Int. J. Appl. Math. Comput. Sci., 2008, Vol. 18, No. 3, 377–387
DOI: 10.2478/v10006-008-0034-7

amcs

ON THE DEVELOPMENT OF SCILAB COMPATIBLE SOFTWARE FOR THE ANALYSIS AND CONTROL OF REPETITIVE PROCESSES

ŁUKASZ HŁADOWSKI *, BŁAŻEJ CICHY *, KRZYSZTOF GAŁKOWSKI *, ERIC ROGERS **

* Institute of Control and Computation Engineering
University of Zielona Góra, ul. Podgórna 50, 65–246 Zielona Góra, Poland
e-mail: {L.Hladowski,B.Cichy,K.Galkowski}@issi.uz.zgora.pl

** School of Electronics and Computer Science
University of Southampton, Southampton, SO17 1BJ, UK
e-mail: etar@ecs.soton.ac.uk

In this paper further results on the development of a SCILAB compatible software package for the analysis and control of repetitive processes is described. The core of the package consists of a simulation tool which enables the user to inspect the response of a given example to an input, design a control law for stability and/or performance, and also simulate the response of a controlled process to a specified reference signal.

Keywords: repetitive processes, simulation, SCILAB.

1. Introduction

Repetitive processes are a distinct class of 2D systems (i.e., information propagation in two independent directions) of both system theoretic and application interest. Their unique feature is a series of sweeps, termed passes, through a set of dynamics, defined over a fixed finite duration known as the pass length (denoted by $\alpha < +\infty$). If we denote the pass index by the integer $k \geq 0$ and use p as the along-the-pass variable, then these processes evolve over the domain $0 \leq p \leq \alpha < \infty$, $k \geq 0$. Also, on each pass an output, termed the pass profile, is produced which acts as a forcing function on, and hence contributes to, the dynamics of the next pass profile. This, in turn, leads to the unique control problem in that the output sequence of the pass profiles generated can contain oscillations that increase in amplitude in the pass-to-pass (i.e., that indexed by k) direction.

Physical examples include long-wall coal cutting and metal rolling operations (see, for example, (Rogers and Owens, 1992)). Also, in recent years applications have arisen where adopting a repetitive process setting for analysis has distinct advantages over alternatives. For example, they can be used to analyze an important class of iterative learning control (ILC) schemes (Owens *et al.*, 2000). More recently, another application has arisen in the context of self-servowriting in disk drives (Melkote *et al.*, 2003), and there are as yet unexploited links with one approach to the analysis of spatially interconnected systems (D'Andrea and Dullerud, 2003) or, generally, systems described by partial differential equations (Rabenstein and Trautmann, 2003).

Attempts to control these processes using standard ('classical' 1D) systems theory/algorithms fail (except in several very restrictive special cases) precisely because such approaches ignore their inherent 2D structure, i.e., information propagation occurs in two independent directions. For example, early work (for details, see, for example, the references cited in Chapter 2 of (Rogers and Owens, 1992)) advocated converting the dynamics of linear processes into those of a standard linear system using the so-called total distance traversed variable. Such an approach ignores the finite length nature of these processes and also the fact that the boundary conditions are reset before the start of each new pass. In particular, this 1D systems approach fails to provide a correct interpretation of what stability means for these processes and hence cannot then be used to design control laws to guarantee this most basic property (and onwards to achieve the desired performance). Hence the only way forward is to develop a control and systems theory, and hence control algorithms,

taking full account of the underlying process dynamics.

In this paper we consider the so-called discrete linear repetitive processes which can arise either from direct modelling of a physical process or as a result of sampling the dynamics of a differential process in the along-the-pass direction. Their state-space model (Rogers and Owens, 1992) has the following form over $0 \leq p \leq \alpha,\ k \geq 0$, where k denotes the pass number or index:

$$\begin{aligned} x_{k+1}(p+1) &= Ax_{k+1}(p) + Bu_{k+1}(p) + B_0 y_k(p), \\ y_{k+1}(p) &= Cx_{k+1}(p) + Du_{k+1}(p) + D_0 y_k(p). \end{aligned} \tag{1}$$

Here, on the pass k, $x_k(p) \in \mathbb{R}^n$ is the state vector, $y_k(p) \in \mathbb{R}^m$ is the pass profile vector, and $u_k(p) \in \mathbb{R}^r$ is the vector of control inputs.

To complete the process description, it is necessary to specify the boundary conditions, i.e., the initial state vector on each pass and the initial pass profile. Here, no loss of generality arises from assuming

$$\begin{aligned} x_{k+1}(0) &= d_{k+1}, \qquad k \geq 0, \\ y_0(p) &= f(p), \quad 0 \leq p \leq \alpha \end{aligned} \tag{2}$$

where the $n \times 1$ vector d_{k+1} contains known constant entries and $f(p)$ is an $m \times 1$ vector whose entries are known functions of p.

The development of a control systems theory for these processes, with a follow through to control systems design, for discrete linear repetitive processes has been the subject of much profitable research effort. For an overview of the major areas of processes see, for example, (Rogers *et al.*, 2007) and the references cited in this monograph. As noted above, new applications are still arising and in such cases there is clearly a need for high quality software to assist in the analysis and design of control schemes for such applications in addition to supporting the ongoing theoretical developments. The purpose of this paper is to report further extensions to the capabilities of software for this purpose focusing, in particular, on stability testing, control law design, and the simulation of the resulting controlled process dynamics. The next section summarizes the necessary background theory.

