Extended systems for Delay-differential equations as implemented in the extensions to DDE BifTool

Jan Sieber

Contents

1 Background 1

2 Folds of periodic orbits — constant delays 2

3 Period doublings and torus bifurcations — constant delays 4

4 Systems with state-dependent delays 5

4.1 The general variational problem 7

5 Fold of periodic orbits — state-dependent delays 8

6 Period doublings and torus bifurcations — state-dependent delays 10

Abstract This document lists the extended systems defining bifurcations of periodic orbits in delay differential equations (DDEs) with finitely many point delays. The delays are permitted to be system parameters or dependent on the state at arbitrary points in the history. All systems are boundary-value problems with periodic boundary conditions.

The proposed systems have been implemented as extensions to DDE-BifTool, a numerical bifurcation analysis package for DDEs for Matlab or Octave. The document contains also practical details of the implementation, pointing to relevant routines.

1 Background

The references [1, 2, 3, 4] give background information on the defining systems for bifurcations of periodic orbits in ODEs. The systems defining the periodic-orbit bifurcations for constant or state-dependent delay can be written as periodic boundary-value problems (BVPs) of DDEs that can be tracked using the `'psol'` routines of DDE-BifTool[1]. The theory for bifurcations of periodic orbits in DDEs with constant delay is covered in textbooks [5, 6, 7]. Details of the algorithms and theory behind DDE-BifTool are discussed in [8, 9, 10, 11, 12]. The review [11] also refers to `knut`[2], an

[1] `http://twr.cs.kuleuven.be/research/software/delay/ddebiftool.shtml`

[2] `http://gitorious.org/knut/pages/Home`

alternative C++ package with a stand-alone user interface. The review [13] lists difficulties and open problems occuring with DDEs with state-dependent delays. However, periodic BVPs can be reduced to smooth finite-dimensional systems of algebraic equations [14]. This means that the general bifurcation theory for DDEs with state-dependent delay works as expected as far as the Hopf bifurcation, regular branches of periodic orbits, the period doubling bifurcation and Arnol'd tongues branching off at resonant points at a torus bifurcation are concerned.

In the following all DDEs are assumed to have periodic boundary conditions on the interval $[0,1]$.

2 Folds of periodic orbits — constant delays

In DDE-BifTool the system defining a periodic orbit is the periodic DDE with n_τ delays

$$0 = \frac{M}{T}\dot{x}(t) - f\left(x(t), \frac{x^{(k_1)}(t-\tau_1/T)}{T^{k_1}}, \ldots, \frac{x^{(k_1)}(t-\tau_{n_\tau}/T)}{T^{k_{n_\tau}}}, p\right), \tag{1}$$

$$0 = \int_0^1 \dot{x}_{\mathrm{ref}}(t)^T x(t)\,\mathrm{d}t \tag{2}$$

where the unknowns are $x \in C^n_{\mathrm{p}} := C_{\mathrm{per}}([0,1]; R^n)$ (space of periodic functions on $[0,1]$) and T. Using the operators

$$[E^\ell_{\tau,T}x](t) := \frac{x^{(\ell)}(t-\tau/T)}{T^\ell}, \qquad F(x_0, \ldots x_{n_\tau}, p)(t) := f(x_0(t), \ldots x_{n_\tau}(t), p),$$

equation (1) reads as an identity for functions in C^n_{p} (defining $\tau_0 = 0$, $\ell_0 = 0$):

$$0 = ME^1_{0,T}x - F\left(\left(E^{\ell_k}_{\tau_k,T}x\right)_{k=1}^{n_\tau}, p\right). \tag{3}$$

The linearizations of E w.r.t. τ and T are

$$\partial_\tau E^\ell_{\tau,T} := -E^{\ell+1}_{\tau,T}, \qquad \partial_T E^\ell_{\tau,T} = \frac{1}{T}\left[\tau E^{\ell+1}_{\tau,T} - \ell E^\ell_{\tau,T}\right].$$

The criterion for a fold is that system (3), (2) has a simple singularity in $x \in C^n_{\mathrm{p}}$, that is, its linearization has a simple nullvector. Some problems (for example, with rotational symmetry) have additional free parameters that we consider part of p. Let us assume that those free n_q parameters are selected by $n_p \times n_q$ matrix Δ_p, such that the derivative of f with respect to the free parameters is $\partial_p f(\cdot)\Delta_p$. Hence ,the nullvector is $(\delta_x, \delta_T, \delta_p) \in C^n_p \times \mathbb{R} \times \mathbb{R}^{n_q}$. This nullvector satisfies the linear system of equations:

