MEMORY-BASED PREDICTION IN CONTROL AND OPTIMISATION [1]

Rudolf Kulhavý and Petya Ivanova

Honeywell Technology Centre Europe
Institute of Information Theory and Automation, AS CR
Pod vodarenskou vezi 4, Prague, Czech Republic
Tel: +420 2 6605 2313, Fax: +420 22 688 4903
kulhavy@htc.honeywell.cz,ivanova@htc.honeywell.cz

Abstract: The recent progress in database and computer technology makes it possible to use for control and optimisation of technological processes the whole process history. Traditionally, prediction uses either global models covering the total process behaviour or local models fitting just the recent process behaviour. The concept of memory-based prediction combines the advantages of global and local modelling. First, data 'similar' to the predicted situation are retrieved from the process database, usually as a result of a SQL query. Second, the target value is predicted by applying a proper smoothing methodology to the retrieved data. Being more flexible and faster than a global model and more precise and robust than a local (recent-data) model, memory-based approach offers an attractive alternative to the current paradigms. The paper outlines the traditional view of local regression, shows its Bayesian extension and discusses practical choices affecting data retrieval and smoothing.

Keywords: Prediction, regression, local structures, databases, SQL.

1. INTRODUCTION

Modern algorithms for control and optimisation of technological processes are largely model-based. The model serves as a vehicle for prediction of the process response to a particular combination of input signals affecting the process behaviour. The quality of prediction determines the limit performance of control; control cannot be better than prediction is. This explains the steady interest of control community in both classical and new modelling paradigms.

It is generally agreed that the future development of control will be to a large extent determined by the progress in computer technology. For years the control paradigms have been tied up by the dictum to implement the algorithms in a strictly limited working memory and an affordable number of operations. With modern storage media available at lower prices and computer performance growing at steady pace, the whole process history can be stored now in a single repository and used in principle for improved prediction. This option calls, however, for a proper modelling paradigm, complementing in a sense the traditional global and local approaches to modelling.

The *global* approach calls for estimation of comprehensive models, such as neural networks or other non-linear parametric models (Sjöberg *et al.* 1995). The advantage of global modelling is that once a proper model is estimated, prediction of future data can be done fairly quickly using a single model covering all situations of interest. The drawback is that time necessary for estimation of unknown model parameters can be very

[1] Partially supported by the Grant Agency of Academy of Sciences of the Czech Republic under grant No. A2075603.

long for huge data sets. Moreover, global models are increasingly sensitive to changes in the process data behaviour. The development of proper models may also be time-consuming, especially if one insists on preserving at least a partial physical interpretation of the resulting model.

The *local* approach is justified by the observation that for the purpose of control and optimisation it is often sufficient to fit the process behaviour in a neighbourhood of the current working point only. Traditionally, local modelling has been identified with *recent-data* fitting. Linear regression (Box and Jenkins 1970) and Kalman filtering (Kalman 1960) have become *de facto* standards. With simple recursive estimation algorithms available, the linear, local-in-time models are a solution of choice. But, the simplicity does not come for free. Adaptation of local-in-time models is driven solely by the prediction error. Whenever one of the previously visited working points is approached, learning starts from scratch — repeating the same mistakes again and again. Clearly, by eliminating the time delay necessary for re-tuning the local model, the quality of prediction could be significantly improved as well as subsequent control and optimisation.

The extension of recent-data fitting to *relevant-data* fitting raises technological and methodological challenges. To retrieve historical data relevant to the predicted situation, access to all past data is necessary. A data repository is thus a technological must. To smooth the retrieved data, a proper modelling and estimation framework is required, capable of coping with massive computational burden. Generally, there is no simple connection between the models constructed at the neighbouring time instants that could be used to avoid duplicate computations. This is the price paid for the possibility to compute prediction *on demand* or *just in time*.

The approach is not new. It has always been implicitly present in the non-parametric, kernel-based techniques. As the kernels assign practically important weights to the points within a local neighbourhood of the central point, only the data points close to the predicted situation determine the kernel-based prediction. Similar schemes have been proposed also in machine learning and system identification (cf. Section 3).

