Comparison of FDA based Time Series Prediction Methods

Tuomas Kärnä and Amaury Lendasse

Helsinki University of Technology - Neural Networks Research Center
P.O. Box 5400 FI-02015 - FINLAND

Abstract. Functional Data Analysis (FDA) provides an important addition to traditional data analysis methods. In FDA a function is fitted to the data and the fitting coefficients are examined instead of the original data. The function fitting, however, is not a straight forward task. The choice of the function space is often crucial and the fitting may involve unknown parameters that need to be determined. This paper presents a comparison of different FDA based methods for time series prediction. The experimented function types are B-splines, Wavelets and Gaussian kernels. In all cases k Nearest Neighbor (k-NN) model is used for the prediction. Furthermore, some input selection methods are experimented to improve the k-NN performance.

1 Introduction

Common time series prediction methods operate in time space and estimate future values directly. A rather different approach is to utilize Functional Data Analysis (FDA) [5] in this task. The idea is to fit some function to the data points and work with the fitting coefficients instead of the original data. In this case, however, one needs to find a suitable set of functions. Smooth functions might be a good guess, if the data is smooth or if the data is very noisy. In the latter case the fitting might result in some noise cancellation. However, it is very difficult to know a priori which basis is suitable for certain data. Furthermore, the fitting usually involves some unknown parameters that need to be determined.

In this paper, we present a comparison of three function types, B-splines, Wavelets and Gaussian kernels in FDA time series prediction task. For all of the functions a k-Nearest Neighbor [3] (k-NN) prediction model is trained. k-NN is suitable for this task because of its robustness and small computational load. However, the drawback of k-NN is that it is sensitive to scaling of the inputs. For this purpose some input selection methods are also examined [8].

The proposed FDA time series method is described in Section 2. Section 3 briefly presents the function types and some of their properties along with a few notions related to FDA applications. The experiments are outlined in Section 4 followed by results and discussion in Section 5. Finally, Section 6 concludes the paper.

2 FDA for Time Series Prediction

When applying FDA to time series prediction, the basic idea is to cast the original prediction problem from time space to some function space. In practice this

means that we work with the function coefficients instead of original data points. The functional representation has some advantages in comparison to traditional prediction schemes, such as dimensionality reduction and the possibility to work with irregularly sampled data.

Consider a set of observations $(x_i, y_i)_{i=1}^n$ in a closed interval $x_i \in [a, b]$. First, the data is divided into input windows I_h and output windows O_h of equal length L. Output is the strictly following window of the corresponding input. The sets are defined as,

$$\begin{aligned} I_h := (x_i^h, y_i^h)_{i=1}^{m_h} &= \{(x_i, y_i) \mid a + (h-1)\delta \quad \leq x_i < a + (h-1)\delta + L\} \\ O_h := (\hat{x}_i^h, \hat{y}_i^h)_{i=1}^{\hat{m}_h} &= \{(x_i, y_i) \mid a + (h-1)\delta + L \leq x_i < a + (h-1)\delta + 2L\}, \end{aligned}$$

where $h = 1, \ldots, N$ and δ stands for the shift between two sequential windows. The number of windows is $N = \lfloor (b - a - 2L)/\delta \rfloor + 1$. The x_i^h values are shifted so that they all belong to an interval $[0, L)$ for all h. Thus, the sample sets are located in the same interval in the time space

If the data is regularly sampled, the sampling time is a natural choice for δ. In this case the sets I_h and O_h intersect, given by $I_{h+L/\delta+1} = O_h$ for $h = 1, \ldots, (b - a - 3l)/\delta$. Regular sampling also implies that there is exactly L data pairs in each set, i.e. $m_h = \hat{m}_h = L$. Furthermore, the x_i coincide and we can write $x_i^h = x_i$ for all h. For simplicity, it is assumed from now on that the data is regularly sampled.

