Working with Time-Delay Systems in MATLAB®

Suat Gumussoy [*], **Bora Eryilmaz** [*], **Pascal Gahinet** [*],

[*] *The MathWorks,*
3 Apple Hill, Natick, MA, 01760,
(e-mails: {suat.gumussoy, bora.eryilmaz, pascal.gahinet}@mathworks.com).

Abstract: This paper presents the functionality of Control System Toolbox in MATLAB regarding systems with time-delays. The toolbox allows the user to define their systems in different system representations and to make complex system interconnections. The user can analyze the overall system in time and frequency domains with different tools and design PID controllers satisfying design requirements. Various visualization tools are available for analysis and design verification. This paper aims to introduce these functionalities to researchers and engineers and to discuss the open directions in computer algorithms for control system design.

Keywords: time delay systems, representation, interconnection, analysis, controller design, control system toolbox, matlab.

1. INTRODUCTION

Time delays frequently appear in many control applications such as process control, communication networks, automotive and aerospace, Dugard and Verriest (1998); Niculescu (2001); Erneux (2009). Depending on the delay length, they may limit or degrade the performance of control systems unless they are considered in the design, Gu et al. (2003); Atay (2010). Although considerable research effort is devoted to extend classical and modern control techniques to accommodate delays, most available packages for delay differential equations (DDE) Hairer and Wanner (1995); Enright and Hayashi (1997); Engelborghs et al. (2001); Breda et al. (2009) are restrictive and not developed for control design purposes.

This paper presents the currently implemented framework and available functionality in MATLAB for computer-aided manipulation of linear time-invariant (LTI) models with delays. It reviews important steps in every practical control design: system representation, interconnections, analysis and design techniques for time-delay systems. Our goal is to introduce available functionality in Control System Toolbox to researchers and engineers to facilitate the design of control systems with delays and by discussing possible enhancements, to illustrate the gap between the desired analysis / design techniques and the current control software implementation.

At the heart of this framework is an LFT-based representation of such systems (LFT stands for linear fractional transformation, see Skogestad and Postlethwaite (1996) and references therein). This representation handles delays in feedback loops and is general enough for most control applications. In addition most classical software tools for analyzing delay-free LTI systems are extended to this class of LTI systems with delays. Given the widespread use of linear techniques in control system design, this framework and the accompanying software tools should facilitate the computer-aided analysis and design of control systems in the presence of delays, as well as stimulate more research into efficient numerical algorithms for assessing the properties and performance of such systems.

The paper is organized as follows. First we define the class of delayed LTI systems under consideration which covers most practical needs in control applications. We discuss how to connect various time-delay systems to obtain the closed-loop system with plants and controllers. We overview analysis and visualization tools in time and frequency domains which are used to understand the closed-loop system behavior and properties. We demonstrate controller design methods with and without delay approximation. Finally, we discuss possible enhancements for Control System Toolbox regarding time-delay systems.

2. MOTIVATION

A standard PI control example is given in Ingimundarson and Hagglund (2001) where the plant is a chemical tank and a single-input-single-output system with an input-output (I/O) delay (i.e., dead-time system),

$$P(s) = e^{-93.9s}\frac{5.6}{40.2s+1}. \tag{1}$$

In the classical feedback configuration in Figure 1, the standard PI controller is chosen as

$$C(s) = K(1 + \frac{1}{T_i s}) \tag{2}$$

where $K = 0.1$ and $T_i = 100$. The closed-loop transfer function from y_{sp} to y is

$$T_{PI}(s) = \frac{4020s^2 + 100s}{4020s^2 + 100s + (56s + 0.56)e^{-93.9s}}.$$

Note that this transfer function has an *internal* delay which can not be represented by input or output delays. Therefore, the representation for time-delay systems has to capture this type of systems and to be *closed* under block diagram of operations.

$$\frac{dx}{dt} = x(t) - x(t-1.2) + 2u(t-0.5) \tag{3}$$
$$y(t) = x(t-0.5) + u(t) \tag{4}$$
$$\tag{5}$$

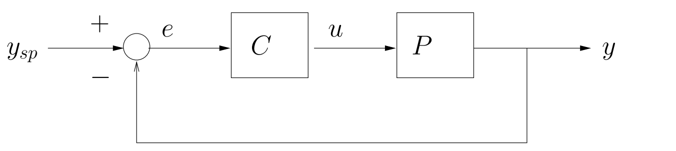

Fig. 1. PI Control Loop.

