pde2path

Submission to DSWEB 2018 Contest – Software on Dynamical Systems, June 28, 2018

Hannes Uecker, Institut für Mathematik, Universität Oldenburg, Senior Faculty
Jens Rademacher, Fachbereich Mathematik & Informatik, University of Bremen, Senior Faculty
hannes.uecker@uni-oldenburg.de, jdmr@uni-bremen.de

1 Type of software, key features and short description

The software tool pde2path is a Matlab package for numerical continuation and bifurcation analysis of FEM based numerical discretizations for PDE systems on one-, two- or three-dimensional domains, possibly coupled with auxiliary constraint equations. pde2path can be used for the continuation of equilibria, traveling waves and time-periodic solutions. It can detect stationary bifurcations, possibly of higher multiplicity, as well as Hopf-bifurcations, and perform a branch switching. Depending on the system type, pde2path also allows for an easy switch between time-simulations and continuation.

The PDE system can have the form

$$M\partial_t u = -G(u,\lambda) := \nabla\cdot(c\otimes\nabla u) - au + b\otimes\nabla u + f, \tag{1}$$

where (the possibly singular) $M{\in}\mathbb{R}^{N\times N}$ can be seen as a mass matrix, $u{=}u(x,t) \in \mathbb{R}^N$ (N components), $t \geq 0$, $x \in \Omega \subset \mathbb{R}^d$, $d = 1,2,3$ (1D, 2D and 3D case, respectively), and $\lambda \in \mathbb{R}^p$ is a parameter vector. The coefficients c, a, b and the 'nonlinearity' f may depend on x, u, λ. The boundary conditions (BCs) for (1) can be (multiply-)periodic BC, if the domain is suitable, or be of the form

$$\mathbf{n}\cdot(c\otimes\nabla u) + qu = g, \tag{2}$$

where $\mathbf{n}$ is the outer normal so that Dirichlet BC can be approximated by (2) using a 'stiff spring' approximation. In (1), for instance in 2D, $[\nabla\cdot(c\otimes\nabla u)]_i := \sum_{j=1}^N[\partial_x c_{ij11}\partial_x + \partial_x c_{ij12}\partial_y + \partial_y c_{ij21}\partial_x + \partial_y c_{ij22}\partial_y]u_j$ (i^{th} component), and similarly $[au]_i = \sum_{j=1}^N a_{ij}u_j$, $[b\otimes\nabla u]_i := \sum_{j=1}^N[b_{ij1}\partial_x + b_{ij2}\partial_y]u_j$, and $f = (f_1,\ldots,f_N)$. In addition, $n_Q \geq 1$ auxiliary equations (typically constraints)

$$Q_j(u,\lambda) = 0, \quad j = 1,\ldots,n_Q \tag{3}$$

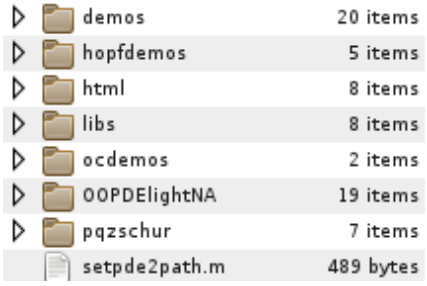
can be coupled to the PDE. In the following, when we refer to (1) this always includes the BC, i.e., periodic or (2), and (if applicable) the constraints (3).

The package pde2path is based on a spatial FEM discretization of (1), and arclength continuation similar as in the well-known continuation package AUTO, [DCF+97], which uses collocation and is designed for 1D general BVP. The FEM uses either Matlab's pdetoolbox (2D) or the FEM implementation OOPDE [Prü16] (1D, 2D and 3D, with unified user interfaces). Typical applications deal with systems leading to 10^2—10^3 (1D steady states), 10^4 (2D steady states, 1+1D Hopf), and 10^5 (3D steady states, 2+1D Hopf) degrees of freedom (DoF), though in some applications (in 3D) we use up to 10^6 DoF.

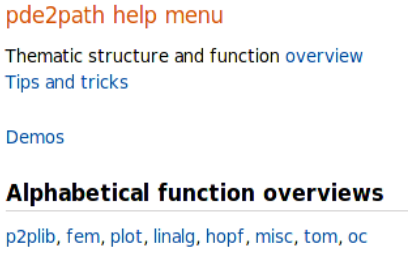
pde2path is designed to be a general and easy to use (and modify and extend) toolbox to investigate bifurcations in PDEs of the (rather large) class given by (1). Thus, pde2path is intended as and used as both, a research tool to study a given PDE system, and as a modular platform to implement and test new algorithms in the context of bifurcations and dynamic stability of solutions of PDEs. Included with the software download at [Uec18d] we provide about 40 demo directories dealing with a variety of systems, a concise Matlab (html) help system, and additionally, also for download at [Uec18d], a Quickstart Guide and a number of tutorials (about 150 pages altogether) explaining the demos and a number of algorithms behind pde2path.