2. Background

Consider the case when the dynamics of a repetitive process are linear. Then the following model includes the process models which arise in most of the currently known applications:

$$y_{k+1} = L_\alpha y_k + b_{k+1}, k \geq 0. \tag{3}$$

In this model, $y_k \in E_\alpha$ denotes the pass profile on the pass k, L_α is a bounded linear operator which maps E_α into itself and $b_{k+1} \in W_\alpha$, where W_α is a linear subspace of E_α. Also, the term $L_\alpha y_k$ describes the contribution of the pass k to the pass $k+1$, and b_{k+1} represents the inputs and other effects which enter on the current pass. The advantage of using such a model is that the stability theory can be developed in terms of this general model and then specialized to the particular one of interest.

Recall now that the unique control problem for repetitive processes is that the sequence of the pass profiles y_k, $k \geq 0$, generated can contain oscillations that can increase in amplitude in the pass-to-pass direction (k). Hence a natural definition of stability is to require that bounded input sequences produce bounded output (pass profiles) sequences. This leads to the following definition.

Definition 1. (Rogers and Owens, 1992) Suppose that $||\cdot||$ denotes the norm on E_α. Then linear repetitive processes described by (3) are asymptotically stable provided there exist real numbers $M_\alpha > 0$ and $\lambda_\alpha \in (0,1)$ such that $||L_\alpha^k|| \leq M_\alpha \lambda_\alpha^k$, $k \geq 0$ (where $||\cdot||$ is also used to denote the induced operator norm).

If this property holds then the sequence of pass profiles generated converge strongly in k to the so-called limit profile $y_\infty := \lim_{k\to\infty} y_k$, which is the unique solution of the linear equation

$$y_\infty = L_\alpha y_\infty + b_\infty, \tag{4}$$

where $b_\infty = \lim_{k\to\infty} b_k$.

Consider now a discrete linear repetitive process described by (1) and (2). Then in this case asymptotic stability holds if, and only if, all eigenvalues of the matrix D_0 lie in the open unit circle in the complex plane. Also the corresponding limit profile is a 1D discrete linear system with a state matrix (with $D = 0$ for simplicity) $A_{lp} := A + B_0(I_m - D_0)^{-1}C$. Hence if an example is asymptotically stable then its repetitive dynamics, after a sufficiently large number of passes have elapsed, can be replaced by those of a 1D discrete linear system—this property is of obvious interest in terms of applications but does not mean that only 1D systems concepts need to be applied to these processes. Note, however, that this property is, in effect, independent of the process state dynamics and, in particular, of the state matrix A. This is due to the finite pass length and can result in the limit profile having unacceptable dynamics along the pass, e.g., the case when $A = -0.5 + \beta,\ B = 0, B_0 = 0.5 + \beta,\ C = 1,\ D = D_0 = 0$, where β is a real scalar. In this case $A_{lp} = \beta$ and hence the limit profile is unstable along the pass if $|\beta| \geq 1$.

To avoid an unstable limit profile, it is necessary to strengthen the concept of stability and this can be achieved by demanding that the property of the above definition holds for all possible values of the pass length. This is termed stability along the pass, and for the processes considered here the necessary and sufficient properties are that (i) all eigenvalues of the matrices D_0 and A lie in the open

unit circle in the complex plane, and (ii) all eigenvalues of the transfer-function matrix

$$G_0(z) := C(zI_n - A)^{-1}B_0 + D_0$$

lie in the open unit circle in the complex plane for all $|z| = 1$. As the example above demonstrates, 1D stability of the state matrix A is also only a necessary condition for stability along the pass (another reason why these processes cannot be analyzed by direct application of 1D linear systems theory).

The three necessary and sufficient conditions for stability along the pass given above can be tested by direct application of 1D linear systems stability tests, e.g., Nyquist plots. A major drawback, however, is that such tests do not provide a basis on which to also address the question of control law design for stability and/or performance. This has led in recent years to the use of Linear Matrix Inequality (LMI) techniques (see, e.g., (Boyd *et al*., 1994)), and there now exists a large volume of results on the design of physically implementable control laws (for a detailed description refer, for example, to (Sulikowski, 2006; Gałkowski, Lam, Rogers, Xu, Sulikowski, Paszke and Owens, 2003) and the references therein).

The basic LMI based condition for stability along the pass is as follows, where from this point onwards a symmetric positive definite (resp. negative definite) matrix, say M, is denoted by $M \succ 0$ (resp. $M \prec 0$.)

Theorem 1. (Gałkowski, Lam, Rogers, Xu, Sulikowski, Paszke and Owens, 2003) *A discrete linear repetitive process described by (1) and (2) is stable along the pass if there exist matrices $P \succ 0$ and $Q \succ 0$ satisfying the following LMI:*

$$\begin{bmatrix} \widehat{A}_1^T P \widehat{A}_1 + Q - P & \widehat{A}_1^T P \widehat{A}_2 \\ \widehat{A}_2^T P \widehat{A}_1 & \widehat{A}_2^T P \widehat{A}_2 - Q \end{bmatrix} \prec 0. \qquad (5)$$

This last result is a sufficient, but not necessary and sufficient, condition for stability along the pass and hence it can be conservative. Unlike alternatives, however, it provides a direct route to control law design. For example, if stability along the pass does not hold for a given example, then a stabilizing control law is the minimum needed. From the discussion above, it is clear that a control law which only uses current pass information alone (e.g., the state feedback law $u_{k+1}(p) = Fx_{k+1}(p)$) cannot guarantee stability along the pass, i.e., these processes cannot be controlled by direct application of 1D linear systems control laws. Instead, the control law must be activated by a combination of current and previous pass information, and one particular case is

$$u_{k+1}(p) = K_1 x_{k+1}(p) + K_2 y_k(p) = K \begin{bmatrix} x_{k+1}(p) \\ y_k(p) \end{bmatrix}, \qquad (6)$$

where K_1 and K_2 are matrices to be computed.