This plant and the controller in MATLAB are defined as

```
P = tf(5.6,[40.2 1],'OutputDelay',93.9); % plant
C = 0.1 * (1 + tf(1,[100 0]));           % PI controller
```

and the closed-loop system T_{PI} is obtained by the `feedback` command:

```
% Closed-loop transfer, ysp -> y
Tpi = feedback(P*C,1);
```

Note that the commands for systems with delays are natural extensions of delay-free case.

The MIMO time-delay systems may have different transport delays for each input-output (I/O), i.e.,

$$H(s) = \begin{pmatrix} e^{-0.1s}\frac{2}{s} & e^{-0.3s}\frac{s+1}{s+10} \\ 10 & e^{-0.2s}\frac{s-1}{s+5} \end{pmatrix}$$

which is defined in MATLAB by the following commands:

```
s = tf('s');
H = [2/s (s+1)/(s+10); 10 (s-1)/(s+5)];
H.ioDelay = [0.1 0.3; 0 0.2];
```

The representation for time-delay systems has to address these challenges. Next section presents the LFT-based representation of time-delay systems and discusses its advantages.

3. SYSTEM REPRESENTATION

The time-delay system is represented by the linear-fractional transformation (LFT). Recall that the LFT is defined for matrices by

$$\mathcal{L}(\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}, \Theta) := M_{11} + M_{12}\Theta(I - M_{22}\Theta)^{-1}M_{21} \; .$$

The LFT has been extensively used in robust control theory for representing models with uncertainty, see Skogestad and Postlethwaite (1996) for details.

Consider the class *generalized LTI* (GLTI) of continuous-time LTI systems whose transfer function is of the form

$$H(s,\tau) = \mathcal{L}(\underbrace{\begin{pmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{pmatrix}}_{H(s)}, \Theta(s,\tau))$$
$$\Theta(s,\tau) := \text{Diag}\left(e^{-\tau_1 s}, \ldots, e^{-\tau_N s}\right) \tag{6}$$

where $H(s)$ is a rational (delay free) MIMO transfer function, and $\tau = (\tau_1, \ldots, \tau_N)$ is a vector of nonnegative time delays. Systems in this class are modeled as the LFT

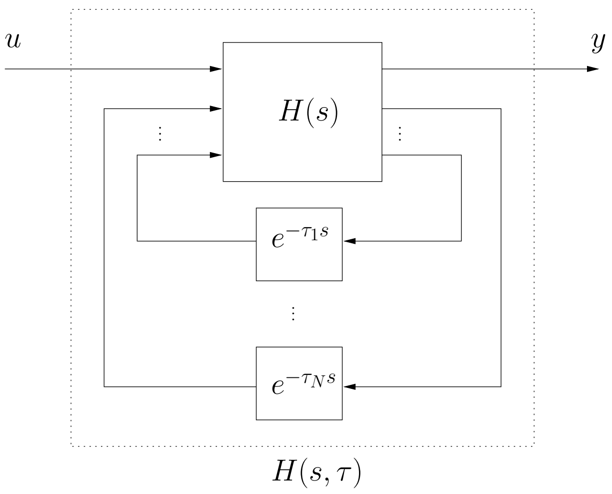

Fig. 2. LFT-based modeling of LTI systems with delays.

interconnection of a delay-free LTI model and a bank of pure delays (see Figure 2). As such, they are clearly linear time-invariant. Also, pure delays are in this class since $e^{-\tau s} = \mathcal{L}(\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, e^{-\tau s})$.

