USAGE OF SELF-TUNING CONTROLLERS SIMULINK LIBRARY FOR REAL-TIME CONTROL

Vladimír Bobál, Petr Chalupa, Petr Dostál

Department of Control Theory, Institute of Information Technologies,
Tomas Bata University in Zlin, Nám. TGM 275, 762 72 Zlín, Czech Republic,
tel.: +420 57 603 3217, E-mail: bobal@ft.utb.cz

Abstract: This contribution presents a usage of Self-Tuning Controllers Simulink Library (STCSL) for real-time control. The STCSL was created for design, simulation verification and especially real-time implementation of single input - single output (SISO) digital self-tuning controllers. The proposed adaptive controllers what are included into a Library can be divided into three groups. The first group covers PID adaptive algorithms using traditional Ziegler-Nichols method for the setting of the controller parameters, the second group of described controllers is based on the polynomial approach and the third group contains the controllers derived on the other approaches (minimum variance etc.). The controllers are implemented as an encapsulated Simulink blocks and thus allows users simple integration into existing Simulink schemas. The process of developing real-time applications using MATLAB Real-Time Workshop and several control courses is also presented. This Library is very successfully used in Adaptive Control Course in education practice for design and verification of self-tuning control systems in real-time conditions. It is suitable also for design and verification of the industrial digital controllers. The STCLS is available free of charge at Internet site - **http://www.utb.cz/stctool/**.

Key words: Self-tuning control; ARX model; Recursive least squares; PID control; Polynomial approach; Real-time control; MATLAB–Simulink.

1. INTRODUCTION

Self-tuning controllers belong by their character to the class of adaptive control systems. The main aim of the adaptive control approach is to solve the control problem in cases where the characteristics of a controlled system are unknown or time variable. The basic principle of the adaptive control system is to change the controller characteristics on base of the characteristics of control process. Self-tuning controllers use the combination of the recursive process identification on base of a selected model process and the controller synthesis based on knowledge of parameter estimates of controlled process.

The general task of optimal adaptive control with on-line process identification is very complicated and thus the method of **the forced separation of the identification and the control** is often used for the design of self-tuning controllers. The principle of this simplification is in the cyclic repeating of the following steps:

1. The process parameters are assumed to be known for the current control step and equal to their current estimations.
2. The control strategy for selected criterion of control quality is designed on base of previous assumption and controller output is calculated.
3. The next identification step is performed after the obtaining the new sample of the controlled variable (eventually the external measured disturbance) – the parameters of controlled process model are recomputed using a recursive identification algorithm.

Also the aim of this contribution is to inform of potential users about Self-Tuning Controllers Simulink Library - STCSL (see Bobál and Chalupa, 2002) and to help for practical usage in the simulation and real time conditions. The individual self-tuning algorithms are introduced in the brief form in User's Guide that is attached into the STCSL. This library can be also the suitable bridge between theory and practice by presenting some digital controller algorithms in a form acceptable for industrial users. The theoretical background of self-tuning controllers which are contained in the STCSL are given in Bobál, *et al.* (1999a, 1999b).

All the explicit self-tuning controllers that are included into STCSL have been algorithmically modified in the form of mathematical relations or as flow diagrams so as to make them easy to program and apply. Some are original algorithms based on a modified Ziegler-Nichols criterion, others have been culled from publications and adapted to make them more accessible to the user.

2. RECURSIVE IDENTIFICATION

The regression (ARX) model of the following form

$$y(k)=\boldsymbol{\Theta}^T(k)\cdot\boldsymbol{\Phi}(k-1)+n(k) \qquad (1)$$

is used in the identification part of the designed controller algorithms, where

$$\boldsymbol{\Theta}^T(k)=\left[a_1,a_2,...,a_{na},b_1,b_2,...,b_{nb}\right] \qquad (2)$$

is the vector of the parameters and

$$\boldsymbol{\Phi}^T(k-1)=\left[-y(k-1),-y(k-2),\dots,-y(k-na),\right.$$
$$\left.u(k-1),u(k-2),\dots,u(k-nb)\right] \qquad (3)$$

is the regression vector ($y(k)$ is the process output variable, $u(k)$ is the controller output variable). The non-measurable random component $n(k)$ is assumed to have zero mean value $E[n(k)] = 0$ and constant covariance (dispersion) $R = E[n^2(k)]$.

