CONDITIONAL PREDICTION OF MARKOV PROCESSES USING NON PARAMETRIC VITERBI ALGORITHM - COMPARISON WITH MLP AND GRNN MODELS

P.F. Marteau, V. Monbet

VALORIA, SABRES
Université de Bretagne Sud
Campus de Tohannic, 56000 Vannes, France
{pierre-francois.marteau, valerie.monbet}@univ-ubs.fr

ABSTRACT

This paper deals with conditional prediction of Markov processes. An algorithm referred as Non Parametric Viterbi (NPV) and based on Hidden Markov Chain theory is proposed and compared to Multi-layer Perceptron predictive models and General Regression Neural Networks. The evaluation is firstly carried out on stationary chaotic time series describing the Lorentz attractor. It is shown that, although neural network models generate very low instantaneous errors, the non parametric Viterbi approach allows to better take into account the dynamical structure of the underlying dynamical process.

KEYWORDS: Nonlinear Time Series, Markov chains, Prediction, Viterbi algorithm, General Regression Neural Network, Multi-layer Perceptron, Non Parametric Estimation.

Introduction

We address the non parametric modeling of multivariate Markovian processes using a continuous state space and discrete time Hidden Markov Model (HMM) for which all necessary densities functions are approximated using samples. The proposed NPV (Non Parametric Viterbi) algorithm is compared to neural networks models: a Multi Layer Perceptron and a General Regression Neural Network.

The proposed NPV approach to prediction is based on HMM modeling. Three classical problems have been solved for discrete HMM that make this kind of model quite useful [12]:

1. the estimation of the probability that the model generates the observation sequence $\{O_t, t \geq 1\}$: the forward-backward algorithm as been developed;

2. the recovery of the most likely hidden state sequence corresponding to these observations: dynamic programming (Viterbi algorithm [4]) is commonly used;

3. the estimation of the parameters of the HMM (transition matrix, prior state distribution, observation conditional distribution) to better account for the observations: the EM-algorithm or Baum-Welch algorithm have been proposed for this task.

In continuous state space situations, the problem of parameter estimation is much more complex [16]. In this paper we address mainly three issues:

1. the local discretization of the observation and state spaces in which the process is handled.

2. the estimation of the parameters of the HMM model, namely the probability density functions involved in the proposed model according to the proposed discretization of observation and state spaces.

3. the estimation of the hidden state sequence $\{S_t, t \geq 1\}$ conditionally to the observation sequence $\{O_t, t \geq 1\}$ and the model M: $P(\{S_t, t \geq 1\}|\{O_t, t \geq 1\}, M)$.

The proposed model approximates the initial multivariate process (IMP) by decomposing it into a Lower Dimensional stationary Markov Chain (LDMC) for which state transitions are hidden. Indeed, state process is indirectly observed through a second stochastic process that generates a multivariate observation from any state of the LDMC. The hidden LDMC state vector coincides with the first coordinates of the state vector of the IMP, while the multivariate observation vector coincides with the last coordinates of the IMP state vector. This decomposition induces a dimension reduction that allows to handle more complex processes, at a computational cost that can be estimated from the data. A Viterbi algorithm referred as Non Parametric Viterbi Algorithm (NPV algorithm) is proposed to extract most likely LDMC state trajectories from sequences of observation vectors. This approach can be used to predict state trajectories when the underlying multivariate dynamical process is partially observed. The method is thus of practical use in case of partially missing or noisy sample data. It can also be used

as a bootstrap technique when the dimension reduction of the state space is necessary for complexity management.

In the first section the Hidden Markov Model for general stationary discrete time continuous state space processes is defined, NPV algorithm is described and shortly discussed. Then, in the following sections, this algorithm is evaluated against neural network models for conditional prediction of a component of a chaotic process.

1. HIDDEN MARKOVIAN MODEL FOR STATIONARY PROCESS MODELIZATION

1.1. Notations, definitions and hypothesis

Let $\{X_t, t \geq 1\}$ be a d-dimensional stochastic process in $(\mathbf{R}^d, \mathcal{B}_d)$ where $\mathcal{B}_d$ is the Borel σ algebra over $\mathbf{R}^d$. We call this process the Initial Multivariate Process (IMP). This process can be decomposed in two dependent lower dimensional processes: $\{S_t, t \geq 1\}$ being a u-dimensional stochastic process in $(\mathbf{R}^u, \mathcal{B}_u)$, and $\{O_t, t \geq 1\}$ being a v-dimensional stochastic process in $(\mathbf{R}^v, \mathcal{B}_v)$, with $u+v = d$, and $\forall t, X_t = (S_t, O_t)$.

