Presented as late breaking results contribution
21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Cascade affine constant recursive algorithm for model-based control

Gregor Černe [*] **Igor Škrjanc** [**]

[*] *Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia (e-mail: gregor.cerne@fe.uni-lj.si)*
[**] *Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia (e-mail: igor.skrjanc@fe.uni-lj.si)*

Abstract: The paper tackles trade-off between slow parameter adaptation and parameter variance of recursive least square estimation (rLSE) after a system change. In this paper, the cascade affine constant (CAC) estimation for linear systems is presented, which uses rLSE parameters as an apriori knowledge for affine constant estimation, as it can be estimated faster with lower variance as the result of its simple structure. In this configuration, rLSE uses a slower forgetting rate for more accurate dynamics estimation, while affine constant is used to react faster to changes in the system. The developed method is compared to recursive least squares in predictive functional control, in which all metrics are better or at least equal.

Keywords: cascade estimation, recursive least squares, identification, predictive control, affine

1. INTRODUCTION

The most widely used identification method for the time-variant system is forgetting factor recursive least squares estimation (rLSE) (see Plackett (1950)), which was successfully deployed in many areas (see Ding et al. (2016)). Even though many shortcomings of rLSE were already solved, the paper is tackling the problem of the sudden system parameter change: with high forgetting rate the biased samples before the change are quickly forgotten, but the variance of estimated parameters is increased, resulting in unstable control. Vise-versa, with slower forgetting rate the variance is lower, but it has a slower convergence rate which could again lead to unstable control. Both outcomes are greatly affected by the number of estimated parameters - for models with 1 parameter forgetting rate can be set much higher than for models with 6 parameters (3rd order linear system with affine constant). But on the other hand, using low-parameter models introduces enormous structural bias.

This paper explores the idea of estimating affine constant in cascade to normal rLSE algorithm - the goal is to use stable dynamic parameters using the rLSE algorithm with a slow forgetting rate as apriori knowledge for affine constant estimation, which can have faster convergence and therefore faster adaptation. The cascaded model is thought to combine the structural complexity of the high-parameter model estimated by rLSE with a fast adaptation of affine constant estimation.

The main contributions of this paper are (1) the cascade affine constant (CAC) estimation and (2) the novel training set definition which detected and excludes biased samples.

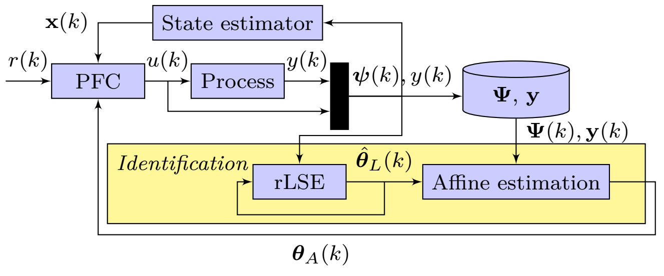
$\mathbf{x}(k)$
State estimator
$r(k)$
PFC
$u(k)$
Process
$y(k)$
$\boldsymbol{\psi}(k), y(k)$
$\boldsymbol{\Psi}, \mathbf{y}$
$\boldsymbol{\Psi}(k), \mathbf{y}(k)$
Identification
rLSE
$\hat{\boldsymbol{\theta}}_L(k)$
Affine estimation
$\boldsymbol{\theta}_A(k)$

Fig. 1. Block diagram of PFC control using CAC estimation

2. METHODOLOGY

In this section, the methodology behind the methodology of the cascade affine constant estimation is presented. In subsection 2.1 the forgetting factor recursive least squared is presented, after which the cascade affine constant estimation is explained in subsection 2.2. The section is concluded with description of used predictive functional control 2.3. The block diagram of the complete control algorithm is shown in figure 1.

2.1 Recursive least squares estimation

Recursive least squares estimation is most used algorithm for estimating linear model parameters. The output of the linear model in this paper is defined with (1).

$$\hat{y} = \boldsymbol{\psi}^T \boldsymbol{\theta}_L \quad , \tag{1}$$

where $\boldsymbol{\theta}_L = [\theta_{L,1}, \dots, \theta_{L,r}]^T$ is the vector of linear model parameters, $\hat{y}$ represents the output of the linear model, $\boldsymbol{\psi} = [\psi_1, \dots, \psi_{r-1}, 1]^T$ represents the regression vector. The parameters $\boldsymbol{\theta}_L$ are estimated using (2). The symbol $\mathbf{P}(k)$ stands for correlation matrix and $\hat{\boldsymbol{\theta}}_L(k+1)$ presents estimated parameters using rLSE in $(k+1)$ time-step.