Download and installation. The package download `pde2path.tar.gz` (or `pde2path.zip`) unpacks to the directory `pde2path`, which contains the directory tree shown in Fig. 1(a). In this tree, `demos` and `hopfdemos` contain a number of stationary and Hopf `pde2path` demos, respectively, `html` contains help, `libs` contains the `pde2path` libraries, `ocdemos` contains optimal control demos, `pqzschur` (included with permission) contains the periodic QZ decomposition from [Kre01], and `OOPDElightNA` (included with permission) is our "light" version of `OOPDE` [Prü16], without abstract classes.

(a) Directory tree



(b) root help menu



(c) demo overview (start)

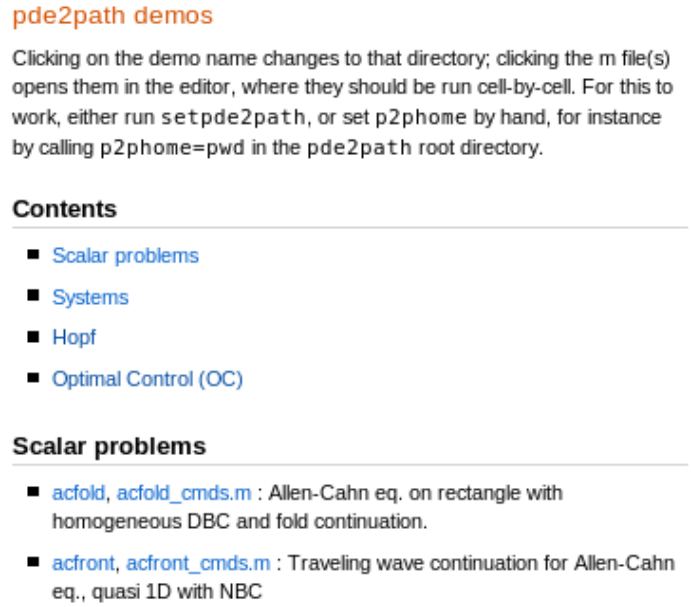


Figure 1: Directory tree, Root help menu, and starting part of html demo overview.

For advanced features, `pde2path` uses two third party softwares which need to mexed (see, e.g., README in directory `pqzschur` for comments on mexing and further instructions):

- `pqzschur` [Kre01]: a `Matlab` driver for a fortran routine which computes a periodic Schur decomposition of a set of matrices; optionally used to compute Floquet multipliers.
- `ilupack` [Bol11]: a preconditioned iterative linear system solver toolbox, which is not included in the download; it is optionally used for solving large scale linear systems.

We have tested `pde2path` on a variety of standard PCs (with different linux distributions, MacOS and Windows), and on some HPC clusters, and under various `Matlab` versions (R2009a and later).

Getting started. Within `Matlab` change into the `pde2path` directory and run `setpde2path`, which also makes available the help system. Calling `p2phelp` yields the main help menu shown in Fig. 1(b). The first two topics are short thematic overviews of the data structures and main functions in `pde2path`, while clicking `p2plib`, ..., `tom` yields complete alphabetic function overviews of these `pde2path` libraries, with a short description of each function, which can then be clicked for further documentation.[1] Similarly, clicking on `demos` opens the demo overview in (c), with brief descriptions of and links to the demo directories and script files.

The basic flow of running a model with `pde2path` is sketched in Fig. 2. For new users, we believe that the best way to understand this scheme is to work through a number of demo problems, where

[1]Help on any `pde2path` function `foo` is also given by typing `help foo` or `doc foo`, but in practice we find the alphabetic library overviews such as in Fig. 1(c) most convenient. To keep the help system functional, `clear all` should be avoided, i.e., replaced by `keep pphome`.

the `demos/acsuite` with the tutorial [RU18] is the easiest place to start. In any demo, open the `cmdsXX.m` script and execute cell after cell, rather than running directly the entire script.

To set up your own problem, create a copy of the demo directory closest to your problem and modify its functions and scripts.