We suppose that there exists a positive integer $p < \infty$ such that the stochastic process $\{S_t,\ t > p\}$ with $Y_t = \{S_t, S_{t-1}, ..., S_{t-p+1}\}$ forms a stationary Markov chain on $(\mathbf{R}^{up}, \mathcal{B}_{up})$ with transition kernel $P(y, A)$ where $A \subset \mathbf{R}^u$. We assume that for each $t \in [0, T]$ the Markov kernel $P(y, A)$ admits a stationary distribution π with continuous probability density function f_Y with respects to Lebesgue measure.

Furthermore, we define the stochastic process $\{Z_t,\ t > p\}$ where $Z_t = \{O_t, O_{t-1}, ..., O_{t-q+1}\}$ and $q < \infty$ is a positive integer.

We consider that the $\{O_t, t \geq 1\}$ stochastic process is made available, through measurements or any kind of simulation procedure (for instance by means of LGB re-sampling [10]). The question that we address is to undercover the underlying Markov process, namely, $\{S_t, t \geq 1\}$ conditionally to the observed $\{O_t, t \geq 1\}$ stochastic process in order to approximate at best the initial process (IMP).

1.2. Local discretization of the state spaces

Let VY_t be the neighborhood of Y_t defined as the set of $Y_l \in \{\tilde{Y}_\tau | d(\tilde{Y}_\tau, Y_t) \leq \sigma_T/2\}$ where $\tau \in [0, T]\}$. We note $I[VY_t]$ the set of time index such that for all $l \in I[VY_t], Y_l \in VY_t$. Furthermore, we define the image $(VY_t)^+$ of VY_t by $(VY_t)^+ = \{S_{l+1}, l \in I[VY_t]\} \subset \mathbf{R}^u$. We define VZ_t and $I[VZ_t]$ similarly. Finally we define $VY_t|Z_t$ the set $\{\tilde{Y}_l | \tilde{Z}_l \in VZ_t\}$.

The local discretization of the state space is obtained according to the schema presented in Figure 1: for each observation Z_t, we evaluate the neighborhood VZ_t that defines the set $VY_t|Z_t$. This set defines the local discretization of the state space at step t.

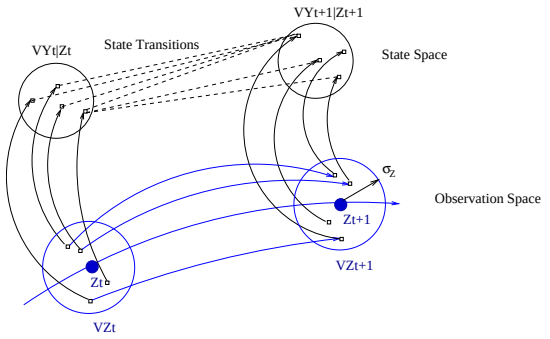

Fig. 1. Discretization of the state space given the observations VZ_t and VZ_{t+1}

1.3. Parameters estimation of the HMM model

The density probability functions f_Y and g_Y of the stationary distributions as well as the transition probability distributions will be approximated from data using appropriate density kernel estimates. In particular, we suppose that some learning data $\{\tilde{X}_t = (\tilde{Y}_t, \tilde{Z}_t), t \geq 1\}$ is available to estimate non parametrically the transition probability density functions for the subprocess $\{Y_t, t \geq 1\}$: the dynamics of $\{Y_t, t \geq 1\}$ is not hidden at this stage.

Let K_u and K_v be probability density functions on $\mathbf{R}^u$ and $\mathbf{R}^v$ respectively, K_{up} and K_{vq} probability density functions on $\mathbf{R}^{up}$ and $\mathbf{R}^{vq}$ respectively. Let $\{h_T, T = 0, 1, 2, ...\}$ and $\{h'_T, T = 0, 1, 2, ...\}$ sequences of positive numbers such that $h_T \to 0$ and $h'_T \to 0$ as $T \to \infty$. We suppose that the density kernels K_u, K_v, K_{up} and K_{vq} satisfy the usual conditions (See for instance Bosq (1998) [**?**]).

