COMPUTING POLES AND ZEROS OF TIME DELAY SYSTEM BY CHARACTERISTIC FUNCTION MAPPING

Tomáš Vyhlídal & Pavel Zítek

Centre for Applied Cybernetics, Inst. of Instrumentation and Control Eng., Czech Technical University in Prague,
vyhlidal@fsid.cvut.cz, zitek@fsid.cvut.cz

Abstract: Two original algorithms for computing the roots of low order quasipolynomials are introduced in the paper. The first algorithm is based on Weyl's construction combined with argument principle rule. The second algorithm is based on quasipolynomial function mapping in the complex plane. Both algorithms provide the approximate positions of the roots located inside a selected region. The prime application of designed quasipolynomial rootfinders is in computing poles and zeros of time delay systems.

Keywords: time delay system, system poles and zeros, characteristic function, quasipolynomial roots,

1. POLES AND ZEROS OF TIME DELAY SYSTEMS

Description of plants by means of models with time delays has become common in control engineering. Unlike the classical system description, this kind of models allows describing the basic features of real systems, i.e., transport phenomenon, after-effects or distributed parameters without necessity of using higher order models, see, e.g., Zítek (1998). Let us consider the general form of linear SISO time delay system (TDS) based on the Stieltjes integrals (this special form of integrals allow to involve both lumped and distributed delays into one convolution formula)

$$\begin{aligned} &\frac{d}{dt}\left[\mathbf{x}(t)+\int_0^T d\mathbf{D}(\tau)\mathbf{x}(t-\tau)\right]=\int_0^T d\mathbf{A}(\tau)\mathbf{x}(t-\tau)+\int_0^T d\mathbf{B}(\tau)u(t-\tau), \\ &y(t)=\mathbf{C}\mathbf{x}(t) \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the vector of system state variables, u and y are the system input and system output, respectively. The matrices $\mathbf{A}(\tau)$, $\mathbf{B}(\tau)$ and $\mathbf{D}(\tau)$ are functional matrices of delay distribution with the upper bound T. $\mathbf{C}$ is coefficient matrix as a rule. Unlike the state of the classical system with the coefficient matrices $\mathbf{A}$ and $\mathbf{B}$, which is given by the actual values of the state variables, the state of system (1) is given by the trajectories of the state variables on the interval $[t\text{-}T, t]$, see Zítek (1998), Diekman, et al, (1995). If $\mathbf{D}(\tau)=\mathbf{0}$ the system is

called retarded while if $\mathbf{D}(\tau) \neq \mathbf{0}$ the system is called neutral (Hale and Verduyn Lunel, 1993). The retarded systems are more common in control engineering than neutral systems and their theory is more developed. The retarded systems are used for describing plants with after-effects, latencies, transportation phenomenon or distributed parameters, e.g., heat-transfer or chemical processes. Using the delay-less systems for description of such plants usually results in high order models with many artificial state variables (unmeasured, often without physical meaning). Involving the delays in the model results in considerably lower order model with the state variables often identical with the available system outputs. Neutral systems are largely used for describing lossless propagation phenomena, see Niculescu, (2001), which is encountered, e.g., in modelling of distributed networks.

In case of zero initial conditions the Laplace transform of (1) can be expressed in the following form

$$\begin{aligned} s\left[\mathbf{I}+\mathbf{D}(s)\right]\mathbf{x}(s) &= \mathbf{A}(s)\mathbf{x}(s)+\mathbf{B}(s)u(s), \\ y(s) &= \mathbf{C}\mathbf{x}(s) \end{aligned} \tag{2}$$

where

$$\mathbf{A}(s)=\int_0^T \exp(-s\tau)d\mathbf{A}(\tau),\quad \mathbf{B}(s)=\int_0^T \exp(-s\tau)d\mathbf{B}(\tau),\quad \mathbf{D}(s)=\int_0^T \exp(-s\tau)d\mathbf{D}(\tau)$$