Correlation matrix $\mathbf{P}$ is updated every sample (3). The λ stands for model forgetting factor (see Åström and Wittenmark (1994)). In order to cope with steady-state operation with low to zero excitement, the dead-band is introduced: if the model have small error (4), the rLSE in k-th time-step update is not executed. The constant k_σ is user-defined empirically.

$$\begin{aligned}\hat{\boldsymbol{\theta}}_L(k{+}1) = \hat{\boldsymbol{\theta}}_L(k) + \mathbf{P}(k{+}1)\boldsymbol{\psi}(k{+}1)\\ \cdot (y(k{+}1) - \boldsymbol{\psi}(k{+}1)\boldsymbol{\theta}_L(k)),\end{aligned} \tag{2}$$

$$\mathbf{P}(k{+}1) = \frac{\mathbf{P}(k)}{\lambda}\left(\mathbf{I} - \frac{\mathbf{P}(k)\boldsymbol{\psi}(k{+}1)\boldsymbol{\psi}(k{+}1)^T\mathbf{P}(k)}{\lambda + \boldsymbol{\psi}(k{+}1)^T\mathbf{P}(k)\boldsymbol{\psi}(k{+}1)}\right), \tag{3}$$

$$(y(k) - \mathbf{x}(k)\boldsymbol{\theta}(k))^2 < k_\sigma^2 \tag{4}$$

2.2 Cascade affine constant (CAC) estimation

When presented with only a limited number of unbiased samples after the process change, less complicated models are generally performing better. Even though the structural bias is increased, the model variance decreases more significantly, which results in decreased overall expected squared error (the statement is one of the bases for the field of feature selection, see Peng et al. (2005)). At the same time, using quality apriori knowledge (such as model parameters before system change) can improve the accuracy of parameters (for example, the knowledge that most of the parameters can be set to 0, see Ledoit and Wolf (2003)). Therefore, this paper proposes novel cascade affine constant (CAC) estimation using rLSE estimated model parameters as the apriori knowledge, for more accurate model parameter estimation after system change and faster control response.

The output equation of CAS model with cascade affine constant is in the following form (5).

$$y(k) = \mathbf{x}(k)\hat{\boldsymbol{\theta}}_L(k) + o(k) \tag{5}$$

where $o(k)$ presents cascade affine constant in k-th time step. Affine constant $o(k)$ is estimated by minimization of squared error (7) between affine constant $\hat{o}$ and the rLSE model error $\hat{e}_o$ (6) between measured output $y(k)$ and rLSE identified model output $\mathbf{x}(k)\boldsymbol{\theta}_L(k)$ on the training set. The result (8) can be interpreted as mean of the error of rLSE estimated model output on training dataset of last N_t samples.

$$\hat{e}_o(k) = y(k) - \mathbf{x}(k)\boldsymbol{\theta}_L(k) \tag{6}$$

$$J = \frac{1}{N_t}\sum_{j=0}^{N_t-1}\left(\hat{e}_o(k-j) - o\right)^2 \tag{7}$$

$$\hat{o}(k) = \frac{1}{N_t}\sum_{j=0}^{N_t-1}\hat{e}_o(k-j) \tag{8}$$

The number of training samples N_t can greatly influence the behavior of the model-based control identically as the forgetting rate influence rLSE parameters: the more samples are taken for the identification, a variance is decreased but bias can be introduced if the system changed (and vise-versa if a small number of samples is used). In section (2.2.1) the method for optimal merging of multiple estimator, which is later used in the algorithm for choosing optimal N_t, presented in section 2.2.2.