$$0 = ME^1_{0,T}\delta_x - \sum_{k=0}^{n_\tau} \partial_k F E^{\ell_k}_{\tau_k,T}\delta_x + \frac{\delta_T}{T}\cdot\left[\sum_{k=0}^{n_\tau}\partial_k F\left[\ell_k E^{\ell_k}_{\tau_k,T} - \tau_k E^{\ell_k+1}_{\tau_k,T}\right] - ME^1_{0,T}\right]x - \partial_p F\Delta_p\delta_p, \tag{4}$$

$$0 = \int_0^1 \dot{x}_{\mathrm{ref}}(t)^T \delta_x(t)\,\mathrm{d}t \tag{5}$$

Note that the factor for δ_T (in brackets) in (4) is the derivative of the original DDE (1) with respect to T. In (4) the terms $\partial_k F_*$ stand for (using $\partial_k f$ for the derivative w.r.t. the argument $[E^{\ell_k}_{\tau_k,T}x](t)$ and $\partial_p f$ for the derivative of f w.r.t. parameter p)

$$[\partial_k F v](t) = \partial_k f(\cdot)v(t), \quad [\partial_p F q](t) = \partial_p f(\cdot)q, \text{ where } \quad (\cdot) = \left(\left(\frac{x^{(\ell_k)}\left(t-\frac{\tau_k}{T}\right)}{T^{\ell_k}}\right)_{k=1}^{n_\tau}, p\right) \tag{6}$$

Position in `parameter`	**parameter**
`1:npar`	user system parameters
`npar+1`	$\beta := \delta_T/T$
`npar+1+nnull`	additional derivatives δ_p w.r.t. other free parameters
`npar+1+nnull+(1:ntau)`	$\tau_{n_\tau+k} = \tau_k$ $(k=1,\ldots,n_\tau)$, delays for $x^{(\ell_k+1)}$

Table 1: Parameters of extended system for fold (`ntau=length(sys_tau())`$=n_\tau$)

Extended system The function variables (stored in `point.profile`) are $(x,\delta_x) \in C_p^{2n} = C_p^n \times C_p^n$. Inside `pfuncs.sys_rhs` they are accessed as `x=xx(1:n,:)` and `v=xx(n+1:end,:)`. The system parameters of the extended system, assuming that the user-defined system has `npar` parameters, are given in Table 1. The DDE defined by `sys_rhs_POEV1(xx,par)` consists of (using the convention $\tau_0 = 0$, $\ell_0 = 0$, $\beta = \delta_T/T$)

$$M\dot{x}(t) = f\left((x^{(\ell_k)}(t-\tau_k))_{k=0}^{n_\tau}, p\right), \tag{7}$$

$$M\dot{v}(t) = \partial f\left((x^{(\ell_k)}(t-\tau_k))_{k=0}^{n_\tau}, p\right)\left[(z_k(t))_{k=0}^{n_\tau}, \Delta_p\delta_p\right] + \beta M\dot{x}(t), \quad \text{where} \tag{8}$$

$$z_k(t) = v^{(\ell_k)}(t-\tau_k) + \beta\tau_k x^{(\ell_k+1)}(t-\tau_k) - \beta\ell_k x^{(\ell_k)}(t-\tau_k).$$

Note that there are only $2n_\tau$ delays, since ther term $\tau_0 = 0$ in (8). The term $\partial f(\ldots)[(z_k)_{k=1}^{n_\tau}, q]$ is the directional derivative of f in $(x(t), x^{(\ell_1)}(t-\tau_1), \ldots, x^{(\ell_{n_\tau})}(t-\tau_{n_\tau}), p)$. It has $n_\tau + 1$ linear arguments z_k of length n_x and 1 linear argument q of length n_p. The parameters in its argument `par` are ordered as in Table 1. The additional $2 + n_{q\cap\tau}$ system conditions implemented in `sys_cond_POfold` are:

$$0 = \int_0^1 v(t)^T v(t)\,\mathrm{d}t + T^2\beta^2 + \sum_{i=1}^{n_q} q_i^2 - 1 \tag{9}$$

$$0 = \int_0^1 \dot{x}_{\mathrm{ref}}(t)^T v(t)\,\mathrm{d}t \tag{10}$$

$$0 = \tau_{k+n_\tau} - \tau_k, \quad \text{for delays } k \text{ that are free parameters.} \tag{11}$$

An auxiliary routine `p_dot` is included in the folder `ddebiftool` to compute scalar products between `'psol'` structures of the type (10) (not directly called by the user but may be useful in other problems).