Currently the only effective approach for the computation of the control law matrices here to ensure stability along the pass for the controlled process is through the use of LMIs. In particular, introduce

$$\hat{A}_1 = \begin{bmatrix} A & B_0 \\ 0 & 0 \end{bmatrix}$$

and

$$\hat{A}_2 = \begin{bmatrix} 0 & 0 \\ C & D_0 \end{bmatrix}.$$

Then the following result gives an LMI based sufficient condition for stability along the pass of the controlled process where from this point onwards we use the notation $X \succ 0$ (respectively $X \prec 0$) to denote a symmetric positive-definite (respectively negative-definite) matrix.

Theorem 2. (Gałkowski, Lam, Rogers, Xu, Sulikowski, Paszke and Owens, 2003) *Suppose that a control of the form (6) is applied to a discrete linear repetitive process described by (1). Then the resulting controlled process is stable along the pass if there exist matrices $Y \succ 0$, $Z \succ 0$ and N such that the following LMI is feasible:*

$$\begin{bmatrix} Z - Y & 0 \\ 0 & -Z \\ \hat{A}_1 Y + \hat{B}_1 N & \hat{A}_2 Y + \hat{B}_2 N \\ Y\hat{A}_1^T + N^T \hat{B}_1^T & Y\hat{A}_2^T + N^T \hat{B}_2^T \\ Y\hat{A}_2^T + N^T \hat{B}_2^T & -Y \end{bmatrix} \succ 0,$$

where

$$\hat{B}_1 = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad \hat{B}_2 = \begin{bmatrix} 0 \\ D \end{bmatrix}.$$

If this condition holds, then

$$K = NY^{-1}. \qquad (7)$$

An alternative route is based on the following result.

Theorem 3. (Gałkowski, Lam, Rogers, Xu, Sulikowski, Paszke and Owens, 2003) *Suppose that a control of the form (6) is applied to a discrete linear repetitive process described by (1). Then the resulting controlled process is stable along the pass if there exist matrices $P_1 \succ 0$,*

$P_2 \succ 0$, $P = \mathrm{diag}\{P_1, P_2\} \succ 0$, N_1 and N_2 such that the following LMI is feasible:

$$\begin{bmatrix} -P & \Phi P + RN \\ P\Phi^T + N^T R^T & -P \end{bmatrix} \succ 0,$$

where

$$\Phi = \hat{A}_1 + \hat{A}_2,$$

$$N = \begin{bmatrix} N_1 & N_2 \\ N_1 & N_2 \end{bmatrix}.$$

If this condition holds, then

$$K = NP^{-1}. \tag{8}$$

Necessary and sufficient conditions for stability along the pass of the controlled process under this and alternative control laws can be written down but this does not lead, in almost all cases, to control law design tools. In contrast, computations with LMIs are well understood and easily implemented. Moreover, for design studies, there is clearly a need to produce a toolkit which also includes options to simulate the controlled process and graphically display the results. In fact, there has been previous work in this area—it is first reviewed in the next section, and then substantial need features added in recent work are detailed and illustrated.

3. Toolkit

A core problem encountered during control related analysis of repetitive processes is how to visualize the process dynamics. This problem was considered in, e.g., (Gramacki, 1999; Gramacki *et al*., 2005; Hładowski *et al*., 2006) but the resulting software was difficult to extend and/or based on commercial environments.

To overcome such limitations, the development of a new toolkit has been initiated in the SCILAB (Scilab, 2008) environment. The main advantage of this option over alternatives is the open-source license of SCILAB and a rapidly growing number of users. An introduction to the SCILAB environment is given in (Gomez *et al*., 1998) and (Campbell *et al*., 2006) and the LMI component in (Nikoukhah, Delebecque and Ghaoui, 2008). The overall aim is to produce high quality reliable software to support the analysis and design of control laws for use with, for example, experimental facilities (Ratcliffe *et al*., 2005) where ILC control laws designed in a repetitive process setting have been experimentally verified.

The remainder of this paper describes how this toolkit has been developed/enhanced in the following major areas:

- The visualization of the process dynamics: 3D plots and 2D plots (both along-the-pass and pass-to-pass).
- Stability analysis in an LMI setting.
- Control law design based on the use of LMIs.
- Options to use a pre-specified input or reference vector (for the controlled case) or to specify such prior to simulating the process response.
- A user-friendly interface.

A first version of this toolkit is described in (Hładowski *et al*., 2006), where the major feature was the simulation of the response of an example (together with some very basic control law design algorithms). This provided the starting point for the developments reported here where the first of these was the rewriting of functions that do not depend on the model structure to accept much more general parameters and hence facilitate easier extensions to the toolkit. Moreover, after the initial release it became obvious that much stronger type checks are required. This is motivated by the fact that the linear repetitive process model contains many variables that are error prone. This has led to the development of new functions for dealing with this task.

Effort was also directed to the development of a new, vastly improved help system. It is based on standard SCILAB templates but contains many more illustrative examples. Moreover, a number of potential pitfalls are explained. In-depth attention was also paid to the presentation of the results. In the new version, the drawing engine was rewritten to allow much easier use in scripts—all the functions have a much clearer syntax. Moreover, the plotting routines were extended to handle degenerate cases (such as a plot of a single point on a single pass). In order to ensure readability, a maximum of 32 passes can be drawn simultaneously using different colors for clarity. Additionally, due to the extended LaTeX support, creating multiple plots is much faster (from $O(n)$ to $O(1)$ calculations of plot surfaces), and the plot surface is calculated only for data required for plotting—when, for example, user requests an "along-the-pass" plot for points $7, \ldots, 18$ for $\alpha = 100$ only the first 18 points are calculated. If necessary, it can be requested to calculate the entire surface.

Compared with the previous version, the 3D plots are now given in full color to better visualize the range of values in them. The main toolkit window is shown in Fig. 1. Moreover, the LaTeX support has been greatly extended.