The theory of non-parametric smoothing is well-developed and largely ready-to-use. Yet, when dealing with complex real-life problems, both theoretical and practical issues emerge that need to be resolved. These are the subject of the present paper. Section 2 illustrates on the problem of load prediction the need for hybrid, continuous-discrete data models. Section 3 presents a traditional view of kernel-based polynomial regression that applies to the hybrid case. Section 4 outlines Bayesian extension that makes it possible to take prior information into account. Section 5 discusses the practical choices affecting data retrieval and data smoothing. Finally, Section 6 points out benefits and limitations of memory-based prediction.

2. EXAMPLE: LOAD PREDICTION

In the sequel the problem of load prediction is considered as a typical example of problems where memory-based prediction can outperform the current solutions. Prediction of load in transmission and distribution networks is a crucial part of any energy management system. The problem of load prediction is largely independent of the object of transport. Be it electricity, heat, gas, or water, one needs to decide upon the variables z_{ik},$i = 1, \ldots, n$, that affect the load y_k at time k and then estimate the functional dependence

$$y_k = f(z_{1k}, z_{2k}, \ldots, z_{nk}) + e_k \qquad (1)$$

using the previously observed data y_k, z_k, $k = 1, \ldots, N$.

The variables z_i include:
- *time attributes* such as (serial) time, time of day, time of year, day of week, type of day (working/holiday),
- *weather attributes* such as outdoor temperature, wind velocity, wind direction, solar irradiation, air humidity.

The vector $z' = (z_1, z_2, \ldots, z_n)$ can be decomposed into vector of continuous variables $(z^c)' = (z_1^c, z_2^c, \ldots, z_{n_c}^c)$ and vector of discrete (ordinal or symbolic) variables $(z^d)' = (z_1^d, z_2^d, \ldots, z_{n_d}^d)$. Typical examples of the latter include discrete time (time stamp of particular measurements), day of week, holiday (Yes/No). While the continuous variables are defined on real intervals, $z_i^c \in I_i \subset \mathbf{R}$, the discrete variables can be labelled without loss of generality by integers, $z_i^d \in \{1, 2, \ldots, M_i\}$. Note that the vector z can contain time-lagged values of the same quantity so that (1) covers dynamic models as well.

3. LOCALLY WEIGHTED REGRESSION

As any smooth function can be approximated locally by a low order polynomial, the polynomial regression model is a natural choice for a local fit of the mapping $z \mapsto y$. The model can be simplified even more assuming that the response is additive in the predictor effects of particular z_i^c variables (Hastie and Tibshirani 1990)

$$y_k = f_\theta(z_k^c) + e_k$$
$$= \theta_0 + \sum_{i=1}^{n_c} \sum_{j=1}^{m_j} \theta_{ij} (z_{ik}^c)^j + e_k \qquad (2)$$

Here k denotes the index of the data point z, m_j is the order of the polynomial in z_i^c and θ_0, θ_{ij} stand for the unknown parameters. Although the model is linear in parameters, it can fit *locally* even strongly nonlinear mappings $z \mapsto y$.

To estimate the unknown parameters θ_0, θ_{ij} locally-weighted least squares can be used

$$\min_\theta \sum_{k=1}^{N} \frac{W(d(z, z_k))}{\int \sum_{z^d} W\left(d(z, z_k)\right) dz^c} \left(y_k - f_\theta(z_k^c)\right)^2$$

where z is the target point for which the prediction $y(z)$ is to be computed, (z_k, y_k) are the stored data points, $W(\cdot)$ stands for a Gaussian kernel $W(d) = \exp(-d^2)$ and $d(z, z_k)$ denotes a distance function that is assumed to satisfy the Pythagorean relationship

$$d(z, z_k)^2 = \sum_{i=1}^{n} d_i(z_i, z_{ik})^2.$$

A typical distance for continuous variables is the Euclidean one

$$d_i(z_i^c, z_{ik}^c) = \left| \frac{z_i^c - z_{ik}^c}{h_i} \right| \qquad (3)$$

where $h_i, i = 1, \ldots, n_c$ are the *smoothing parameters* of the estimator. The distance function for a discrete variable is defined by the set of all point-to-point distances.