Initialization The Jacobian J of the linearization of (1)–(2) in a point `x` can be obtained with the DDE-BifTool routine `dde_psol_jac` (calling it with free parameters in the range of Δ_p: `free_par=nullparind(:,1)`). The phase condition of the periodicBVP is included in `dde_psol_jac`. Linearizations of additional q `sys_cond` type conditions must be appended. If $J = USV^T$ is the singular-value decomposition of J then `V(:,end)` is the approximate nullvector of J. This column vector only has to be reshaped and rescaled to achieve (9):

```
v.profile=reshape(V(1:size(x.profile),end),size(x.profile));
v.parameter=0*x.parameter;
extrapars=[ip.beta,ip.nullparind(:,1)'];
v.parameter(extrapars)=V(size(x.profile)+1:end,end);
v.parameter(ip.beta)=v.parameter(ip.beta)/x.period;
vnorm=sqrt(p_dot(v,v,'free_par_ind',extrapars);
v.parameter=v.parameter/vnorm;
v.profile=v.profile/vnorm;
x.profile(dim+(1:dim),:)=v.profile;
x.parameter(ip.beta)=v.parameter(ip.beta);
x.parameter(ip.nullparind(:,2))=v.parameter(ip.nullparind(:,1));
```

Then `x.profile(dim+(1:dim),:)` is the approximation for $v(t)$, `x.parameter(ip.beta)` is the approximation for β, and `x.parameter(ip.nullparind(:,2))` is the approximation for q.

3 Period doublings and torus bifurcations — constant delays

The extended system has the function components x, u and v (all in C^n_p) and the additional parameters ω (angle of critical Floquet multipiers) and T_{copy} (equal to the period of the orbit). The linearized system is written in Floquet exponent form. This means that, if $\exp(\mathrm{i}\omega\pi)$ is the Floquet multiplier and z the corresponding eigenfunction (satisfying $z(1) = z(0)\exp(\mathrm{i}\omega\pi)$), then

$$u(t) + \mathrm{i}v(t) = \exp(-\mathrm{i}\omega\pi t/T)z(t)$$

such that u and v are periodic (and real). The linearized system in Floquet exponent form (as implemented in `sys_rhs_TorusBif`) is

$$\dot u(t) = \frac{\pi\omega}{T}v(t) + \sum_{j=0}^{n_\tau}\partial_{j+1}f(t)\big[\cos(\pi\omega\tau_j/T)u(t-\tau_j) + \sin(\pi\omega\tau_j/T)v(t-\tau_j)\big] \tag{12}$$

$$\dot v(t) = -\frac{\pi\omega}{T}u(t) + \sum_{j=0}^{n_\tau}\partial_{j+1}f(t)\big[-\sin(\pi\omega\tau_j/T)u(t-\tau_j) + \cos(\pi\omega\tau_j/T)v(t-\tau_j)\big]. \tag{13}$$

Again, we used the convention $\tau_0 = 0$ and $\partial_k f(t)$ as defined in (6). The additional system conditions (in `sys_cond_TorusBif`) are

$$0 = \int_0^1 u(t)^T u(t) + v(t)^T v(t)\,\mathrm{d}t - 1 \tag{14}$$

$$0 = \int_0^1 u(t)^T v(t)\,\mathrm{d}t \tag{15}$$

$$0 = \texttt{point.parameter(npar+2)-point.period} \tag{16}$$

where `npar` is the number of parameters defined by the user.

Initialization The auxiliary routine `mult_crit` extracts the critical Floquet multiplier μ and the corresponding eigenfunction $z(t)$ for the approximate bifurcation point `x`. Then the periodic Floquet mode $y(t)$ is obtained as

$$y(t) = y_r(t) + \mathrm{i}y_i(t) = \exp(-\log(\mu)t)z(t),$$

where $t \in [0,1]$ is already rescaled, and $\omega =$ `atan2(imag(mu),real(mu))`$/\pi$. Finally, one has to rescale y to make it of unit length, and make its real and imaginary part orthogonal. Define

$$\begin{aligned} r &= \sqrt{\langle y_r, y_r\rangle + \langle y_i, y_i\rangle}, \quad \text{and} \\ \gamma &= \frac{1}{2}\mathtt{atan2}(2\langle y_r, y_i\rangle, \langle y_r, y_r\rangle - \langle y_i, y_i\rangle), \quad \text{then} \\ u(t) &= [y_r(t)\cos\gamma - y_i \sin\gamma]/r, \\ v(t) &= [y_r(t)\sin\gamma + y_i \cos\gamma]/r \end{aligned}$$

satisfy $\langle u, v\rangle = 0$ and $\langle u, u\rangle + \langle v, v\rangle = 1$, giving the profiles of u and v, needed for the extended system.

4 Systems with state-dependent delays

For state-dependent delays the user may provide the argument `'sys_tau_seq'`, for example,

```
funcs=set_funcs(...,'sys_tau_seq',{1:3,4},'sys_ntau',@()4);
```

to indicate that `sys_tau` can be called as `sys_tau(1:3,xx0,par)` with `xx0`$=x(t)$ of format $n_x \times 1$, returning the first 3 delays, and `sys_tau(4,xx2,par)` with `xx1` of format $n_x \times 4$ (consisting of $(x(t), x^{\ell)}(t-\tau_1), x^{(\ell)}(t-\tau_2), x^{(\ell)}(t-\tau_3))$ for some order $\ell \geq 0$. This possibility of returning multiple delays at once makes the treatment of a large number of state-dependent delays (approximating distributed delays) computationally feasible. If `'sys_tau_seq'` is not provided, it defaults to `num2cell(1:sys_ntau())`.

