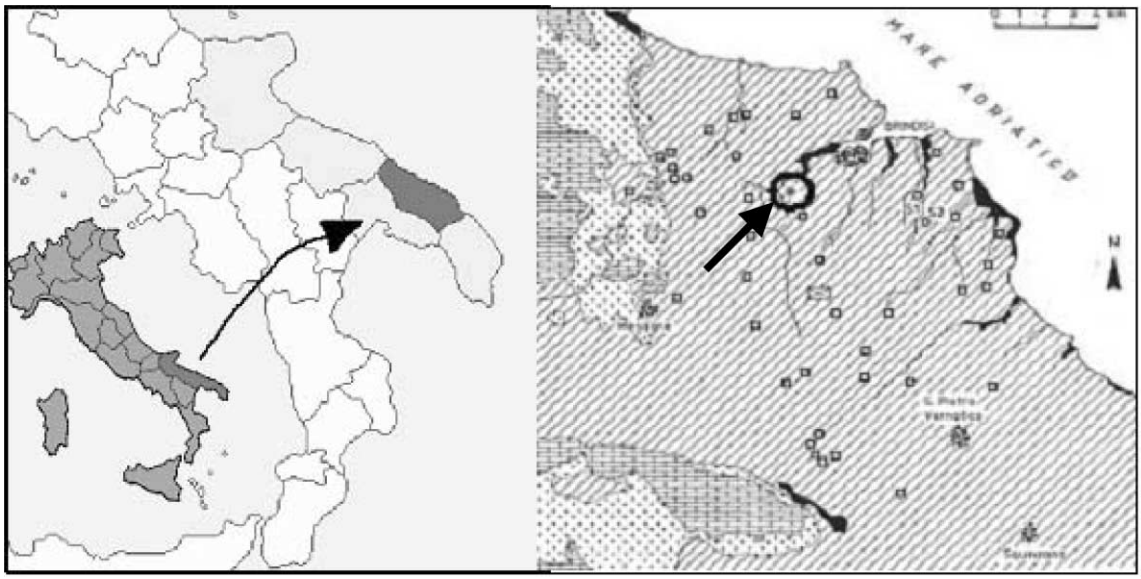

Figure 4 | Location of the sampling well.

time series and (2) a groundwater level time series. Each one of these data collections consists of 528 data points. The rainfall data series consists of monthly cumulative heights measured in cm. The groundwater series consists of the average monthly values of the piezometric head of water referred to the average sea level: the mouth of the well is located at 35.92 m a.s.l. (above sea level). Both the rainfall and groundwater data series cover a 44-year period: between January 1953 to December 1996. Figures 5 and 6 show the time plot of rainfall and groundwater levels, respectively.

PRELIMINARY ASPECTS OF MODELLING

Before modelling, both time series were split into two subsets: the first made up of 300 samples, called the *training set*, and the second made up of 228 samples, called the *test set*. Furthermore, the *training set* was split in two other subsets: the first made up of 216 samples (75%), called the *evaluation subset*, and the second made up of 84 samples (25%), called the *validation subset*.

The *evaluation subset* was used for weight estimation and *the validation subset* was used for fitness evaluation for

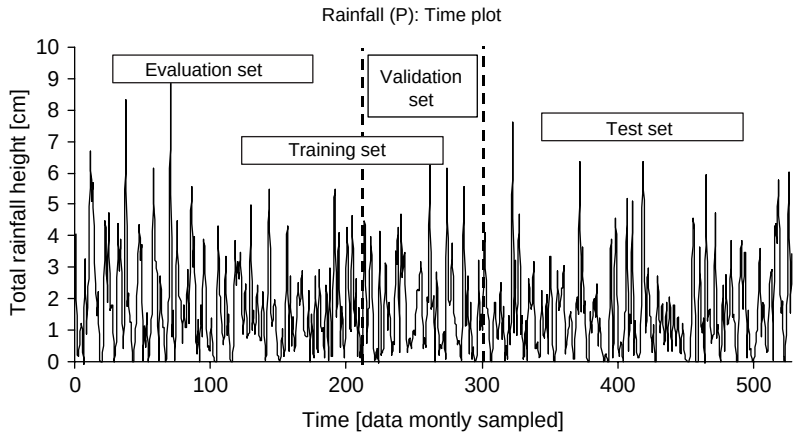

Figure 5 | Rainfall time plot of available data.