Laplace form (2) is advantageous, since it allows to use equivalent matrix operations like in the case of classical system description. System formulation (2) can be transformed into input-output transfer function

$$G(s)=\frac{y(s)}{u(s)}=\frac{N(s)d(s)}{M(s)} \tag{3}$$

where

$$N(s)=\frac{1}{d(s)}\mathbf{C}\,\mathrm{adj}\left\{s[\mathbf{I}+\mathbf{D}(s)]-\mathbf{A}(s)\right\}\mathbf{B}(s) \tag{4}$$

$$M(s)=\det\left\{s[\mathbf{I}+\mathbf{D}(s)]-\mathbf{A}(s)\right\} \tag{5}$$

are quasipolynomials as a rule and term d(s) represents the distribution of system input delay. Analogously to the case of classical systems the poles and zeros of TDS determine the features of the system dynamics. The system poles are the solutions of the characteristic equation

$$M(s)=\sum_{i=0}^{n} s^i Q_i(s)=0 \tag{6}$$

where n is system order and $Q_i(s)$, i=1..n are the delay distribution terms of quasipolynomial M(s). The system zeros are the solutions of the equation

$$N(s)=\sum_{i=0}^{m} s^i P_i(s)=0 \tag{7}$$

where $m \leq n$, and $P_i(s)$, i=1..m are the delay distribution terms of quasipolynomial N(s). Unlike delay free systems, TDS has infinitely many poles because the characteristic equation

(6) is transcendental. The number of zeros depends on the form of $N(s)$, which does not have to be quasipolynomial as a rule (depending on the matrices $\mathbf{B}(\tau)$ and $\mathbf{C}$). If at least one of the terms $P_i(s)$ is not equal to zero, equation (7) is also transcendental with infinitely many roots.

Provided that no zero-pole cancellation occurs, every pole generates a natural mode in the system response. Even though the numbers of poles and zeros are infinite, only few of them are decisive in the system dynamics. As regards the system poles, their roles in the system dynamics depend on the distances of the poles from the imaginary axis and from the *s*-plane origin. The dominant poles are those located in the closest positions to the *s*-plane origin and to the stability boundary. Nevertheless, it is difficult to claim either the distance from the origin or the distance from the imaginary axis is more important in determining the particular pole significance. An alternative method for evaluating pole dominance based on residue evaluation has been proposed in Zítek and Vyhlídal, (2002). Anyway, if at least one pole is located to the right from the stability boundary, the system is unstable. As regards the system zeros, the analogous evaluation to the system poles can be done. The most important zeros from the infinite set are those located in the closest positions to the stability boundary. The sign of the zeros is also important, however, not from the stability point of view. The zeros with plus sign are called nonminimum-phase zeros and they are responsible for the undershoots in the system responses.

The distribution of system poles differs significantly with respect to the character of TDS. If the system is retarded, the number of poles located to the left from the vertical line drawn in any real α is always finite, see Hale and Verduyn Lunel, (1993). The poles of a retarded system are usually distributed as a finite number of chains asymptotically departing to the upper-left direction with increasing distance of the poles from the *s*-plane origin. The poles of neutral systems are distributed within the lines drawn in some specific real α and β. The poles of a neutral system are also distributed as a finite number of asymptotic chains. However, these chains asymptote to the chains of essential spectrum, see , e.g., Avelar and Hale, (1980). The essential spectrum corresponds to the spectrum of difference equation

$$\mathbf{x}(t) + \int_0^T d\mathbf{D}(\tau)\mathbf{x}(t-\tau) = 0 \tag{8}$$

which is given by the solutions of the following equation

$$M_e(s) = \det\{\mathbf{I} + \mathbf{D}(s)\} = 0 \tag{9}$$

where $M_e(s)$ is called exponential polynomial. Since the stability of some difference equations is very sensitive to the infinitesimal changes in delays, the concept of so-called strong stability has been introduced in analysis of neutral systems, see Hale and Verduyn Lunel, (2002). The difference equation is strongly stable if it is stable independently on the changes in delays. The analysis of stability of difference equation (8) is very important, since the essential spectrum determines asymptotically the spectrum of the neutral system. If the difference equation (8) is unstable, the neutral system is not only unstable but it is unstable with infinitely many unstable poles.