Optimal estimator merge (OEM) In this section the algorithm for estimating optimal merge between multiple estimators into one with lowest expected value of squared error is presented. Given three estimators and their variances, currently best estimator named *apriori estimator* $\hat{o}_a$ (which holds already known information), estimator which holds additional information for improving apriori estimator is named *potential estimator* $\hat{o}_p$ and *unbiased estimator* with larger variance $\hat{o}_\mathrm{b}$ (which is used to estimate biases of apriori and potential estimators), the optimal estimator merge searches such variable a to minimize expected value of squared error of combined estimator, defined as combination of apriori and potential estimator (9).

$$\hat{o} = (1-a)\hat{o}_a + a\hat{o}_p \tag{9}$$

The objective function is defined as expected value of squared difference between estimator $\hat{o}_p$ and real value of affine constant o (10).

$$\begin{aligned}J(a) &= E\Big[(\hat{o} - o)^2\Big] = \\ &= E\Big[((1{-}a)(o{+}{}_{\Delta}\hat{o}_{a,}{+}\mathrm{b}_{\hat{o}_a}){+}a(o{+}{}_{\Delta}\hat{o}_p{+}\mathrm{b}_{\hat{o}_p}) - o)^2\Big]\end{aligned} \tag{10}$$

$$\begin{aligned}E\Big[(\hat{o}_p - o)^2\Big] &= (1-a)^2E\Big[({}_{\Delta}\hat{o}_{a,+}\mathrm{b}_{\hat{o}_a})^2\Big] + 2(1-a)\cdot\\ &\cdot aE\Big[({}_{\Delta}\hat{o}_{a,+}\mathrm{b}_{\hat{o}_a})({}_{\Delta}\hat{o}_p + \mathrm{b}_{\hat{o}_p})\Big] + a^2E\Big[({}_{\Delta}\hat{o}_p + \mathrm{b}_{\hat{o}_p})^2\Big]\end{aligned} \tag{11}$$

where ${}_{\Delta}\hat{o}_a/{}_{\Delta}\hat{o}_p$ presents noise, $\mathrm{b}_{\hat{o}_a}/\mathrm{b}_{\hat{o}_p}$ presents bias of the estimator $\hat{o}_a/\hat{o}_p$. Because the true values of biases $\mathrm{b}_{\hat{o}_a}$ and $\mathrm{b}_{\hat{o}_p}$ are not known, they must be estimated using (12). The bias estimators have also additional Gaussian noise (13-14), which also need to be taken into the account when estimating expressions in (11). For the purpose of this section, the noise variances of estimators $\hat{\sigma}^2_{\hat{o}_a}, \hat{\sigma}^2_{\hat{o}_\mathrm{b}}, \hat{\sigma}^2_{\hat{o}_p}$ are known. Cross correlations between any noises in this paper are neglected. Each expected value expression is computed in (15 - 17), and after the expressions are inserted into (11), the final objective function is obtained (18).

$$E\big[\mathrm{b}_{\hat{o}_p}\big] = \hat{\mathrm{b}}_{\hat{o}_p} = \hat{o}_p - \hat{o}_\mathrm{b}, \quad E\big[\mathrm{b}_{\hat{o}_a}\big] = \hat{\mathrm{b}}_{\hat{o}_a} = \hat{o}_a - \hat{o}_\mathrm{b} \tag{12}$$

$$\mathrm{b}_{\hat{o}_a} = \hat{\mathrm{b}}_{\hat{o}_a} + {}_{\Delta}\mathrm{b}_{\hat{o}_a}, \quad \mathrm{b}_{\hat{o}_p} = \hat{\mathrm{b}}_{\hat{o}_p} + {}_{\Delta}\mathrm{b}_{\hat{o}_p} \tag{13}$$

$${}_{\Delta}\mathrm{b}_{\hat{o}_a} \sim \mathcal{N}(0, \hat{\sigma}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}}), \quad {}_{\Delta}\mathrm{b}_{\hat{o}_p} \sim \mathcal{N}(0, \hat{\sigma}^2_{\hat{o}_p} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}}) \tag{14}$$

$$E\Big[({}_{\Delta}\hat{o}_a + \mathrm{b}_{\hat{o}_a})({}_{\Delta}\hat{o}_p + \mathrm{b}_{\hat{o}_p})\Big] = \hat{\mathrm{b}}_{\hat{o}_a}\hat{\mathrm{b}}_{\hat{o}_p} \tag{15}$$

$$E\Big[({}_{\Delta}\hat{o}_a + \mathrm{b}_{\hat{o}_a})^2\Big] = \hat{\sigma}^2_{\hat{o}_a} + \hat{\mathrm{b}}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}} \tag{16}$$

$$E\Big[({}_{\Delta}\hat{o}_p + \mathrm{b}_{\hat{o}_p})^2\Big] = \hat{\sigma}^2_{\hat{o}_p} + \hat{\mathrm{b}}^2_{\hat{o}_p} + \hat{\sigma}^2_{\hat{o}_p} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}} \tag{17}$$