For this reason we unify the notation by accepting the entire introduce a mildly different notation for the system defining a periodic orbit by grouping the elements of `'sys_tau_seq'` into vectors of length m_k, each corresponding to m_k delays with the same arguments, with $m_0 = 1$ and $\sum_{k=1}^{m_\tau} m_k = n_\tau$:

$$0 = M\frac{\dot{x}(t)}{T} - f\left([x_0, \dots, x_{m_\tau}], p\right), \quad \text{where } x_k \in \mathbb{R}^{n_x \times m_k}, \tag{17}$$

$$x_k = \left(x^{(\ell_k)}\left(t - \frac{\tau_{k,j}([x_0, \dots, x_{k-1}], p)}{T}\right)\Big/T^{\ell_k}\right)_{j=1}^{m_k}, \tag{18}$$

$$\tau_0 = 0,\ m_0 = 1,\ k = 0, \dots, m_\tau, \quad \sum_{j=1}^{m_\tau} m_k = n_\tau,$$

augmented with the phase condition (2) to define the period T. In (20) the delays are functions

$$\tau_k : \mathbb{R}^{n_x \times \mu_k} \times \mathbb{R}^{n_p} \mapsto \mathbb{R}^{m_k}, \text{ where } \quad \mu_k = \sum_{j=0}^{k-1} m_k, \text{ and } 0 = \mu_0 < \mu_1 < \ldots < \mu_{m_\tau} = n_\tau + 1.$$

In the extreme case that all delays only depend on $x(t)$ and p, we have the arguments

```
funcs=set_funcs(...'sys_tau_seq',{1:ntau},'sys_ntau',@()ntau,...);
```

where `sys_tau(1:ntau,xx0,par)` is called once with `xx0` of size $n_x \times 1$, while `sys_rhs(xx,par)` is called with `xx` of size $n_x \times (1+n_\tau)$ ($[x(t), x(t-\tau_1(x(t),p)), \ldots, x(t-\tau_{n_\tau}(x(t),p))]$), such that $m_\tau = 1$, $m_0 = 0$, $m_1 = n_\tau$.

Remark Every system of form

$$0 = M\frac{\dot{x}(t)}{T} - f\left([x_0, \ldots, x_{n_\tau}], p\right), \quad \text{where } x_0 = x(t), \tag{19}$$

$$x_k = x^{(\ell_k)}(t - \tau_k([x_0, \ldots, x_{k-1}], p)/T)\big/T^{\ell_k}, \ (k = 1, \ldots, n_\tau) \tag{20}$$

(the default scenario of (19), (20), where `'sys_tau_seq'` equals `num2cell(1:sys_ntau())` can be transformed into a system where all delays depend on $x(t)$ and parameter p only. To do this one introduces the delays explictly as additional time-dependent variables $x_{n_x+1}(t), \ldots, x_{n_x+n_\tau}(t)$ and adding the n_τ algebraic equations

$$0 = \tau_1(x_{1,\ldots,n_x}(t), p) - x_{n_x+1}(t), \text{ and for } k = 2\ldots, n_\tau \tag{21}$$

$$0 = \tau_k([x_{1,\ldots,n_x}(t), x_{1,\ldots,n_x}(t - x_{n_x+1}(t)), \ldots, x_{1,\ldots,n_x}(t - x_{n_x+k-1}(t))], p) - x_{n_x+k}(t). \tag{22}$$

This reduces the complexity of the format of calling `sys_tau`, but adds n_τ variables to the system, which may not be appropriate if the number of delays is large.

Operator notation We correspondingly generalize our notation E to multi-column vectors $\hat{E}$. For a vector $\tau \in \mathbb{R}^m$, integer $\ell \geq 0$, scaling $T > 0$,

$$[\hat{E}^\ell_{\tau,T} x](t) := \left(x^{(\ell)}(t - \tau_1/T), \ldots, x^{(\ell)}(t - \tau_m/T)\right)/T^\ell \in \mathbb{R}^{n_x \times m},$$

$$F(x_0, \ldots x_{m_\tau}, p)(t) := f(x_0(t), \ldots x_{m_\tau}(t), p) \in \mathbb{R}^{n_x},$$

the linearizations of $\hat{E}$ are identical to those in the constant delay case

$$\partial_\tau \hat{E}^\ell_{\tau,T} x(t)\delta_\tau := -\hat{E}^{\ell+1}_{\tau,T} x(t)\,\mathrm{diag}(\delta_\tau) \in \mathbb{R}^{n_x \times m} \quad \text{for } \delta_\tau \in \mathbb{R}^{m \times 1}, \mathrm{diag}(\delta_\tau) \in \mathbb{R}^{m \times m},$$

$$\partial_T \hat{E}^\ell_{\tau,T} x(t) = \frac{1}{T}\left[\hat{E}^{\ell+1}_{\tau,T} x(t)\,\mathrm{diag}(\tau) - \ell \hat{E}^\ell_{\tau,T} x(t)\right] \in \mathbb{R}^{n_x \times m}.$$