2. COMPUTING POLES OF RETARDED TDS

The problem of computing poles of TDS has remained unsolved untill the nineties of the last century when two algorithms appeared both based on discretization of the continuous TDS. The first algorithm is based on discretization of the solution operator (Engelborghs and

Roose, 1999) in which the discretization is performed by means of linear multi-step methods. The second algorithm is based on discretization of the infinitesimal generator of the semigroup, (Ford and Wulf, 1998), where the Euler explicit method and trapezoidal rule is used for the discretization. The solution operator and the infinitesimal generator of the semigroup are the concepts used in functional theory of TDS, in which the functional state of the system is directly involved, see Hale and Verduyn Lunel, (1993) or Diekman, et al, (1995). The method based on on discretization of the solution operator has been implemented in the Matlab package DDE-BIFTOOL (Engelborghs, 2001).

The drawback of the mentioned methods based on discretization is given by the fact that only the rightmost poles of the system are approximated. The accuracy of the approximation depends on the chosen numerical method and particularly on the chosen step of the discretization. On the other hand the rightmost poles are the most important ones in the system dynamics, therefore the methods often provide satisfactory results for evaluation of the dominant modes of retarded TDS. However, to the best of the authors knowledge, the analogous methods for computing the poles of neutral systems has not appeared yet. The objective of the paper is to introduce two algorithms that can be used for computing both poles and zeros of continuous TDS located inside a selected region of the complex plane. Both algorithms are based on computing the quasipolynomial roots.

3. COMPUTING QUASIPOLYNOMIAL ROOTS

3.1 Weyl's construction with argument principle based test

Originally, Weyl's construction is used for locating the roots of polynomials, see Pan, (1997). The method consists in recursive partitioning the suspect region $\mathcal{D}=[\beta_{\min},\beta_{\max}]\times[\omega_{\min},\omega_{\max}]$ into subregions and performing proximity test that provides information whether or not the particular subregion contains any roots, see Fig.1. If yes, the region is partitioned and if not, the subregion is discarded. Let us use the Weyl's construction for locating the roots of quasipolynomials. Instead of the proximity test, let us use the argument principle, which holds for any analytic function including quasi-polynomials, see ,e.g. , (El'sgol'ts and Norkin, 1973). Considering a quasipolynomial in the form of $M(s)$ from (6), the number of its roots located inside the region $\mathcal{D}$ with the boundary φ is given by the following formula

$$N_{\mathcal{D}} = \frac{1}{2\pi}\Delta_{\varphi}\arg M(s) = \frac{1}{2\pi \mathrm{j}}\int_{\varphi}\frac{M'(s)}{M(s)}ds \tag{10}$$

where $M'(s)=dM(s)/ds$. As can be seen, the number of roots $N_{\mathcal{D}}$ can be computed directly by evaluating the integral in (10), see Zítek and Vyhlídal, (2000). However, the numerical evaluation of the integral becomes unacceptably time and memory demanding in case that M'/M acquires complicated form. It usually happens if $M(s)$ is of higher order with the delays of different distributions. The other possibility to evaluate $N_{\mathcal{D}}$ is based on a graphical evaluation of formula (10) which claims that $N_{\mathcal{D}}$ is equal $1/2\pi$ times of the variation of the argument $\Phi_M(s)$ given by

$$\tan\Phi_M(s) = \frac{\mathrm{Im}(M(s))}{\mathrm{Re}(M(s))} \tag{11}$$

as s moves once around φ in the counter-clockwise sense.