This GLTI class has two key properties, Gahinet and Shampine (2004):

- Any block diagram interconnection of GLTI systems is a GLTI system. In other words, the class of GLTI systems is closed under series, parallel, and feedback connections as well as branching/summing junctions.
- The linearization of any nonlinear block diagram with time delays is a GLTI system.

These two properties show that the GLTI class is general enough to model any (linearized) system with a finite number of delays, including delays in the feedback path. For further motivation of this representation and equivalent case of discrete time systems, see Gahinet and Shampine (2004).

The GLTI class is represented in state-space equations as follows. Let

$$\begin{pmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{pmatrix} = \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} + \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} (sI - A)^{-1} \begin{pmatrix} B_1 & B_2 \end{pmatrix}$$

be a minimal realization of $H(s)$ in (6). State-space equations for $H(s,\tau) = \mathcal{L}(H(s), \Theta(s,\tau))$ are readily obtained as

$$\frac{dx}{dt} = Ax(t) + B_1u(t) + B_2w(t) \tag{7}$$
$$y(t) = C_1x(t) + D_{11}u(t) + D_{12}w(t) \tag{8}$$
$$z(t) = C_2x(t) + D_{21}u(t) + D_{22}w(t) \tag{9}$$
$$w(t) = (\Delta_\tau z)(t) \tag{10}$$

where

- $u(t)$ and $y(t)$ are the input and output vectors, respectively
- $w(t)$ and $z(t)$ are internal signals commensurate with the vector τ of time delays
- $\Delta_\tau z$ is the vector-valued signal defined by $(\Delta_\tau z)(t) := (z_1^T(t-\tau_1), \ldots, z_N^T(t-\tau_N))^T$.

Note that standard delay-free state-space models are just a special case of (7)-(10) corresponding to $N = 0$, a handy fact when it comes to integrating GLTI models with existing software for manipulating delay-free state-space models.

Delay LTI systems of the form

$$\frac{dx}{dt} = A_0x(t) + B_0u(t) + \sum_{j=1}^{M}(A_jx(t-\theta_j) + B_ju(t-\theta_j))$$

$$y = C_0x(t) + D_0u(t) + \sum_{j=1}^{M}(C_jx(t-\theta_j) + D_ju(t-\theta_j))$$

are often considered in the literature with various restrictions on the number and locations of the delays $\theta_1, \ldots, \theta_M$. It turns out that any model of this form belongs to the class GLTI as shown in Gahinet and Shampine (2004). It is possible to define a large class of time-delay systems in MATLAB, both in time and frequency domains. For further details on representation of time-delay systems, see Control System Toolbox (2011).

4. INTERCONNECTIONS

Control systems, in general, are built up by interconnecting other subsystems. The most typical configuration is a feedback loop with a plant and a controller as shown in Section 2; whereas more complex configurations may have distributed systems with multiple plants, controllers and transport / internal delays.

A standard way to to obtain the closed-loop model of interconnections of systems in MATLAB is to use the `connect` command. This function requires all systems to have input and output names and summation blocks. It automatically builds the resulting closed-loop system with the given inputs and outputs. Consider the Smith Predictor control structure given in Figure 3 for the same dead-time system $P(s)$ in (1). The Smith Predictor uses an internal model to predict the delay-free response $y_p(t)$ of the plant, and seeks to correct discrepancies between this prediction and the setpoint $y_{sp}(t)$, rather than between the delayed output measurement $y(t)$ and $y_{sp}(t)$. To prevent drifting, an additional compensator $F(s)$ is used to eliminate steady-state drifts and disturbance-induced offsets.

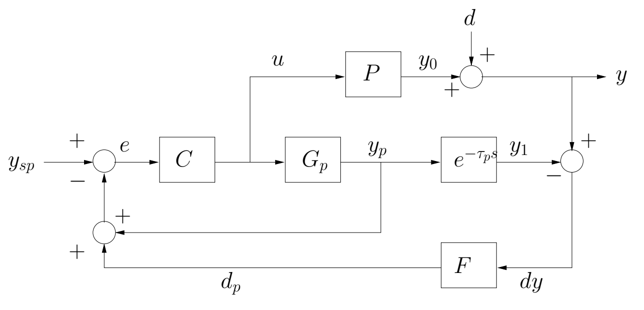

Fig. 3. Smith Predictor.

