Numerical continuation for the spatial model of vegetation-water-herbivore system

Joydeep Singha[1], Hannes Uecker[2], and Ehud Meron[1]

[1]The Swiss Institute for Dryland Environmental and Energy Research, BIDR, Ben-Gurion University of the Negev, Sede Boqer Campus, Israel
[2]Institut für Mathematik, Universität Oldenburg, Germany

Corresponding authors: joydeepsingha105@gmail.com

Abstract

We describes the codes for the manuscript *"Traveling vegetation–herbivore waves may sustain ecosystems threatened by droughts and population growth"* by Singha et al. The methods include numerical continuation using `pde2path`.

1 Brief description of the model

In [SUM24] we consider a spatially explicit model that describes the coupled dynamics of vegetation, soil water, and herbivores in dryland ecosystems. The model captures two critical stressors—limited water availability and herbivory—and incorporates feedback mechanisms that can give rise to spatial self-organization. In particular, the model accounts for a behavioral component of herbivore movement, referred to as "vegetaxis," in which herbivores are attracted to denser vegetation patches. This feature plays a crucial role in shaping vegetation–herbivore patterns.

Let $B(x,t)$, $W(x,t)$, and $H(x,t)$ represent the vegetation biomass density, soil water content, and herbivore biomass density, respectively, at spatial location x and time t. The governing equations are:

$$
\begin{aligned}
\partial_t B &= \Lambda BW(1+EB)^2\left(1-\frac{B}{K_B}\right) - M_B B + D_B\nabla^2 B - G(B)H,\\
\partial_t W &= P - \frac{NW}{1+RB/K_B} - \Gamma BW(1+EB)^2 + D_W\nabla^2 W,\\
\partial_t H &= -M_H H + AG(B)H\left(1-\frac{H}{K_H}\right) - \nabla\cdot\mathbf{J}_H.
\end{aligned}
$$

Vegetation growth depends on local water availability and is enhanced by root development, modeled through the nonlinear term $\Lambda BW(1+EB)^2$. Growth is also limited by saturation effects and competition, captured by the factor $1-B/K_B$. Vegetation biomass decreases due to natural mortality at rate M_B and through consumption by herbivores, where the consumption rate $G(B)$ follows a saturating function:

$$
G(B) = \frac{\alpha B}{\beta + B},
$$

with α representing the maximum per capita consumption rate and β the biomass at which half-maximum consumption occurs.

The soil water balance includes a constant precipitation input P, reduced by evaporation, which is itself mitigated by shading from vegetation ($NW/(1+RB/K_B)$). Plants extract water from the soil via a nonlinear uptake term proportional to $BW(1+EB)^2$, while water diffuses laterally at a rate D_W.

Herbivores grow by consuming vegetation, reproducing at a rate proportional to the product $G(B)H$, moderated by density dependence via the factor $1-H/K_H$. Herbivore losses are due to natural mortality at rate M_H and spatial redistribution. Movement is governed by the flux term:

$$\mathbf{J}_H = -D_R(B)\nabla H + HD_V(B)\nabla B,$$

where $D_R(B)$ describes biomass-dependent random movement:

$$D_R(B) = D_{HH}\frac{H\xi^2}{\xi^2+B^2},$$

and $D_V(B)$ describes biased movement up vegetation gradients (vegetaxis):

$$D_V(B) = D_{HB}\frac{B}{\kappa+B}.$$

In this framework, herbivores tend to move more rapidly in bare-soil regions and slow down as they approach vegetation, while also being drawn toward denser patches. This behavior mimics an exploitation strategy, allowing herbivores to efficiently locate and graze on vegetation.

This system of equations can give rise to a rich variety of spatial and temporal patterns, including stationary vegetation stripes, localized herbivore aggregations, and traveling vegetation–herbivore waves. These emergent behaviors are critical for understanding the resilience of dryland ecosystems under increasing stress due to climate change and population-driven grazing pressure.