The operation $\mathrm{diag}(\tau)$ for a vector τ results in a diagonal matrix D with diagonal entries $D_{i,i} = \tau_i$. For a function $\tau : \mathbb{R}^{n_x \times \mu} \times \mathbb{R}^{n_p} \to \mathbb{R}^m$, we have the linearization of

$$\hat{E}^\ell_{\tau(y,p),T} x(t) = \left(x^{(\ell)}(t - \tau_1/T)/T^\ell, \ldots, x^{(\ell)}(t - \tau_m/T)/T^\ell\right) \in \mathbb{R}^{n_x \times m}$$

with respect to deviation $(\delta_x, \delta_y, \delta_p, \delta_T)$:

$$\partial \hat{E}^{\ell}_{\tau(y,p),T} x[\delta_x, \delta_y, \delta_p, \delta_T](t) = \hat{E}^{\ell}_{\tau(y,p),T} \delta_x(t) - \ell \frac{\delta_T}{T} \hat{E}^{\ell}_{\tau(y,p),T} x(t) + \hat{E}^{\ell+1}_{\tau(y,p),T} x(t) \left[\operatorname{diag}\left(\tau(y,p) \frac{\delta_T}{T} - \partial \tau(y,p)[\delta_y, \delta_p] \right) \right].$$

Using the oprators $\hat{E}$ and F the differential equation reads

$$0 = M \hat{E}^1_{0,T} x - F\left([x_0, \dots, x_{m_\tau}], p\right), \quad \text{where} \quad x_k = \hat{E}^{\ell_k}_{\tau_k([x_0,\dots,x_{m_{k-1}}],p),T} x, \tag{23}$$

with $\ell_0 = 0$, $\tau_0 = 0$ for $k = 0 \dots, m_\tau$.

4.1 The general variational problem

The variational problem of (19)—(20) linearized in $(x(\cdot), T, p)$ with respect to $(\delta_x(\cdot), \delta_T, \delta_p)$ is:

$$0 = M \hat{E}^1_{0,T} \delta_x - M \hat{E}^1_{0,T} x \frac{\delta_T}{T} - \partial F(y_{m_\tau}, p)[Y_{m_\tau}, \delta_p], \qquad \text{where for } k = 0, \dots, m_\tau, \tag{24}$$

$$x_k = \hat{E}^{\ell_k}_{\tau_k(y_{k-1},p),T} x, \quad y_k = [x_0, \dots, x_k], \quad Y_k = [X_0, \dots, X_k],$$

$$X_k = \hat{E}^{\ell_k}_{\tau_k(y_{k-1},p),T} \delta_x - \ell_k \frac{\delta_T}{T} \hat{E}^{\ell_k}_{\tau_k(y_{k-1},p),T} x + \hat{E}^{\ell_k+1}_{\tau_k(y_{k-1},p),T} x \left[\operatorname{diag}\left(\tau_k(y_{k-1}, p) \frac{\delta_T}{T} - \partial \tau_k(y_{k-1}, p)[Y_{k-1}, \delta_p] \right) \right]. \tag{25}$$

Since $\ell_0 = 0$, $\tau_0 = 0$ and $\partial \tau_0 = 0$, we have in particular that $x_0 = E^0_{0,T} x = x$ and $X_0 = \hat{E}^0_{0,T} \delta_x = \delta_x$. At each time $t \in [0, 1]$ the $X_k(t)$ have shape $n_x \times m_k$ and $Y_k(t)$ have shape $n_x \times \mu_k = n_x \times (m_0 + \ldots + m_k)$. The matrix functions $t \mapsto X_k(t) \in \mathbb{R}^{n_x \times m_k}$ are the directional derivatives of $t \mapsto x_k(t) \in \mathbb{R}^{n_x \times m_k}$ in direction $(t \mapsto \delta_x(t), \delta_T, \delta_p)$.