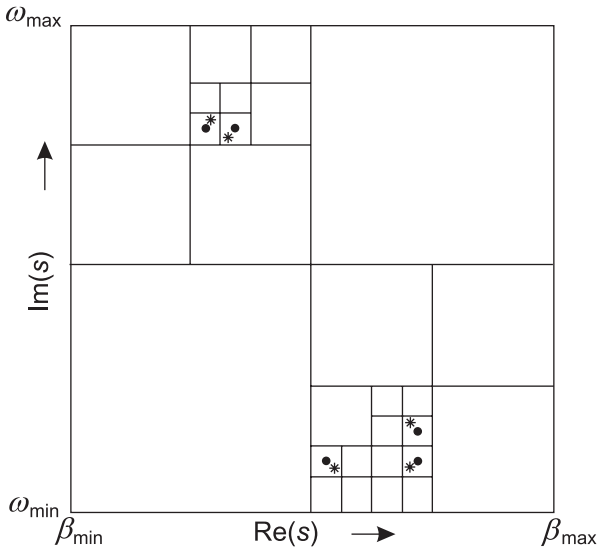

Fig. 1 Locating the roots by Weyl's algorithm, asterisks - quasipolynomial roots, black dots - approximations of the roots

Applying the argument principle in the particular suspect subregion, the result is not only whether or not there are some roots in the subregion, but also the number of the roots located in the subregion is obtained. The procedure of recursive partitioning of the suspect regions is shown in Fig. 1. There are 5 roots in the region $\mathfrak{D}$, marked by asterisks and their approximations obtained using Weyl's construction are marked by the black dots. The first step of the algorithm consists in computing the number of roots in $\mathfrak{D} = [\beta_{\min}, \beta_{\max}] \times [\omega_{\min}, \omega_{\max}]$ using argument principle rule. Consequently Weyl's construction with the argument principle based test is applied recursively until the root approximations with required accuracy are achieved. Thus, in k iteration steps, the approximation errors do not exceed $0.5\text{diag}(\mathfrak{D}_k)/2^k$, where $\mathfrak{D}_k$ represents the actual suspect region.

The accuracy of the final root approximations can be enhanced by means of applying Newton's iteration method

$$s_{i,k+1} = s_{i,k} - \frac{M(s_{i,k})}{M'(s_{i,k})} \tag{12}$$

where s_i, i=1, 2, ... are the roots of $M(s)$ and k=0,1,... is the step of the Newton's iteration. Suppose that the root approximations $s_{i,0}$ resulting from the algorithm based on Weil's construction are close to the roots s_i then the approximations $s_{i,k}$ are likely to converge much faster to the roots s_i than in case of further carrying on Weil's algorithm.

3.2 Algorithm based on quasipolynomial function mapping

The quasipolynomial $M(s)$ as a function of the complex variable $s = \beta + \mathrm{j}\omega$ can be split up into real and imaginary parts

$$M(\beta,\omega) = R(\beta,\omega) + \mathrm{j}I(\beta,\omega) \tag{13}$$

where $R(\beta,\omega) = \mathrm{Re}\{M(\beta,\omega)\}$ and $I(\beta,\omega) = \mathrm{Im}\{M(\beta,\omega)\}$. Consequently, equation (6) can be split up into

$$R(\beta,\omega)=0$$
$$I(\beta,\omega)=0 \tag{14}$$

From the geometric point of view, the roots of $M(s)$ are the intersection points of the curves described by the implicit functions $R(\beta,\omega)=0$ and $I(\beta,\omega)=0$. Mapping the surfaces $R(\beta,\omega)$ and $I(\beta,\omega)$ over the region $\mathcal{D}=[\beta_{\min},\beta_{\max}]\times[\omega_{\min},\omega_{\max}]$ the zero-level contours are given by the intersections of the surfaces with the s-plane, see Fig. 2.

