Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Adaptive Process Control Using Controlled Finite Markov Chains Based on Multiple Models

Enso Ikonen * **Urpo Kortela** *

* *Systems Engineering Laboratory, FIN-90014 University of Oulu, POB 4300 (fax +358-8 5532439; e-mail: enso.ikonen@oulu.fi)*

Abstract: Controlled finite Markov chain (CFMC) approach can deal with a large variety of signals and systems with multivariable, non-linear and stochastic nature. In this paper, adaptive control based on multiple models is considered. For a set of candidate plant models, CFMC models (and controllers) are constructed off-line. The state transitions predicted by the CFMC models are compared with frequentist information obtained from on-line data. The best model (and controller) is chosen based on the Kullback–Leibler distance. This approach to adaptive control emphasizes the use of physical models as the basis of reliable plant identification. Three series of simulations are conducted: to examine the performance of the developed Matlab-tools; to illustrate the approach in the control of a non-linear non-minimum phase van der Vusse CSTR plant; and to examine the suggested model selection method for the adaptive control.

Keywords: dynamic programming; Markov decision processes; nonlinear control; optimal control; physical models; probabilistic models.

1. INTRODUCTION

Development of physical plant models typically results in models that do not easily lend themselves for process control design. This can be due to nonlinearities, complexity of the model structure, uncertainties or vagueness in model parameters, etc. Therefore, various grey/black-box plant approximations are commonly used in control design. For example, linear models have turned out to be most useful in industrial practice. These techniques are commonly extended by considering local linear approximations (cf. gain scheduling, indirect adaptive control, piecewise (multi)linear multimodel systems, Wiener and Hammerstein systems, etc.), where linear descriptions vary with state and/or time.

This paper focuses on an approach that can cope with a large class of nonlinear systems: the finite Markov chains (Puterman, 1994; Häggström, 2002; Poznyak et al., 2002; Hsu, 1987; Lunze et al., 2001; Kaelbling et al., 1996; Bertsekas, 2007). The basic idea is simple. The system state space is quantized (discretized, partitioned, granulated) into a finite set of states (cells), and the evolution of system state in time is mapped in a probabilistic (frequentist) manner, by specifying the transition probabilities (counting the observed transitions) from domain cells to image cells. With controlled Markov chains (CFMC), the mappings from each domain cell–action pair are constructed. It is straightforward to construct such a model by simulation of a physical model, for example. The finite Markov chain modelling techniques provide uncomplicated means for learning from data and handling of system uncertainties.

Once equipped with such a model, a control policy can be obtained by minimizing a cost function defined in a future horizon, based on a specification of immediate costs for each cell–action pair. Immediate costs allow versatile means for characterising the desired control behaviour. Dynamic programming, studied in the field of Markov decision processes (MDP), offers a way to solve various types of expected costs in an optimal control framework.

The focus in this paper is in applications in the field of process engineering. These are applications characterized by: availability of rough process models, slow sampling rates, significant nonlinearities (that are typically either smooth or appear as few discontinuities), expensive experimentation (large-scale systems running in production), and substantial on-site tuning (due to uniqueness of products and local operating conditions). Clearly, this type of requirements differ from those encountered, e.g., in the field of economics (lack of reliable models), robotics (very precise models may be available), consumer electronics (mass production of low cost products), or telecommunication (extensive use of test signals, fast sampling). As pointed out, e.g., by Lee and Lee (2004), applications of MDP in process control have been few. Instead, the model predictive control paradigm is very popular in the process control community. Whereas not-so-many years ago the computations associated with finite Markov chains were prohibitive, the computing power available today using cheap office-pc's encourages the re-exploration of these techniques.

The contribution of the paper is manifold: First, we propose (up to our knowledge) a new adaptive control technique based on a combination of multiple models and CFMC design. Second, new simulation results illustrate and characterize the feasible domain of applications for the approach. Finally, we wish to promote discussion

on techniques and applications of CFMC-based process control. From process control philosophy point of view, a significant point in the considered approach is that it supports adaptation based on a physico-chemical plant description. In this way, many of the problems encountered in structure selection and parameter estimation of black-box models are avoided. Instead, physical arguments and *a priori* information can be used to back up the control design.

This paper is organized as follows: The process models, control design and closed-loop analysis based on CFMC are briefly considered in the next section, while section 3 outlines their implementation in the developed Matlab-toolbox. The adaptive CFMC is presented in section 4; some theoretical issues concerning the Kullback–Leibler distance are presented in the Appendix. Section 5 gives numerical examples. Discussion and future research topics end the paper.