3.1. Stability analysis and control law design. In terms of stability analysis, asymptotic stability (and hence the construction of the resulting limit profile for the processes considered in this paper) is simple to check as it requires that all eigenvalues of the matrix D_0 in (1) have modulus strictly less than unity (Benton, 2000). A test for this property is implemented in the toolkit as

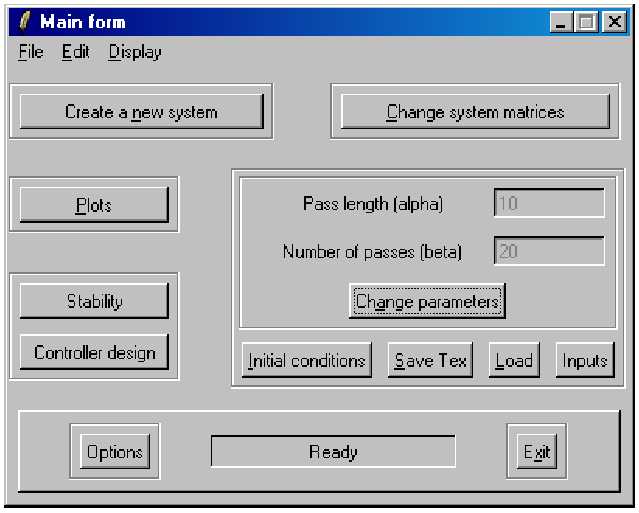

Fig. 1. Main toolkit window.

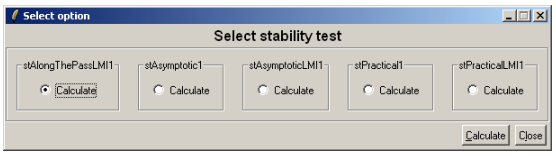

Fig. 2. Stability analysis window.

`stAsymptotic1`. An LMI interpretation is also implemented as `stAsymptoticLMI1`. (This is based on the fact that all eigenvalues of the matrix D_0 lie in the open unit circle in the complex plane if, and only if, there exists a matrix $Q \succ 0$ such that $D_0^T Q D_0 - Q \prec 0$ which has an obvious LMI interpretation.)

Asymptotic stability is always a necessary condition for stability along the pass and hence should be established for a given example before proceeding to stability along the pass where in the toolkit the condition of Theorem 2 can be implemented using `stAlongThePassLMI1`.

One of the main reasons for selecting SCILAB as the host platform for the toolkit was the excellent LMI solver available for this platform as LMITOOL*: A Package for LMI Optimization in* SCILAB (Nikoukhah, Delebecque and Ghaoui, 2008). Note also that provision is available to easily include the existing or newly developed tests by simply implementing a single SCILAB function with no need to change the GUI, as illustrated in Fig. 2. (Practical stability is another property which lies between asymptotic stability and stability along the pass and is not discussed here.)

In the case of control law design, one approach would be to use the LMI tools already available in SCILAB and code the methods by hand. However, this approach requires substantial knowledge about the internal data structures used by the LMI solver and this is not ideal. Consequently, a set of ready-to-use control law design templates have been incorporated into the toolkit. Moreover, expanding it to include additional control law design methods is a simple task.

3.2. LaTeX export. The presentation of simulation results can be a time consuming task, especially when we have to compute with large dimensioned matrices, which are often encountered in this area. To simplify this tedious task, the toolkit was equipped with LaTeX export capabilities. An essential novelty is the fact that the user needs to write the LaTeX file, adding the tags that will be replaced by the simulation results, instead of using a complicated syntax of the previous toolkit version. It is possible to include any number of plots, both 2D and 3D. On each plot any number of points/passes can be displayed, which is an essential difference with the "interactive" plots discussed earlier. Note, however, that 32 points/passes on each plot can be displayed in unique colors—a substantial improvement over the previous version. The process of plot selection is simplified by the use of an interactive wizard. It is also possible to export simulation data from the script.

3.3. Usability enhancements. One of the design goals was to make this new toolkit as user friendly as possible. To achieve this, a "new system wizard" for entering various process parameters was implemented. Since most of the model parameters are matrices, the basic method to define these is the SCILAB convention for entering matrices (exactly the same as in MATLAB). To simplify this process, it is also possible to enter the matrix element by element (see Fig. 3).

To make the end product available for a broader audience, the Windows operating system version employs an easy to use multilingual (currently Polish and English) installer based on the *Nullsoft Install System* (*NSIS*) (Nullsoft, 2008). This system is widely regarded to be a very reliable, free solution that produces a small overhead code. Additionally, the *NSIS* can package and verify all the files included into the prepared compilation. Moreover, as an additional safety measure, for each installed file the Message-Digest Algorithm 5 (better known as MD5) checksum is calculated using the *MD5 library* ("IGx*" Lieder, 2008). This value is used when upgrading and uninstalling the toolkit—if a change is detected, the user can choose to leave the file intact. Essentially, this feature provides protection against accidental deletion of manual changes.

During the installation phase, an existing LaTeX installation is automatically detected. Currently, the most popular *MiKTeX* distribution is supported by the installer, but any standard LaTeX can be used.

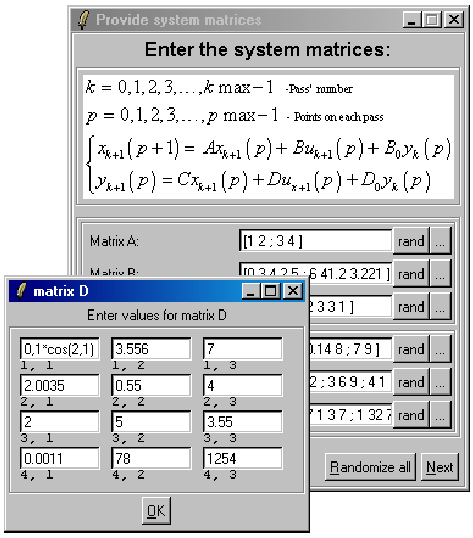

Fig. 3. Matrix wizard windows: main and the matrix entry.