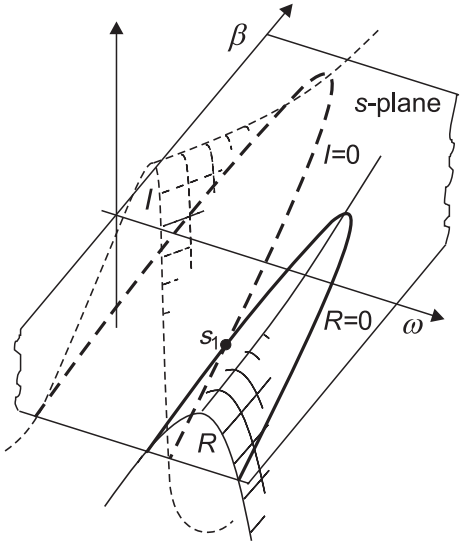

Fig. 2 The principle of locating $M(s)$ roots
R=Re(M)=0 - solid, I=Im(M)=0 - dashed

Taking into account this geometric interpretation of equations (14) the algorithm for locating the roots of $M(s)$ can be summarized as follows: Firstly, the region $\mathcal{D}$ is covered by a grid of nodes $d_{ij} = \beta_j \times \omega_i$, i =1.. $|\omega_{\max}-\omega_{\min}|/\Delta$, j = 1.. $|\beta_{\max}-\beta_{\min}|/\Delta$ with stepwise incrementing co-ordinates β and ω, where Δ is the chosen step of the grid. Secondly, For each node d_{ij}, the values of $R(\beta_i,\omega_j)$ and $I(\beta_i,\omega_j)$ are evaluated. Using a contour plotting method, see, e.g., Cottafava and Le Moli, (1969) , the intersections of the surfaces R and I with the s-plane, i.e., the contours $R(\beta,\omega)=0$ and $I(\beta,\omega)=0$, are mapped. The intersection points of the contours indicate the approximate positions of the roots of $M(s)$. Finally, the accuracy of the root approximations is enhanced by means of Newton's iteration method. Regarding the computational effort, the algorithm is rather demanding. A lot of calculation has to be done, especially if the region $\mathcal{D}$ is chosen too large and there are many roots located inside the region. However, using the Matlab functions defined for the matrices and especially the function *contour*, the approximate locations of the roots given as the intersections of the contours can be found relatively fast. The crucial role in the length of the root finding process is played by the increment Δ, which should be chosen according to the expected frequency range of the contours and particularly to the size of the region $\mathcal{D}$. The smaller Δ is, the more precise is the estimate of the positions of the roots. On the other hand, too small Δ results in too long duration of the rootfinding process. To solve the problem of choosing Δ, it is necessary to balance these two contradictory requirements. With respect to the size of the region $\mathcal{D}$ the increment Δ should not be less than $0.01\min\{(\beta_{\max}-\beta_{\min}),(\omega_{\max}-\omega_{\min})\}$. If the increment is chosen too large, the contours are not approximated well. It may result in omitting some of the roots. In such a case, the increment should be decreased. It is also advisable to divide the original region into smaller ones and to carry out the computation separately for each new region with the decreased increment.

4. APPLICATION EXAMPLE

Let us consider a retarded system with the following functional matrix

$$\mathbf{A}(s)=\begin{bmatrix} -\exp(-8.4s) & \exp(-4.1s) & \exp(-6.6s) \\ \dfrac{\exp(-5.2s)-\exp(-12.5s)}{7.3s} & -\exp(-4.3s) & \exp(-3.7s) \\ \exp(-7.8s) & \dfrac{\exp(-6.5s)-\exp(-18.9s)}{12.4s} & \exp(-5.2s) \end{bmatrix} \qquad (15)$$

It can be seen that there are encountered both lumped and linear distributed delays in matrix **A**(*s*). Using linear distributed delays is advantageous in modelling heat transfer processes in long pipe-lines, see, Zítek and Vyhlídal, (2000). In order to analyse the system dynamics the system poles, i.e., the roots of quasipolynomial $M(s)=\det\{s\mathbf{I}-\mathbf{A}(s)\}$ are to be found. Using the second described method for computing the roots of *M*(*s*) the maps shown in Fig. 3 result.