2. CONTROLLED FINITE MARKOV CHAINS

In this section an introduction to using controlled finite Markov chains is given. The section is brief, due to obvious space restrictions, and mainly intended for introduction of the notation used. For the many issues related to modelling, identification (learning), prediction, state-estimation, control design and analysis, see the references given earlier.

Let the process under study be described by the following discrete-time dynamic system and measurement equations

$$\mathbf{x}(k) = f(\mathbf{x}(k-1), \mathbf{u}(k-1), \mathbf{w}(k-1)) \quad (1)$$

$$\mathbf{y}(k) = h(\mathbf{x}(k), \mathbf{v}(k)) \quad (2)$$

where $f : \Re^{n_\mathrm{x}} \times \Re^{n_\mathrm{u}} \times \Re^{n_\mathrm{w}} \to \Re^{n_\mathrm{x}}$ and $h : \Re^{n_\mathrm{x}} \times \Re^{n_\mathrm{v}} \to \Re^{n_\mathrm{y}}$ are nonlinear functions, $w_k \in \Re^{n_\mathrm{w}}$ and $v_k \in \Re^{n_\mathrm{v}}$ are i.i.d. white noise with pdf's p_w and p_v. The initial condition is known via $p_\mathrm{X}(0)$. (1) describes a Markov process.

Let the state space be discretized into a finite number of sets called (state) cells, indexed by $s \in \mathcal{S} = \{1, 2, ..., S\}$. The index s is determined from

$$s = \arg\min_{s \in \mathcal{S}} \left\| \mathbf{x} - \mathbf{x}_s^{\mathrm{ref}} \right\| \quad (3)$$

where $\mathbf{x}_s^{\mathrm{ref}}$ are called reference points; a sink cell may be defined to cover the space outside of the region of interest. Similarly, let the control action and measurement spaces be partitioned into cells indexed by $a \in \mathcal{A} = \{1, 2, ..., A\}$ and $m \in \mathcal{M} = \{1, 2, ..., M\}$, respectively, and determined using reference points $\mathbf{u}_a^{\mathrm{ref}}$ and $\mathbf{y}_m^{\mathrm{ref}}$. The discretization results in $\mathcal{X} = \cup_{s=1}^{S} \mathcal{X}_s$, $\mathcal{U} = \cup_{a=1}^{A} \mathcal{U}_a$ and $\mathcal{Y} = \cup_{m=1}^{M} \mathcal{Y}_m$. The number and distribution of the reference points determines the resolution of the CFMC model.

The evolution of the system can now be approximated as a finite state controlled Markov chain over the cell space (however, see Lunze (1998)). Let the state pdf be approximated as a $S \times 1$ cell probability vector $\mathbf{p}_\mathrm{X}(k) = [p_{\mathrm{X,s}}(k)]$ where $p_{\mathrm{X,s}}(k)$ is the cell probability mass. The evolution of cell probability vectors is described by a Markov chain represented by a set of linear equations

$$\mathbf{p}_\mathrm{X}(k+1) = \mathbf{P}^{a(k)} \mathbf{p}_\mathrm{X}(k) \quad (4)$$

where $\mathbf{P}^a$ is the $S \times S$ transition probability matrix under action a, $\mathbf{P}^a = \left[p^a_{s',s}\right]$

$$p^a_{s',s} = \int_{\mathcal{X}_s} p(\mathbf{x}(k+1) \in \mathcal{X}_{s'} | \mathbf{x}(k) \in \mathcal{X}_s, \mathbf{u}(k) \in \mathcal{U}_a)\, d\mathbf{x}. \quad (5)$$

A CFMC model of (1) will consist of A transition probability matrices.

In general, the state may not be measurable, in which case a state estimator is required. The likelihood of obtaining a measurement cell m, when the system state cell is s, is given by the likelihood matrix $\mathbf{L}$, $\mathbf{L} = [l_{m,s}]$

$$l_{m,s} = \int_{\mathcal{Y}_m} p(\mathbf{y} \in \mathcal{Y}_m | \mathbf{x} \in \mathcal{X}_s)\, d\mathbf{y} \quad (6)$$

Let a row in the likelihood matrix be denoted as a likelihood vector $\mathbf{l}_m$. Given the current likelihood vector and the previous posterior probability vector $\mathbf{p}_\mathrm{X}(k-1)$, a Bayesian estimate of the cell probability can be constructed,

$$\mathbf{p}_\mathrm{X}(k) \propto \mathbf{l}_m \otimes \mathbf{P}^{a(k)} \mathbf{p}_\mathrm{X}(k-1), \quad (7)$$

where $\otimes$ is the Haddamard product (component multiplication).