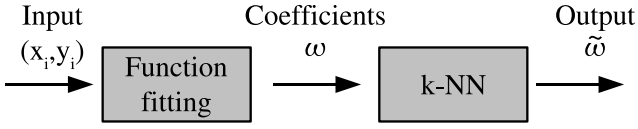

Fig. 1: Prediction method. A function is fitted to the data points and the prediction is done in the function space using k-NN. The output is also a set of coefficients.

2.1 Function Fitting

The outline of the prediction method is presented in Figure 1. An interpolating function is fitted for each window I_h and O_h. To be more specific, it is assumed that there exits some regular function $f \in L^2$ so that $y_i = f(x_i) + s_i$, where s_i stands for observation noise. Working with L^2 in practice, however, is impossible because it is infinite dimensional. For this reason it is necessary to take some finite dimensional subspace $\mathcal{A} \subset L^2$ instead. Knowing the truncated basis φ_i of $\mathcal{A}$ we can approximate f by minimizing the mean square fitting error,

$$\min J(\mathbf{w}) \quad = \quad \sum_{i=1}^{m} (y_i - \hat{f}(x_i))^2 \text{ with } \hat{f}(x) = \sum_{l=1}^{q} w_l \varphi_l(x) \tag{1}$$

where w_l stands for the fitting coefficients and q is the dimension of $\mathcal{A}$. The coefficients w_l define the approximation $\hat{f}$ uniquely and can thus be used as "new" data in further analysis.

When working with the function coefficients w_l, orthogonality of the basis must be taken into consideration. One good property of functional spaces is that the Euclidian operations (such as norm or innerproduct) can also be extended to the functions. Therefore it is natural to require that the innerproduct of the coefficients (i.e. $\langle \mathbf{w}, \mathbf{v} \rangle = \mathbf{w}^T\mathbf{v}$) is equal to the innerproduct of the functions ($\int_{\mathbb{R}} f(x)g(x)dx$). A direct substitution yields,

$$\int_{\mathbb{R}} f(x)g(x)dx = \int_{\mathbb{R}} \sum_{l=1}^{q} w_l\varphi_l(x) \sum_{k=1}^{q} v_k\varphi_k(x)dx = \mathbf{w}^T\Phi\mathbf{v}$$

where $\Phi_{i,j} = \int_{\mathbb{R}} \varphi_i(x)\varphi_j(x)dx$. This implies that if the basis is orthonormal, the matrix Φ becomes an identity matrix and the requirement is automatically met. Otherwise, a linear transformation $\boldsymbol{\omega} = \mathbf{U}\mathbf{w}$ must be applied. Here the matrix $\mathbf{U}$ is a Cholesky decomposition of $\boldsymbol{\Phi} = \mathbf{U}^T\mathbf{U}$ and we get $\boldsymbol{\omega}^T\boldsymbol{\nu} = (\mathbf{U}\mathbf{w})^T\mathbf{U}\mathbf{v} = \mathbf{w}^T\mathbf{U}^T\mathbf{U}\mathbf{v}$. The obtained input and output coefficient sets are denoted as $\mathcal{I}_h = \boldsymbol{\omega}(I_h)$ and $\mathcal{O}_h = \boldsymbol{\omega}(O_h)$, respectively.

2.2 k-NN prediction

k-NN clustering is a widely used for example in pattern recognition [3] and time series prediction [8]. Here, k-NN prediction for functional data is presented. For working with original data directly, see [8].