All scripts can simply be run by calling them from the `Matlab` command line; however, we strongly recommend running them in cell-mode from the `Matlab` editor, to see the results of individual cells and commands. For easy online use, the folder `bwhcont` contain cmds.m with cell blocks beginning with a brief description in each block.

The folder `bwhcont` contains all the necessary files to reproduce the bifurcation diagrams presented in the manuscript (see Table 1 for an overview). Before making any modifications, we recommend consulting the relevant documentation or references [Uec21], [Uec25] to become familiar with the `pde2path` command structure and workflow.

The function `bwinit.m` initialises the problem (see listing 1), with the inputs the domain size `lx`, the number of discretization points `nx`, the model parameter `par`, an initial homogeneous solution `[B, W, H]`, and the output folder name `'bwh'`. The basic procedure is to use `p = stanparam` to general a problem structure with standard values of numerical parameters and control switches to obtain the bifurcation branch. The default value of are often overwritten to make it suitable for the problem description, for example we have modified `hobra.m` to `hobrax.m` as function handler for output. The function `oosetfemops.m` creates the necessary finite element matrices such as stiffness matrix in `p.mat.K` and the mass matrix `p.mat.M` appropriate for periodic boundary condition (see listing 2).

```
function p=bwinit(lx,nx,par,varargin)
p=stanparam;  % initialize fields with defaults, then overwrite some:
p.nc.neq=3; p.sw.sfem=-1;
p.fuha.outfu=@hobrax; % mod of llibrary function hobra
p.pdeo=stanpdeo1D(lx,2*lx/nx); p.vol=2*lx; % standard 1D PDE object
p.np=p.pdeo.grid.nPoints; p.nu=p.np*p.nc.neq; % # PDE unknowns
p.sol.xi=1/p.nu; p.file.smod=1; p.sw.para=2; p.sw.foldcheck=1;
p.nc.ilam=6;  % use par(6) (P) for continuation
p.sol.ds=0.1; p.nc.dsmax=10.0; % initial and maximal stepsize
p.nc.lammin=0.0001; p.nc.lammax=500; % min and max values for cont parameter
p.nc.dlammax=10; %max step size in lam (bifurcation param)
p.file.smod=20; % save each smod'th cont.step, increase saves less data points
p.sw.bifcheck=2; %
p.nc.tol=1e-6; % tolerance
```

Table 1: Scripts and functions in `bwhcont/`

file	purpose, remarks
`cmds1`	continuation of homogeneous branches and branch switching to and continuation of Turing branches and travelling wave branches; to cont. speed
`cmds2`	branch, fold and hopf point continuation
`cmdsplot`	plot commands for the bifurcation diagrams
`cmdsdatext`	to extract data to text files
`bwhinit`	Initialisation of problem struct p with the required parameter values; generates the FEM matrices by calling `oosetfemops`; sets the `pde2path` parameters according the requirement of the problem
`oosetfemops`	creates the mass matrix M and the laplacian K for periodic boundary condition
`nodalf`	kinetic terms without the spatial derivatives
`sG`	right hand side of the PDE with the spatial terms in the moving co-ordinate system
`hobrax`	mod of library function hobra
`twswibrax`	mod of library function twswibra
`qf, qf1, qf1der, qjac`	standard phase condition for translational invariance in 'x'

```
p.sw.jac=0;    % switch for jacbian, zero as we use numerical jacobian (no sGjac)
p.nc.mu1=1;    % tolerance for entering BP localization by bisection
%default initial condition
p.sw.verb=2;
p.nc.neig=30; % #eigenvalues to compute, and reference point for this
p.nc.eigref=-4;
np=p.np; B=zeros(np,1); W=(par(6)/par(7))*ones(np,1); H=zeros(np,1);
p.u=[B; W; H; par']; % initial solution (bare soil) and parameters
p=box2per(p,1);      % switch on periodic BCs, this also calls oosetfemops
p.plot.pcmp=[1 2 3]; p.plot.cl={'k','b','r'}; % plotting settings
screenlayout(p); p.sw.verb=2; % place windows; choose verbosity of output
```