Using a CFMC model of the plant, an optimal control action for each state can be solved by minimizing a cost function. In optimal control, the control task is to find an appropriate mapping (optimal policy or control table) π from cells s to control actions a, given the immediate costs for each cell–action pair, r_s^a. The infinite-horizon discounted model attempts to minimize the geometrically discounted immediate costs, $J(s) = \sum_{k=0}^{\infty} \gamma^k r_s^{\pi(s)}$, under initial conditions at $k = 0$. The optimal control policy π^* is the one that minimizes J. According to Bellman's principle of optimality, an optimal path has the property that whatever the initial conditions and control variables, the control chosen over the remaining period must be optimal for the remaining problem, with the state resulting from the early decisions taken to be the initial condition. For the infinite-horizon discounted model, we have

$$J^*(s) = \min_a \left[r_s^a + \gamma \sum_{s' \in \mathcal{S}} p^a_{s',s} J^*(s') \right]. \quad (8)$$

Application of the Bellman equation leads to methods of dynamic programming. In value iteration, for example, the optimal value function is determined by a simple iterative algorithm derived directly from the Bellman equation.

A set of powerful tools exists for analysing the behaviour of a CFMC, based on the matrices $\mathbf{P}^a$. Recall that a CFMC description of a closed loop system is simple to obtain by constructing the transition probabilities under policy π, $p^{\mathrm{cl}}_{s',s} = p^{\pi(s)}_{s',s}$. The tools include classification of states into transient, recurrent and absorbing, communicating classes and their basins of attraction, assessment of system stability using the concept of sink cell, expected times of transitions, simulations of state trajectories (realizations and probabilities) etc. In particular, tools for global analysis of the (nonlinear, stochastic, multidimensional) closed-loop system exist, in addition to local and simulation-based verifications of the system behaviour.

3. MATLAB-TOOLBOX

CFMC modelling and control design/analysis tools were implemented on a MATLAB platform (Matlab 6, R12). A set point control design problem set-up was envisioned, assuming that an *a priori* state-space model of the plant (1) is available (a set of ordinary differential equations based on physico–chemical process knowledge, for example), and that a decision on input, state, and output variables has been made. A typical CFMC control design procedure would then involve the following (iterative) steps:

(1) Set model resolution by specifying discretization of plant inputs, states, outputs and output set points; and sampling time.
(2) Set control targets by specifying immediate costs (based on deviation from target, constraints on state and control, stability, etc).
(3) Build a CFMC plant model (by successive evaluations of the *a priori* model, and counting the occurred state transitions) and analyze its behaviour.
(4) Solve a controller based on the CFMC plant model (e.g., via value iteration).
(5) Build a CFMC closed-loop model and analyze its behaviour.

The steps involved in model building and controller design may be time and memory intensive. The following list outlines the essential components affecting computational load, time and memory requirements (S = number of cells, A = number of actions, n_{x} state dimension):

- $S \times n_{\mathrm{x}}$ state reference points and $S \times A$ immediate costs need to be stored. Assuming a grid-partitioning, the number of cells S will be Πn_i, where n_i is the resolution (number of partitions) for the i'th dimension of the state.
- When building a CFMC plant model the model (1) needs to be evaluated for all $S \times A$ discrete state–action pairs. To have a 1%-unit precision, at least $100SA$ evaluations of the original model are required. The $S \times S \times A$ elements $p^a_{s',s}$of the probability transition matrices need to be stored, as well as $S \times A$ counter values.
- In optimal control, the Bellman equation needs to be solved for each set point, where the Q-matrix is of size $S \times A$.

It clear that the memory requirements are far from being feasible. The remedy, however, is in that the probability transition matrices are sparse. This can be used to dramatically reduce the memory requirements. A grid-based partitioning greatly reduces the time required for mapping states to cells; the remaining computation times will depend mainly on the complexity of the plant model.

The toolbox is freely available at `http://cc.oulu.fi/~iko/MGCM.htm`.

4. ADAPTIVE CFMC CONTROL

Following Kárný et al. (2007), adaptive systems use local (in time and data spaces) approximations around behaviour realizations. In other words, one set of system parameters is not sufficient to adequately describe the process over its operating region (Shah and Cluett, 1991). A multiple models approach (Narendra et al., 1995; Filev, 2002) is based on the simple idea of using a bank of a priori defined models. Given appropriate control specifications, a model-based control design technique can then be applied for each model (off-line). The selection of which model/controller to use (on-line) is based on a comparison of on-line observations and corresponding predictions by the models in the bank.