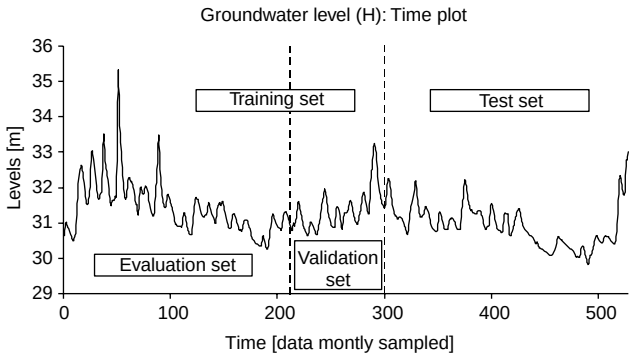

Figure 6 | Groundwater level time plot of available data.

a population-based (OPTIMOGA or OPTISOGA) strategy. The test set is considered in a latter phase, when the optimiser has already generated the set of best models (SVMs). In this phase it is important to test the generalisation capabilities of the models, i.e. to assess how these models perform when fed with an input data set different from that used to identify them (unseen data). Thus, the model's performance was evaluated using the coefficient of determination (CoD) computed on the test set. The CoD is defined as

$$\begin{aligned} \mathrm{CoD} &= 1 - \frac{\sum_{N}\left(\hat{H} - H_{\exp}\right)^2}{\sum_{N}\left(H_{\exp} - \mathrm{avg}(H_{\exp})\right)^2} = 1 - m \cdot \mathrm{SSE} \\ &= 1 - (\mathrm{NRMSE})^2 \\ m &= \frac{N}{\sum_{N}\left(H_{\exp} - \mathrm{avg}(H_{\exp})\right)^2}. \end{aligned} \tag{11}$$

where N is the number of samples, $\hat{H}$ is the value of groundwater level returned by the model, $H_{\exp}$ is the measured value of groundwater level and finally $\mathrm{avg}(H_{\exp})$ represents the average value of measured groundwater levels evaluated on the N samples. We should emphasise from Equation (11) that CoD, SSE and NRMSE (normalized root mean squared error) are strictly correlated (note that the value m does not depend on the particular model), belonging to the same membership for fitness evaluation.

The approach can support any objective function (Hall 2001) given by the user. In fact, while for SVM the error function for parameter estimation is that reported in Equation (6) or Figure 2, the fitness evaluation for a population-based (OPTIMOGA or OPTISOGA) strategy by the validation set and the performance evaluated by the test set for model selection are user-defined according to the specific purpose of the model.

Regarding the MO/SO-SVM technique, the software was implemented in the MATLAB environment. The features used to perform MO/SO-SVM construction were:

- training of the SVMs by 1-norm paradigm for weight estimation,
- two different kernels, see Equations (10): inverse multi-quadratic function and Gaussian function,
- one decision variable b for both the kernels, see Equations (10),
- one decision variable ϵ for the training paradigm,
- maximum na and $nb + nk$, respectively, equal to 4 and 13 (note that nk derives from selection of the lowest index of the P components); thus the candidate inputs were H_{t-1}, H_{t-2}, H_{t-3}, H_{t-4}, P_t, P_{t-1}, P_{t-2}, P_{t-3}, P_{t-4}, P_{t-5}, P_{t-6}, P_{t-7}, P_{t-8}, P_{t-9}, P_{t-10}, P_{t-11}, P_{t-12},
- linear scaling of the time series in the range [0 1] when used in the SVM,
- OPTIMOGA search for the Pareto front considering minimisation of 1-CoD (computed by the validation subset) and the number of support vectors,
- OPTISOGA search for the best SVM by minimising 1-CoD (computed by the validation subset).