We first assume that the prediction model $P_p(s) = e^{-\tau_p s}G_p(s)$ matches the plant model $P(s)$ in (1), and use the following compensator settings:

$$C(s) = 0.5(1 + \frac{1}{40s}), \quad F(s) = \frac{1}{20s+1}.$$

By defining summation blocks and input and outputs names of systems, we obtain the closed-loop model T_{SP} from the input signal y_{sp} to the output signal y:

```
s = tf('s');

% LTI blocks
P = exp(-93.9*s) * 5.6/(40.2*s+1);
P.InputName = 'u'; P.OutputName = 'y';

Gp = 5.6/(40.2*s+1);
Gp.InputName = 'u'; Gp.OutputName = 'yp';

Dp = exp(-93.9*s);
Dp.InputName = 'yp'; Dp.OutputName = 'y1';

C = 0.5 * (1 + 1/(40*s));
C.InputName = 'e';  C.OutputName = 'u';

F = 1/(20*s+1);
F.InputName = 'dy'; F.OutputName = 'dp';

% Sum blocks
Sum1 = ss([1,-1,-1],'InputName',...
       {'ysp','yp','dp'},'OutputName','e');
Sum2 = ss([1,-1],...
       'InputN',{'y','y1'},'OutputN','dy');

% Build interconnection model
Tsp = connect(P,Gp,Dp,C,F,Sum1,Sum2,'ysp','y');
```

Alternatively, various types of connections can be constructed such as connecting in parallel and series (`parallel` and `series`); grouping systems by appending their inputs and outputs (`append`); forming the linear fractional transformation (`lft`). Note that standard system operations are also valid for time-delay systems such as addition, subtraction, multiplication, division.

By defining and connecting systems, we easily construct the closed-loop model with time-delays. Our next goal is to analyze the characteristics of the resulting closed-loop models with visualizations and calculate some of system properties.

5. TIME / FREQUENCY DOMAIN ANALYSES AND VISUALIZATIONS

The analysis of the plant, either a single system or a closed-loop model with various interconnections and systems, is important to understand the characteristics and some properties of the overall system. The simulation of the time-domain response of the plant to certain inputs such as a step or tracking signals allows us to observe some time-domain characteristics, i.e., rise and settling times, overshoot. On the other hand, frequency domain analysis gives information on, for example, gain and phase margins, bandwidth and resonant peak.

Given the dead-time plant in the previous section, the responses of the closed-loop models with the Smith Predictor and PI to the tracking signal, `ref` are obtained by the following commands and the resulting plot is given in Figure 4.

```
% time and reference signal
time = 0:.1:2000;
ref = (time>=0 & time<1000)*4 + (time>=1000 & time<=2000)*8;
```

```
% compare responses
lsim(Tsp,Tpi,ref,time);
```

Simulation results show that PI controller has a slower response time with oscillations and the Smith Predictor has better tracking performance.

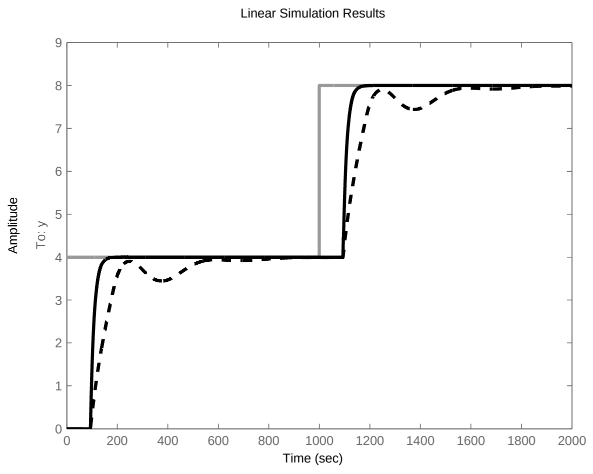

Fig. 4. Responses of the Smith Predictor (–) and PI (- -) to the reference signal (gray colored).