Estimation of the transition probability density

The transition probability function of state S_{t+1} given observation Y_t is estimated non parametrically using a kernel estimate. The kernel estimate is based on the product of three terms. Two first terms measure the distance between respectively the state vectors and the observation vectors of the current neighborhood.

$$p(S_{t+1}|Y_t) = \sum_{i \in I[VY_t]} K_u\left(\frac{S_{t+1} - \tilde{S}_{i+1}}{h_T}\right) K_{up}\left(\frac{Y_t - \tilde{Y}_i}{h_T}\right) \quad (1)$$

From this density of probability, we define the probability mass for the discrete random variable J taking its values in $I[VZ_{t+1}] = \{k \in \mathbf{N},\ \tilde{Z}_k \in VZ_{t+1}\}$, with probability mass function given by:

$$P(J = k) = \frac{p(S_{t+1} = \tilde{S}_k|Y_t)}{\sum_{j \in I[VZ_{t+1}]} p(S_{t+1} = \tilde{S}_j|Y_t)}, \forall k \in I[VZ_{t+1}] \quad (2)$$

Estimation of the observation probability density
Observation probability density functions are estimated similarly as transition probability functions just above.

$$p(Z_t|Y_t) = \sum_{i \in I[VY_t]} K_{vq}\left(\frac{Z_t - \tilde{Z}_i}{{h'}_T}\right) K_{up}\left(\frac{Y_t - \tilde{Y}_i}{h_T}\right) \quad (3)$$

Given this density of probability, we define the probability mass for the discrete random variable L tacking its values in $I[VY_t] = \{k \in \mathbf{N},\ \tilde{Y}_k \in VY_t\}$, with probability mass function given by:

$$P(L = k) = \frac{p(Z_t = \tilde{Z}_k|Y_t)}{\sum_{j \in I[VY_t]} p(Z_t = \tilde{Z}_j|Y_t)}, \forall k \in I[VY_t] \quad (4)$$

1.4. Estimation of the most likely state sequence

Given an observed sequence ${Z_1}^T = (Z_1, \cdots, Z_T)$, the inference of the most likely state sequence $\hat{Y_1}^T = (\hat{Y}_1, \cdots, \hat{Y}_T)$ is achieved using algorithms that perform the following maximization:

$$\hat{Y_1}^T = \max_{{Y_1}^T} P({Y_1}^T|{Z_1}^T, M) = \max_{{Y_1}^T} P({Y_1}^T, {Z_1}^T, M)$$

The Viterbi algorithm [4] finds the above maximum with a relatively efficient recursive solution: its computational cost is proportional to the number of non-zero transitions probabilities multiplied by the sequence length. First $\delta(i,t)$ is defined as:

$$\delta(i,t) = \max_{{Y_1}^{t-1}} P({Z_1}^t, {Y_1}^{t-1}, Y_t = \tilde{Y}_i, M)$$

which can be computed recursively as follows, using the usual Markov conditional independence assumptions:

$$\begin{aligned} \delta(i,t) &= P(Z_t|Y_t = \tilde{Y}_i) \\ &\times \max_j P(Y_t = \tilde{Y}_i|Y_{t-1} = \tilde{Y}_j, M) \\ &\times \delta(j, t-1) \end{aligned} \quad (5)$$

with the initialization:

$$\delta(i,1) = P(Z_1|Y_1 = \tilde{Y}_i)P(\tilde{Y}_i)$$

If we define:

$$\mathcal{I}(i,t) = arg\max_j P(Y_t = \tilde{Y}_i|Y_{t-1} = \tilde{Y}_j, M)\delta(j, t-1)$$

then we obtain the optimal state sequence $\hat{Y_1}^T$ using the following backward recursion:

$$\begin{aligned} \hat{j}_T &= arg\max_i \delta(i,T), \\ \hat{j}_{t-1} &= \mathcal{I}(\hat{j}_t, t) \text{ and } \forall t, \hat{Y}_t = Y_{\hat{j}_t} \end{aligned} \quad (6)$$

1.5. Non Parametric Viterbi Algorithm

The proposed Non Parametric Viterbi algorithm can be described as follows.
Initialization step -

- Select the initial observation Z_0, the bandwidth parameters h_T and h'_T, the width σ_Y (respectively σ_Z) of the neighborhood of a given state (respectively of a given observation).
- From Z_0 (at $t = 0$) determine the set of initial states $VY_0|Z_0$ as specified in Figure 1 at $t = 0$.
- Initialize prior probabilities according to the mass function defined in equation (4).