4. Implementation details

The toolkit consists of a TCL/TK GUI frontend and a number of SCILAB script files and functions. All the basic process parameters are included in the `lrp` structure which is implemented as a new type based on `tlist` (typed list; native SCILAB datatype) with the following fields:

- `lrp_Sys_Cla` — (`string`), the name of the data type.
- The `lrp.mat` field of the type `lrp_Sys_Cla_Mat` used for storing the model matrices—see (1). Currently, it contains the values of `A`, `B`, `B0`, `C`, `D`, `D0`.
- The `lrp.dim` field of the type `lrp_Sys _Cla_Dim` used for storing the model dimensions. Currently, it has the following fields:
  - `alpha` — (`positive integer`), the pass length (number of points on each pass), denoted by α in (1),
  - `beta` — (`positive integer`), the number of passes over which the simulation will run,
  - `n`, `r`, `m` — (`positive integers`), the dimensions of state, input and output vectors, respectively.
- The `lrp.ini` field of the type `lrp_Sys _Cla_Ini` used for storing the initial conditions. Currently, it has two fields.
- `x0`, `y0` — (`real matrices`), the boundary conditions, see (2). Note that $x_{k+1}(0)$ in (2) is here denoted by `x0` for simplicity.
- `controller` — (`list`), the list of known control laws for the process; see below.
- `indController` — (`positive integer`), the index of the current control law. This field contains the index of the currently active control law. If `indController=1`, then no control law is applied.
- `stability` — (`list`), the list of performed stability tests; see below.

Note that the model of (1) does not impose any constraints on the number of passes, and hence to simulate the process response it is necessary to bound it by some finite value selected by the user—hence the parameter `beta` in the `lrp` structure.

4.1. lrp.controller field. The `controller` field of the `lrp_Sys_Cla` datatype is a dynamically increasing `list` that contains a number of `tlist` structures. Each element holds the results of control law calculations. By design the first element of the `controller` list (i.e, `lrp.controller(1)`) is a copy of all the system matrices. This "controller" is necessary in order to retain the matrices defining the uncontrolled process.

When new control law matrices are computed, new `tlist` is added to the `lrp.controller` field. This field is defined by the user and the toolkit does not enforce any constraints on its structure. The only requirement is that the following three fields must be present:

- `functionName`, which holds the function name without the `.sci` suffix,
- `displayName`, which is used when the user-friendly description of the method is needed,
- `solutionExists` of the `boolean` type, which informs the user as whether or not it is possible to obtain control law matrices for the example under consideration by the design method being considered.

In order to introduce a new control law, the user must (i) give it a name (e.g., `controllerExample`) and (ii) implement a set of three functions and one `.tex` file:

- `controllerExample` (the same name as a control law name)—the main function used for calculating the (constant) control law matrices given a process state-space model. This approach allows faster calculations but also imposes an important drawback: it is not possible (without changes to the core

toolkit files) to simulate the controlled process to asses the effects of varying the control law matrix (K). (LMI designs produce a family of such K.) The addition of this feature is a subject for future work.

- `setcontrollerExample` (`set`+controller name)—changes the system matrices to those for the controlled process (e.g., replace the A matrix by $A+B\cdot K$, where K is the calculated (constant) control law matrix).
- `writecontrollerExample` (`write`+controller name)—a SCILAB function for exporting the results (e.g., controller matrices, parameters, etc.) to LaTeX; it can be blank if no export is required.
- `describecontrollerExample.tex` (`describe`+controller name)—an introductory text in LaTeX to be inserted when using the LaTeX export capabilities; it can be left blank.

It must be stressed that the toolkit files written by the user must be placed in the appropriate directories. This is explained in the help file. Note also that SCILAB enforces a maximum function name length of 25 characters.

4.2. lrp.stability field. The `stability` field of the `lrp_Sys_Cla` datatype is a dynamically increasing `list` that contains a number of `tlist` structures. Each element represents a performed stability test.

When the user checks a new stability condition, new `tlist` is added to the `lrp.stability` field. This field is defined by the programmer and the toolkit does not enforce any constraints on its structure. The only requirement is that there must be a field `solutionExists` of an integer type (note here the difference between `lrp.stability` and the `lrp.controller` field where a `boolean` type is used instead) which informs the user whether the system is stable, unstable or the method used is inconclusive. The `solutionExists` field can have the following values:

- -1 – the test is inconclusive. In this case the only alternative is to use another test.
- 0 – the process is unstable.
- 1 – the process is stable.

Obviously, SCILAB works in finite precision arithmetic and hence numerical errors can influence the results.

To implement a stability test, the user should write a function that returns an `lrp` structure and is given 1 as the only argument. Note here that a better solution would be to use a "by variable" (or by pointer) passing method, but this is not implemented in SCILAB. This function must be placed in the `stability` directory of the toolkit.

To save memory, the `lrp.stability` field is dynamically created. If the user does not complete any stability tests, then this field does not exist.

4.3. Boundary conditions. The boundary conditions of (2) can be entered either as a set of values or by providing a function that returns the appropriate value given the pass k and the point number p.

By default, if the user does not supply the boundary conditions, they are zero state vector on each pass and a unit step applied at $p = 0$ in each channel of the initial pass profile.

5. CoolMatrix type

One of the difficulties of implementing system theoretic results in the SCILAB code is that very often the matrices are indexed from 0 or even by a negative value (a case commonly encountered in the analysis of the so-called "wave" processes). SCILAB natively supports matrices and vectors indexed only from 1. While changing all the indices to start from 1 may seem simple, it can lead to many difficult-to-detect errors. To overcome this difficulty, the new version of the toolkit includes the $CoolMatrix$ matrix type that allows the user to index the matrices as required (from 0 or any other value, including negative numbers). The functions used for this type are designed for fast prototyping and hence provide a strict error checking—any attempt to use a wrong index causes an error. This type has been made compatible with the standard SCILAB matrix type.