The above scheme has been presented in various contexts, e.g., as
- *locally-weighted smoothing* (Cleveland 1979)
- *non-parametric regression* (Härdle 1990, Hastie and Tibshirani 1990)
- *local learning* (Bottou and Vapnik 1992)
- *memory-based learning* (Schaal and Atkeson 1994)
- *just-in-time estimation* (Cybenko 1996, Stenman 1997)

Here, the 'memory-based' adjective is chosen to stress the fact that any practical implementation of local regression for typical process databases entails the use of a database management system.

4. BAYESIAN LOCAL REGRESSION

With relatively simple model structures applied to a bunch of local data, memory-based prediction becomes crucially dependent on the amount and quality of data available. The occasional 'gaps' in data can be cured to some extent by adaptation of the neighbourhood size (see Section 5). But the only systematic way of ensuring that the memory-based prediction always yields sensible answers (not contradicting common sense) is to incorporate prior knowledge into the memory-based concept. The Bayesian generalisation of local regression is presented below.

4.1 *Model Class*

Consider a system on which two sequences of continuous random variables are measured, $Y^{N+m} = (Y_1, \ldots, Y_{N+m})$, $U^{N+m} = (U_1, \ldots, U_{N+m})$, taking values in subsets $\mathbf{Y}$ and $\mathbf{U}$ of R^{dimY} and R^{dimU}, respectively. U_k is defined as a directly manipulated input to the system at time k. Y_k is the output, i.e., response of the system at time k to the past history of data represented by the sequences Y^{k-1} and U^k. The above sequences form a *sample* of data. A sequence of observed values $y^{N+m} = (y_1, \ldots, y_{N+m})$, $u^{N+m} = (u_1, \ldots, u_{N+m})$ is a *realisation* of the sample Y^{N+m},U^{N+m} or an *observed sample*.

Suppose that the output values Y_k depend on the past data Y_{k-m}^{k-1} and U_{k-m}^k only through a known vector function, *regressor* $Z_k = z(U^k, Y^{k-1})$, taking values in a subset $\mathbf{Z}$ of R^{dimZ}. The assumption implies $s_k(y_k|y^{k-1}, u^k) = s_k(y_k|z_k)$ for $k = m+1, \ldots, N+m$. In addition, it is assumed that the conditional density of Y_k given $Z_k = z_k$ is identical, $s_k(y_k|z_k) = s(y_k|z_k)$, for all k.

Furthermore, let assume that the density $s(y|z)$ comes from a given family $\mathbf{S} = \{s_\theta(y|z) : \theta \in \mathbf{T}\}$ parametrized by the parameter θ taking values in a subset $\mathbf{T}$ of a given dimension. Finally, consider that $s_\theta(y|z) > 0$ for all $(y, z) \in \mathbf{Y} \times \mathbf{Z}$ and all $\theta \in \mathbf{T}$.

4.2 *Bayesian Estimation*

Let the dependence of the input U_k on the past data Y^{k-1},U^{k-1} and the parameter θ be expressed through a conditional density $\gamma_k(u_k|y^{k-1}, u^{k-1}, \theta)$. Consider also that the only information about θ used for computation of the new input is the information contained in the past data. More precisely, assume that at $k = m+1, \ldots, N+m$ (see 'natural conditions of control' in (Peterka 1981))

$$\gamma_k(u_k|y^{k-1}, u^{k-1}, \theta) = \gamma_k(u_k|y^{k-1}, u^{k-1}).$$

Interpreting the unknown parameter θ as a random variable Θ, its uncertainty can be described through the posterior probability density function conditional on the observed sample y^{N+m},u^{N+m}

$$p_N(\theta) \stackrel{\text{def}}{=} p(\theta|y^{N+m}, u^{N+m}).$$

Here the subscript N indicates conditioning on N data points $(y_{m+1}, z_{m+1}), \ldots (y_{N+m}, z_{N+m})$. Given a prior density conditional on available prior information and m initial values y^m,u^m

$$p_0(\theta) \stackrel{\text{def}}{=} p(\theta | y^m, u^m),$$

the posterior density follows by application of Bayes theorem and natural conditions of control

$$p_N(\theta) \propto p_0(\theta) \prod_{k=m+1}^{N+m} s_\theta(y_k | z_k)$$

where $\propto$ stands for equality up to a normalising factor.

4.3 *Estimation via Inaccuracy*

The last formula can be rewritten so as to emphasise the role of the observed data (Kulhavý 1996).