$$\begin{aligned}E\Big[(\hat{o} - o)^2\Big] &= (1-a)^2(2\hat{\sigma}^2_{\hat{o}_a} + \hat{\mathrm{b}}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}}) + \\ &+ 2(1-a)a(\hat{\mathrm{b}}_{\hat{o}_a}\hat{\mathrm{b}}_{\hat{o}_p}) + a^2(\hat{\mathrm{b}}^2_{\hat{o}_p} + 2\hat{\sigma}^2_{\hat{o}_p} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}})\end{aligned} \tag{18}$$

In respect to weight a, the objective function represents second order polynomial, consequentially the minimum can be analytically solved by equaling derivative of J in respect to a to 0 (19).

$$\begin{aligned}\frac{\partial}{\partial a}E\Big[(\hat{o} - o)^2\Big] &= -2(1-a)(2\hat{\sigma}^2_{\hat{o}_a} + \hat{\mathrm{b}}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}}) + \\ &+ 2(1-2a)(\hat{\mathrm{b}}_{\hat{o}_a}\hat{\mathrm{b}}_{\hat{o}_p}) + 2a(\hat{\mathrm{b}}^2_{\hat{o}_p} + 2\hat{\sigma}^2_{\hat{o}_p} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}}) = 0\end{aligned} \tag{19}$$

The final equation for weight a is (20). The second derivative of objective function is always positive, therefore the a always represents the minimum point.

$$a = \frac{2\hat{\sigma}^2_{\hat{o}_a} + \hat{\sigma}^2_{\hat{o}_\mathrm{b}} + \hat{\mathrm{b}}^2_{\hat{o}_a} - \hat{\mathrm{b}}_{\hat{o}_a}\hat{\mathrm{b}}_{\hat{o}_p}}{2\hat{\sigma}^2_{\hat{o}_a} + 2\hat{\sigma}^2_{\hat{o}_\mathrm{b}} + 2\hat{\sigma}^2_{\hat{o}_p} + \hat{\mathrm{b}}^2_{\hat{o}_a} + \hat{\mathrm{b}}^2_{\hat{o}_a} - 2\hat{\mathrm{b}}_{\hat{o}_a}\hat{\mathrm{b}}_{\hat{o}_p}} \tag{20}$$

Defining training set In this section, the incremental algorithm for defining the training set for CAC estimation is presented. The goal of the algorithm is to define optimal training samples size N_t with regard to the minimal expected squared error of the estimator (minimal sum of variance and bias).

In i-th iteration, the algorithm checks if the samples used for potential estimator $\hat{o}_{p,i}$ can improve the squared error of the best estimator in previous iteration $\hat{o}_{i-1}(k)$ using OEM. In this paper the potential estimator (22) and its variance (23) is defined as a weighted average of samples preceding $k-i$, where weights are set linearly using (21).

$$w_{p,i,j}(k) = \begin{cases} \dfrac{N_s-(k-i-j)}{\sum(N_s-(k-i-j))} = \dfrac{2(N_s-(k-i-j))}{N_s(N_s+1)}, & \begin{array}{l} k-i \geq j > \\ > k-i-N_s \end{array} \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

$$\hat{o}_{p,i}(k) = \sum_{j=1}^{i+N_s} w_{p,i,j} e_o(j) \tag{22}$$

$$\mathrm{Var}[\hat{o}_{p,i}(k)] = \hat{\sigma}^2_{\hat{o}_{p,i}}(k) = \sum_{j=1}^{N_s} w^2_{p,i,j} \tag{23}$$

The weights of j-th sample in i-th iteration $\tilde{w}_{i,j}$ is introduced. The weights in i-th iteration are derived directly from estimators definitions (24). The result is straightforward (25).