The disadvantage of this addition to the toolkit is the overhead caused by this type. This is not very significant, as in the early/prototyping phase of a design exercise, small or medium dimensions will often be encountered. Moreover, experience has shown that the efficiency of the toolkit is greatly dependent on the LMI solver. Obviously, the user is not forced to utilise this data type; standard SCILAB types can be used without limitation.

An example of using CoolMatrix is given below.

```
--> x=new_mtx([-2,-3],[1 2; 3 4])
 x  =
is a CoolMatrix (-2..-1, -3..-2)
    1.    2.
    3.    4.

--> x(-2,-3:-2)
 ans  =
is a CoolMatrix (-2..-2, -3..-2)

    1.    2.

--> x(1,:)
 !--error 10000
Wrong index of dimension 1. Is: 1,
Allowed range: <-2, -1>, inclusive.
```

```
at line   64 of function %mtx_e called by :
x(1,:)
```

Here the symbol "−−>" denotes the SCILAB prompt. Note that the SCILAB notation of "extending" the size of the matrix when called with an index larger than its former size is not supported (see the above example), as this is very error prone.

6. Illustrating the toolkit in iterative learning control

Iterative learning control is a technique for controlling systems operating in a repetitive (or pass-to-pass) mode with the requirement that a reference trajectory $r_{\rm ref}(p)$ defined over a finite interval $0 \leq p \leq \alpha - 1$ be followed to high precision. Examples of such systems include robotic manipulators that are required to repeat a given task to high precision, chemical batch processes or, more generally, the class of tracking systems.

Since the original work (Arimoto *et al.*, 1984), the general area of ILC has been the subject of intense research effort. A possible initial source for the literature here is the survey paper (Ahn *et al.*, 2007). One approach in ILC is to construct the input to the plant or process from the input used on the last trial plus an additive increment which is typically a function of the past values of the measured output error, i.e., the difference between the achieved output on the current pass and the desired plant output. As such, it places the analysis of ILC schemes firmly outside the standard (or 1D) control theory—although it is still has a significant role to play in certain cases of practical interest.

In essence, ILC has the structure of a repetitive process and to illustrate some of the main features of the toolkit (as it currently stands) we now detail the design of an ILC control law based on an LMI setting and then give the results of performance assessment via simulation. The starting point is a process described by a 1D discrete linear time invariant state-space model of the form

$$\begin{aligned} x(p+1) &= Ax(p) + Bu(p), \\ y(p) &= Cx(p), \\ & 0 \leq p \leq \alpha - 1, \end{aligned} \tag{9}$$

where $x \in \mathbb{R}^n$ ($u \in \mathbb{R}^r$, $y \in \mathbb{R}^m$) denotes the state, input and output vectors, respectively. The initial state vector is taken as $x(0) = d_0$. In many practical cases, a physical process is required to repeat the same task over a finite duration. One example here is a gantry robot whose task is to place items on a moving conveyer under synchronization, i.e., to collect an object (or a workpiece) from a specified location and place it on the conveyer and then return to pick up the next one and so on. Once the operation has been completed for, say, the k-th item, then (in principle at least) all input, output and state dynamics generated during this and all previous trials are available for use to update the control input vector to be applied for the item $k+1$ and so on. This is the core ILC problem, i.e., use information from the previous trial (or trials) to update the control input vector applied from trial to trial in order to sequentially improve performance and, in particular, force the system to produce a desired trajectory, say $y_{\rm ref}(p)$, with "acceptable" dynamics along each trial.

To introduce the ILC setting, we use the integer subscript $k \geq 0$ to denote the current trial and rewrite the model of (9) as

$$\begin{aligned} x_k(p+1) &= Ax_k(p) + Bu_k(p), \\ y_k(p) &= Cx_k(p), \end{aligned} \tag{10}$$

and take the control objective to force the tracking error on the trial k as

$$e_k(p) = y_{\mathrm{ref}(p)} - y_k(p), \tag{11}$$

$p = 0, 1, \cdots, \alpha - 1, k \geq 0$, where $y_{\rm ref}(p)$ denotes the reference signal to be learnt. Then a known result is that, in its strongest form, the convergence of the ILC scheme is equivalent to the property of linear constant pass length repetitive processes known as stability along the pass. In the repetitive process case, this is equivalent to uniform bounded input bounded output stability (defined in terms of the norm on the underlying function space), i.e., independent of the pass length.

Introduce now the following additional variables, (i.e., the state increment from trial-to-trial $\eta_{k+1}(p)$ and the input update $\Delta u_{k+1}(p)$):

$$\begin{aligned} \eta_{k+1}(p+1) &= x_{k+1}(p) - x_k(p), \\ \Delta u_{k+1}(p) &= u_{k+1}(p) - u_k(p). \end{aligned} \tag{12}$$

Then

$$\eta_{k+1}(p+1) = A\eta_{k+1}(p) + B\Delta u_{k+1}(p-1), \tag{13}$$

and consider also a control law of the form

$$\Delta u_{k+1}(p) = K_1\eta_{k+1}(p+1) + K_2 e_k(p+1). \tag{14}$$

Then we can write the following state-space model for the error dynamics of the ILC scheme considered:

$$\begin{aligned} \eta_{k+1}(p+1) &= \hat{A}\eta_{k+1}(p) + \hat{B}_0 e_k(p), \\ e_{k+1}(p) &= \hat{C}\eta_{k+1}(p) + \hat{D}_0 e_k(p), \end{aligned} \tag{15}$$

where

$$\begin{aligned} \hat{A} &= A + BK_1, \\ \hat{B}_0 &= BK_2, \\ \hat{C} &= -C(A + BK_1), \\ \hat{D}_0 &= (I - CBK_2). \end{aligned}$$

Following the analysis in (Hładowski *et al.*, 2007), the solution of the ILC problem here is equivalent to control law design for stability along the pass in the repetitive process interpretation of the system dynamics. The following result is an LMI based solution to this problem.