The recursive least squares method for calculating of the parameter estimates $\hat{\boldsymbol{\Theta}}(k)$ is utilized. Using the pure least squares method, the influence of all pairs of identified system inputs and outputs to the parameters estimates is the same. This property can be inconvenient for example when identifying the system with time-varying parameters. In this case, it is better to use least squares method with exponential forgetting where the influence of latter data to the calculation of the parameter estimates is greater then the influence of older data.

The exponential forgetting method can be further improved by adaptive directional forgetting (Kulhavý, 1987) which changes forgetting coefficient with respect to changes of input and output signal.

2. CONTROLLER ALGORITHMS

The proposed self-tuning controllers what are included into a Library are divided into three groups:

- classical digital PI and PID controllers tuned on base of Ziegler-Nichols method, pole assignment method and its modifications,
- controllers based on polynomial approach (dead-beat strong and weak version, pole assignment, controllers based on minimization of quadratic criterion),
- controllers based on other approaches (minimum variance controllers, Dahlin's controller, Bányász-Keviczky's controller etc.).

2.1 Algorithms of digital PID Ziegler-Nichols controllers

The PID controllers are still widely used in industry. These types of controllers are more convenient for users owing to their simplicity of implementation, which is generally well known. Provided the controller parameters are well chosen they can control a considerable part of continuous technological processes. To get a digital version of the PID controller, it is necessary to discretize the integral and derivative component of the continuous-time controller. For discretizing the integral component we usually employ the forward rectangular method (FRM), backward rectangular method (BRM) or trapezoidal method (TRM). The derivative component is mostly replaced by the 1st order difference (two-point difference). The recurrent control algorithms which compute the actual value of the controller output $u(k)$ from the previous value $u(k\text{-}1)$ and from compensation increment seem to be suitable for practical use

$$u(k)=q_0e(k)+q_1e(k-1)+q_2e(k-2)+u(k-1) \quad (4)$$

where q_0, q_1, q_2 are the controller parameters. The advantage of algorithm (4) is matter of fact that it is not necessary to storage last input and output data in the computer storage.

It is subsequently possible to derive further variants of digital PID controllers. First group of proposed self-tuning PID controllers is based on the classical Ziegler and Nichols (1942) method. In this well-known approach the parameters of the controller are calculated from the ultimate (critical) gain K_{pu} and the ultimate period of oscillations T_u of the closed loop system. The analytical expressions for computing of these critical parameters are derived in Bobál, *et al.* (1999a, 1999b).

2.2 Algorithms of PID pole assignment controllers

A controller based on the pole assignment method in a closed feedback control loop is designed to stabilise the closed control loop whilst the characteristic polynomial should have a previously determined pole. Digital controllers is possible can be expressed in the form of a discrete transfer function

$$G_R(z) = \frac{U(z)}{E(z)} = \frac{Q(z^{-1})}{P(z^{-1})} \qquad (5)$$

Let the controlled process be given by the transfer function

$$G_P(z) = \frac{Y(z)}{U(z)} = \frac{B(z^{-1})}{A(z^{-1})} \qquad (6)$$

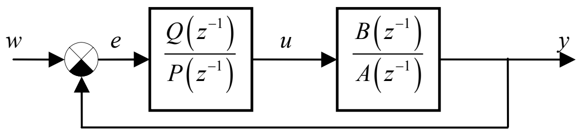

Fig. 1 Control loop with PID-A controller

PID - A controller structure. This controller structure is shown in Fig. 1, then the transfer function of closed loop circuit is

$$G_W(z) = \frac{Y(z^{-1})}{W(z^{-1})} = \frac{B(z^{-1})Q(z^{-1})}{A(z^{-1})P(z^{-1}) + B(z^{-1})Q(z^{-1})} \qquad (7)$$

and the characteristic polynomial of the closed-loop system with a PID-A controller is in the form

$$A(z^{-1})P(z^{-1}) + B(z^{-1})Q(z^{-1}) = D(z^{-1}) \qquad (8)$$

The pole placement of characteristic polynomial (8) determines the dynamic behaviour of the closed - loop system. The characteristic polynomial $D(z^{-1})$ can be specified by different methods.