Regarding the OPTIMOGA/OPTISOGA, their main features were:

- GA mixing binary strings for the model's input component selection and integer strings for coding the two real numbers (b and ε),
- $2 \leq b \leq 10^{-5}$ and $1 < \varepsilon < 10^{-5}$ bounds for real decision variables,
- multi-point crossover with probability rate equal to 0.4,
- single-point mutation with probability rate equal to 0.1,
- maximum population in evolution equal to 40 individuals,
- generation number equal to 500.

Finally, from a practical viewpoint it is important to report that a MO/GA-SVM strategy implemented in the MATLAB environment took about 80–100 min (depending on the kernel: Gaussian or inverse multi-quadratic) for 500 generation runs using a PC with an Intel Pentium IV, 2600 MHz processor and Windows XP operating system.

RESULTS AND COMPARISONS

Table 1 reports the results (actually more than one run was performed, obtaining similar results) concerning the implementation of two kernels (Gaussian and inverse multi-quadratic) and the MOGA and SOGA population-based strategy.

In order to obtain a more reliable estimate of the CoD evaluated on the test set ([1 month; 3 months; 6 months; 9 months; 12 months] ahead predictions as in Table 1), data were resampled 1000 times according to a bootstrap scheme (Efron 1979). Therefore, the CoD values in Table 1 are average values. Moreover, the CoD of the naive model equal to [0.920, 0.583, 0.226, 0.032, -0.136] for 1 month, 3 months, 6 months, 9 months and 12 months ahead predictions was computed as a benchmark for values in Table 1.

In the comparison between MOGA and SOGA strategies, no significant difference between their run-times was observed. The results imply:

- For the Gaussian kernel the SOGA strategy provides a SVM having 10 support vectors (10-SV machine) and for this reason a lower tolerance ε (i.e. higher accuracy with respect to the estimation set) compared to the less parsimonious SVM of the MOGA strategy. The model's input is equal to the 9-SV machine of the MOGA strategy and the fitness to the validation subset (*Obj Fun*: *1-CoD*) is comparable to the same 9-SV machine. The performance on the test set is similar until 3-months ahead predictions, becoming worse for the SOGA approach and increasing the prediction horizon.
- for the inverse multi-quadratic kernel the SOGA strategy provides a 14-SV machine and for this reason a lower tolerance ε (i.e. higher accuracy with respect to the estimation set) compared to the less parsimonious SVM of the MOGA strategy. The model's input is very different from all the SVMs of the MOGA strategy and much longer. The fitness to the validation subset (*Obj Fun*: *1-CoD*) is comparable to a 5-SV machine of the MOGA strategy. The performance on test proves itself similar to the best MOGA-SVMs, SOGA-SVM not being comparatively satisfactory for short-time predictions.
- SOGA generally provides a SVM that is less parsimonious with respect to all the MOGA-SVMs, without offering a better generalisation performance. The kernel parameter result is similar in both the MOGA and SOGA approach.

Now, looking at the comparison between the performance of the Gaussian and the inverse multi-quadratic kernels:

- The best Gaussian SVMs looks generally similar performing on the test set (excluding in the analysis the trivial 2-SV machines) compared to the best inverse multi-quadratic SVMs, but the Gaussian kernels prove to have a steadier generalisation performance across the SVMs, making more parsimonious structures generally available both in the SOGA and MOGA cases.
- When the Gaussian kernel is used, the model's input sometime involves the oldest past rainfall components (i.e. P_{t-8}, P_{t-9}, P_{t-10}, P_{t-12}).
- When the Gaussian kernel is used, the model's input (2-SV machine excluded) always involves H_{t-1}, H_{t-2}, P_t, P_{t-1}, as closer past components, while when the inverse multi-quadratic is used, there is never H_{t-2}, and sometimes H_{t-3}, H_{t-4} appear.
- The inverse multi-quadratic kernel always requires a lower parameter b value than the Gaussian kernel.