Step t -

- Discretize the state space using $VY_t|Z_t$ as specified in Figure 1 at time t (we suppose that the state space is discretized at step $t-1$, using $VY_{t-1}|Z_{t-1}$).
- Evaluate observation probabilities at step t according to the mass function defined in equation (4).
- Evaluate state transition probabilities from step $t-1$ to step t according to the mass function defined in equation (2).
- Compute $\delta(i,t)$ according to equation (6) for $i \in I[VY_t|Z_t]$

Stop condition $Step\ T$ -

- At $t = T$ compute the most likely state sequence according to equation (6).

In next section, performance of NPV algorithm is compared to performance of neural network models in the context of conditional prediction of Lorentz oscillator.

2. EVALUATION

2.1. General Regression Neural Network

GRNN or "General Regression Neural Networks" have been proposed by Donald Specht [15]. They are relevant to Nadaraya-Watson kernel regression method [**?**], or Parzen window methods [14]. GRNN is a normalized Radial Basis Function (RBF) network for which a hidden unit is centered at every training sample. The RBF units of a GRNN architecture are usually characterized by Gaussian kernels. The hidden-to-output weights are identified to the target values, so that the output is a weighted average of the target values of the training samples close to the given input case. The only parameters of the networks are the widths of the kernels associated to the RBF units. These widths (often a single width

is used) are called "smoothing parameters" or "bandwidths" [18] [17]. They are usually chosen by cross-validation or by ad-hoc methods not well-described. GRNN is a universal approximator for smooth functions, so it should be able to solve any smooth function-approximation problem provided enough data is given. The main drawback of GRNN is that, like kernel methods in general, it suffers badly from the curse of dimensionality. For the following evaluations, the bandwidth of the RBF units is chosen as the size of the neighborhood of the considered input case containing at least 5 neighbors

2.2. Multi-layer perceptron networks

The MLP model [13] [2] is a very general feed-forward neural network. It has often been found to provide compact representations of nonlinear mappings in real-world problems. The MLP network is composed of input and output layers separated with one or more hidden layer(s). Each layer is composed with neurons characterized with a nonlinear activation function (usually hyperbolic tangent or logistic sigmoid functions). It has been shown that one layer of suitable nonlinear neurons followed by a linear layer can approximate any nonlinear function with arbitrary accuracy, given enough nonlinear neurons are provided [5]. MLP networks are thus universal function approximators.

2.3. Conditional prediction of Lorentz chaotic systems

The Lorentz attractor [6] is chosen as example for comparison of several predictive models. This oscillator model has been used the first by Lorentz in meteorology to modelize turbulence consisting of rollers of parallel convection that appear in a horizontal layer of fluid heated in its inferior part. Simplified equations are given by:

$$\begin{cases} \dot{x} = N_{Pr}(y - x) \\ \dot{y} = -xz + rx - y \\ \dot{z} = xy - bz \end{cases} \tag{7}$$

where N_{Pr} is the number of Prandtl and parameters r and b depends on Rayleigh number. This oscillator exhibits a chaotic behavior characterized by an attractor with two lobes as shown in figure 2. The chaotic series has been computed with $N_{Pr} = 16$, $b = 4$, $r = 45.92$ integrating equations (7) by a fourth-order Runge-Kutta method with time step $\Delta_t = 0.02$. The dimension of the attractor is 2.06.

NPV algorithm is compared with several classical algorithms:

1. Nearest Neighbor (NN): using the same structure as NPV algorithm it is straightforward to return the state corresponding to the nearest neighbor of the current observation.