PID - B controller structure. The structure of the control loop with controller PID-B is shown in Fig. 2. The characteristic polynomial has in this case form

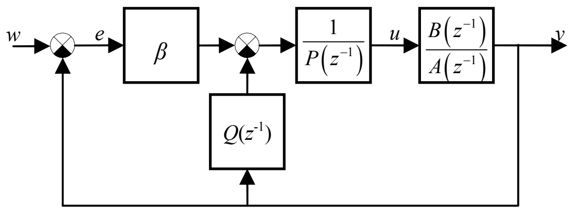

Fig. 2 Control loop with PID-B controller

$$A(z^{-1})P(z^{-1}) + B(z^{-1})\left[Q(z^{-1}) + \beta\right] = D(z^{-1}) \qquad (9)$$

2.3 Self-tuning controllers based on polynomial approach

Different forms of self-tuning controllers could be suggested according to the chosen type of a plant model (and, consequently according to the chosen identification method), according to a criterion of quality, according to a mathematical procedure during deriving of controller equations, etc. Next group of described controllers is based on the polynomial approach. These algorithms succeed various criterions for a control process course – dead-beat method, pole assignment method and LQ (linear-quadratic) control method. The design of these controllers results from the basic block diagram of 2DOF (two-degree freedom) configuration (see Fig. 3). From Fig. 3 is obvious that the controller equation has the form

$$P(z^{-1})K(z^{-1})u(k) = R(z^{-1})w(k) - Q(z^{-1})y(k) \qquad (10)$$

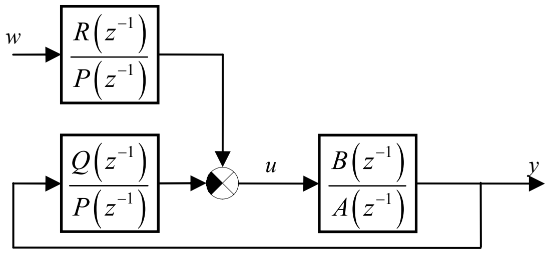

Fig. 3 Closed loop 2DOF control system

Then the transfer function of closed loop 2DOF circuit is

$$G_W(z) = \frac{Y(z^{-1})}{W(z^{-1})} = \frac{R(z^{-1})B(z^{-1})}{A(z^{-1})P(z^{-1}) + B(z^{-1})Q(z^{-1})} \qquad (11)$$

The special case is a controller 1DOF (one degree of freedom - see Fig. 1)), which is valid for $R(z^{-1}) = Q(z^{-1})$ and which works with a tracking error $e(k) = w(k) - y(k)$. It could be proved that such a controller is only suboptimal for tasks of a reference value tracking.

The process of computing coefficients of polynomials $Q\left(z^{-1}\right)$ and $P\left(z^{-1}\right)$ is similar to 1DOF case. The presence of feedforward part brings the possibility of simpler processing of reference signal. If the 1DOF and 2DOF controllers have to fulfil the same requirements when controlling the same system, the degrees of polynomials in 1DOF control are usually higher then in 2DOF control.

3. SELF-TUNING CONTROLLERS SIMULINK LIBRARY

The Simulink is nowadays a word-wide standard in simulation, testing, and verification of behaviour of various dynamic systems. Simulink is a part of MATLAB system and supports linear or nonlinear systems modelled in continuous time, sampled time or a hybrid of the two. Systems can also be multirate, i.e. have different parts that are sampled or updated at different rates.

On base on monograph Bobál, *et al.* (1999a) was created a library of self-tuning controllers in MATLAB/Simulink environment. The purpose was to create a framework suitable for creating and testing of self-tuning controllers. The library is available free of charge at internet site of Tomas Bata University in Zlin – www.utb.cz/stctool (see Bobál and Chalupa, 2002). The library was created using MATLAB version 6.1 (Release 12.1), but it can be ported with some changes to lower MATLAB versions. Controllers are implemented in the library as standalone Simulink blocks, which allow an easy incorporation into existing simulation schemes and an easy creation of new simulation circuits. Only standard techniques of Simulink environment were used when creating the controller blocks and thus just basic knowledge of this environment is required for the start of work with the library. Controllers can be implemented to simulation schemes just by the copy or drag & drop operation and their parameters are set using dialog windows. Another advantage of used approach is a relatively easy implementation of user-defined controllers by modifying some suitable controller in the library.