DISCUSSION ON MODEL SELECTION

The model selection is a very important issue. In fact, the decision to use or not to use a model after construction is a critical choice for the user. What usually drives the selection of the model in a single-objective strategy of parameter design is the statistical performance evaluation on a test set of data (unseen data) by means of user-selected objective functions (Hall 2001). Unfortunately, this test can only provide an estimation of the generalisation performance of the model. This estimate may be biased by the finite length of the test set, besides being corrupted by errors. Thus, the single realisation of errors may bias the estimate because the particular model fits it. On the other hand, the performance required for a model is not necessarily the highest considering the estimated prediction performance, but a certain robustness of the selection is equally important looking at the future use on new sets of data of the model being selected. In this scenario, the multi-objective strategy may be considered as a decision support tool for a more robust selection of the SVM through considerations that are

Table 1 | MO/SO-SVMs' parameters and estimated performance on test set

Decision variables																			Obj Fun		CoD on Test Set at different horizon predictions				
b	ε	H_{t-1}	H_{t-2}	H_{t-3}	H_{t-4}	P_t	P_{t-1}	P_{t-2}	P_{t-3}	P_{t-4}	P_{t-5}	P_{t-6}	P_{t-7}	P_{t-8}	P_{t-9}	P_{t-10}	P_{t-11}	P_{t-12}	1-CoD	SV	1 month	3 months	6 months	9 months	12 months
Gaussian kernel (MOGA)																									
0.304	0.0088	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	0	1	0.04123	9	0.94912	0.78738	0.68378	0.69183	0.68294
0.284	0.00774	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	0	0	0.042118	8	0.95644	0.81192	0.71938	0.70099	0.67863
0.33888	0.0075	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	0.042186	7	0.95211	0.79416	0.68982	0.69496	0.69771
0.31716	0.00695	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0.043941	6	0.9606	0.82868	0.71005	0.69239	0.6583
0.33668	0.0047	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0.045263	5	0.95661	0.81096	0.70038	0.68243	0.68719
0.27576	0.0077	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0.056004	4	0.95669	0.81121	0.66295	0.56144	0.47617
0.2	0.00769	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.072711	3	0.94031	0.58288	0.067037	−0.26703	−0.55752
0.292	0.01677	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.116	2	0.91826	0.57311	0.22261	0.046933	−0.08796
Gaussian kernel (SOGA)																									
0.26	0.00273	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	0	1	0.04139	10	0.95481	0.79388	0.5835	0.39022	0.10487
Inverse multi-quadratic kernel (MOGA)																									
0.9798	0.0006	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0.043256	12	0.96385	0.81747	0.64014	0.48396	0.33785
0.1016	0.00054	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0.046488	7	0.9615	0.83485	0.71972	0.68166	0.64775
0.1656	0.00064	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0.047896	6	0.94283	0.69607	0.43122	0.30875	0.25809
0.18	0.00271	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0.052133	5	0.95217	0.79294	0.65577	0.6004	0.55701
0.072526	0.00157	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0.063851	4	0.94206	0.69114	0.35779	0.16798	0.063839
0.060002	0.0015	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0.080888	3	0.94529	0.78261	0.62353	0.53122	0.46672
0.1628	0.14248	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.11603	2	0.91369	0.55293	0.19614	0.011989	−0.12118
Inverse multi-quadratic kernel (SOGA)																									
0.20001	0.00035	1	0	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	0.053713	14	0.94136	0.78662	0.68183	0.63319	0.58391

not only statistically based. In fact, the Pareto front of the best (most parsimonious and accurate) SVMs provides a description of the SVMs' family properties in modelling the groundwater data, see Table 1. Looking at the Gaussian kernel in Table 1, the model's input presents some similarities across the SVMs as well as b. In this way, the user may observe from the Pareto front of the SVMs that increasing the number of support vectors means the length of the input increases, the oldest past rainfall components P_{t-8}, P_{t-9}, P_{t-10}, P_{t-12} now being involved. Thus, there is a sort of contiguity of the SVMs. Looking at the CoD performance on the test set in Table 1, they increase in all the prediction horizons from 2-SV to 6-SV machines. When the number of SV ranges between 7 and 9 the performance decreases for some line horizon predictions.