Listing 1: `/bwhcont/bwinit.m`

```
function p=bwinit(lx,nx,par,varargin)
p=stanparam;  % initialize fields with defaults, then overwrite some:
p.nc.neq=3; p.sw.sfem=-1;
p.fuha.outfu=@hobrax; % mod of llibrary function hobra
p.pdeo=stanpdeo1D(lx,2*lx/nx); p.vol=2*lx; % standard 1D PDE object
p.np=p.pdeo.grid.nPoints; p.nu=p.np*p.nc.neq; % # PDE unknowns
p.sol.xi=1/p.nu; p.file.smod=1; p.sw.para=2; p.sw.foldcheck=1;
p.nc.ilam=6;  % use par(6) (P) for continuation
p.sol.ds=0.1; p.nc.dsmax=10.0; % initial and maximal stepsize
```

Listing 2: `/bwhcont/oosetfemops.m`

2 Demo bifurcation diagrams

Using the bare soil solution as the initial guess, we first continue first on the homogeneous branch, (see listing 3).

```
%% Please each section of this code to get the branches of
close all; keep pphome;rng(42);
%% specification of the system, parameter values, domain size, grid size
Lambda=0.5;             % rate of vegetation growth per unit soil water
E=10;                   % root to shoot ratio
```

```
K=0.9;                      % maximal standing biomass per unit area
M=11.4;                    % plant mortality rate
DB=1.2;                    % seed dispersal rate
P=0.0;                     % precipitation
N=20.0;                    % evaporation rate
R=0.01;                    % Reduction in evaporation due to shading (dimensionless)
GamW=10.0;                 % water uptake rate
DW=150;                    % lateral soil water diffusion
mH=0.06;                    % herbivore mortality rate
GamH=0.3;                  % fraction of consumed biomass used in herbivore production
AlphaH=0.603;                 % maximum rate of plant consumption per unit herbivore
BetaH=0.82;                 % satiaton biomass
DHH=400;               % maximal random motility
Zeta=0.001;                % reference biomass at which the motility drops to 50%
DHB=700;                 % maximal vegetaxis motility
k=0.0001;                  % reference biomass at which the motility drops to 50%
K_H=175;                   % maximum herbivore capacity per unit area
s=0;                       % speed
% parameters are refered to by their number in p.nc.ilam; note numbers here!
par=[Lambda, E, K, M, DB, P, N, R, GamW, DW, mH, GamH, AlphaH, BetaH, DHH, Zeta, DHB, k, K_H, s];
%       1                 6                       12                              17
nx=256; lx=pi/2;
p=bwinit(lx,nx,par); p=setfn(p,'bwh3/BS');% init and initial folder-name
% Bare soil solution branch (BS)
p=cont(p,1000);     %continuation with number of steps
```

Listing 3: `/bwhcont/cmds1.m` continued; swibra to UV;

From the first branch point of the `BS` branch, we continue the `UV` branch (see Listing 4).

```
%% branch switching to uniform vegetation solution branch without herbivore (UV)
% arguments: input dir, branch point no, output dir, initial stepsize
p=swibra('bwh3/BS','bpt1','bwh3/UV',-0.05); pause; % pause to inspect kernel in Fig.6
p.nc.dsmax=1.0; p=cont(p,100000);
```

Listing 4: `/bwhcont/cmds1.m` continued; swibra to UV;

From the `UV` we then continue to the `UH` branch (see listing 5).

```
%% Uniform solution branch with vegetation and herbivore (UH) using swibra command
p=swibra('bwh3/UV','bpt3','bwh3/UH',-0.01); p.nc.dsmax=5.0;
p.file.smod=20; p=cont(p,10000);
```

Listing 5: `/bwhcont/cmds1.m` continued; swibra to UH;