Robustness to mismatch between the prediction and plant models is easily investigated with these tools. For example, consider two perturbed plant models

$$P_1(s) = e^{-90s}\frac{5}{38s+1}, \;\; P_2(s) = e^{-100s}\frac{6}{42s+1} \; .$$

To assess the Smith predictor robustness when the true plant model is $P_1(s)$ or $P_2(s)$ rather than the prediction model $P(s)$, simply bundle P, P_1, P_2 into an LTI array, rebuild the closed-loop model(s), and replot the response for the tracking signal:

```
P1 = exp(-90*s) * 5/(38*s+1);
P2 = exp(-100*s) * 6/(42*s+1);
Plants = stack(1,P,P1,P2);
T = connect(Plants,Gp,Dp,C,F,Sum1,Sum2,'ysp','y');

lsim(T,Tpi,ref,time);
```

The resulting plot in Figure 5 shows a slight performance degradation, but the Smith predictor still retains an edge over the pure PI design.

The closed-loop frequency response for the nominal and perturbed plants is obtained by `bode(T)` and shown in Figure 6. Note that the phase behavior of systems with internal delays is quite different than systems with I / O delays.

The bandwidth of the responses is computed numerically by `bandwidth(T)` which returns 0.0695, 0.0565, 0.0767. The gain and phase margins of the responses are calculated by `[gm, pm] = margin(T)` and their values are

$$\mathrm{gm} = [1.0838 \;\; 1.1570 \;\; 1.0317], \mathrm{pm} = [180 \;\; 180 \;\; 7.8858].$$

Other frequency-domain based tools are also available for the GLTI class such as `bandwidth`, `dcgain`, `nyquist`, `allmargin`.

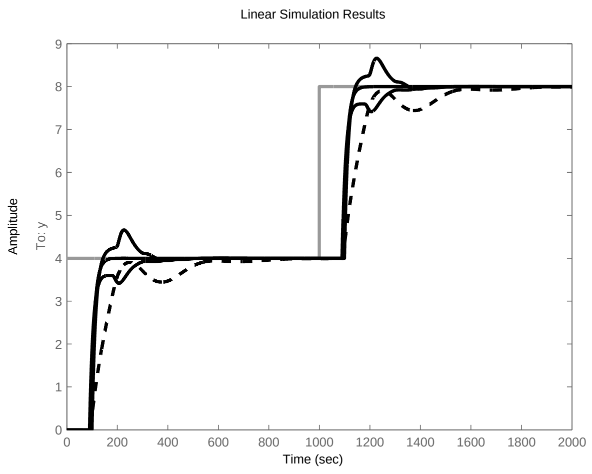

Fig. 5. Robustness of the Smith Predictor (–) to Model Mismatch

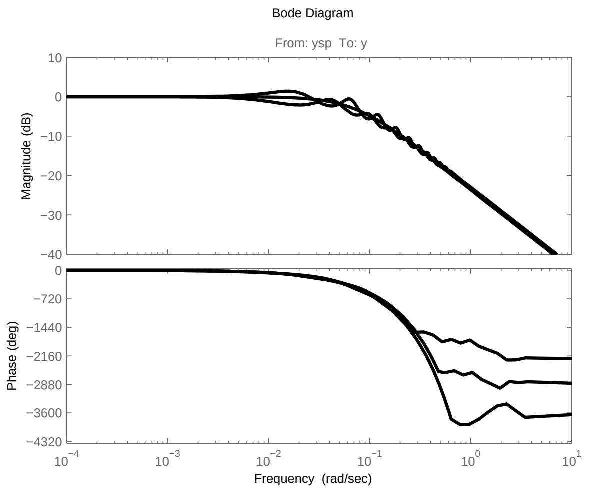

Fig. 6. Closed-Loop Response from y_{sp} to y.