It is possible to observe that this situation corresponds to the use of the new components P_{t-8}, P_{t-9}, P_{t-10}, P_{t-12}. Thus, the decision support table indicate as candidates for the SVM selection the 5-SV or 6-SV machines. The difference between the two machines is in one hidden neuron (support vector) and the introduction of the P_{t-9} component in the model's input for the less parsimonious SVM. This seems to slightly improve the estimation of the prediction performance. For example, the author's choice is for the most parsimonious SVM, whose prediction performance on the test set is depicted in Figure 7, thus considering the pressure for simplicity as very important for model robustness. Moreover Figure 7 shows that the months from 470–480 are critical for SVM prediction. This is due to the attitude of SVMs, and in general black-box models, at interpolating or slightly extrapolating with respect to training data. Clearly, the poorness in prediction may be in the model construction or caused by test set events far from the training ones. However, Figure 7 shows that the selected model generally has good performance on "unseen data" (test set) and never failing (consider that 228 data means 19 years of testing).

Therefore, the selected SVM is not the best performing on the test set, but has a good trade-off between parsimony and accuracy. Moreover, the fact that performance is comparable from 6 months to 12 months indicates that machine has picked up the "deterministic" relations between the event's input and output (Giustolisi 2000; Giustolisi & Laucelli 2005).

In fact, the SVM may be seen as a two-component model: the first component allows modelling of the relationship between rainfalls and groundwater levels (it is named deterministic, since it models a determinate input–output relation), the latter component (named probabilistic) models short-time effects due to the unknown or secondary extra inputs and/or noise correlations in a statistical way, in order to improve on-line groundwater level forecasting using past measured data (Giustolisi 2000; Giustolisi & Laucelli 2005).

The fact that occasionally the CoD of the 12-months line horizon prediction is higher than the 9-months one is

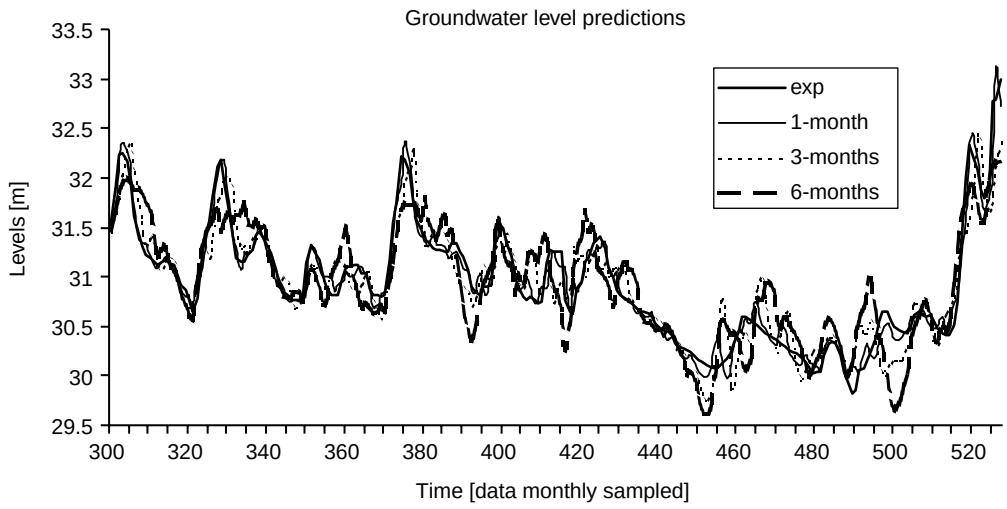

Figure 7 | Measured groundwater level vs predicted at different line horizons.

theoretically not correct, but is caused by the estimation scenario. Therefore, the prediction performance can be assumed comparable and the prediction error constant increasing the horizon forecasting. Thus, the deterministic component is working well since for horizon predictions longer than short time effects (modelled by the probabilistic component) the prediction error should become constant as well as in the physically based models.