In the case of CFMC models, the model prediction is a sequence of multivariate discrete distributions. A straightforward approach would then be to use on-line data to construct a frequentist distribution, and compare it against the predictions by the various models. In what follows, Kullback–Leibler distance is considered.

4.1 Comparison of distributions

The Kullback–Leibler distance (see, e.g., Kulhavy, 1995) is a measure of difference between two probability distributions

$$D\left(R||S\right) = \sum_{s \in S} R\left(s\right) \log \frac{R\left(s\right)}{S\left(s\right)} \tag{9}$$

where R represents data and S represents a model: $R\left(s\right) = \frac{N(s)}{k}$ where $N\left(s\right)$ counts the number of occurrences s in the data and k is the length of the data sequence; $S\left(s\right) = \Pr\left\{s\right\}$.

In the CFMC context, the transitions from a state cell to another can be counted from a sequence of plant operating data, $\left\{\mathbf{x}\left(i\right), \mathbf{u}\left(i\right)\right\}$, $i = 0, 1, ..., k$, thereby obtaining empirical probability mass functions (pmf's): the data R. This information can be compared with predictions from the different models: the S_θ's, $\theta = 1, 2, ..., \Theta$. For CFMC, an empirical expectation of Kullback–Leibler distance between conditional distributions can be written as (see Appendix)

$$\overline{D}\left(R||S_\theta\right) = \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} R\left(s,a\right) D\left(R^{\mathrm{sa}}||S^{\mathrm{sa}}_\theta\right) \tag{10}$$

where $R^{\mathrm{sa}}\left(s'\right) = R\left(s'|s,a\right)$, $S^{\mathrm{sa}}_\theta\left(s'\right) = S_\theta\left(s'|s,a\right)$; R are the relative frequencies representing the empirical pmf obtained from the sequence of plant operating data, $R\left(s,a\right) = \frac{N(s,a)}{k}$, $R\left(s'|s,a\right) = \frac{N\left(s'|s,a\right)}{k}$; and $S_\theta\left(s'|s,a\right)$ is the corresponding prediction using the model θ.

Note that for CFMC, the Kullback–Leibler distance consists of a sum of Kullback–Leibler distances for the conditional distributions. In what follows, the frequencies $R\left(s,a\right)$ are replaced by equal weights. The heuristic argument is that in this way the importance of wide spatial validity of the estimation is emphasized, as all domain cell–action pairs in the data will have an equal weight regardless of their relative frequencies. The 'best' model is then obtained from

$$\theta^* = \arg\min_\theta \sum_{(s,a)} D\left(R^{\mathrm{sa}}||S^{\mathrm{sa}}_\theta\right) \tag{11}$$

where the summation is over all cell–action pairs found from the observed plant operating data sequence.

4.2 Adaptive control based on multiple models

The adaptive control procedure (with main design parameters) is outlined as follows:

(1) Select Θ possible plant descriptions, $\theta = 1, 2, ..., \Theta$ (selection of models in the bank)
(2) Build CFMC maps for each Θ plant models (discretization).
(3) Set up and solve the optimal control problem for all models in the bank (immediate costs, discounting factor).
(4) Select the model/controller to be applied (length of data, use of prior estimates for θ).

Since stages 1-3 can be conducted off-line, the on-line computations (stage 4) remain feasible.

A number of highly non-trivial questions remains to be solved, such as how to select the Θ plant descriptions, the CFMC discretization and controller tuning parameters, how to decide when to switch to another controller, etc. At this stage, we wish to point out that the approach assumes the existence of prior plant model(s). Consequently, a wealth of information on the plant behaviour must exists already, recall also the characterization of our focus area of applications made in Section 1. Therefore it is very likely that reasonable basic choices can be based on prior physical and experimental knowledge about the plant. Of course, this does not exclude application of the many data-based techniques proposed in the literature dealing with problems of clustering/classification and active learning, for example. Also extensions to multiple model learning control such as based on N fixed and two adaptive models (Narendra et al., 1995) are straightforward, noticing however that practical techniques of adaptation in a large-dimensional cell space are far from trivial. These considerations, however, are outside of the scope of this paper, and we only briefly return to them in the section dedicated to discussions.