First, a joint *empirical* density of (Y, Z) for a sample y^{N+m},u^{N+m} can be defined as

$$r_N(y, z) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{k=m+1}^{N+m} \delta(y - y_k, z - z_k)$$

where $\delta(y, z)$ is a Dirac function satisfying $\delta(y, z) = 0$ for $y \neq 0$ or $z \neq 0$ and $\iint \delta(y, z)\, dy\, dz = 1$. Next, a (conditional) inaccuracy of the empirical density $r_N(y, z)$ relative to the model density $s_\theta(y|z)$ is introduced as

$$K(r_N : s_\theta) \stackrel{\text{def}}{=} \iint r_N(y, z) \log \frac{1}{s_\theta(y|z)}\, dy\, dz.$$

Combining the definitions of the empirical density and conditional inaccuracy, an alternative expression of the posterior density can be obtained

$$p_N(\theta) \propto p_0(\theta)\ \exp\left(-N\, K(r_N : s_\theta)\right) \qquad (4)$$

which stresses the specific roles of the prior density of the parameter θ, the amount of data N, the empirical (joint) density $r_N(y, z)$ and the theoretical (conditional) density $s_\theta(y|z)$.

4.4 *Prior Information via Inaccuracy*

The expression (4) can be further simplified if the prior density is chosen in the following *conjugate* form

$$p_0(\theta) \propto \exp\left(-\nu_0 K(\rho_0 : \theta)\right) \qquad (5)$$

where $\rho_0(y, z)$ stands for a 'prior' density of (Y, Z) built upon prior information. The scalar ν_0 counts the number of actual or fictitious observations $\rho(y, z)$ is built on. Hence, the density (5) can be regarded as a 'posterior' density given a uniform prior and ν_0 observations with the empirical density $\rho_0(y, z)$. In general, the scalar $\nu_0 \geq 0$ needs not be integer — its practical meaning is to put an appropriate weight on prior knowledge expressed through $\rho_0(y, z)$.

With the conjugate prior density (5), the posterior density (4) takes the form

$$p_N(\theta) \propto \exp\left(-\nu_N\ K(r_N : s_\theta)\right) \qquad (6)$$

where the scalar $\nu_N > 0$ counts the total 'number' of data and ρ_N is a convex combination of the 'prior' density ρ_0 and the empirical density r_N

$$\begin{aligned} \nu_N &= \nu_0 + N, \\ \rho_N(y, z) &= \frac{\nu_0}{\nu_N} \rho_0(y, z) + \frac{N}{\nu_N} r_N(y, z). \end{aligned}$$

4.5 *Locally Weighted Smoothing*

When the kernel weighting introduced in Section 3 is applied to predict the process response for a particular value z of the regressor vector, the statistic takes the following form

$$\begin{aligned} \nu_N &= \nu_0 + \sum_{k=m+1}^{N+m} W\left(d(z, z_k)\right) \\ \rho_N(y, z) &= \frac{\nu_0}{\nu_N} \rho_0(y, z) + \\ &\quad + \nu_N \sum_{k=m+1}^{N+m} W\left(d(z, z_k)\right) \delta(y - y_k, z - z_k) \end{aligned}$$

Note that Bayesian local regression replaces the true empirical distribution of data with a locally-weighted distribution that stresses the data points close to the regressor of interest whereas the occasional lack of data is compensated by the 'prior' distribution ρ_0. More information on specification of prior information via the 'prior' density can be found in (Kulhavý and Tesař 1997). As there are many ways of locally weighting the data, the smoothing needs to be properly optimised. Cross-validation, bias-variance trade-off and minimisation of Mallow's C_p statistic are among the most frequently used tools (Hastie and Tibshirani 1990).

5. PRACTICAL CHOICES

5.1 *Data Retrieval*

As in local regression the prediction is affected only by the data points z_k, y_k close to the target regressor z, in order to compute the prediction $y(z)$ for a particular vector z, it is not necessary

to process all historical data — it is sufficient to retrieve only data within a local neighbourhood of z.