Nowadays the library contains over 30 simple single input single output discrete self-tuning controllers, which use discrete ARX models of second and third order for the on-line system identification. All of these controllers use discrete control laws, where controller parameters are computed by various methods. Many methods calculate controller parameters on base of the ultimate parameters of controlled system (Ziegler-Nichols approach) or on base of pole assignment approach. The library package contains not only the controllers but also reference manual with simple description of the algorithm and the internal structure of each controller.

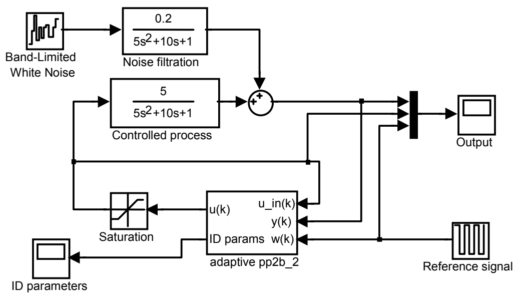

Fig. 4 Control circuit in Simulink environment

The typical wiring of any library controller is shown in Fig. 4. Each self-tuning controller from the library uses 3 input signals and provides 2 outputs. The inputs are the reference signal (w) and the actual output of controlled process (y). The last controller input is the current input of controlled process – the control signal (u_in). The value of this signal does not have to be the same as controller output e.g. due to the saturation of controller output. The main controller output is, of course, control signal – the input signal of the controlled process. The second controller output consists of the current parameter estimates of the controlled process model. The number of parameters this output consists of depends on the model used by on-line identification.

A scheme analogous to the scheme in Fig. 4 can be used to simulate the control process of both a discrete and a continuous controlled process with much more complicated structure. It is possible to implement processes with time variable parameters, processes described by non-linear differential equations etc.

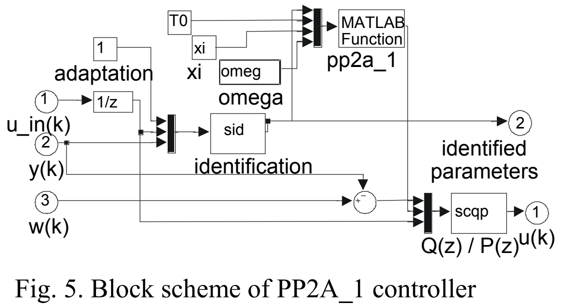

Fig. 5. Block scheme of PP2A_1 controller

Each library controller is constructed as a mask of a subsystem, which consists of Simulink blocks and has inputs, outputs and parameters. Internal controller structure consists of Simulink blocks which provide, among others, the possibility of easy creation of a new controller by a modification of some suitable library controller. The structure of controller *pp2a_1* is presented in Fig. 5 as an example. Each library controller consists of three basic parts:

– on-line identification block,
– block computing controller parameters,
– block computing controller output.

4. REAL - TIME CONTROL

When using the library to control some real-time model or equipment the first phase consists of selecting appropriate controller and some simulations are performed to tune controller parameters. In the next phase the testing the controller using real model is performed. In the last phase the Simulink environment is used to generate source codes of program working outside MATLAB/Simulink environment. This code can be compiled, linked and loaded into industrial controllers.

Practical verification of library controllers has been performed using several laboratory models. One of them, the coupled servomotors CE 108, is shown in Fig. 6. The decentralized approach using logical supervisor was applicated for control of this multivariable system.