5. NUMERICAL EXAMPLES

In this section, three sets of numerical results are presented and analysed. The first series of simulations is used for examination of feasible CFMC model sizes when using an average PC as a computing device. The second example illustrates the use of CFMC in the control of a nonlinear non-minimum phase CSTR. The third series examines adaptive control based on selection of a proper model by comparison of model-based and data-based probability distributions.

5.1 Computational load

A series of simulations was made in order to examine the feasibility of the CFMC approach on an average office PC (Matlab R12, Windows XP, 3GHz Pentium 4 CPU with 1GB RAM). The plant model was a second order linear system $\frac{Y(s)}{U(s)} = \frac{-10s+1}{20s^2+12s+1}$, with a sampling time of 1 we obtain a discrete state-space model $\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k)$, with $n_{\mathrm{x}} = 2$. Table 1 shows the time spent in different parts of the algorithm for various model grid resolutions ($n_1 = n_2 = \sqrt{S}$). In all simulations, the number of control actions (A) and set points (Q) was 10; value iteration with $\gamma = 0.98$ was used for solving the DP problems.

The initialization stages ($\mathrm{m_{ini}}$, $\mathrm{c_{ini}}$) consist mainly of initialization of grid structures and reserving space for the sparse $S \times S$ probability transition matrices, with $6S$ nonzeros each. In the modelling stage, the plant system equation (1) is evaluated one time step ahead for each S cells and for each A control actions. The first evaluation ($\mathrm{m_1}$) uses the reference points as initial domain states ($\mathbf{x} = \mathbf{x}_s^{\mathrm{ref}}$); the subsequent evaluations ($\mathrm{m_2}$, $\mathrm{m_3}$, ...) take slightly longer as a random point within the supporting hypercube of each domain cell ($\mathbf{x} \in \mathcal{X}_s$) needs to be generated. The first solution of the controller ($\mathrm{c_1}$) takes a considerable time due to poor initial conditions for the costs-to-go, while the remaining solutions ($\mathrm{c_2}$, ...) are obtained much faster.

Table 1. Time spent for solving CFMC model and controller construction tasks (in minutes).

S	$\mathrm{m_{ini}}$	$\mathrm{m_1}$	$\mathrm{m_2}$	$\mathrm{c_{ini}}$	$\mathrm{c_1}$	$\mathrm{m_3}$	$\mathrm{c_2}$
100	0.01	0.01	0.00	0.00	0.01	0.00	0.00
2 500	0.00	0.02	0.02	0.00	0.07	0.02	0.03
10 000	0.01	0.12	0.15	0.00	0.31	0.15	0.11
250 000	0.45	24	32	0.09	13	35	3.4
1 000 000	3.8	380	480	–	–	–	–

The observed computation times were quite affordable for coarse resolutions ($S \leq 10000$). For example, for the 100×100 system, the model update (simulation and update of the probability transition matrices using one sample for each cell) and solving the optimal control policies (using value iteration) took less than 10 seconds per task. For denser resolutions, the computation times soon grow very large. For example, for $S = 250000$ the time needed model updates exceeded half an hour for one batch.

For such a simple linear model the time required for computing a one-step ahead solution for (1) is practically negligible. Adding an estimate of model execution times for a particular model, and relating to the A and Q for the particular problem, the simulation results in Table 1 can be used to give an indication of the computational load for other plant models.

It can be concluded that the computing times can become prohibitive already in moderately-sized problems, taking into account that one-step-ahead predictions for complex plant models may take a substantially long time to be solved. However, even overnight computations may be feasible if they can be performed off-line. Notice also that possibilities to parallelize the computation of plant predictions are good. Finally, observe that 250000 states equals a 2-dimensional system of size 500×500, or roughly a $60 \times 60 \times 60$ 3-dimensional system, or a $30 \times 30 \times 30 \times 2 \times 2 \times 2$ 6-dimensional system with three binary variables. All these resolutions (with 10 control actions, and 10 set points) may already be completely feasible for solving a real life problem. An out-of-memory error stopped the experiment with one million states at the control design stage.

5.2 CFMC control of van der Vusse CSTR

The van der Vusse CSTR benchmark (Chen et al., 1995) was used to illustrate the behaviour of the CFMC. This problem involves control of a highly nonlinear chemical reactor that exhibits interesting properties, such as non-monotone static gain and non-minimum phase dynamics.