Local Neighbourhood. The Euclidean distance functions define ellipsoidal neighbourhoods

$$\left\{ \tilde{z}^c \,:\, \sum_{i=1}^{n_c} d_i(z_i^c, \tilde{z}_i^c)^2 = \sum_{i=1}^{n_c} \left(\frac{z_i^c - \tilde{z}_i^c}{h_i} \right)^2 < C \right\}.$$

The constant C must be large enough so that the probability assigned to the neighbourhood of z^c is practically equal to 1. But, to retrieve data from such a neighbourhood would mean to compute the distances of all points z_k^c stored in the database from a given z^c. Even for databases of moderate size, the data retrieval would take extremely long time.

An alternative is to approximate the local ellipsoid by a cube - shaped neighbourhood

$$\{ \tilde{z}^c \,:\, z_i^c - 3h_i \le \tilde{z}_i^c \le z_i^c + 3h_i, i = 1, \ldots, n_c \}.$$

Retrieval of discrete variables z_i^d is done simply by specifying the values of interest (e.g., working days or holidays).

SQL Query. To retrieve data from the local cube, we can use standard SQL (Structure Query Language) queries. The fast querying is a crucial prerequisite for efficient prediction. The retrieval time is affected by the performance of both the computer and database management system. Attention should be paid to indexing of the variables z_i within the database. As indexing accelerates querying but at the same time slows down updating of database records, the optimum indexing is usually a compromise. The use of multiple-field index may be a good option.

Nearest Neighbour Method. If the neighbourhood of z is chosen too small, the query does not return enough data points for reliable prediction. On the other hand, if the neighbourhood is chosen too large, the query returns unnecessarily many data points which slows down the computation.

To relate the neighbourhood size to the number of retrieved points, the N-th nearest neighbour method is recommended in theory. Implementation of this method would require, however, time-exhausting computation of Euclidean distances $d(z, z_k)$ for all points in the database. A time-efficient alternative is to use a cube-shaped neighbourhood and tune its size iteratively, i.e., to repeat query-driven retrievals until the number of retrieved data reaches a predefined value with predefined precision. Obviously, there is a lot or room here for 'smart' tricks.

Curse of Dimensionality. The neighbourhood size is affected drastically by the dimension of the vector z; to retrieve the same number of points in more dimensions, the neighbourhood size must be correspondingly bigger. The fact that neighbourhoods with a fixed number of points become 'less local' as the dimension of z increases (so called 'curse of dimensionality' phenomenon) can be addressed in two major ways. First, we can replace the polynomials in (2) by, say, ridge basis functions or radial basis functions. The model becomes more global in scope but at the same time non-linear in the parameters θ_{ij}. The data smoothing step thus consumes considerably more time compared with the weighted least squares. Alternatively, the curse of dimensionality can be moderated by allowing radically different smoothing parameters h_i in different dimensions z_i^c. The variables that are known to have a strong, non-linear predictor effect (such as outdoor temperature in load prediction) are fitted locally, i.e., using relatively small h_i values. The variables that are relatively less important (such as wind velocity) are fitted using larger h_i values.

5.2 *Data Smoothing*

The smoothing of data is affected by:
- the *model structure*, namely the orders m_i of polynomials in z_i^c for $i = 1, \ldots, n_c$,
- the *kernel type*, namely the definition of component distance functions $d_i(z_i, z_{ik})$ for $i = 1, \ldots, n$.

The distance functions (3) for the continuous variables are determined by the smoothing parameters h_i. The choice of $h_i, i = 1, \ldots, n_c$ is equivalent to the choice of a local neighbourhood over which the data behaviour is fitted. When the neighbourhood is determined by the approximate nearest neighbour method as outlined above, the smoothing parameters h_i become a function of the required number N of retrieved data points.

Intuitively, proper m_i, h_i, N should be found as a trade-off between retrieving enough data (to limit the variance of estimation) and keeping the neighbourhood size small enough (to limit the bias of estimation due to the local model fit).

Standard methods for balancing the estimation bias and variance could be used in principle at every prediction step (Stenman 1997). The extra computational burden is, however, prohibitively large in real problems. Exhaustive off-line analysis of data takes typically too much time as well. For instance, off-line computation of one-day-ahead predictions (based on local polynomial fits) every 15 minutes for a period of 18 months takes about a week using Microsoft Access on Pentium II 266 MHz. To tune the smoothing parameters so as to minimise the deviation of predictions from the actual data, tens to hundreds of such iterations would be needed.