Theorem 1. *The ILC problem considered here is stable along the pass if there exist matrices* $X_1 = X_1^T \succ 0$ *and* $X_2 = X_2^T \succ 0$, R_1 *and* R_2 *for which the following LMI is feasible:*

$$M = \begin{bmatrix} -X_1 & 0 \\ 0 & -X_2 \\ AX_1 + BR_1 & BR_2 \\ -CAX_1 - CBR_1 & X_2 - CBR_2 \\ X_1A^T + R_1^TB^T & -X_1A^TC^T - R_1^TB^TC^T \\ R_2^TB^T & X_2 - R_2^TB^TC^T \\ -X_1 & 0 \\ 0 & -X_2 \end{bmatrix} \prec 0.$$

If this condition holds, the control matrices K_1 *and* K_2 *in (14) are given by*

$$K_1 = R_1X_1^{-1},$$
$$K_2 = R_2X_2^{-1}.$$

As an example consider the gantry robot ILC experimental test facility described in (Ratcliffe *et al.*, 2006). For this facility, approximations for the dynamics of each axis have been obtained by frequency response tests and then approximation based on approximate Bode gain plots. If in the case of the X-axis the resulting model is sampled with a period $T_s = 0.02$ seconds, the resulting state-space model is of the form (15) with

$$\hat{A} = \begin{bmatrix} -0.24 & 1 & 0 & 0 & 0 \\ -0.0032 & -0.24 & -0.2 & -0.57 & -0.37 \\ 0 & 0 & -0.77 & 1 & 0 \\ 0 & 0 & -0.06 & -0.77 & -0.31 \\ 0 & 0 & 0 & 0 & -0.0088 \\ 0 & 0 & 0 & 0 & -0.62 \\ 0.18 & -0.65 & -0.14 & -0.054 & 0.5 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 \\ 0.77 & 0.85 \\ 0 & 0 \\ 0.64 & 0.7 \\ 1 & 0 \\ -0.0088 & 1 \\ 0.56 & -0.79 \end{bmatrix},$$

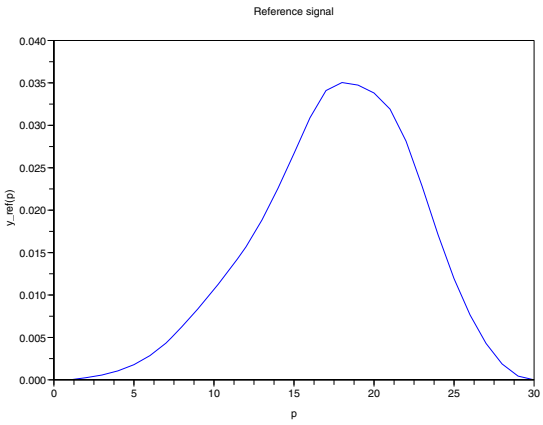

Fig. 4. Reference signal for the X-axis.

$$\hat{B}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.21 \end{bmatrix}, \quad \hat{D}_0 = \begin{bmatrix} 2.6 \cdot 10^{-15} \end{bmatrix},$$

$$\hat{C} = \begin{bmatrix} 6.3 \cdot 10^{-15} & -9.8 \cdot 10^{-15} & -1.1 \cdot 10^{-15} \\ 1.1 \cdot 10^{-15} & -8.9 \cdot 10^{-16} & 8.9 \cdot 10^{-15} \\ 7.5 \cdot 10^{-15} \end{bmatrix}.$$

The application of Theorem 1 now gives (with computations performed within the toolkit)

$$K_1 = \begin{bmatrix} 0.0228188 & -0.0816981 & -0.0171825 \\ -0.0066907 & 0.0629192 & 0.0697179 \\ -0.2238190 \end{bmatrix},$$
$$K_2 = 0.0259195.$$

In order to assess the performance of the resulting design, consider the reference signal shown in Fig. 4.

Simulating this process in the toolkit for $\alpha = 30$ points and $\beta = 5$ trials gives Fig. 5, from which it is clear that the ILC error has decayed to 0, i.e., the reference signal is achieved by this design.

7. Conclusions and future work

The SCILAB toolkit whose development was described in this paper has already proven useful in the analysis and control law design for discrete linear repetitive processes of the form considered here. Its basic functions include

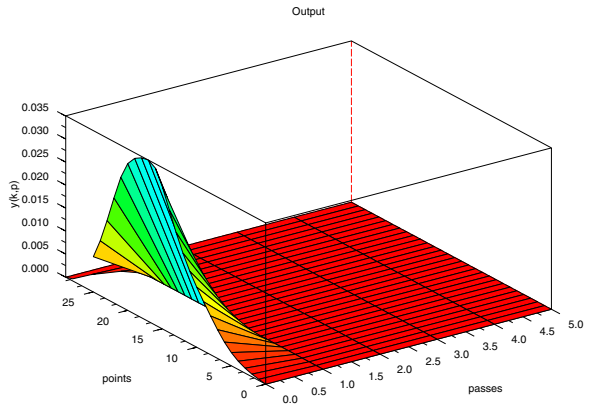

Fig. 5. Evolution of the ILC error.

the simulation engine and the stability analysis/control law design abilities. Moreover, the user is able to export the results to the valid LaTeX compatible format text file.