Our goal is to build a prediction model $P : \mathcal{I}_h \mapsto \mathcal{O}_h$. With k-NN, the estimate for the future is obtained as a mean of the k most similar outputs. I.e. given the training pairs $(\boldsymbol{\omega}_i, \boldsymbol{\nu}_i)$, $\boldsymbol{\omega}_i \in \mathcal{I}_h$, $\boldsymbol{\nu}_i \in \mathcal{O}_h$ and a previously unknown input $\boldsymbol{\omega}$ we get the estimate,

$$\tilde{\boldsymbol{\nu}} \quad = \quad P(\boldsymbol{\omega}) = \frac{1}{k} \sum_{\{\boldsymbol{\nu}_i | \boldsymbol{\omega}_i \in \mathcal{N}_k(\boldsymbol{\omega})\}} \boldsymbol{\nu}_i,$$

where the set $\mathcal{N}_k(\boldsymbol{\omega}) \subset \mathcal{I}_h$ is the k nearest neighbors of $\boldsymbol{\omega}$. Usually the Euclidian metric is used to compute the distance between samples. The number of neighbors, k, is unknown and must be validated separately. Quality of the prediction is measured by evaluating the predicted function at the data locations and taking mean square of the errors.

2.2.1 Input Selection

Due to its simplicity, k-NN is computationally very cheap which makes it suitable for handling large data sets and time consuming parameter optimization tasks. However, it is well known that the performance of k-NN can be greatly decreased if the data is noisy or if there exists a lot of irrelevant information.

Because k-NN is based on pair wise distances, it may be useful to normalize the inputs to zero mean and unit variance. This ensures that each dimension

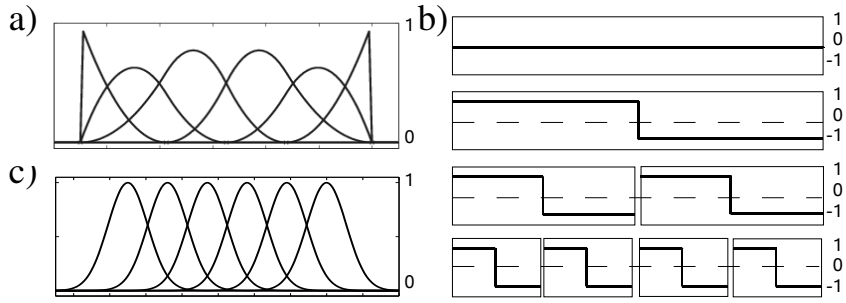

Fig. 2: Basis functions. a) B-Splines b) Haar Wavelets for 8 dimensional data c) Gaussian kernels

will contribute equally to the distances. Normalization is helpful in situations were certain dimensions have large variance and thus dominate the choice of neighbors. However, it does not resolve the problem of irrelevant information.

As another choice, input selection [8] or scaling [9] can be used. In the case of input selection the pair wise distances $|\boldsymbol{\omega} - \boldsymbol{\omega_i}|$ are calculated using only a subset of the data dimensions. In scaling, on the other hand, each dimension is scaled separately before the computation of the distances. The latter can be formulated as $[\omega_1, \omega_2, \ldots, \omega_q] \hookleftarrow [\alpha_1\omega_1, \alpha_2\omega_2, \ldots, \alpha_q\omega_q], \quad \alpha_i \in [0, 1]$. Thus input selection is actually a special case of the scaling where the scaling parameters α_i are restricted to be either 1 or 0.

Input selection increases the computational load significantly. For example, running an exhaustive search, i.e. trying out all the possible combinations of $\alpha_i \in \{0, 1\}$, increases the work load by a factor $2^q - 1$, which quickly leads into unthinkable running times as q grows. For scaling, unfortunately, the situation is not any better, because the optimization problem is very difficult.

3 Functional Spaces

3.1 B-splines

Splines are piecewise polynomials developed for data interpolation. This section presents a short revision of splines in the B-form. B-splines provide a convenient basis function representation to the piecewise polynomials. For more detailed information and discussion on algorithms see [2].