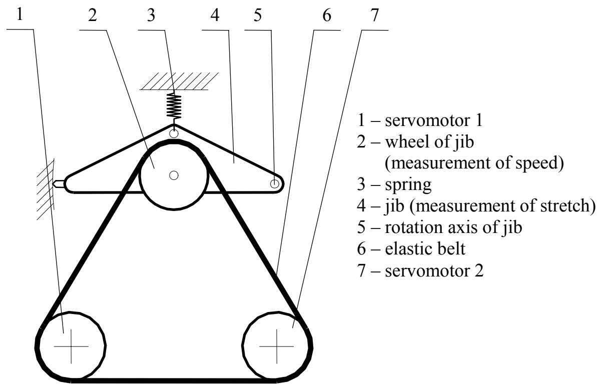

Fig. 6 Laboratory model of coupled motors CE 108

This system has two inputs (rotations of servomotors) and two outputs: the speed of the belt measured by the rotations of wheel and a stretch of the belt measured by deviation of the jib. The system was divided into two input-output pairs, where the first pair consists of the rotations of servomotor 1 as an input and the stretch of a belt as an output. The second pair consists of the rotation of servomotor 2 as an input and the speed of the belt as an output. This system is strongly nonlinear with great interaction between loops and thus adaptive controllers were used to archive satisfactory control courses.

The connection of the laboratory model and the Simulink environment has been realized through control and measurement PC card Advantech PCL-812 PG. Blocks for reading analogue inputs and for writing to the analogue outputs on the PC card were used to communicate with the model. The sample of control course using Ziegler-Nichols based controllers is plot in Fig. 7. The first part of control course (approximately 0-40s) is used to adapt controller to the system because initial estimations of model parameters are set without using prior information about the controlled system.

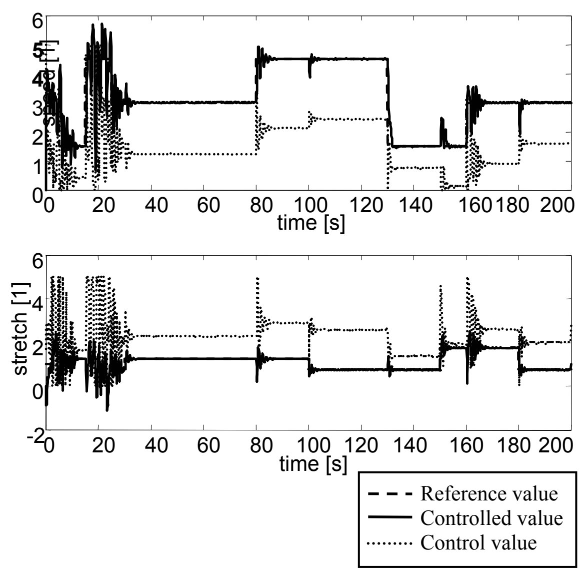

Fig.7 Control courses using Ziegler-Nichols based controllers (zn2fr)

Another control courses are shown in Fig. 8. The pole placement controllers were used in this case. The poles were set so that the closed loop behaves like second order continuous system.

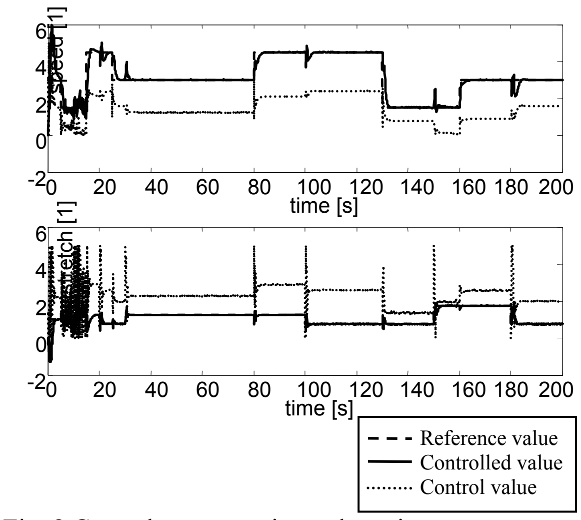

Fig. 8 Control courses using pole assignment controllers (pp2a1)

The last courses presented in Fig. were obtained using minimum variance controllers in both control loops. Minimum variance controllers provided the smoothest control courses while best control process was obtained using pole placement controllers. The integral criterion was used to compare the quality of control process.