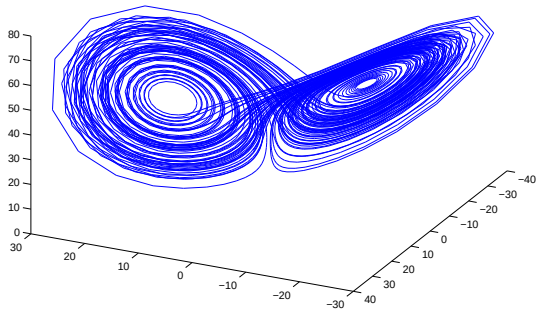

Fig. 2. The 3D Lorentz attractor

2. Multi Layer Perceptron (MLP) with 2 layers and 20 hidden units on each layer. The activation function is the hyperbolic tangent defined by $f(x) = \frac{1-\exp(-x)}{1+\exp(-x)}$. And the model is learned by maximizing the likelihood of the residuals that are supposed to be independent and Gaussian. The network is trained by back-propagation.

3. General regression Neural Network (GRNN)

Each models are trained with the same sample of variable length (from 10000 points to 97000 points) and validate on 3000 points samples. For each test, the mean absolute error between x-component of the validation set and the corresponding prediction is computed. Results are given in table 2.3. Results of table 2.3 show that NPV algorithm better predict x-component of the Lorentz attractor as Nearest Neighbor and General Regression Neural Network models. In fact, Viterbi algorithm take into account the whole trajectory of the process for the prediction and that permits to better restore the dynamical structure of the system. One observes that NPV needs long training samples to attain MLP error. Some numerical tests have demonstrated that NPV error still decreases when the sample size increases. MLP produces a low mean absolute error. This error is high correlated to the complexity of the MLP architecture. We remark also that MLP errors does not decrease anymore

	1e4 pts	5e4 pts	8e4 pts
NPV	.183	.069	.054
NN	.289	.112	.084
GRNN	.693	.218	.0812
MLP	.062	.054	.055

Table 1. Mean of absolute error between prediction and reference according to the length of the training sample

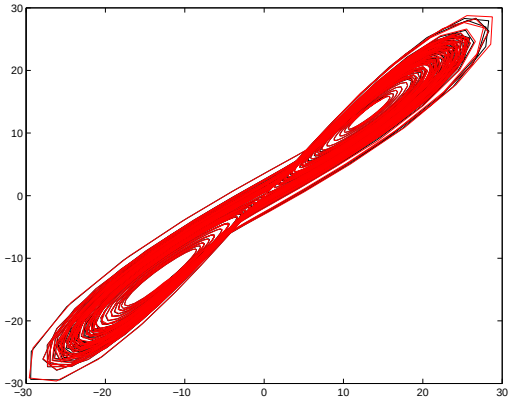

Fig. 3. black curve: reference reconstructed trajectory, red curve: NPV prediction

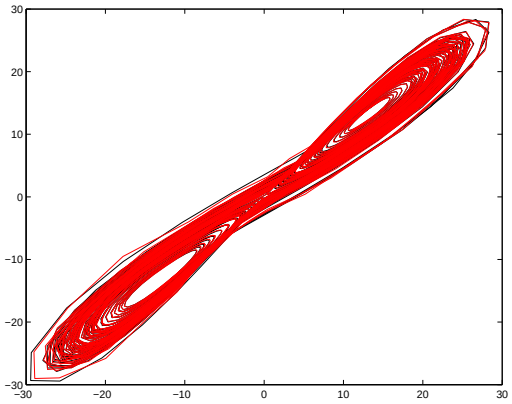

Fig. 4. black curve: reference reconstructed trajectory, red curve: MLP prediction

when the size of the training sample increase after a sample size level between 10000 and 50000. It is due to the parametric structure of MLP. Indeed, parametric models need generally smaller training sample to learn the model than non parametric models so that the model is all ready optimally learned with 50000 points samples. The residual error of MLP reported in table 2.3 is likely due to a bias. More complex structures of networks could probably improve this result.

Figures 3 and 4 show the reconstructed attractor in a 2-D space from the predicted x-component of Lorentz attractor. One can observe that NPV prediction is superimposed on the reference trajectory most of time while MLP is often lightly shift. Figure 5 plots the absolute residuals. One can remark that NPV generates more high errors than MLP but for NPV the small errors are smaller in mean than for MLP.