Consequently, the hyper-parameters of the memory-based predictor (polynomial orders, smoothing parameters, size of retrieved data set) are currently found by performing a limited study on a selection of representative data. As the choice affects significantly the prediction quality, optimisation of hyper-parameters is a subject of intensive research. In a nutshell, the problem is: "If someone gave you just n opportunities to try different models on the data set, how would you design your experiments?"

6. CONCLUSIONS

There are good reasons that speak for memory-based prediction:
- It computes only what is needed and when it is needed (just-in-time feature).
- It makes use of all relevant data, being a prerequisite for improved prediction.
- It can cope with huge process databases as long as SQL queries are fast enough.
- It handles complex, heterogeneous databases due to separation of data retrieval.
- It is relatively cheap to develop and maintain because a single predictor can serve different tasks.
- It is easy to implement as standard commercial database management systems can be used for data retrieval.
- It copes simply with missing and poor data — by omitting the incomplete or assigning smaller weights to the unreliable data.

The major limitations of memory-based prediction are the following:
- A sufficient amount of representative data must be collected beforehand.
- Each prediction requires a fair amount of computations — at least one query to the database and solving a least-squares problem for a bunch of retrieved data.
- It may fail in high dimensions when retrieval of enough data calls for too large neighbourhoods (to fit the behaviour of data over such neighbourhoods, models non-linear in parameters are generally needed).

The availability of data is usually not a problem. The memory-based concept was born with the advent of huge process databases and the problem is actually what to do with the amount of data.

The computational time may be a problem. To give an idea, memory-based prediction for a 6-dimensional vector z retrieving about 500 records out of a database of 150,000 records takes 0.6-0.8 sec using Microsoft Access on Pentium II 266 MHz. This time needs to be multiplied by the number of predicted points; thus the computation of 15-minute averages on 24 hours ahead takes about a minute. This time, quite acceptable for prediction in distribution networks by the way, is likely to drop in the near future with the further improvement of database and computer performance.

The curse of dimensionality is a problem. The memory-based paradigm addresses directly only the size of the existing repositories of process data. To describe complex, multivariate relationships among data may call for model structures more sophisticated than local polynomial regression is. Repetitive fitting of such models to the retrieved data does not seem viable today although this may change in future.

7. REFERENCES

Bottou, L. and V. Vapnik (1992). Local learning algorithms. *Neural Computation* **4**, 888–900.

Box, G.E.P. and G. M. Jenkins (1970). *Time Series Analysis, Forecasting and Control.* Holden-Day. San Francisco.

Cleveland, W.S. (1979). Robust locally-weighted regression and smoothing scatterplots. *J. Amer. Statist. Assoc.* **74**, 829–836.

Cybenko, G. (1996). *Identification, Adaptation, Learning.* Chap. Just-in-time learning and estimation, pp. 423–434. NATO ASI Series. Springer-Verlag.

Härdle, W. (1990). *Applied Non-parametric Regression.* Cambridge University Press.

Hastie, T.J. and R. J. Tibshirani (1990). *Generalized Additive Models.* Chapman & Hall. London.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *J. Basic Engin.* **82**, 35–45.

Kulhavý, R. (1996). *Recursive Nonlinear Estimation: A Geometric Approach.* Springer. London.

Kulhavý, R. and L. Tesař (1997). On dual expression of prior information in bayesian parameter estimation. In: *Proceedings of the 11th IFAC Symposium on System Identification.* Vol. 2. Fukuoka, Japan. pp. 451–456.

Peterka, V. (1981). *Trends and Progress in System Identification.* Chap. Bayesian approach to system identification, pp. 239–304. Number 8. Pergamon Press. Elmsford, N.Y.

Schaal, S. and C. Atkeson (1994). Robot juggling: an implementation of memory-based learning. *Control Systems Magazine* **14**, 57–71.

Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Deylon, P.-Y. Glorennec, H. Hjalmarsson and A. Juditsky (1995). Nonlinear black-box modeling in system identification: a unified overview. *Automatica* **31**, 1691–1724.

Stenman, A. (1997). Just-in-Time Models with Applications to Dynamical Systems. PhD thesis. Linkoping University.