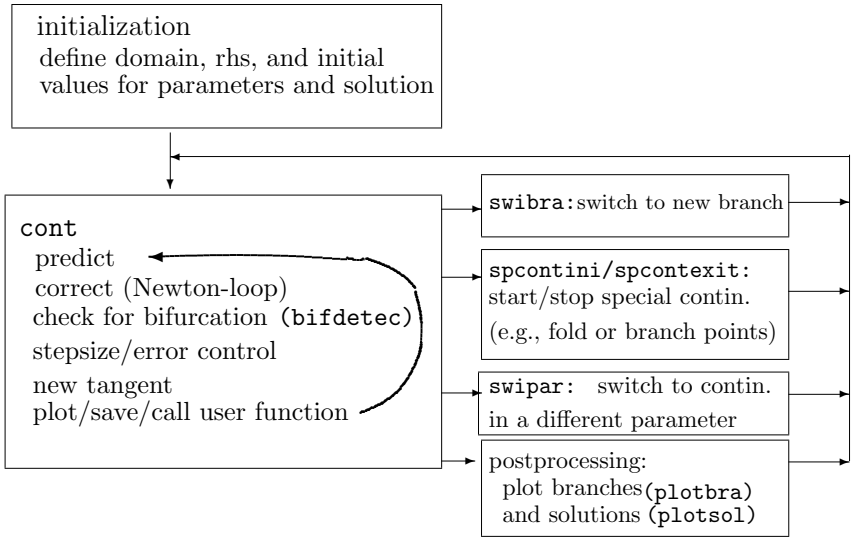


Figure 2: Basic flow diagram of using `pde2path`. The initialization block is typically put into a function `*init`, where `*` is the name of the problem, and it usually starts with a call of `p=stanparam(p)`, setting all `pde2path` parameters to standard values, after which the user should redefine the pertinent problem parameters. The `cont` block gives a schematic overview of the main steps in the continuation routine `cont`, where the loop is executed for a defined number of steps, or until some other criterion is fulfilled. To the right there are four typical next steps after an initial (or subsequent) run of `cont`, where (a), (b) naturally assume that in some run of `cont` a branch point (BP) or fold point (FP) has been found. The postprocessing is an integral part of `pde2path` and makes use of the powerful plotting functionality of `Matlab`. After each of these commands, `cont` can be called again to continue new branches (or further extend those already given). For convenience, all these commands are typically put into a script file `cmds.m` in "cell mode", i.e., where (groups of) commands are executed individually. This scheme basically applies to all demo directories, with some modifications, e.g.: for BPs of higher multiplicity we use `qswibra` or `cswibra` for branch switching, and in the Hopf demos `swibra` and `plotsol` are replaced by the Hopf versions `hoswibra` and `hoplot`.

History and contributors. The first version of `pde2path` was launched in November 2013, was restricted to simple steady bifurcation in 2D, and has essentially been documented in [UWR14]. Since then, the development has been rather fast, with a software upgrade about every 12 months, and we somewhat changed the documentation method and put an emphasis on tutorials, which are also available at [Uec18d], and if necessary are updated with the releases. The main additions have been:

- Implementation of the constraints (3), in particular used for fold–continuation [DRUW14], of periodic BC [DRUW14, DU17], and of continuous symmetries [RU17]. This also includes the option for time-simulation with 'freezing' in order to time-step orthogonal to the group orbit.
- Linking with `OOPDE` to efficiently treat the 1D, 2D and 3D cases, with unified interfaces [RU18].
- Treatment of infinite time horizon PDE-constrained optimal control problems [Uec17a].
- Treatment of Hopf bifurcation and periodic orbit continuation [Uec18a, Uec17b], and of steady bifurcation points of higher multiplicity [Uec18c, Uec18b].

The core developer team of `pde2path` consists of us two (HU, JR); major contributions are also due to

- Prof Tomas Dohnal, University of Halle, Senior Faculty;
- Lars Siemer, PhD student, University of Bremen;
- Dr Daniel Wetzel, PostDoc, and Hannes de Witt, PhD student, both University of Oldenburg.

We estimate that `pde2path` is currently used as a standard tool in about 10-15 research groups worldwide, and that about 8-10 PhD thesis have been written or are currently written using `pde2path` as a major tool.

Hints to detailed instructions, tutorials and manuals. For detailed instructions on installation, and a demo and library overview we refer to [dWDR+18]* (all tutorials and preprints are available at [Uec18d], and documents marked with * are also included with this submission). To get started, we then recommend the tutorials as follows:

- [RU18]* deals with scalar (Allen-Cahn type) problems. We start with a simple semilinear 1D problem with homogeneous Neumann BCs, and then proceed step by step to more complicated problems in 1D, 2D and 3D, with x–dependent terms, with various (including nonlinear) BCs, and quasilinear versions. See also [DU17] for similar setups with periodic BC in 1D–3D.
- [Uec18c]* extends [RU18] to more complicated systems such as reaction–diffusion systems and Swift–Hohenberg type of equations, including the handling of steady bifurcations of higher multiplicity (see also [Uec18b]), which naturally occur for pattern formation problems in $d \geq 2$ space dimensions.
- [Uec17b]* explains how to deal with Hopf bifurcations, see also [Uec18a].

A number of further more specialized tutorials is available at [Uec18d], for instance on (pattern formation in) optimal control problems [Uec17a], and on the handling of continuous symmetries by setting up suitable constraints (3) [RU17]. Additionally, [EGU+19] contains a review of a variety of `pde2path` applications to scalar problems, in particular thin–film equations.