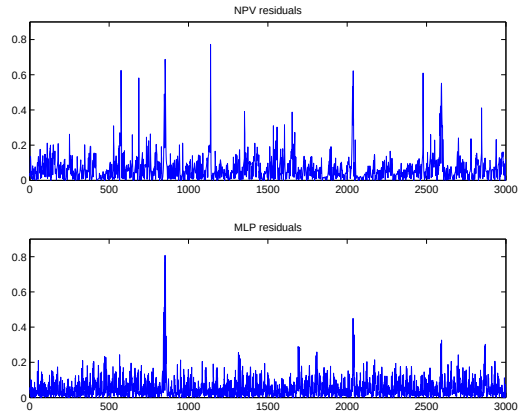

Fig. 5.

Concluding remarks

In this paper, a non parametric version of Viterbi algorithm is proposed to modelize and predict part of a multivariate stationary discrete time continuous space Markov processes conditionally to the remaining part. The performances of this algorithm are compared to those obtained with classical neural networks models (MLP and GRNN) for the prediction of a component of a chaotic system.

It is shown that, although neural networks as for instance MLP allows to achieve a good prediction for the Lorentz attractor, NPV algorithm shows the same order of error when the training sample set is large enough. Furthermore, Viterbi algorithm take into account the whole trajectory for the prediction and seems to better restore the dynamical structure of the process than other algorithms. But it must be underline here that more complex MLP structure would probably improve the results.

NPV algorithm runs quickly and is quite easy to use for various types of multivariate data. There is essentially one critical parameters in the algorithm: the bandwidth parameter used in the non parametric estimations of the probability density functions. This parameter can be regulated empirically. A version of NPV has been developed to predict cyclostationary processes that as meteo time series for instance [7].

3. REFERENCES

[1] Athreya, K.B., Atuncar, G.S., 1998. Kernel estimation for real-valued Markov chains. Sankhya: The Indian J. Stat. 60, Series A, Pt.1, 1-17.

[2] C. M. Bishop, Neural Networks for Pattern Recognition. Clarendon Press, 1995.

[3] Bosq D., 1998. Non parametric statistics for Stochastic Processes. Springer, New York.

[4] Forney, G.D.Jr., 1973.The Viterbi algorithm Proc. of the IEEE Vol. 61, No. 3, pp. 268-278.

[5] Hornik K., Stinchcombe M. and White H. , “Multilayer feedforward networks are universal approximators,” Neural Networks, vol. 2, no. 5, pp. 359-366, 1989.

[6] Lorenz, E. N. ”Deterministic Nonperiodic Flow.” J. Atmos. Sci. 20, 130-141, 1963.

[7] Marteau P.F., Monbet V., (2004). Non Parametric Modelling of Cyclo-Stationary Markovian Processes, Part II: prediction and dimension reduction. Proc. ISOPE Conf. 2004

[8] Meyn, S.P., Tweedie, R.L., 1993. Markov Chains and Stochastic stability. Springer, London.

[9] Monbet V., Marteau P.F., 2001. Continuous Space Discrete Time Markov Models for Multivariate Sea State Parameter Processes. Proc. ISOPE Conf. 2001.

[10] Monbet V., Marteau P.F., 2004. Non Parametric Modelling of Cyclo-Stationary Markovian Processes Part I: Simulation of multivariate sea state processes, Proc. ISOPE 2004 Conf.

[11] Nadaraya, E.A. (1964) ”On estimating regression”, Theory Probab. Applic. 10, 186-90.

[12] Rabiner 1989. A tutorial on hidden markov models and selected applications in speech recognition. In Proceedings of the IEEE, 1989.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error backpropagation,” in Parallel distributed processing (D. E. Rumelhart and J. L. McClelland, eds.), vol. 1, pp. 318-362, The MIT Press, 1986.

[14] Schioler, H. and Hartmann, U. (1992) ”Mapping Neural Network Derived from the Parzen Window Estimator”, Neural Networks, 5, 903-909.

[15] Specht, D.F. (1991) ”A Generalized Regression Neural Network”, IEEE Transactions on Neural Networks, 2, Nov. 1991, 568-576.

[16] Thrun S., Langford J.C., Fox D. , 1999. Monte Carlo Hidden Markov Models: Learning Non-Parametric Models of Partially Observable Stochastic Processes Proc. 16th International Conf. on Machine Learning. 1999.

[17] Wand, M.P., and Jones, M.C. (1995), Kernel Smoothing, London: Chapman and Hall.

[18] Watson, G.S. (1964) ”Smooth regression analysis”, Sankhy¯a, Series A, 26, 359-72.