6. CONTROLLER DESIGN

The Control System Toolbox allows user to design a PID controller for finite dimensional and time-delay plants using `pidtune` function.

This function aims to satisfy different PID objectives such as

- closed-loop stability,
- adequate performance by tracking reference changes and suppressing disturbances as rapidly as possible,
- adequate robustness by designing enough phase and margins for modeling errors or variations in system dynamics.

The algorithm for tuning PID controllers helps you meet these objectives by automatically tuning the PID gains to balance performance (response time) and robustness (stability margins). By default, the algorithm chooses a crossover frequency (loop bandwidth) based upon the plant dynamics, and designs for a target phase margin of 60°.

The finite dimensional transfer function $P_a(s)$ of the dead-time system $P(s)$ can be obtained by Padé approximation. Using `pidtune`, a PID controller $C_a(s)$ is designed for the approximate finite dimensional plant by the following commands:

```
Pa = pade(P,8);          % approximate 8th order plant
Ca = pidtune(Pa,'pid'); % design PID for Pa
Ta = feedback(P*Ca,1);  % closed-loop for Ca
```

The function `pidtune` also designs PID controller for time-delay systems without any approximation,

```
Cpid = pidtune(P,'pid');    % design PID for P
Tpid = feedback(P*Cpid,1); % closed-loop for Cpid
```

The closed-loop responses of the Smith Predictor, the designed PI controllers for the approximate plant $P_a(s)$ and the original plant $P(s)$ are obtained by

```
lsim(Tsp,Ta,Tpid,ref,time);
```

and shown in Figure 7. The simulation result shows that the designed PID controller for the original plant offers a good compromise between the simplicity of the controller and good tracking performance compared to the Smith predictor.

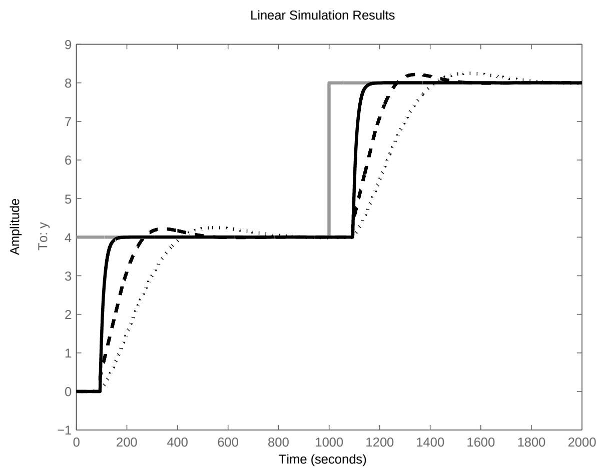

Fig. 7. Responses of the Smith Predictor (–), PID (- -) for $P(s)$ and PID (:) for $P_a(s)$ to the reference signal (gray colored).

7. POSSIBLE ENHANCEMENTS IN CONTROL SYSTEM TOOLBOX

In this section, we discuss possible enhancements for Control System Toolbox regarding time-delay systems. Most of the functions in Control System Toolbox are extended for time-delay systems. There is an on-going research on numerical methods for

- deciding the *stability* of a time-delay system,
- computation of system *poles* and *zeros*,
- computation of $\mathcal{H}_\infty$ and $\mathcal{H}_2$ *norms* of a time-delay system.

There are various numerical methods to determine the stability of LTI systems with constant delays Dugard and Verriest (1998); Gu et al. (2003); Michiels and Niculescu (2007) via detection of characteristic roots through imaginary axis; approximating the right-most characteristic roots; using Lyapunov theory. Each method has different pitfalls. Some of them are restricted to the special type of time-delay systems, i.e., retarded type, quasi-polynomials; while others are conservative or they are based on some heuristic methods with possible failing points. Most methods can not handle time-delay systems with high orders.