2 Recent highlights

Modulated traveling waves. In the demo `demos/symtut/modfro` we compute modulated traveling waves via Hopf bifurcations from traveling waves (TWs) in a model from [BCM99], namely

$$\partial_t u = a\partial_x^2 u - uf(v), \quad \partial_t v = \partial_x^2 v + uf(v), \tag{4}$$

on the interval $\Omega = (-l_x, l_x)$ with BC $(u, v) = (0, 1)$ at $x = -l_x$ and $(u, v) = (1, 0)$ at $x = l_x$, which fixes a boost invariance of (4). Here $f(v) = v^m$ for $v \geq 0$ and 0 otherwise, $m \geq 2$ (here $m = 9$). In order to approximate the behavior on the real line we use $l_x = 25$, which seems large enough. The numerical results described next are summarized in Figure 3 (from [RU17]).

We use time-simulation with freezing [BT07] in order fix the location in the domain and identify the TW speed, which converges to a TW for $a \approx 0.2$ with (constant) speed s. Since the real line problem possesses spatial translation symmetry, we add the phase condition $0 = q(u) := \langle \partial_x u_0, u - u_0 \rangle$ (with a reference profile u_0), which enforces continuation orthogonal to the corresponding near translation mode on the domain with Dirichlet BC. Continuation to lower values of a (with adaptive mesh refinement to deal with sharp interfaces) yields a Hopf bifurcation point and branch switching gives a modulated TW. Here the speed s is an average speed corresponding to the constraint

$$q_H(u) := \sum_{i=1}^{m-1} \langle \partial_x u_0, u(t_i) \rangle \stackrel{!}{=} 0, \tag{5}$$

where $t_1, \ldots, t_m$ denotes the gridpoints of the time discretization. See [RU17] for details. In the specific example, for, e.g., $a = 0.12$, also time simulation converges to the stable modulated TW;

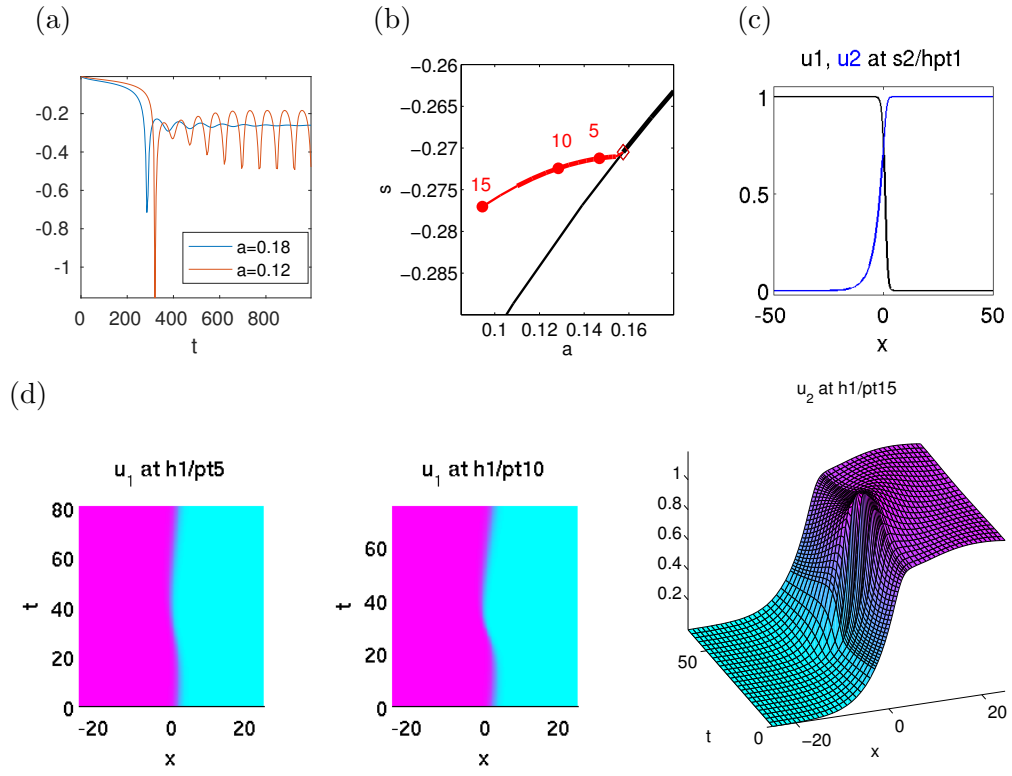


Figure 3: Results for (4) from demo `symtut/modfro` [RU17]. (a) Speed $s(t)$ returned from time simulation with freezing for $a = 0.18$ (convergence to steady TW) and for $a = 0.12$ (modulated TW). (b) Bifurcation diagram of steady TW (black) with speed=average speed s and modulated TW (red); stable branches as thicker lines. (c) Profile at Hopf bifurcation point in (b). (d) Solutions on modulated TW branch.

notably, the oscillations in the speed s are rather large, see Fig.3(a). For sufficiently fine time–discretization, the stability obtained from the Floquet multipliers indicated by thick lines in (b) agrees with the stability obtained from the time simulation with freezing, as do the instantaneous speeds $s(t) = \dfrac{\langle \partial_x u_0, G(u(t))\rangle}{\langle \partial_x u_0, \partial_x u\rangle}$, computed a posteriori.