We observe that the definition of X_k recursively requires $Y_{k-1} = [x_0, \dots, x_{k-1}]$ such that the X_k could be computed iteratively over k from 0 to m_τ. Alternatively, introducing the notation

$$\hat{\tau}_k(y_{m_\tau}, p) = \hat{\tau}_k(y_k, p), \qquad \partial \hat{\tau}_k(y_{m_\tau}, p)[Y_{m_\tau}, \delta_p] = \partial \tau_k(y_k, p)[Y_k, \delta_p],$$

$$\hat{\tau} = [\hat{\tau}_0, \dots, \hat{\tau}_{m_\tau}], \qquad \hat{E}^{\ell}_{\hat{\tau}(y,p),T} x = \left[\hat{E}^{\ell_l}_{\hat{\tau}_k(y,p),T} x \right]_{k=0}^{m_\tau},$$

(permitting the longer argument vectors y_{m_τ} and Y_{m_τ} to be passed on to $\hat{\tau}_k$ and $\partial \hat{\tau}_k$ such that all $\hat{\tau}_k$ have the same arguments), we observe that Y_{m_τ} satisfies the linear equation

$$[I - \Theta] Y = Y_{\text{rhs}} \quad \text{where} \tag{26}$$

$$Y_{\text{rhs}} = \hat{E}^{\ell}_{\hat{\tau}(y,p)} \delta_x + \left[\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \operatorname{diag}(\hat{\tau}(y,p)) - \hat{E}^{\ell}_{\hat{\tau}(y,p)} x \operatorname{diag}(\ell) \right] \frac{\delta_T}{T} - \hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \partial \hat{\tau}(y,p)[0, \delta_p],$$

$$\Theta Y = -\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \partial \hat{\tau}(y,p)[Y, 0], \text{ with the abbreviations} \quad Y = Y_{m_\tau}, \quad y = y_{m_\tau}.$$

When the solution x is discretized into n_{int} collocation intervals with collocation polynomial order n_{cd}, such that we have $n_{\text{bp}} = n_{\text{int}} n_{\text{cd}} + 1$ base time points (typically in $[0, 1]$, but for stability computations the base interval is extended to $[-\tau_{\max}, 1]$)) and $n_{\rm eqs} = n_{\rm int}n_{\rm cd}$ collocation time points (where equations are imposed), then the dimension of the terms involved is

$$\begin{aligned}
\hat{E}^{\ell}_{\hat{\tau}(y,p)} &: n_x n_{\rm eqs}(n_\tau+1) \times n_x n_{\rm bp},\\
\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \operatorname{diag}(\hat{\tau}(y,p)) - \hat{E}^{\ell}_{\hat{\tau}(y,p)} x \operatorname{diag}(\ell) &: n_x n_{\rm eqs}(n_\tau+1) \times 1 \quad (\operatorname{diag}(\cdot) \text{ applied to 3rd dim.})\\
\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \partial\tau(y,p)[0,(\cdot)] &: n_x n_{\rm eqs}(n_\tau+1) \times n_{\rm fp},\\
\Theta &: n_x n_{\rm eqs}(n_\tau+1) \times n_x n_{\rm eqs}(n_\tau+1).
\end{aligned}$$

In the dimension specification the order of terms in the products indicates the ordering of the vectors/matrices. The matrix Θ is nilpotent with $\Theta^{m_\tau+1} = 0$ such that the iteration given in (25) is equivalent to a backsubstitution, or, if one uses matrix free directional derivatives for $\partial\hat{\tau}$,

$$Y = \sum_{k=0}^{m_\tau} \Theta^k Y_{\rm rhs}.$$

Using the above notation, the general variational problem can be written as

$$0 = M\hat{E}^1_{0,T}\delta_x - M\hat{E}^1_{0,T}x\frac{\delta_T}{T} - \partial F(y,p)[Y,\delta_p], \quad \text{where} \quad Y = [I-\Theta]^{-1}Y_{\rm rhs} = \sum_{k=0}^{m_\tau}\Theta^k Y_{\rm rhs}, \tag{27}$$

$$\Theta = -\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \partial_y \hat{\tau}(y,p) \tag{28}$$

$$Y_{\rm rhs} = \hat{E}^{\ell}_{\hat{\tau}(y,p)}\delta_x + \left[\hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \operatorname{diag}(\hat{\tau}(y,p)) - \hat{E}^{\ell}_{\hat{\tau}(y,p)} x \operatorname{diag}(\ell)\right]\frac{\delta_T}{T} - \hat{E}^{\ell+1}_{\hat{\tau}(y,p)} x \partial_p \hat{\tau}(y,p)\delta_p. \tag{29}$$

The constant-delay case is the special case where $m_\tau = 0$ such that $Y = Y_{\rm rhs}$.