The B-spline basis $B_{i,d}$ is uniquely defined by a order d and a non-decreasing knots sequence t_i with possible multiplicities. The basis can be computed recursively [2],

$$
\begin{aligned}
B_{i,1}(x) &= \begin{cases} 1 & \text{if } t_i \leq x < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \\
B_{i,d}(x) &= \frac{x - t_i}{t_{i+d-1} - t_i} B_{i,d-1}(x) + \frac{t_{i+d} - x}{t_{i+d} - t_{i+1}} B_{i,d-1}(x).
\end{aligned}
$$

The basis has small support, since $b_{i,d}(x) = 0$ for all $x \notin [t_i, t_{i+d}]$. The order d is related to the smoothness constraints of the spline. The larger the d, the smoother the fitting. In fact, if a knot has multiplicity r, i.e. $t_j = t_{j+1} = \ldots = t_{j+r-1}$, then the number of continuous constraints at t_j equals to $d - r$. So, if $r = d$, the spline becomes discontinuous.

It is convenient to set the spline to be discontinuous at the first and last data location and as smooth as possible in between. Thus the first and the last knot has multiplicity d and the rest are simple. In this case, we have a handy relation,

$$number\ of\ knots = q + d$$

where q is the number of basis functions. The location of the remaining knots are set so that the fitting becomes as accurate as possible. There are several methods available, the one used in this paper is presented in [7]. An example of 3rd order B-spline basis is presented in Figure 2.

In FDA the B-splines have some clear advantages; The fitting is guaranteed to be smooth by definition and it is well suited for reducing data dimensionality. However, the two unknown parameters q and d need to be to optimized. Furthermore, the basis is not orthonormal, so the Cholesky decomposition described in Section 2.1 must be applied.

3.2 Wavelets

Wavelet transformation resembles Fourier transform in that both provide a time-frequency description of the data [1]. Wavelets, however, are able to encode spatial information as well, which make them appealing for multiresolutional signal processing. Given some mother wavelet function ψ, the basis can be written as,

$$\psi_{i,j}(x) = 2^{-i/2}\psi(2^{-i}x - j),$$

where i and j are integers that represent variation in frequency and spatial location, respectively. The wavelet basis functions form an orthonormal basis for L^2. An example of the simplest wavelet, Haar (or Daubechies 1) is presented in Figure 2. There are many wavelet families available with different properties, such as Daubechies (in [1] these are called compactly supported wavelets), Biorthogonal and Mayer wavelets.

Usually there is no need to compute the transformations using the basis functions. Instead the transformation can be obtained efficiently using digital sub band filtering [1]. In this case, however, we have to assume that the data has a constant sampling rate. Furthermore, due to the nature of the basis, data length should be a power of 2. Otherwise some data points must be generated during the filtering process, and the transformation actually becomes longer than the original data. Thus discrete wavelet decomposition cannot reduce dimensionality.

Although dimensionality cannot be reduced, the wavelets have some good properties for FDA. First of all, orthonormality of the basis is always desirable, and on the other hand there are no additional parameters involved. Thus only the window length and wavelet type must be tested.

3.3 Gaussian kernels

The Gaussian fitting is the simplest function type presented here. The basis functions are Gaussian kernels,

$$\varphi_i(x) = e^{\frac{\|x-t_i\|^2}{2\sigma_i^2}}, \tag{2}$$

centered at t_i and with width parameter σ_i, $\quad i = 1, \ldots, q$. Thus there are two unknown parameters for each kernel. In total we end up with $2 + 2q$ free parameters, (window length, number of kernels, centers and widths) which is considerably more than with Wavelets or B-splines.

One way to ease the parameter optimization is to fix the locations t_i so that they are equally distributed on the interval of the data. This is justified in the case of regularly sampled data, although it may be far away from the optimal. Furthermore, all the kernels can be share the same width parameter σ. This reduces the number of unknown parameters down to 3 which is feasible for a grid search, for example. The grid search optimization was presented in [10]. An example of such Gaussian functions is presented in Figure 2.