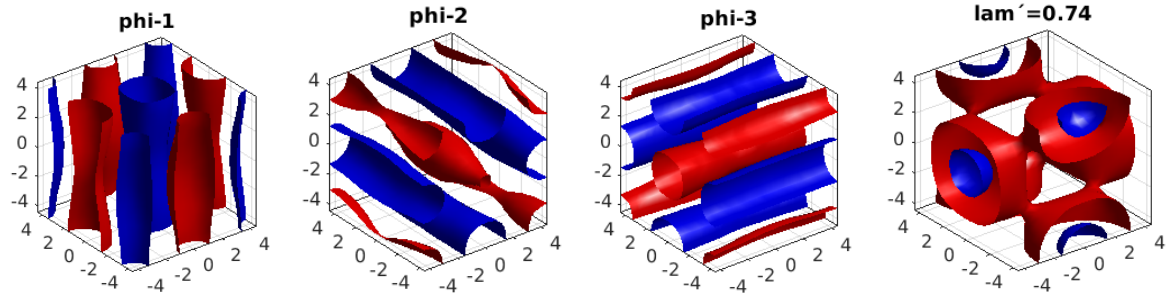
Bifurcation of Turing patterns in 3D under symmetry. One of the main original goals of `pde2path` was to study Turing patterns in reaction–diffusion systems and other pattern forming systems, and this continues to be a key application area. For 2D and 3D domains with symmetries, such as squares and cubes, an interesting issue is the competition between different types of patterns, e.g., 'spots vs stripes' [Erm91], see also [GS02]. The primary bifurcation from a trivial branch then usually is associated to eigenvalues of higher multiplicity. We recently added support for these to `pde2path`, explained in detail in [Uec18b] and [Uec18c]. In Fig. 4 we show just one example, namely the primary bifurcations at $\lambda = 0$ from the trivial branch $u \equiv 0$ in the (quadratic-cubic) Swift-Hohenberg (SH) equation

$$\partial_t u = -(1+\Delta)^2 u + \lambda u + \nu u^2 - u^3, \quad u = u(x,t) \in \mathbb{R},\ x \in \Omega \subset \mathbb{R}^3, \tag{6}$$

with Neumann BC $\partial_n u|_{\partial\Omega} = \partial_n(\Delta u)|_{\partial\Omega} = 0$. Due to the flexibility gained by the mass matrix M on the left hand side of (1), (6) can equivalently formulated as a second order system in the form (1). The demo directory `demos/sh` contains scripts to study a multitude of patterns for (6), including various snaking branches of localized patterns in 1D, 2D, and 3D. In Fig. 4 we use $\Omega = (-\sqrt{2}\pi, \sqrt{2}\pi)^3$, i.e., a cube corresponding to a so–called body-centered-cubic (BCC) wave vector lattice. In this situation, the kernel of the linearization $L = -(1+\Delta)^2$ of (6) at $(u,\lambda) = (0,0)$ is three dimensional, for instance

spanned by the tube (or more precisely square prism) $\phi_1 \sim \cos((x+y)/\sqrt{2}) + \cos((x-y)/\sqrt{2})$ and its two other orientations, see Fig. 4(a). Then, in total there are 15 branches bifurcating at $(u, \lambda) = (0, 0)$, but only 2 different equivariance classes defined by symmetries, i.e., tubes and BCCs (right-most plot in (a)), where the BCC correspond to an equal amplitude superposition of these tubes; see Fig. 4(b) for some branches obtained.

(a) three kernel vectors on the BCC (sub)lattice, and bifurcation direction for a BCC