The right-most system poles and zeros of an time-delay system can be computed by approximating the spectrum of the time-delay system. The computations are based on either discretization of the solution operator of a delay differential equation or the infinitesimal generator of the solution operator semigroup. The solution operator approach by linear-multi-step time integration for retarded type delay differential equations is given in Engelborghs and Roose (2002); Engelborghs et al. (2001). The infinitesimal generator approach discretizes the derivative in abstract delay differential equation by Runge-Kutta or linear multi-step methods and approximates into a matrix Breda et al. (2005, 2009) for retarded type delay differential equations with multiple discrete and distributed delays. Extensions to neutral type delay differential equations and mixed-type functional differential equations are done in Breda et al. (2006). Numerically stable implementation of spectral methods with some heuristics is given in Wu and Michiels (2011). The computation of system poles and zeros is closely connected with the nonlinear eigenvalue problem and an eigenvalue algorithm for this is presented in Jarlebring et al. (2011b). A numerical method to compute all characteristic roots of a retarded or neutral quasi-polynomial on a large region in the complex-plane is proposed in Vyhlídal and Zítek (2003). The characteristic roots are calculated by finding the intersection of real and imaginary part of the characteristic equation on certain regions in complex-plane. This approach is further improved and accelerated by removing the regions outside of asymptotic chain roots in Vyhlídal and Zítek (2009). These methods consider the transfer function representation of delay differential equations which can be written as a ratio of quasi-polynomials. As noted in Vyhlídal and Zítek (2009), when delay differential equations have state-space representations, transforming these systems into transfer function representation is not numerically desired, therefore in this case discretization approaches may be preferred.

The computation of $\mathcal{H}_\infty$ norm of a time-delay system requires solving a nonlinear eigenvalue problem where the recent developments are applicable, Michiels and Gumussoy (2010). The computation of $\mathcal{H}_2$ norm requires solving the delay Lyapunov equation, Jarlebring et al. (2011c).

Another challenging task is the compensator design for time-delay systems and extension of classical control methods to time-delay systems such as

- LQG,
- $\mathcal{H}_2$ control,
- $\mathcal{H}_\infty$ control,
- root-locus technique,
- model reduction methods.

There are continuing research efforts to solve these problems such as Vanbiervliet et al. (2011); Gumussoy and Michiels (2011, 2012); Jarlebring et al. (2011a). The remaining main task is to determine numerically stable algorithms to solve control design problems for high dimensional plant with the least user interactions.

8. CONCLUDING REMARKS

We have shown that the GLTI class is suitable for computer-aided manipulation. We discussed various representations and interconnections of time-delay systems on MATLAB. We presented the functionality to do analysis and design of control systems with delays, regardless of the control structure and number of delays. Most Control System Toolbox functions have been extended to work on GLTI models, all this without additional complexity or new syntax for the user. We hope that these new tools will facilitate the design of control systems with delays and bring new insights into their behavior.

REFERENCES

Atay, F.M. (ed.) (2010). *Complex Time-Delay Systems: Theory and Applications*. Understanding Complex Systems. Springer.

Breda, D., Maset, S., and Vermiglio, R. (2005). Pseudospectral differencing methods for characteristic roots of delay differential equations. *SIAM Journal on Scientific Computing*, 27(2), 482–495.

Breda, D., Maset, S., and Vermiglio, R. (2006). Pseudospectral approximation of eigenvalues of derivative operators with non-local boundary conditions. *Applied Numerical Mathematics*, 56(3-4), 318–331.

Breda, D., Maset, S., and Vermiglio, R. (2009). *TRACE-DDE: a Tool for Robust Analysis and Characteristic Equations for Delay Differential Equations*, volume 388 of *Lecture Notes in Control and Information Sciences*. Springer.

Control System Toolbox (2011). MathWorks Inc., Natick.

Dugard, L. and Verriest, E. (eds.) (1998). *Stability and control of time-delay systems*, volume 228 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag.