When the locations and widths are known, the fitting weights are obtained by solving the problem (1). The solution is the well-known pseudo inverse $\mathbf{w} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{y}$ [6], where $\mathbf{y} = [y_1, y_2, \ldots, y_m]^T$ are the values to be fitted and $\mathbf{G}_{i,j} = \varphi_j(x_i)$. In practice the matrix $\mathbf{G}^T\mathbf{G}$ tends to become singular on some occasions. In that case it can be replaced with $\mathbf{G}^T\mathbf{G} + \epsilon\mathbf{I}$ using a small ϵ to ensure the existence of the inverse.

3.3.1 Optimizing widths and locations

The locations and widths has a major role in the quality of the fitting, so it would be desirable to relax the rather artificial constraints given above. Optimizing the average fitting error, is a non-supervised and relatively fast way to find good parameter values. The optimization methods, however, require gradient that exists in closed form since all the functions in (1) are analytical.

To derive the gradient, we first define the error functional. Squared fitting error of all the functions $h = 1, \ldots, N$ can be written as,

$$\begin{aligned} E &= \frac{1}{2}\sum_{h=1}^{N}\big(\mathbf{G}\mathbf{w}_h - \mathbf{y}_h\big)^T\big(\mathbf{G}\mathbf{w}_h - \mathbf{y}_h\big) \\ &= \frac{1}{2}\sum_{h=1}^{N}\big(\mathbf{w}_h^T\mathbf{G}^T\mathbf{G}\mathbf{w}_h - 2\mathbf{y}_h^T\mathbf{G}\mathbf{w}_h + \mathbf{y}_h^T\mathbf{y}_h\big) \end{aligned}$$

The columns of $\mathbf{G}$ are denoted as $\mathbf{G}_k = [\varphi_k(x_1), \varphi_k(x_2), \ldots, \varphi_k(x_m)]^T$ and it's derivative with respect to t_k and σ_k,

$$\begin{aligned} \mathbf{G}_k^{(t)} &= \Big[\frac{x_1 - t_k}{\sigma_k^2}\varphi_k(x_1), \ldots, \frac{x_m - t_k}{\sigma_k^2}\varphi_k(x_m)\Big]^T \\ \mathbf{G}_k^{(\sigma)} &= \Big[\frac{(x_1 - t_k)^2}{\sigma_k^3}\varphi_k(x_1), \ldots, \frac{(x_m - t_k)^2}{\sigma_k^3}\varphi_k(x_m)\Big]^T, \end{aligned}$$

respectively. With this notation we obtain,

$$\begin{aligned}\frac{\partial}{\partial t_k}\left(\mathbf{y}_h^T\mathbf{G}\mathbf{w}_h\right) &= \mathbf{y}_h^T\mathbf{G}_k^{(t)}w_{h,k}\\ \frac{\partial}{\partial t_k}\left(\mathbf{w}_h^T\mathbf{G}^T\mathbf{G}\mathbf{w}_h\right) &= 2\mathbf{w}_h^T\mathbf{G}^T\mathbf{G}_k^{(t)}w_{h,k},\end{aligned}$$

which finally yields,

$$\frac{\partial E}{\partial t_k} = \sum_{h=1}^{n}\left(\mathbf{G}\mathbf{w}_h-\mathbf{y}_h\right)^T\mathbf{G}_{\mathbf{k}}^{(\mathbf{t})}w_{h,k}.$$

Following similar steps for $\partial/\partial\sigma_k$ we get,

$$\frac{\partial E}{\partial \sigma_k} = \sum_{h=1}^{n}\left(\mathbf{G}\mathbf{w}_h-\mathbf{y}_h\right)^T\mathbf{G}_{\mathbf{k}}^{(\sigma)}w_{h,k}.$$

When the gradient is known, the locations and the widths are optimized using unconstrained non-linear optimization. Actually, the problem is constrained because σ cannot be negative. But the kernel (2) is an even function with respect to σ so negative values can be accepted as well. In this paper a Broyden-Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton method with line search is used [4]. BFGS method resembles quadratic algorithms, such as Newton method, in the sense that it assumes that the problem is quadratic, but with BFGS there is no need to actually compute the Hessian matrix at any stage.