The van der Vusse reaction is described by the following reaction scheme

$$A \overset{k_1}{\rightarrow} B \overset{k_2}{\rightarrow} C, 2A \overset{k_3}{\rightarrow} D \tag{12}$$

The main reaction is given by the transformation of substance A to the product B. There is also an unwanted consecutive reaction to C and a side reaction to D. The normalized input flow $\dot{V}/V_{\mathrm{R}}$ contains substance A with concentration c_{A0}. Assuming constant temperatures, the dynamics of the reactor are described by

$$\dot{c}_{\mathrm{A}} = \frac{\dot{V}}{V_{\mathrm{R}}}(c_{\mathrm{A0}} - c_{\mathrm{A}}) - k_1 c_{\mathrm{A}} - k_3 c_{\mathrm{A}}^2 \tag{13}$$

$$\dot{c}_{\mathrm{B}} = -\frac{\dot{V}}{V_{\mathrm{R}}} c_{\mathrm{B}} + k_1 c_{\mathrm{A}} - k_2 c_{\mathrm{B}} \tag{14}$$

where c_{A} and c_{B} are the concentrations of substances A and B. The reaction velocities are assumed to depend on the temperature via the Arrhenius law, the parameters used in the simulation were $k_1 = k_2 = 1.2870 \times 10^{12}, k3 = 0.9 \times 10^{10}$ (reactor temperature 115 °C); $c_{\mathrm{A0}} = 5.1$; $V_{\mathrm{R}} = 0.01$.

The CFMC model was constructed using the following discretization: states $c_{\mathrm{A}} = \{1.00, 1.05, ..., 3.00\}, c_{\mathrm{B}} = \{0.5, 0.55, ..., 1.5\}$ resulting in $S = 862$ states (including a sink cell); the control action $\dot{V}/V_{\mathrm{R}} = \{2, 6, 10, 11, 12, 12.5, 13.0,... ,15.0, 15.5, 16, 17, 18, 20, 26\}$ ($A = 17$). 100 evaluations per each cell-action pair were generated to build the probability transition matrices.

In control design, five set points for c_{B} were considered, $c_{\mathrm{B}}^{\mathrm{ref}} = \{0.80, 1.00, 1.05, 1.10, 1.15\}$ ($Q = 5$). The immediate costs were generated using Euclidean distance from the set point. The optimal control policy was solved using value iteration and discount factor $\gamma = 0.96$. Examination of the probabilities of entering the sink cell indicated that the system was stable at all initial cells (initial states) and set points.

Examination of the sizes of basins of attraction (stationary probabilities for entering a recurrent cell, summed over all cells) revealed that set points were reached with probabilities ranging from 49% (set point at $c_{\mathrm{B}}^{\mathrm{ref}}$=0.8) to 100% ($c_{\mathrm{B}}^{\mathrm{ref}}$= 1.05), given a random initial cell. Table 2 summarizes the results. All controllers (see next paragraph, however) converged to a communicating class which included the set point. With $c_{\mathrm{B}}^{\mathrm{ref}}$= 1.05 all cells in the communicating class had c_{B}= 1.05, while the value of c_{A} ranged from 1.7 to 2.75; with other controllers some jittering at the output existed.

An exception was the closed loop controlled with the controller for set point c_{B}= 0.80. In addition to the sink cell, two communicating classes were found: a set of 15 cells in the range of c_{A} =[2.6,2.75], c_{B}=[1,1.15] (not reaching the set point), and a set of 11 cells in the range of c_{A} =[1.00,1.20], c_{B}=[0.8,0.9] (succesful). The size of the basin-of-attraction for entering the latter (success) was substantially larger than for entering the former (failure).

A more detailed analysis, e.g., on the basin-of-attraction of the unsuccessful controls; on the expected times of transition to a particular communicating class, etc., could easily be further conducted. Finally, also simulations of the closed loop performance can be computed, of course, propagating either plant realizations or state/measurement probability distributions.

Table 2. Size of basin-of-attraction for each of the set points.

$c_{\mathrm{B}} \backslash c_{\mathrm{B}}^{\mathrm{ref}}$	0.8	1.00	1.05	1.10	1.15
0.5	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮
0.75	0	0	0	0	0
0.80	**425.7**	0	0	0	0
0.85	151.9	0	0	0	0
0.90	25.1	0	0	0	0.4
0.95	0	0	0	2.9	2.1
1.00	167.9	**559.6**	0	7.9	12.9
1.05	56.1	187.0	**861.0**	25.4	77.4
1.10	31.3	104.2	0	**824.7**	255.9
1.15	3.1	10.3	0	0	**512.4**
1.20	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮
1.50	0	0	0	0	0

From this example it can be concluded that important and useful information about the closed-loop system performance can be obtained using the CFMC control design techniques. Tuning of the controller may then be based on altering controller parameters such as discount factor or immediate costs, or on altering the system discretization (with the cost of needing to re-generate CFMC plant models). In the considered example, the system was not particularly sensitive for density of the discretization (provided dense enough), discretization of the control actions, or number of evaluations of the plant ode's.