$$\hat{o}_{i,j} = (1-a_i)\hat{o}_{a,i-1} + a_i\hat{o}_{p,i} = \sum_{j=0}^{\infty} \tilde{w}_{i,j} e_o(j) \tag{24}$$

$$\tilde{w}_{i,j} = (1-a_i)\tilde{w}_{i-1,j} + a_i w_{p,i,j} \tag{25}$$

In order to use OEM algorithm, the unbiased estimator $\hat{o}_\mathrm{b}$ needs to be defined. Because every available estimator have some disadvantage (large variance or possible bias), the paper proposes that the OEM is computed with all past estimators $\hat{o}_{\mathrm{b},i} \in \hat{o}_{a,j}, \forall j$ and take lowest coerced weight $a^*_{ij}, \forall j$. Replacing $\hat{o}_a, \hat{o}_\mathrm{b}, \hat{o}_p$ with iterative estimators $\hat{o}_{a,i}, \hat{o}_{a,j}, \hat{o}_{p,i}$ in (12)(20) results in (26)(27). The a^*_{ij} can be less than 0 and higher than $\tilde{w}_{i,j}$, which contradicts the definition of weights ($a_{ij} \in [0,1]$) and assumption that the $e_o(k)$ is unbiased. Therefore, the coercion of a_{ij} is introduced (28).

$$\hat{\mathrm{b}}_{\hat{o}_{p,ij}} = \hat{o}_{p,i} - \hat{o}_{a,j}, \quad \hat{\mathrm{b}}_{\hat{o}_{a,ij}} = \hat{o}_{a,i} - \hat{o}_{a,j} \tag{26}$$

$$a^*_{ij} = \frac{2\hat{\sigma}^2_{\hat{o}_{a,i}} + \hat{\sigma}^2_{\hat{o}_{a,j}} + \hat{\mathrm{b}}^2_{\hat{o}_{a,ij}} - \hat{\mathrm{b}}_{\hat{o}_{a,ij}}\hat{\mathrm{b}}_{\hat{o}_{p,ij}}}{2\hat{\sigma}^2_{\hat{o}_{a,i}} + 2\hat{\sigma}^2_{\hat{o}_{a,j}} + 2\hat{\sigma}^2_{\hat{o}_{p,i}} + \hat{\mathrm{b}}^2_{\hat{o}_{a,ij}} + \hat{\mathrm{b}}^2_{\hat{o}_{a,ij}} - 2\hat{\mathrm{b}}_{\hat{o}_{a,ij}}\hat{\mathrm{b}}_{\hat{o}_{p,ij}}} \tag{27}$$

$$a_{ij} = \begin{cases} 0, & a^*_{ij} < 0 \\ a^*_{ij}, & 0 \leq a^*_{ij} \leq \dfrac{\tilde{w}_{i-1,1}(k)(1-a^*_{ij})}{W_i} \\ \dfrac{\tilde{w}_{i-1,1}(k)(1-a^*_{ij})}{W_i}, & \dfrac{\tilde{w}_{i-1,1}(k)(1-a^*_{ij})}{W_i} < a^*_{ij} \end{cases} \tag{28}$$

$$a_i = \min\{a_{i1}, \dots a_{ii}\} \tag{29}$$

The algorithm is initialized by setting weights following (30). The initialization ensures that while using (25), the weight of any sample is equal or lower than the weight of the most recent sample $\tilde{w}_{i,k}$. The estimated affine constant is initialized using weights (31), the initial sample variance $\hat{\sigma}^2_{\hat{o}_{a,i}}(k)$ is set to sample variance in previous time step (31) and the estimator variance is computed following (32).

$$w_{k-j} = \begin{cases} \dfrac{(N_s-j)(N_s-j+1)}{N_s(N_s+1)}, & 0 \leq j < N_s \\ 0, & \mathit{otherwise} \end{cases} \tag{30}$$

$$\hat{o}_{a,1}(k) = e_o(k), \quad \hat{\sigma}^2_{y,1}(k) = \hat{\sigma}^2_y(k-1) \tag{31}$$

$$\mathrm{Var}[\hat{o}_i(k)] = \hat{\sigma}^2_{\hat{o}_{a,i}}(k) = \frac{\sum_{j=1}^{N_s+i} w^2_{k-j}}{(\sum_{j=1}^{N_s+i} w_{k-j})^2}\hat{\sigma}^2_{y,i}(k) \tag{32}$$

The algorithm is stopped when two consecutive samples have weight relative to the $\tilde{w}_k$ smaller than the user-defined threshold t_e (33), which means that the bias has been detected (with only one, the noise could bring false positive detection). At that point, the $N_t(k)$ is set to iteration number i.

$$\frac{\max[\tilde{w}_{k-i}, \tilde{w}_{k-i+1}]}{\tilde{w}_k} < t_e \rightarrow N_t(k) = i \tag{33}$$