It should be noted, however, that the optimization of the fitting is not equivalent to running a grid search as described above. With the grid search, the prediction performance is optimized directly, while the goal in here is only a good data fitting. Still, a good fitting is a key for a good prediction, since the prediction error cannot be smaller than the fitting error.

4 Experiments

The different function fittings were experimented with the regularly sampled ESTSP'07 benchmark data. First 465 values were used as a learning set and all the parameters were optimized using Leave-One-Out error (LOO). The remaining 410 data points were used as test set to evaluate the quality of the obtained models. In all the tests only the first 15 predicted values were taken into account.

The tests were run for Wavelet, B-spline and Gaussian fitting. The wavelet families Daubechies {1,2,3,4}, Biorthogonal {1.3,2.2} and discrete Meyer were tested. For B-splines orders 2, 3 and 4 were experimented. The Gaussian kernels were tested with fixed kernel locations and grid-search optimized width (as explained in Section 3.3) but also with the Quasi-Newton optimization. The maximum amount of Quasi-Newton iterations was 100. For a reference, a plain k-NN prediction test was carried out. The neighborhood parameter k ranged from 1 to 15 in all the cases.

The experimented window lengths L varied from 15 to 35. Number of coefficients q ranged from 5 to 29. In the case of fixed Gaussian kernels, the width parameter σ got values from 0.3 to 0.93 with 0.07 step size[1].

For comparison, the effect of input normalization was experimented for all the tests. And finally, input selection was tested by running an exhaustive search. For this purpose the models that gave the best LOO error for each number of coefficients (q) were selected. However, due to a rapid increase in computational load only values $q \leq 11$ were experimented.

5 Results

The results are presented in Table 1. The best setup is 2nd order B-spline with input normalization. Functional approach improves results compared to plain k-NN prediction. B-splines give the best overall results and LOO errors are less than one third compared to plain k-NN. The Gaussian fitting is slightly worse.

Wavelet results, on the other hand, are quite similar to plain k-NN. This may be due to the fact that dimensionality is not reduced. k-NN prediction tends to become more difficult as dimensionality grows.

Normalization is beneficial only for the B-splines. With plain k-NN there is not much difference, while in the case of wavelets it is clearly disadvantageous since LOO error becomes roughly 6 fold.

In the case of Gaussian fitting, the fixed kernels with one width parameter perform better. However, the Quasi-Newton method seem to be more robust since test errors are smaller.

5.1 Input Selection

The exhaustive search results are presented in Table 2. Performance is slightly worse compared to the last results, because number of coefficients were restricted to be less than 12. Due to high dimensionality wavelets were left outside of this test. Based on the previous results, inputs were normalized only for B-splines.

Surprisingly the B-spline results were exactly the same because all the inputs were chosen. This may explain the good performance in the previous test; all the coefficients carry relevant information. There was a slight improvement with the Gaussian fitting. But even in this case almost all inputs were selected; Only 9th input out of 11 was omitted for the fixed kernels and 1st, 9th and 10th for Quasi-Newton.

Based on these results, exhaustive search is not recommended because the benefits are minor compared to the increase in computational load.

[1]The width was normalized so that value 1 stands for the distance between the centers

6 Conclusions

Three function spaces were experimented for 15-step ahead prediction of the ESTSP'07 benchmark data. The best setup was 2nd order B-splines with normalized inputs. Gaussian fitting was quite as good while wavelets performed clearly worse.

The results support the notion that the choice of function space is not trivial in FDA. The fact that normalization was beneficial in some occasions stress the problems related to k-NN; input scaling can have a remarkable effect on the performance.

Functional approach improved performance compared to direct prediction method. Moreover, all the methods presented here can be modified to work with irregularly sampled data, which is out of the scope of the traditional time series prediction methods.