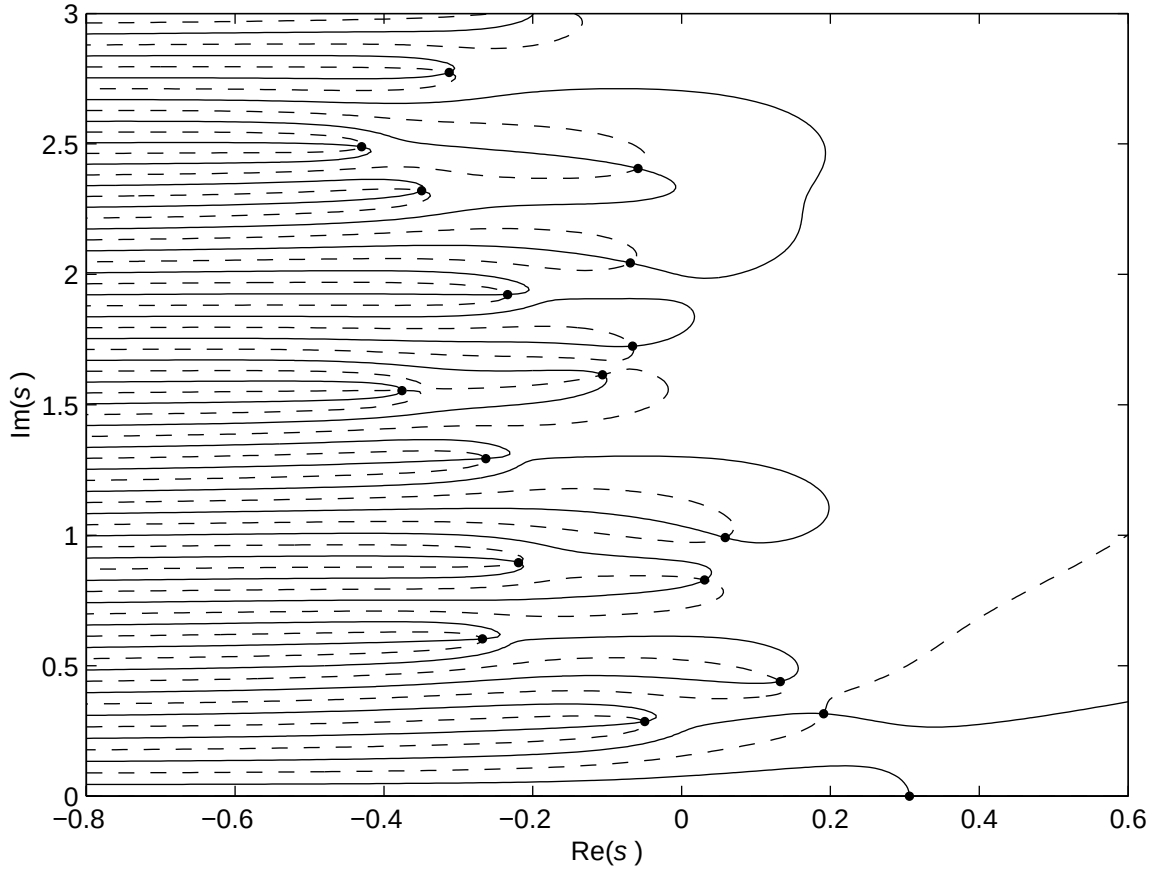

Fig. 3 Distribution of the poles of the retarded system with matrix **A**(*s*) given by (15). $R(\beta,\omega)=0$ (solid), $I(\beta,\omega)=0$ (dashed)