(b) BD of BCCs and square-prisms on BCC lattice, and example solutions

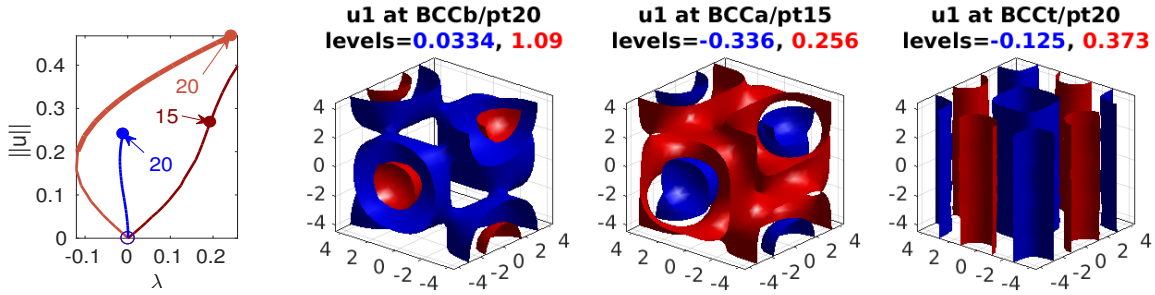


Figure 4: (6) on a "BCC lattice cube" $\Omega = (-\sqrt{2}\pi, \sqrt{2}\pi)^3$, $\nu = 1$. (a) kernel vectors (distorted square prisms) and bifurcation direction for a BCC obtained from `qswibra`. (b) BD, 'hot' and 'cold' BCCs and square prism example solutions. $n_p = 10351$, $n_t = 57024$ tetrahedra. From [Uec18c].

Similarly, the demo `demos/schnakpat` treats Turing patterns in the so–called Schnakenberg RD model. See [Uec18c] for details, which also contains remarks about the careful (symmetry preserving) meshing necessary for 2D and 3D problems with many solutions. A related problem is studied in [UW18], namely 3D patterns in the Brusselator RD system, including snaking branches of localized BCCs, see Fig. 5 for an illustration.

Hopf bifurcation in optimal control. An interesting if somewhat specialized area of application of `pde2path` are patterns in infinite time horizon distributed optimal control problems of the form

$$V(v_0(\cdot)) \stackrel{!}{=} \max_{k(\cdot,\cdot)} J(v_0(\cdot), \kappa(\cdot,\cdot)), \qquad J(v_0(\cdot), \kappa(\cdot,\cdot)) := \int_0^\infty \mathrm{e}^{-\rho t} J_{ca}(v(t), \kappa(t))\, \mathrm{d}t, \tag{7}$$

where $J_{ca}(v(\cdot,t), \kappa(\cdot,t)) = \dfrac{1}{|\Omega|}\displaystyle\int_\Omega J_c(v(x,t), \kappa(x,t))\, \mathrm{d}x$ is the spatially averaged current value function, J_c a given function, $\rho > 0$ is a discount rate, and where the states $v = v(t,x) \in \mathbb{R}^N$ fulfill some PDE of the form $\partial_t v = D\Delta v + g_1(v, \kappa)$ with control κ from some admissible set. Pontryagin's maximum principle leads to so–called canonical systems for the states v and co-states (or Lagrange–multipliers or shadow–prices) $w = w(t,x) \in \mathbb{R}^N$, i.e., formally,

$$\partial_t v = D\Delta v + g_1(v, w), \tag{8a}$$
$$\partial_t w = -D\Delta w + g_2(v, w), \tag{8b}$$

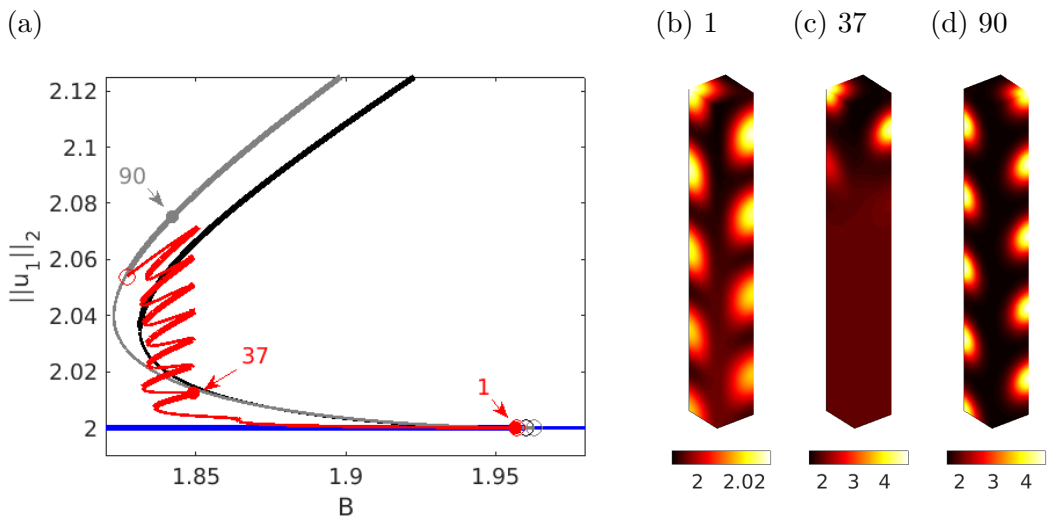


Figure 5: (a) Branches of the homogeneous solution (blue), the BCCs (black), and localized BCCs (red) for a Brusselator RD system over a cuboid $\Omega = (-l_x, l_x)^2 \times (-l_z, l_z)$ with Neumann BC, $l_x = \pi/(\sqrt{2}k_c)$, $l_z = 8l_x$, where k_c is the wave–number of a primary Turing bifurcation from the homogeneous solution branch. From [UW18].