References

[1] I. Daubechies, *CBMS-NSF Regional Conference Series in Applied Mathematics vol.61: Ten Lectures on Wavelets*, Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[2] C. de Boor, *Applied Mathematical Sciences vol.27: A Practical Guide to Splines*, Springer-Verlag, New York, 1978.

[3] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Clarendon, 1995

[4] M. S. Bazaraa, H. D. Sherali, C. M. Shetty, *Nonlinear Programming, Theory and Algorithms*, John Wiley and Sons, New York, 1993.

[5] J. Ramsay and B. Silverman. *Functional Data Analysis*, Springer Series in Statistics, Springer Verlag, 1997.

[6] D.S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, *Complex Systems*, 2:321-355, Complex Systems Publication, 1988

[7] P.W. Gaffney, M.J.D. Powell, Optimal interpolation. In G.A. Watson, editor, *On Numerical Analysis*, Lecture Notes in Mathematics 506, pages 90-99, Springer, Heidelberg, 1976.

[8] A. Sorjamaa, N. Reyhani, A. Lendasse, Input and Structure Selection for k-NN Approximator. In proceedings of IWANN 2005, LNCS 3512, pages 985-991, Springer-Verlag, 2005.

[9] A. Lendasse, F. Corona, J. Hao, N. Reyhani, M. Verleysen, Determination of the Mahalanobis matrix using nonparametric noise estimations. In proceedings of ESANN 2006, pages 227-232, April 26-28, Bruges (Belgium), 2006.

[10] T. Kärnä, F. Rossi, A. Lendasse, LS-SVM functional network for time series prediction. In proceedings of ESANN 2006, pages 473-478, April 26-28, Bruges (Belgium), 2006.

B-splines			Normalized	
Degree	LOO	Test	LOO	Test
2	0.0082	0.2004	* **0.0079**	**0.2004**
3	0.0086	0.2088	**0.0080**	**0.2017**
4	0.0105	* 0.1921	**0.0103**	* **0.1921**
Wavelets			Normalized	
Type	LOO	Test	LOO	Test
Daubechies 1	0.0331	0.2189	0.1815	0.4451
Daubechies 2	0.0414	0.1897	0.2933	0.2537
Daubechies 3	0.0393	0.1948	0.2978	**0.1675**
Daubechies 4	0.0342	0.1948	0.2446	* **0.1609**
Biorthogonal 1.3	0.0325	0.1834	0.1712	0.2323
Biorthogonal 2.2	0.0335	0.1852	0.2680	**0.1851**
Discrete Meyer	0.0328	0.2082	* **0.0313**	0.2665
Gaussian			Normalized	
Type	LOO	Test	LOO	Test
Grid search	0.0110	0.1999	* **0.0093**	0.2029
Quasi-Newton	0.0155	* 0.1820	0.0163	0.1958
Plain k-NN			Normalized	
	LOO	Test	LOO	Test
	0.0328	* 0.2082	* **0.0313**	* **0.2082**

Table 1: Results for 15 step ahead prediction. The values are normed mean square errors. The best values for each function type are marked with an asterisk. The cases were normalization improved the results are in bold face.

B-splines					
Input Selection	Degree	LOO	Test	L	q
Yes	2	0.0102	0.1893	30	11
No	2	0.0102	0.1893	30	11
Yes	3	0.0087	0.1901	34	10
No	3	0.0087	0.1901	34	10
Gaussian					
Input Selection	Type	LOO	Test	L	q
Yes	Quasi-Newton	**0.0145**	0.2142	35	11
No	Quasi-Newton	0.0173	0.2088	35	11
Yes	Grid-Search	**0.0138**	0.2304	30	11
No	Grid-Search	0.0140	0.1936	30	11

Table 2: Input selection results. An exhaustive search was run to best models with $q = 5, \ldots, 11$. Improvements are in bold face. For B-splines all the inputs were selected.