5.3 Adaptive CFMC control of van der Vusse CSTR

In a third series of simulations, a model bank of five models were considered (using different values for the reactor temperature). The performance of the Kullback–Leibler distance $\overline{D}$ as a model/controller selection criterion was examined in a small series of simulations.

Five different values for the CSTR temperature were considered: {105, 107.5, 110, 112.5, 115 (°C)}. For each parameter setting, a CFMC model was constructed using 50 evaluations per cell. The same discretization ($S = 862$) and control specifications were used as in Section 5.2 for generating the five controllers. In order to simulate online operation (Step 4 in Section 4.2), a data sequence of length $k + 1$ was generated using a random action sequence. The empirical and corresponding model-based distributions were then generated and compared using the Kullback–Leibler distance (11).

The results of a series of simulations with different k, averaged over 1000 simulations, are shown in Table 3. Few samples were sufficient for an almost errorless behaviour ($> 95\%$ correct); a window of 20 samples provided the correct selection in all 1000 simulations.

These results indicate that the Kullback–Leibler distance could successfully be used as a basis for model/controller selection in an adaptive CFMC control framework. Note, however, that the excellent performance was under random controls and noiseless data and in practical cases longer

Table 3. Percentage of correct selections.

k	%-ok	# faults	k	%-ok	# faults
1	60	401	8	99	6
2	83	167	9	100	2
3	91	86	10	100	3
4	96	39	15	100	2
5	99	13	20	100	0
6	98	16	25	100	0
7	100	5	50	100	0

sequences would be required due to a smaller information content in a typical (closed-loop) plant operating data.

6. DISCUSSION AND CONCLUSIONS

The work reported in this paper aims at examining the possibilities of controlled finite Markov chains in learning nonlinear process control, with special emphasis on applications in the area of process engineering, such as power plants, pulp and paper mills, etc. In adaptive control, various model structures can be motivated by the availability of convenient ways to solve the associated identification and control design problems. However, robust learning from data may be more appropriate in structures motivated by the plant and the underlying phenomena. In process engineering, the theoretically or experimentally known physical (chemical, mechanical, etc.) phenomena provide a natural basis for models with good interpretability and transparency. The approach taken in this paper aims to promote the use of proper physical models as the basis of reliable adaptive control of industrial processes.

REFERENCES

Bertsekas, D. (2007) *Dynamic Programming and Optimal Control*, Athena Scientific.

Chen, H., A Kremling and F Allgöwer (1995). Nonlinear predictive control of a benchmark CSTR. *Proceedings of the 3rd European Control Conference*, pp. 3247-3258.

Hsu, C. S. (1987) *Cell-to-cell mapping -A method of global analysis for nonlinear systems*. Springer-Verlag, New York.

Häggström, O. (2002) *Finite Markov Chains and Algorithmic Applications*. Cambridge University Press, Cambridge.

Filev, D. (2002) Model bank based intelligent control. *Proceedings of the NAFIPS*, pp 583-586.

Kaelbling, L. P., M. L. Littman and A. W. Moore (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, pp. 237-285.

Kárný, M., J. Kracík and T. V. Guy (2007). Cooperative decision making without facilitator. *IFAC ALCOSP*, St-Petersburg, Russia.

Kulhavy, R (1995). Kullback-Leibler distance approach to system identification. *IFAC Symp. on Adaptive Systems in Control and Signal Processing*, Budapest, 55-66.

Lee, J. M. and J. H. Lee (2004). Approximate dynamic programming strategies and their applicability for process control: A review and future directions. *International Journal of Control, Automation and Systems*, 2 (3), pp. 263-278.

Lunze, J. (1998) On the Markov property of quantised state measurement sequences. *Automatica*, 32 (11), pp. 1439-1444.

Narendra, K. S., J Balakrishnan and M. K. Ciliz (1995) Adaptation and learning using multiple models, switching and tuning. *IEEE Control Systems*, 15 (3), pp. 37-51.

Poznyak, A. S., K. Najim and E. Gómez-Ramírez (2000). *Self-Learning Control of Finite Markov Chains*. Marcel Dekker, New York.

Puterman, M. L. (1994) *Markov Decision Processes - Discrete Stochastic Dynamic Programming*. Wiley et Sons, New York.