Practical considerations Continuation of periodic orbit bifurcations computes the delays $\tau_{j,k}$ and calls the right-hand side f at $(x(t-\tau_{k,0}),\ldots,x(t-\tau_{k,n_\tau}))$. The function `tauSD_ext_ind` creates an array `xtau_ind` which sorts the delays as follows: $\tau_{0,k} = \tau_{k,0} =$ `tau(k)` for $1 \le k \le n_\tau$, $\tau_{j,k} =$ `tau((j-1)*ntau+ntau+1+k)` for $j \ge 1$ and $k = 1,\ldots,n_\tau$. Inside the array `xx`, which is the first argument of `sys_rhs` and `sys_tau`, $(x(t-\tau_{k,0}),\ldots,x(t-\tau_{k,n_\tau}))$ can be found as `xx(:,xtau_ind(k+1,:))`

5 Fold of periodic orbits — state-dependent delays

The function `sys_rhs_SD_POfold` extends the user-defined nonlinear problem with the variational problem (24)—(25) restricted to $q = 0$. As a user-defined function it assumes that its argument $x(\cdot)$ has period T (thus, all delays and derivatives are re-scaled by T). Repeating all definitions, `sys_rhs_SD_POfold` implements the following system:

$$\begin{aligned}
\dot{x}(t) &= f\left(x_0,\ldots,x_{n_\tau},p\right), \\
\dot{v}(t) &= \frac{\beta}{T}f(t)+\sum_{k=0}^{n_\tau}\partial_{x,k}f(t)[V_k+B_k], \qquad \text{where} \\
V_0 &= v(t), \quad B_0=0 \quad (\in\mathbb{R}^n), \\
V_k &= v(t-\tau_k(t))-f_k(t)\sum_{j=0}^{k-1}\partial_{x,j}\tau_k(t)V_j, \quad (k=1\ldots n_\tau) \\
B_k &= f_k(t)\left[\frac{\tau_k(t)}{T}\beta-\sum_{j=0}^{k-1}\partial_{x,j}\tau_k(t)B_j\right], \quad (k=1\ldots n_\tau).
\end{aligned} \tag{30}$$

System (30) uses the notations

$$\begin{aligned}
x_k &= x(t-\tau_k(x_0,\ldots,x_{k-1},p)) && (\tau_0=0),\ k=0,\ldots,n_\tau, \\
f(t) &= f\left(x_0,\ldots,x_{n_\tau},p\right), \\
\tau_k(t) &= \tau_k\left(x_0,\ldots,x_{n_\tau},p\right) && \text{for } k=0,\ldots,n_\tau, \\
\partial_{x,k}f(t) &= \frac{\partial}{\partial x_k}f\left(x_0,\ldots,x_{n_\tau},p\right) && \text{for } k=0,\ldots,n_\tau, \\
\partial_{x,k}\tau_k(t) &= \frac{\partial}{\partial x_k}\tau_k\left(x_0,\ldots,x_{n_\tau},p\right) && \text{for } k=0,\ldots,n_\tau, \\
f_k(t) &= f(x(t-\tau_{k,0}),\ldots,x(t-\tau_{k,n_\tau}),p) && \text{for } k=0,\ldots,n_\tau, \\
\tau_{0,k} &= \tau_{k,0}=\tau_k(t) && \text{for } k=0,\ldots,n_\tau, \\
\tau_{j,k} &= \tau_{j,0}+\tau_k(x(t-\tau_{j,0}),\ldots,x(t-\tau_{j,k-1}),p) && \text{for } k,j=1,\ldots,n_\tau.
\end{aligned} \tag{31}$$

The rows of the argument `xx` for `sys_rhs_SD_POfold` consist of $x(\cdot)$ and $v(\cdot)$. The parameter array is extended by β and T_{copy} (a copy of the period T, the relation $T=T_{\text{copy}}$ is enforced in `sys_cond_POfold`). The two differential equations in (30) are augmented with the phase condition built into DDE-BifTool, and additional system conditions. The additional $3+n_\tau(n_\tau+1)/2$ system conditions implemented in `sys_cond_POfold` are identical to the extra conditions of the constant-delay case (9)–(**??**):

$$0=\int_0^1 v(t)^T v(t)\,\mathrm{d}t+\beta^2-1 \tag{32}$$

$$0=\int_0^1 \dot{x}(t)^T v(t)\,\mathrm{d}t \tag{33}$$

$0=$ `point.parameter(npar+2)-point.period`. (34)

As the additional delays are not parameters, condition (11) is not present (controlled by the parameter `relations` of `sys_cond_POfold` being empty).

6 Period doublings and torus bifurcations — state-dependent delays

The function `sys_rhs_SD_TorusBif` extends the user-defined nonlinear problem with a coupled pair of variational problems (24)—(25) restricted to $q=0$ and $\beta=0$. The approach is identical to the extended system for constant delays outlined in Section 3. However, it includes the derivatives of the time-delays. The extended system for torus bifurcations computes a pair of two functions $u(t)$ and $v(t)$ ($u(t)+\mathrm{i}v(t)$ is the critical complex Floquet mode). Consequently, the extended system for torus bifurcations with state-dependent delays requires a pair of sequences U_k and V_k (instead of V_k and B_k for the fold of periodic orbits). The full extended system for torus bifurcations and period doublings is:

$$
\begin{aligned}
&\dot{x}(t)=f\left(x_0,\ldots,x_{n_\tau},p\right),\\
&\dot{u}(t)=\frac{\pi\omega}{T}v(t)+\sum_{k=0}^{n_\tau}\partial_{x,k}f(t)U_k,\\
&\dot{v}(t)=-\frac{\pi\omega}{T}u(t)+\sum_{k=0}^{n_\tau}\partial_{x,k}f(t)V_k,\qquad\text{where}\\
&\quad U_0=u(t),\quad V_0=v(t),\quad\text{and for } k=1\ldots,n_\tau\\
&\quad U_k=\quad\cos\left(\frac{\pi\omega\tau_k(t)}{T}\right)u(t-\tau_k(t))+\sin\left(\frac{\pi\omega\tau_k(t)}{T}\right)v(t-\tau_k(t))-f_k(t)\sum_{j=0}^{k-1}\partial_{x,j}\tau_k(t)U_j,\\
&\quad V_k=-\sin\left(\frac{\pi\omega\tau_k(t)}{T}\right)u(t-\tau_k(t))+\cos\left(\frac{\pi\omega\tau_k(t)}{T}\right)v(t-\tau_k(t))-f_k(t)\sum_{j=0}^{k-1}\partial_{x,j}\tau_k(t)V_j.
\end{aligned}
\tag{35}
$$

System(35) uses the same set (31) of notations as the system for the fold of periodic orbits. The additional conditions and the initialization are identical to the constant-delay case in Section 3.

References

[1] H. Meijer, F. Dercole, and B. Oldeman. Numerical bifurcation analysis. In R. A. Meyers, editor, *Encyclopedia of Complexity and Systems Science*, pages 6329–6352. Springer New York, 2009.

[2] W.J.F. Govaerts. *Numerical Methods for Bifurcations of Dynamical Equilibria*. Miscellaneous Titles in Applied Mathematics Series. SIAM, 2000.

[3] E J Doedel. Lecture notes on numerical analysis of nonlinear equations. In B Krauskopf, H M Osinga, and J Galán-Vioque, editors, *Numerical Continuation Methods for Dynamical Systems: Path following and boundary value problems*, pages 1–49. Springer-Verlag, Dordrecht, 2007.

[4] E. J. Doedel, A. R. Champneys, T. F. Fairgrieve, Y. A. Kuznetsov, B. Sandstede, and X. Wang. *AUTO97, Continuation and bifurcation software for ordinary differential equations*. Concordia University, 1998.

[5] G Stépán. *Retarded Dynamical Systems: Stability and Characteristic Functions*. Longman Scientific and Technical, Harlow, Essex, 1989.

[6] J. K. Hale and S. M. Verduyn Lunel. *Introduction to Functional Differential Equations*, volume 99 of *Applied Mathematical Sciences*. Springer-Verlag, 1993.

[7] O. Diekmann, S. A. van Gils, S. M. Verduyn Lunel, and H.-O. Walther. *Delay Equations: Functional-, Complex-, and Nonlinear Analysis*, volume 110 of *Applied Mathematical Sciences*. Springer-Verlag, 1995.

[8] K Engelborghs, T Luzyanina, and G Samaey. DDE-BIFTOOL v.2.00: a Matlab package for bifurcation analysis of delay differential equations. Report TW 330, Katholieke Universiteit Leuven, 2001.

[9] K. Engelborghs, T. Luzyanina, and D. Roose. Numerical bifurcation analysis of delay differential equations using DDE-BIFTOOL. *ACM Transactions on Mathematical Software*, 28(1):1–21, 2002.

[10] G. Samaey, K. Engelborghs, and D. Roose. Numerical computation of connecting orbits in delay differential equations. *Numer. Algorithms*, 30:335–352, 2002.

[11] D. Roose and R. Szalai. Continuation and bifurcation analysis of delay differential equations. In B Krauskopf, H M Osinga, and J Galán-Vioque, editors, *Numerical Continuation Methods for Dynamical Systems*: *Path following and boundary value problems*, pages 51–75. Springer-Verlag, Dordrecht, 2007.

[12] K. Verheyden, T. Luzyanina, and D. Roose. Efficient computation of characteristic roots of delay differential equations using lms methods,. *Journal of Computational and Applied Mathematics*, 214:209–226, 2008.

[13] F. Hartung, T. Krisztin, H.-O. Walther, and J. Wu. Functional differential equations with state-dependent delays: theory and applications. In P. Drábek, A. Cañada, and A. Fonda, editors, *Handbook of Differential Equations*: *Ordinary Differential Equations*, volume 3, chapter 5, pages 435–545. North-Holland, 2006.

[14] J. Sieber. Finding periodic orbits in state-dependent delay differential equations as roots of algebraic equations. *Discrete and Continuous Dynamical Systems A*, 32(8):2607–2651, 2012.