Software development for this toolkit is an ongoing process; there are many options which remain to be implemented. Representative samples of the ongoing development work include the following topics:

- support for new classes of processes (such as "wave" or semi-linear) and for a differential process model where the dynamics along the pass are governed by a linear matrix differential equation coupled with discretization methods,
- the addition of new control law design algorithms,
- control law design for stability and performance (e.g., ensuring that a reference signal is tracked, the minimization of the influence of the external disturbances),
- the handling of processes with uncertainty in the state-space model,
- an installer for the Linux/Unix version.

In some applications, it is necessary to use a pass state initial vector sequence which is an explicit function of points along the previous pass profile. It is known that these alone can cause the process to be asymptotically unstable (and hence unstable along the pass). There has been a considerable amount of research effort directed to this case, see (Rogers *et al.*, 2007), which has produced computable stability tests and also some algorithms for stabilizing control law design. Clearly, work is required to include these within the toolkit. Results of this work will be reported in due course.

References

Ahn H.-S., Chen Y. and Moore K. L. (2007). Iterative learning control: Brief survey and categorization, *IEEE Transactions on Systems, Man and Cybernetics, Part C* **37**(6): 1109–1121.

Arimoto S., Kawamura S. and Miyazaki F. (1984). Bettering operations of robots by learning, *Journal of Robotic Systems* **1**(2): 123–140.

Benton S. E. (2000). *Analysis and Control of Linear Repetitive Processes*, Ph. D. thesis, University of Southampton.

Boyd S., Ghaoui L. E., Feron E. and Balakrishnan V. (1994). *Linear Matrix Inequalities*, SIAM, Philadelphia, PA.

Campbell S., Chancelier J. and Nikoukhah R. (2006). *Modeling and Simulation in Scilab/Scicos*, Springer.

D'Andrea R. and Dullerud G. E. (2003). Distributed control design for spatially interconnected systems, *IEEE Transactions on Automatic Control* **48**(1): 1478–1495.

Gałkowski K., Lam J., Rogers E., Xu S., Sulikowski B., Paszke W. and Owens, D. H. (2003). LMI based stability analysis and robust controller design for discrete linear repetitive processes, *International Journal of Robust and Nonlinear Control* **13**(13): 1195–1211.

Gomez C., Delebecque F., Bunks C., Chancelier J.-P., Steer S. and Nikoukhah R. (Eds.) (1998). *Engineering and Scientific Computing with Scilab with CDROM*, Birkhäuser, Boston, MA.

Gramacki, J. (1999). *Stability Analysis Methods and Stabilization of Linear Discrete Repetitive Processes*, Ph. D. thesis, Technical University of Zielona Góra, (in Polish).

Gramacki J., Gramacki A., Gałkowski K. and Rogers E. (2005). Java based toolbox for linear repetitive processes, *Proceedings of the 2nd International Conference on Informatics in Control, Automation and Robotics, ICINCO* **1**: 182–187, Barcelona, Spain, available at: `http://www.uz.zgora.pl/~jgramack/LRP/lrp.html`.

Hładowski Ł., Cichy B., Gałkowski, K., Sulikowski B. and Rogers E. (2006). Scilab compatible software for analysis and control of repetitive processes, *Proceedings of the IEEE International Symposium on Computer-Aided Control Systems Design, CACSD 2006*, Munich, Germany, pp. 3024–3029.

Hładowski Ł., Gałkowski K. and Rogers E. (2007). A new iterative learning control scheme for linear time-varying discrete systems, *Proceedings of the 3rd IFAC Workshop PSYCO'07*, Saint Petersburg, Russia, (on CD-ROM).

Melkote H., Cloke B. and Agarwal V. (2003). Modeling and compensator designs for self-servowriting in disk drives, *Proceedings of the American Control Conference*, Denver, CO, USA, pp. 737–742.

Nikoukhah R., Delebecque F. and Ghaoui L. E. (2008). *LMITOOL: A Package for LMI Optimization in* SCILAB, available at: `http://www.scilab.org/doc/lmidoc/`

Nullsoft (2008). *NSIS Users Manual,* available at: http://nsis.sourceforge.net/Docs/

Owens D. H., Amann N., Rogers E. and French M. (2000). Analysis of linear iterative learning control schemes — A 2D systems/repetitive processes approach, *Multidimensional Systems and Signal Processing* **11**(1-2): 125–177.

"IGx89" Lieder, M. (2008). *MD5 plugin DLL,* derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm, available at: http://nsis.sourceforge.net/MD5_plugin

Rabenstein R. and Trautmann L. (2003). Towards a framework for continuous and discrete multidimensional systems, *International Journal of Applied Mathematics and Computer Science* **13**(1): 73–85.

Ratcliffe J. D., Hatonen J. J., Lewin P. L., Rogers E., Harte T. J. and Owens, D. H. (2005). P-type iterative learning control for systems that contain resonance, *International Journal of Adaptive Control and Signal Processing* **19**(10): 769–796.

Ratcliffe J. D., Lewin P. L., Rogers E., Hatonen J. J. and Owens D. H. (2006). Norm-optimal iterative learning control applied to gantry robots for automation applications, *IEEE Transactions on Robotics* **22**(6): 1303–1307.

Rogers E., Gałkowski K. and Owens D. H. (2007). *Control Systems Theory and Applications for Linear Repetitive Processes*, Springer, Berlin/Heidelberg.

Rogers E. and Owens D. H. (1992). *Stability Analysis for Linear Repetitive Processes*, Springer, New York, NY.

Scilab Group, I. M. P. C. (2008). *Introduction to* SCILAB, available at: http://www.scilab.org/doc/intro/intro.pdf

Sulikowski B. (2006). *Computational Aspects in Analysis and Synthesis of Repetitive Processes*, Ph. D. thesis, University of Zielona Góra.

Received: 17 April 2007
Revised: 30 July 2007
Re-revised: 11 May 2008