where the dependence on the control $\kappa = \kappa(t, x)$ is hidden in the dependence on (v, w). Setting $u = (v, w) \in \mathbb{R}^{2N}$ we formally write (8) as $\partial_t u = -G(u)$. However, (8b) shows 'backward diffusion' such that (8) is not an initial value problem. Given initial *states* $v|_{t=0} = v_0$, a possible strategy to find pertinent (optimal) solutions of (8) is to

1. first compute canonical steady states (CSS), i.e., solutions $u^s = (v^s, w^s)$ of $G(u) = 0$;
2. compute canonical paths, i.e., solutions of (8) that connect some v_0 to some CSS u^s.

In, e.g., [Uec16], this method has been used to identify patterned optimal harvesting strategies in a semi-arid vegetation system.

The computation of steady states for (8) can be seen as a continuation/bifurcation problem for an elliptic system (by multiplying (8b) by -1), and hence as a standard application of `pde2path`. However, the backward diffusion in (8b) makes the computation of periodic orbits and their stability a hard problem, which in particular seems impossible to solve by shooting methods. `pde2path`'s Hopf algorithms are based on collocation boundary value problem solvers, which are indifferent to time reversal, but the computation of Floquet multipliers for Hopf orbits of systems of type (8) needs a particularly stable method for Floquet multiplier computations. In `pde2path`, this can be done using `pqzschur` from [Kre01].

As an example, in Fig. 6 we show results from the demo `hopfdemos/pollution`. Here we consider a problem for optimal pollution mitigation, where the states $v = (v_1, v_2) = (v_1(t, x), v_2(t, x))$ are the emissions of some firms and the pollution stock, respectively, and the control $\kappa = \kappa(t, x)$ models the firms' abatement policies. We refer to [Uec18a, §3.4] for details, and here only remark that:

- For suitable parameters, spatially patterned time-periodic orbits can be candidates for optimal solutions.
- Even on a 1D domain with a coarse spatial discretization of only 17 points the Floquet multipliers of typical Hopf–orbits range from 10^{-40} to 10^{40}, i.e., across 80 orders of magnitude.

References

[BCM99] N. J. Balmforth, R. V. Craster, and S. J. A. Malham. Unsteady fronts in an autocatalytic system. *R. Soc. Lond. Proc. Ser. A Math. Phys. Eng. Sci.*, 455(1984):1401–1433, 1999.

[Bol11] M. Bollhöfer. ILUPACK V2.4, `www.icm.tu-bs.de/~bolle/ilupack/`, 2011.