Looking at the Gaussian kernel, the same use of the Pareto front of SVMs for the inverse multi-quadratic kernel shows that the structures are characterised by a variability of b and model's input.

The 5-SV and 7-SV machines look similar as input (the same length but they differ for H_{t-3} and P_{t-10}) and the best performance on the test set. In this case, we may consider the decision for SVM selection less supported (less robust) because of the slight similarity in the structures of the Pareto front than in the Gaussian kernel case. Moreover, the selection between 5-SV and 7-SV machines is less unequivocal because it may be driven by performance or parsimony.

CONCLUSIONS

This paper proposes a multi-objective strategy, MO-SVM, for automatic design of the support vector machines based on a genetic algorithm. The strategy is compared with a single-objective one performed in the same framework.

For the multi-objective strategy, the OPTIMOGA optimiser found the Pareto front of non-dominated SVMs considering maximisation of the fitness to a validation subset and the minimisation of the VC dimension (by the number of support vectors), while for the single-objective strategy the fitness maximisation was performed. Furthermore, two classical kernels from the literature for the transfer function of the hidden neurons were tested.

Regarding the kernels, the Gaussian function proves to be more robust, compared to the inverse multi-quadratic kernel, providing good steady performance in groundwater modelling accuracy throughout the non-dominated SVMs.

The multi-objective strategy proves to be better than the single-objective strategy because the Pareto front of SVMs are found to be more parsimonious and, contemporarily, more accurate in prediction performance (assuming different prediction horizons) of the groundwater levels than the single SVM (2-SV and 3-SV machines are not considered in this).

Furthermore, the multi-objective strategy seems to be more robust for SVM selection than the single-objective one. Multi-objective makes it possible to look at the similarities and contiguities of the structures belonging to the Pareto front of the non-dominated SVMs as a decision support tool. Besides, the decision for selection in the single-objective strategy may be critically influenced by the specific realisation of the errors in a "finite size" test set when computing performance, being only statistically based on the estimation of the generalisation performance.

Finally, the parameter ε of Vapnik's error function seems to be related to the specific kernel adopted and not to data noise. Mind that data do not change across the runs.

ACKNOWLEDGEMENTS

The author wish to kindly acknowledge Dr Angelo Doglioni for the support provided in the development of the algorithm OPTIMOGA.

REFERENCES

Barrodale, I. & Roberts, F. D. K. 1973 An improved algorithm for discrete L1 linear approximation. Numer. Anal. *SIAM J. Numer. Anal.* **10**, 839–848.

Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. 2002a A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Trans. Evolut. Comput.* **6** (2), 182–197.

Deb, K., Thiele, L. & Zitzler, E. 2002b Scalable multi-objective optimization test problems. In *IEEE Congress on Evolutionary Computation (CEC 2002)*, IEEE Press, Piscataway, NJ. pp. 825–830.

Efron, B. 1979 Bootstrap methods. Another look at the jackknife. *Ann. Stat.* **7**, 1–26.

Fonseca, C. M. & Fleming, P. J. 1993 Genetic algorithms for multi-objective optimisation: formulation, discussion and generalisation. In *Proc. of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, pp. 141–153.

Fonseca, C. M. & Fleming, P. J. 1998 Multiobjective optimisation and multiple constraint handling with evolutionary algorithms I: A unified formulation - PART I - A unified formulation. *IEEE Trans. Syst. Man Cybern.* **28** (1), 26–37.

Giustolisi, O. 2000 Input-output dynamic neural networks simulating inflow-outflow phenomena in a urban hydrological basin. *J. Hydroinf.* **2** (4), 269–279.