Last thing to estimate recursively is sample variance $\hat{\sigma}^2_{y,i}$, which is time-variant because after the system change, estimated rLSE parameters $\hat{\boldsymbol{\theta}}_L$ are introduced with bias, and this bias contributes to increased sample variance. In every step there are 2 $\hat{\sigma}^2_{y,i}$ estimators: one from previous iteration $\hat{\sigma}^2_{y,i-1}$ and measured variance $\hat{\sigma}^2_{m,i}$ (34). Best estimator is the combination of the two (35), where $a_{\hat{\sigma},i}$ is found by minimizing expected squared error of the combined estimator (36). Using approximation of $\mathrm{Var}[\hat{\sigma}_{m,i}]$ (37), the $a_{\hat{\sigma},i}$ is computed using (38).

$$\hat{\sigma}^2_{m,i} = \frac{1}{N_{t,i}-1}\sum_{j=1}^{N_{t,i}}(e_o(k-j) - \hat{o}_i(k))^2 \tag{34}$$

$$\hat{\sigma}^2_{y,i} = a_{\hat{\sigma},i}\hat{\sigma}^2_{m,i} + (1-a_{\hat{\sigma},i})\hat{\sigma}^2_{y,i-1} \tag{35}$$

$$J = E\Big[\sigma^2_{y,i} - \hat{\sigma}^2_{y,i}\Big] \tag{36}$$

$$\mathrm{Var}[\hat{\sigma}_{m,i}] \approx \frac{1}{N_{t,i}}\left(\frac{1}{2} + \frac{1}{8N_{t,i}} + \frac{21}{64N^2_{t,i}}\right)\hat{\sigma}^2_{y,i-1} \tag{37}$$

$$a_{\hat{\sigma},i} = \frac{b_{\hat{\sigma}^2_{y,i-1}}}{b_{\hat{\sigma}^2_{y,i-1}} + \mathrm{Var}[\hat{\sigma}_{m,i}]}, \quad b_{\hat{\sigma}^2_{y,i-1}} = \hat{\sigma}^2_{m,i} - \hat{\sigma}^2_{y,i-1} \tag{38}$$

2.3 State estimation and control

In this paper the control bases on predictive functional control (PFC) described in Karer et al. (2008), to which affine constant was added to the linear model. The resulting control law is (39)(40)

$$\eta = \boldsymbol{C}(\boldsymbol{A}^H - \boldsymbol{I})^{-1}(\boldsymbol{A} - \boldsymbol{I})\boldsymbol{B} \tag{39}$$

$$\begin{aligned} u =& \eta^{-1}\Big((1-a_r^H)(r(k) - y_p(k)) + y_m(k) - \\ &- \boldsymbol{C}\big(\boldsymbol{A}^H\boldsymbol{x}_m + (\boldsymbol{A}^H - \boldsymbol{I})(\boldsymbol{A} - \boldsymbol{I})^{-1}\big)\boldsymbol{O}\Big) \end{aligned} \tag{40}$$

where $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{O}$ presents matrices of state space model with affine constant (41), H presents coincidence horizon, $\mathbf{x}$ presents estimated states, y_m model output, y_p presents measured output and r presents reference.

$$\begin{aligned} \mathbf{x}(k+1) &= \boldsymbol{A}(k)\mathbf{x}(k) + \boldsymbol{B}(k)\boldsymbol{u}(k) + \boldsymbol{O}(k) \\ y(k) &= \boldsymbol{C}(k)\mathbf{x}(k) \end{aligned} \tag{41}$$

In this paper, the state space is defined in observable canonical form, therefore the states can be estimated as the delayed output y (42), where p presents model order.

$$\mathbf{x}(k) = [y(k), \ldots, y(k-p-1)]^T \tag{42}$$

Algorithm 1 Cascade affine constant estimation with rLSE

```
1:  Initialize $\boldsymbol{\theta}_L(0), \mathbf{P}(0)$
2:  **At every timestep** $k$
3:      Update rLSE parameters $\hat{\boldsymbol{\theta}}_L(k)$ (2)
4:      **CAC**
5:          Initialize variables (21 - 32), i = 1
6:          **Until** criteria (33) , i = i + 1
7:              **For each** $0 \leq j \leq i$
8:                  Compute $a_{ij}$ (27 - 28)
9:              Update $w_j$ (25)(29), $\hat{o}_i$ (8), $\hat{\sigma}^2_{y,i}$ (35)(38)
10:         Estimate final $\hat{o}(k)$ (8) using $N_t(k)$ (33)
11:     Compute $u$ using PFC, section 2.3 using $\boldsymbol{\theta}_A$ (5)
```