Engelborghs, K., Luzyanina, T., and Samaey, G. (2001). DDE-BIFTOOL V. 2.00: a Matlab Package for Bifurcation Analysis of Delay Differential Equations. Technical Report TW330, Department of Computer Science, K. U. Leuven, Leuven, Belgium.

Engelborghs, K. and Roose, D. (2002). On stability of LMS methods and characteristic roots of delay differential equations. *SIAM Journal on Numerical Analysis*, 40(2), 629–650.

Enright, W.H. and Hayashi, H. (1997). A delay differential equation solver based on a continuous rungekutta method with defect control. *Numerical Algorithms*, 16, 349–364.

Erneux, T. (2009). *Applied delay differential equations*. Surveys and tutorials in the applied mathematical sciences. Springer.

Gahinet, P. and Shampine, L.F. (2004). Software for modeling and analysis of linear systems with delays. In *Proceedings of the American Control Conference*.

Gu, K., Kharitonov, V., and Chen, J. (2003). *Stability of time-delay systems*. Birkhuser, Boston, MA.

Gumussoy, S. and Michiels, W. (2011). Fixed-Order H-infinity Control for Interconnected Systems using Delay Differential Algebraic Equations. *SIAM Journal on Control and Optimization*, 49(2), 2212–2238.

Gumussoy, S. and Michiels, W. (2012). Root Locus for SISO Dead-Time Systems: A Continuation Based Approach. *In Press, Automatica*.

Hairer, E. and Wanner, G. (1995). RETARD: Software for delay differential equations. `unige.ch/ hairer/software.html`.

Ingimundarson, A. and Hagglund, T. (2001). Robust tuning procedures of dead-time compensating controllers. *Control Engineering Practice*, 9, 1195–1208.

Jarlebring, E., Damm, T., and Michiels, W. (2011a). Model reduction of time-delay systems using position balancing and delay Lyapunov equations. Technical Report TW602, Department of Computer Science, K. U. Leuven, Leuven, Belgium.

Jarlebring, E., Michiels, W., and Meerbergen, K. (2011b). A linear eigenvalue algorithm for the nonlinear eigenvalue problem. Technical Report TW580, Department of Computer Science, K. U. Leuven, Leuven, Belgium.

Jarlebring, E., Vanbiervliet, J., and Michiels, W. (2011c). Characterizing and computing the L2 norm of time-delay systems by solving the delay Lyapunov equation. *IEEE Transactions on Automatic Control*, 56, 814–825.

Michiels, W. and Gumussoy, S. (2010). Characterization and computation of H-infinity norms of time-delay systems. *SIAM Journal on Matrix Analysis and Applications*, 31(4), 2093–2115.

Michiels, W. and Niculescu, S.I. (2007). *Stability and stabilization of time-delay systems. An eigenvalue based approach*, volume 12 of *Advances in design and control*. SIAM, Philadelphia.

Niculescu, S.I. (2001). *Delay effects on stability: A robust control approach*, volume 269 of *Lecture notes in control and information sciences*. Springer-Verlag, London.

Skogestad, S. and Postlethwaite, I. (1996). *Multivariable Feedback Control*. John Wiley.

Vanbiervliet, J., Michiels, W., and Jarlebring, E. (2011). Using spectral discretization for the optimal H2 design of time-delay systems. *International Journal of Control*, 84(2), 228–241.

Vyhlídal, T. and Zítek, P. (2003). Quasipolynomial mapping based rootfinder for analysis of time delay systems. In *Proceedings of the 4th IFAC workshop on Time Delay Systems*, 227–232. Rocquencourt, France.

Vyhlídal, T. and Zítek, P. (2009). Mapping based algorithm for large-scale computation of quasi-polynomial zeros. *IEEE Transactions on Automatic Control*, 54(1), 171–177.

Wu, Z. and Michiels, W. (2011). Reliably computing all characteristic roots of delay differential equations in a given right half plane using a spectral method. Technical Report TW596, Department of Computer Science, K. U. Leuven, Leuven, Belgium.