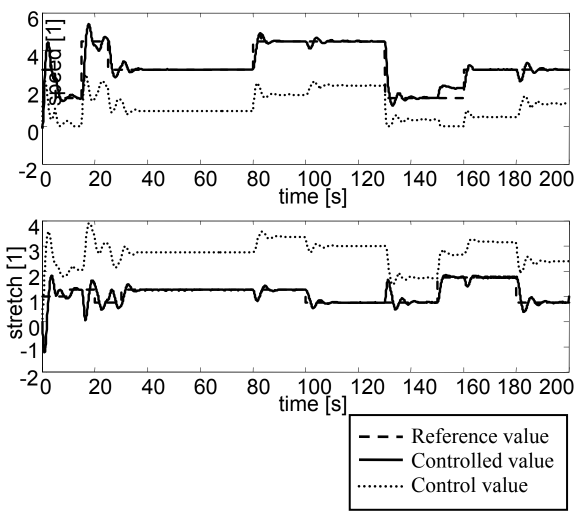

Fig. 9 Control courses using minimum variance controllers (mv2)

5. CREATING APPLICATIONS WITH REAL - TIME WORKSHOP

The MATLAB/Simulink environment can also be used to generate code to be used in controllers in industrial practice. *Real-Time Workshop*, one of the toolboxes shipped with MATLAB, allows generating of source code and programs to be used outside the MATLAB environment. The process of generating the source code is controlled by special compiler files that are interpreted by *Target Language Compiler*. These files are identified by the *.tlc* (target language compiler) extension and describe how to convert Simulink schemes to target language. Thereby source code is generated and after compiling and linking the resulting application is created. Applications for various microprocessors and operating systems can be created by selecting corresponding target language compiler files.

The target language compiler can create applications to be used under the Windows environment, which perform control algorithm and save results in a binary file with the structure acceptable by MATLAB. An analysis of the control process can then be performed using advantages of MATLAB functions and commands. Selecting another *.tlc* file leads to creation of a MS-DOS application or an application to be used on PC based industrial computers without a requirement of any operating system.

Many manufactures of industrial computers and controllers has created their own target language compiler files used to create applications for equipment they produce. *Real-Time Workshop* provides a relatively opened environment for the conversion of block schemes to various platforms where users can create their own target language compiler files for converting the block scheme to a source code and hence reach the compatibility with any hardware.

An application creation consists only of selecting appropriate target language compiler file, eventually setting compiler parameters and then start-up of compilation process.

6. CONCLUSIONS

The Self-Tuning Controllers Simulink Library is used in university course of adaptive control systems. Its architecture enables an easy user orientation in Simulink block schemes and source code of controllers' functions. The controllers provided are suitable for modification and thereby implementation of user-defined controllers. The compatibility with *Real-Time Workshop* ensures not only the possibility of laboratory testing using real time models but also the possibility of creating applications for industrial controllers.

ACKNOWLEDGEMENTS

This work was supported in part by the Grant Agency of the Czech Republic under grants No.102/02/0204, 102/00/0526 and in part by the Ministry of Education of the Czech Republic under grant MSM 281100001.

REFERENCES

Bobál, V., J. Böhm, J. Prokop and J. Fessl (1999a). *Practical Aspects of Self-Tuning Controllers: Algorithms and Implementation*. VUTIUM Press, Brno University of Technology, Brno (in Czech).

Bobál, V., J. Böhm, and R. Prokop (1999b). Practical aspects of self-tuning controllers. *International Journal of Adaptive Control and Signal Processing*, **13**, 1999, 671-690.

Bobál, V. and P. Chalupa (2002). Self-Tuning Controllers Simulink Library. http: //www.utb.cz/stctool/.

Kučera, V. (1979). Discrete Linear Control: The Polynomial Equation Approach. John Wiley, Chichester.

Kučera, V. (1991). Analysis and Design of Discrete Linear Control Systems. London, Prentice Hall.

Kulhavý, R. (1987). Restricted exponential forgetting in real time identification. *Automatica*, **23,** 586-600.

Ziegler, J. G. and N. B. Nichols (1942). Optimum settings for automatic controllers, *Trans. ASME* **64**, 1942, 759 – 768.