Giustolisi, O. 2004 Sparse solution in training artificial neural network. *NeuroComputing 56C*, 284–304.

Giustolisi, O., Doglioni A., Savic, D. & Laucelli, D. B. 2004 *A Proposal for an Effective Multi-objective Non-dominated Genetic Algorithm: the Optimised Multi-Objective Genetic Algorithm: OPTIMOGA*. Report 07. School of Engineering, Computer Science and Mathematics, Centre for Water Systems, University of Exeter.

Giustolisi, O. & Laucelli, D. 2005 Increasing generalisation of input-output artificial neural networks in rainfall-runoff modelling. *Hydrol. Sci. J.* **50** (3), 439–457.

Goldberg, D. 1989 *Genetic Algorithms in Search Optimization and Machine Learning*. Addison Wesley, New York.

Golub, G. H. & Van Loan, C. F. 1993 *Matrix Computations*. The Johns Hopkins University Press, London.

Hall, M. 2001 How well does your model fit the data? *J. Hydroinf.* **3** (1), 49–55.

Hansen, P. C. & Mosegaard, K. 1996 Piecewise polynomial solutions without a priori break points. *Numer. Linear Algebra Appl.* **3**, 513–524.

Haykin, S. 1999 *Neural Networks: A Comprehensive Foundation*, 2nd edn. Prentice-Hall. Englewood Cliffs, NJ.

Kecman, V. 2000 *Learning and Soft Computing by SVM, NN and FLS*. MIT Press, Cambridge, MA.

Knowles, J. & Corne, D. 1999 The Pareto archived evolution strategy: a new baseline algorithm for multiobjective optimisation. In *Proceedings of the Congress on Evolutionary Computation* (ed. Z. Michalewicz), IEEE Press, Piscataway, NJ pp. 82–87.

Ricchetti, E. & Polemio, M. 1996 L'acquifero superficiale del territorio di Brindisi: dati geoidrologici diretti ed immagini radar da satellite. *Memorie Società Geologica Italiana* **51** (2), 1059–1074. (in Italian).

Savic, D. A. 2002 Single-objective vs. multi-objective optimisation for integrated decision support. In *Integrated Assessment and Decision Support, Proceedings of the First Biennial Meeting of the International Environmental Modelling and Software Society* (eds. A. E. Rizzoli & A. J. Jakeman), vol. 1, iEMSs. Manno, Switzerland, pp. 7–12.

Schaffer, J. D. 1984 *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. Doctoral dissertation, Electrical Engineering, Vanderbilt University, TN.

Smola, A., Friess, T. & Schölkopf, B. 1999 Semiparametric support vector and linear programming machines. In *Advances in Neural Information Processing Systems*, vol 11. MIT Press. Cambridge, MA, pp. 585–591.

Srinivas, N. & Deb, K. 1995 Multiobjective optimisation using nondominated sorting in genetic algorithms. *Evolut. Comput.* **2** (3), 221–248.

Tikhonov, A. N. 1963 Solution of incorrectly formulated problems and the regularization method. *Dokl. Akad. Nauk SSSR* **151**, 501–504.

Vapnik, V. N. 1995 *The Nature of Statistical Learning Theory*. Springer Verlag, New York.

Vapnik, V. N. & Chervonenkis, A. Ya. 1971 On uniform convergence of relative frequencies of events to their probabilities. *Theor. Prob. Appl.* **17**, 264–280.

Yu, X., Liong, S. & Babovic, V. 2004 EC-SVM approach for real-time hydrologic forecasting. *J. Hydroinf.* **6** (3), 209–223.

Zitzler, E. & Thiele, L. 1998 Multiobjective optimization using evolutionary algorithms - a comparative case study. In *Parallel Problem Solving from Nature* (eds. Eiben V. A. E., Bäck T., Shoenauer M. & Schwefel H. P.), Springer Verlag. Berlin, pp. 292–301.