3. RESULTS

The developed algorithm was compared to rLSE with different parameters (λ, t_e). Process on which comparison was done was synthetic linear 3rd order system with initial discrete poles at $p_1 = p_2 = 0.9, p_3 = 0.5$, affine constant $o = 0$, and gain $K = 0.15$ with output Gaussian noise $\sigma^2 = 0.05$ and quantization with discrete step of $\Delta = 0.1$ and sampled with $T_s = 0.01s$. The system was changed at time $t_1 = 6s$, where pole is changed $p_1^* = 1.08$. The reference was changing at the interval of $4s$ between 20 and 40. The parameters of PFC are $H = 6, a_r = 0.8$.

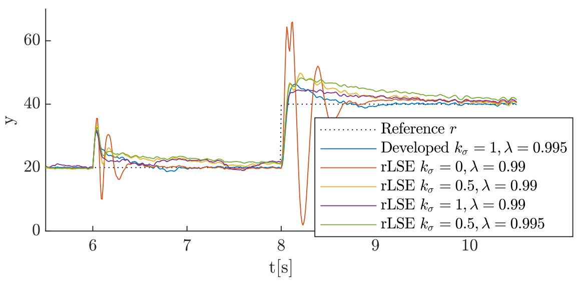

Fig. 2. Response using different identification methods

The area of interest is response just after the system change and the next reference step where the new system information is used (example in figure 2). Models were compared on the sum of squared error, percentage of overshoot (OS), and settling time t_s (when the process gets within 2 of reference). Results after the system change and after first reference change are shown in Table 1.

The developed algorithm matches the performance of fast adaptive models in settling time, and at the same time, it matches squared error and overshoot metric of the slower models. One of the drawbacks of the method is computational complexity because in each time step $\mathcal{O}(N_t^2)$ iterations are computed.

Table 1. Response analysis

	CAC	$k_\sigma = 0$ $\lambda = 0.99$	$k_\sigma = 0.5$ $\lambda = 0.99$	$k_\sigma = 1$ $\lambda = 0.99$	$k_\sigma = 0.5$ $\lambda = 0.995$
After the pole change					
e^2	**5.68**	9.32	9.96	**6.60**	12.44
OS[%]	**11.33**	15.52	13.38	**11.94**	12.59
T_s	**0.39**	**0.31**	1.23	1.92	1.39
First reference step after system change					
e^2	**6.40**	71.18	**6.86**	9.23	8.52
OS[%]	0.87	1.88	0.86	**0.62**	**0.65**
T_s	**0.43**	**0.44**	0.82	1.91	1.21

4. CONCLUSION

In the paper, the novel CAC estimation was presented, which uses stable rLSE with a slow forgetting rate as the apriori knowledge for fast and stable affine constant estimation, which combines parameter stability of rLSE with a fast adaptation of affine constant. The algorithm was compared to rLSE with different forgetting rates in the model predictive control experiment, where the developed algorithm expressed a fast convergence rate in combination with a lower overshoot at step reference change. Future work will focus on computational improvement and possible implementation of other classes of models used in model predictive control.

REFERENCES

Åström, K.J. and Wittenmark, B. (1994). *Adaptive Control*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition.

Ding, F., Liu, X., and Liu, M. (2016). The recursive least squares identification algorithm for a class of Wiener nonlinear systems. *Journal of the Franklin Institute*, 353(7), 1518–1526. doi:10.1016/j.jfranklin.2016.02.013.

Karer, G., Škrjanc, I., and Zupančič, B. (2008). Self-adaptive predictive functional control of the temperature in an exothermic batch reactor. *Chemical Engineering and Processing: Process Intensification*, 47(12), 2379–2385. doi:10.1016/j.cep.2008.01.015.

Ledoit, O. and Wolf, M.N. (2003). Honey, I Shrunk the Sample Covariance Matrix. *SSRN Electronic Journal*, 1–21. doi:10.2139/ssrn.433840.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(8), 1226–1238. doi:10.1109/TPAMI.2005.159.

Plackett, R.L. (1950). Some theorems in least squares. *Biometrika*, 37(1-2), 149–57. doi:10.1093/biomet/37.1-2.149.