Shah, S. and W. Cluett (1991) Recursive least squares based estimation schemes for self-tuning control. *Canadian Journal of Chemical Engineering*, 69, pp. 89-96.

Appendix A. KULLBACK–LEIBLER DISTANCE

Following Kulhavy (1995), suppose that X_1, X_2, ..., X_{k+1} form a controlled Markov chain with a conditional probability mass function $S(y|z,u) = \Pr\{X_{k+1} = y|\ X_k = z, U_k = u\}$. The transition probability distribution is known partially, it is assumed to belong to a family S_θ. The task is to estimate θ.

For a sequence of observations $\mathbf{x} = (x_1, x_2, ..., x_{k+1})$ and control actions $\mathbf{u} = (u_1, u_2, ..., u_{k+1})$ we have

$$S_\theta^k(\mathbf{x}|x_1,\mathbf{u}) = \prod_{i=1}^k S_\theta(x_{i+1}|x_i,u_i)$$
$$= \exp\left\{\sum_{i=1}^k \log S_\theta(x_{i+1}|x_i,u_i)\right\}$$
$$= \exp\left\{k\sum_{(y,z,v)\in\mathcal{X}^2\times\mathcal{U}} R_{\mathbf{x},\mathbf{u}}(y,z,v)\log S_\theta(y|z,v)\right\}$$
$$= \exp\left\{-k\left[\overline{H}(R_\mathbf{x}) + \overline{D}(R_\mathbf{x}||S_\theta)\right]\right\}$$

where

- $R_{\mathbf{x},\mathbf{u}}(a,b,c)$ is the empirical distribution $R_{\mathbf{x},\mathbf{u}}(a,b,c) = \frac{N_{\mathbf{x},\mathbf{u}}(a,b,c)}{k}$, and $N_{\mathbf{x},\mathbf{u}}(a,b,c)$ counts the number of occurences of the triplets (a,b,c) in the sequence formed by $\mathbf{x}$ and $\mathbf{u}$.
- $\overline{H}(R_{\mathbf{x},\mathbf{u}})$ is the conditional Shannon entropy $\overline{H}(R_{\mathbf{x},\mathbf{u}}) = \sum_{(y,z,v)} R_{\mathbf{x},\mathbf{u}}(y,z,v)\log R_{\mathbf{x},\mathbf{u}}(y,z,v) + \sum_{(z,v)} R_{\mathbf{x},\mathbf{u}}(z,v)\log R_{\mathbf{x},\mathbf{u}}(z,v)$ of a random variable Y given another random variable Z and a control action V, described jointly by a probability distribution $R_{\mathbf{x},\mathbf{u}}$, and
- $\overline{D}(R_{\mathbf{x},\mathbf{u}}||S_\theta)$ is the conditional Kullback–Leibler distance $\overline{D}(R_{\mathbf{x},\mathbf{u}}||S_\theta) = \sum_{(y,z,v)} R_{\mathbf{x},\mathbf{u}}(y,z,v)\log\frac{R_{\mathbf{x},\mathbf{u}}(y,z,v)}{S_\theta(y|z,v)R_{\mathbf{x},\mathbf{u}}(z,v)}$ of a joint probability distribution $R_{\mathbf{x},\mathbf{u}}$ and a conditional distribution S_θ.
  Further, we can regard any of the conditional distributions $S_\theta(y|z,v)$ as a set of distributions $S_\theta^{z,v}(y)$, and conditional empirical distribution $R_{\mathbf{x},\mathbf{u}}(y|z,v)$ as a set of points $R_\mathbf{x}^{z,v}(y)$, $z\in\mathcal{X}$, $v\in\mathcal{U}$. We can then write
  $$\overline{D}(R_{\mathbf{x},\mathbf{u}}||S_\theta) = \sum_{(z,v)\in\mathcal{X}\times\mathcal{U}} R_{\mathbf{x},\mathbf{u}}(z,v)\,D\left(R_{\mathbf{x},\mathbf{u}}^{z,v}||S_\theta^{z,v}\right)$$

The posterior distribution of the unknown parameter θ conditional on $\mathbf{x}$ and $\mathbf{u}$ is

$$P_\mathbf{x} \propto P(\theta)\,S_\theta^k(\mathbf{x},\mathbf{u}|x_1) \propto P(\theta)\exp\left\{-k\overline{D}(R_{\mathbf{x},\mathbf{u}}||S_\theta)\right\}$$

since the conditional entropy $\overline{H}(R_\mathbf{x})$ does not depend on θ.