The poles are given by the intersections of the contours described by the functions $R(\beta,\omega)=0$ (solid) and $I(\beta,\omega)=0$ (dashed). As can be seen, there are 18 roots in region $\mathcal{D}=[-0.8,0.6]\times[0,3]$. However, since the complex plane is symmetric, the complex poles have the complex conjugates in negative half plane. Thus, actually, the locations of 35 roots have been found out. The system with matrix (15) is unstable, because there are 5 system poles located to the right from the stability boundary. In agreement with the basic feature of the spectrum of retarded systems, the poles tend to depart to the left with increasing modulus.

5. CONCLUSIONS

Two original algorithms for computing the quasipolynomial roots located inside a selected region in complex plane have been introduced. The first algorithm is based on application of Weyl's construction and argument principle. The second algorithm is based on the geometric interpretation of the quasipolynomial function in the complex plane. By means of the designed quasipolynomial rootfinders, poles as well as zeros of both retarded and neutral time delay systems can be computed. It should be noted that the described techniques for computing quasipolynomial roots are suitable only for relatively low order quasipolynomials (n<10). However, this drawback is fairly unimportant in computing poles or zeros of time delay systems because using time delays in modelling results in low order models as a rule.

Acknowledgement: The presented research was supported by the Ministry of Education of the Czech Republic under Project LN00B096

REFERENCES

Avelar, C.E. and Hale, J.K., (1980), On the zeros of exponential polynomials, *Mathematical analysis and application*, 73, pp. 434-452

Cottafava, G. and Le Moli, G., (1969), Automatic Contour Plotting, *Communications of the ACM*, vol. 12, no. 7, pp. 386-391

Diekmann, O., S.A. Van Gils, S.M. Verduyn Lunel and H.O. Walther (1995) *Delay Equations Functional, Complex and Nonlinear Analysis*. Springer Verlag, AMS Series, No. 110

El'sgol'c, L. E. and Norkin, S. B., (1973), *Introduction to the Theory of Differential Equations with Deviating Argument*, Mathematics in science and Eng., vol 105, Academic Press, New York

Engelborghs, K. (2000). *DDE-BIFTOOL: a Matlab package for bifurcation analysis of delay differential equations*. TW Report 305, Department of Computer Science, Katholike Universiteit Leuven, Belgium.

Engelborghs, K. and Roose, D. (1999). Numerical computation of stability and detection of hopf bifurcations of steady state solutions of delay differential equations. *Advanced in Computational Mathematics,* 10(3-4), pp. 271-289

Ford, N. J. and Wulf, V, (1998), Embedding of a numerical solution of a dde into the numerical solution of a system of odes, Technical Report, Manchester Cntre for Computational Mathematics, University of Manchester

Hale, J. K., and Verduyn Lunel, S. M., (1993), *Introduction to Functional Differential Equations, Vol. 99 of Applied Mathematical Sciences*, Springer Verlag New York Inc.

Hale, J. K., and Verduyn Lunel, S. M., (2002), Strong stabilization of neutral functional differential equations*, IMA Journal Math. Control Inform*, Vol. 19, no. 1, pp. 5-23.

Niculescu, S. I., (2001), *Delay Effects on Stability, A Robust Control Approach*. Springer-Verlag London Limited.

Zítek, P. and Vyhlídal, T. (2002), Dominant eigenvalue placement for Time Delay Systems, In *Proc. of Controlo 2002, 5th Portuguese Conference on Automatic Control, University of Aveiro, Portugal*

Zítek, P., (1998). Time Delay Control System Design Using Functional State Models. *CTU Reports* No. 1, CTU Prague.

Zítek, P., Vyhlídal, T., (2000), State Feedback Control of Time Delay System: Conformal Mapping Aided Design In: *2nd IFAC Workshop on Linear Time Delay Systems*. Ancona, Universita di Ancona, pp. 146-151