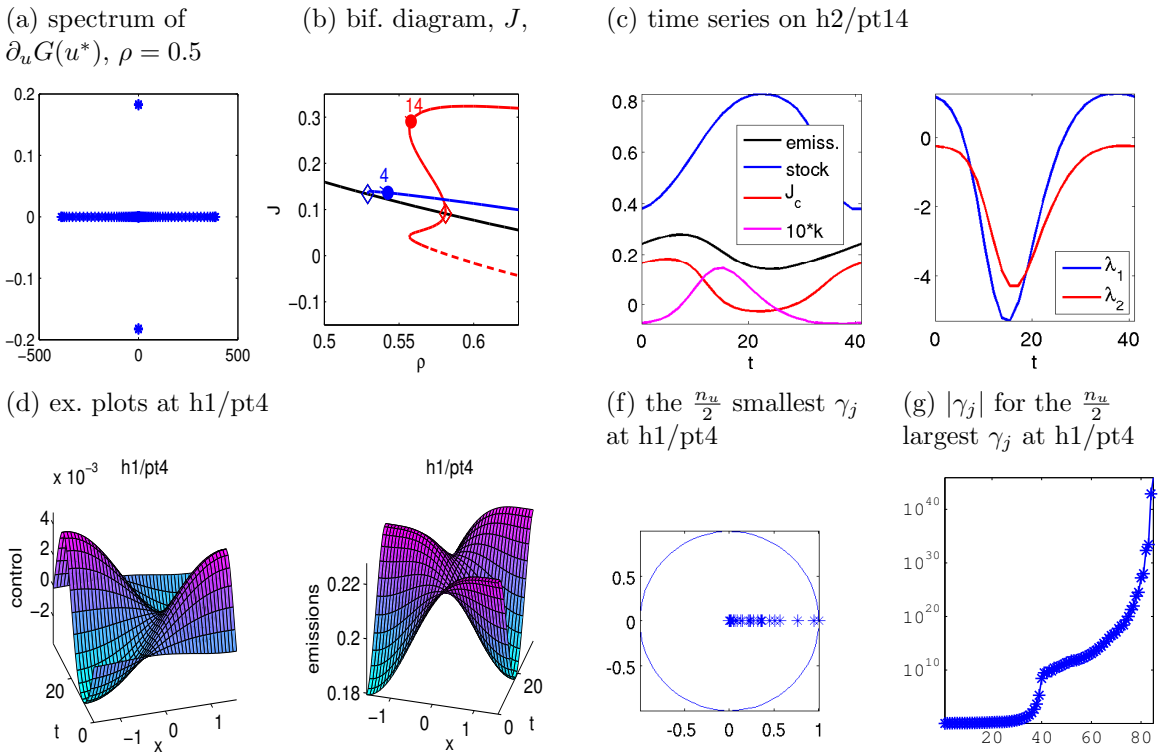


Figure 6: (a) full spectrum of the linearization of the CS around a steady state u^* on a coarse mesh with $n_p = 17$. (b) Bifurcation diagram, value J over ρ. A CSS and two Hopf-orbits. (c) Time series on a spatially homogeneous Hopf-orbit, including current value J_c, control k, and co–states $\lambda_{1,2} = w_{1,2}$. (d,f,g) Example plots and multipliers of some Hopf orbit u_H. From [Uec18a].

[BT07] W.J. Beyn and V. Thümmler. Phase conditions, symmetries, and PDE continuation. In *Numerical continuation methods for dynamical systems*, pages 301–330. Springer, Dordrecht, 2007.

[DCF+97] E. Doedel, A. R. Champneys, Th. F. Fairgrieve, Y. A. Kuznetsov, Bj. Sandstede, and X. Wang. AUTO: Continuation and bifurcation software for ordinary differential equations (with HomCont). `http://indy.cs.concordia.ca/auto/`, 1997.

[DRUW14] T. Dohnal, J.D.M. Rademacher, H. Uecker, and D. Wetzel. pde2path 2.0. In H. Ecker, A. Steindl, and S. Jakubek, editors, *ENOC 2014 - Proceedings of 8th European Nonlinear Dynamics Conference, ISBN: 978-3-200-03433-4*, 2014.

[DU17] T. Dohnal and H. Uecker. Periodic boundary conditions in pde2path, 2017.

[dWDR+18] H. de Witt, T. Dohnal, J.D.M. Rademacher, H. Uecker, and D. Wetzel. pde2path - Quickstart guide and reference card, 2018.

[EGU+19] S. Engelnkemper, S. V. Gurevich, H. Uecker, D. Wetzel, and U. Thiele. Continuation for thin film hydrodynamics and related scalar problems. In *Computational Modeling of Bifurcations and Instabilities in Fluid Mechanics*, Computational Methods in Applied Sciences, 50, pages 459–501. Springer, 2019.

[Erm91] B. Ermentrout. Stripes or spots? Nonlinear effects in bifurcation of reaction-diffusion equations on the square. *Proc. R. Soc. Lond., Ser. A*, 434(1891):413–417, 1991.

[GS02] M. Golubitsky and I. Stewart. *The symmetry perspective*. Birkhäuser, Basel, 2002.

[Kre01] D. Kressner. An efficient and reliable implementation of the periodic qz algorithm. In *IFAC Workshop on Periodic Control Systems*. 2001.

[Prü16] U. Prüfert. OOPDE, `www.mathe.tu-freiberg.de/nmo/mitarbeiter/uwe-pruefert/software`, 2016.

[RU17] J.D.M. Rademacher and H. Uecker. Symmetries, freezing, and Hopf bifurcations of modulated traveling waves in pde2path, 2017.

[RU18] J.D.M. Rademacher and H. Uecker. The OOPDE setting of pde2path – a tutorial via some Allen-Cahn models, 2018.

[Uec16] H. Uecker. Optimal harvesting and spatial patterns in a semi arid vegetation system. *Natural Resource Modelling*, 29(2):229–258, 2016.

[Uec17a] H. Uecker. Infinite time–horizon spatially distributed optimal control problems with pde2path – a tutorial, 2017.

[Uec17b] H. Uecker. User guide on Hopf bifurcation and time periodic orbits with pde2path, 2017.

[Uec18a] H. Uecker. Hopf bifurcation and time periodic orbits with pde2path – algorithms and applications, *Comm. in Comp. Phys*, to appear, 2018.

[Uec18b] H. Uecker. Multiple bifurcation points in pde2path, Preprint, 2018.

[Uec18c] H. Uecker. Pattern formation with pde2path – a tutorial, 2018.

[Uec18d] H. Uecker. `www.staff.uni-oldenburg.de/hannes.uecker/pde2path`, 2018.

[UW18] H. Uecker and D. Wetzel. Snaking branches of localized body-centered cubes, in preparation, 2018.

[UWR14] H. Uecker, D. Wetzel, and J.D.M. Rademacher. pde2path – a Matlab package for continuation and bifurcation in 2D elliptic systems. *NMTMA*, 7:58–106, 2014.