Here if `swibra` does not work, we can manually switch to the `UH` branch by using `loadp` jump from the branch point to `UH` (see listing 6).

```
%% Uniform solution branch with vegetation and herbivore (UH) through manual initialisation if swibra does not work
p=loadp('bwh2/UV','bpt3','bwh2/UH'); %load from a pt in the in dir to an out dir
p.u(p.nu+6)=p.u(p.nu+6)+1.0; %manually increasing the lam
n=p.nu/3; p.u(2*n+1:3*n)=2.0;p=resetc(p); %changing H and resetting struc count
p.sw.bifcheck=2;
p.sol.ds=1.0; p.nc.dsmax=2;
p=cont(p,200);
```

Listing 6: `/bwhcont/cmds1.m` continued; manuall switching to UH;

At low precipitation, `UV` has a turing bifurcation point. Using `swibra` the SP_0 branch is obtained. Here also use the phase condition (see listing 7).

```
%% stationary periodic solution branch of vegetation without herbivore (SP0)
p=swibra('bwh3/UV','bpt2','bwh3/SP0',0.01); pause;
p.sw.bifcheck=0; p.nc.dsmax=0.1; p=cont(p,20);pause;
p.nc.nq=1; p.nc.ilam=[6,20]; %phase condition on and adding speed as another bifurcation parameter
```

```
p.fuha.qf=@qf1; p.fuha.qfder=@qf1der; %standard phase condition and derivative
p.sw.qjac=1;    %phase condition jacobian
p.sw.bifcheck=2;  p.nc.dsmax=5.0;p=cont(p,168);
```

Listing 7: `/bwhcont/cmds1.m` continued; swibra to SP_0;

Using the above settings for $\alpha = 0.48$ and setting the other parameters in set A, we obtain the complete bifurcation diagram (See Fig 1).

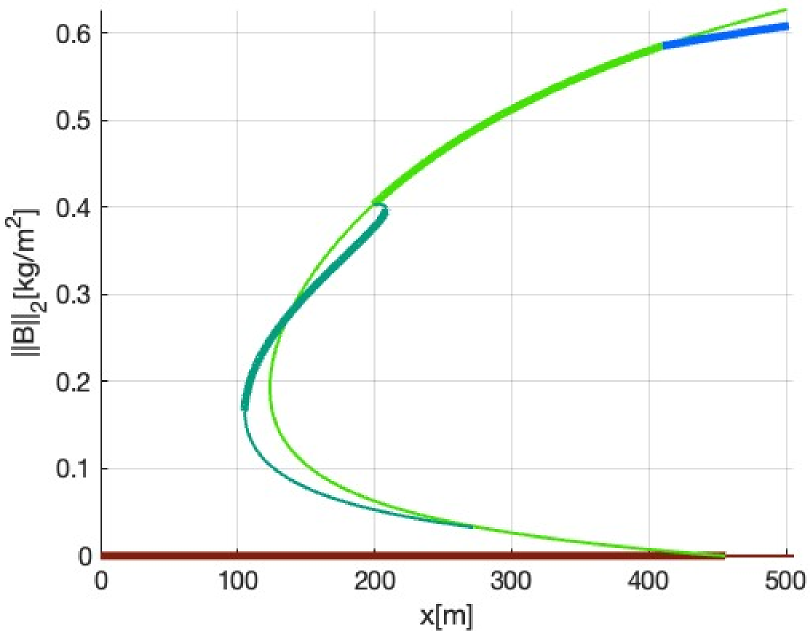

Figure 1: Bifurcation diagram obtained at $\alpha = 0.48$ using `plotbra` for parameters in set A . This is Fig. 3a of [SUM24].

For $\alpha = 0.57$, we get the 4 solution branches as before but SP_0 looses stability at low P. From here we access the SP_H solution branch using **swibra** from the corresponding branch point on SP_0 (see listing 8).

```
%% stationary periodic solution branch vegetation and herbivore (SPH)
p=swibra('bwh3/SP0','bpt1','bwh3/SPH',0.1); pause;
p.nc.tol=1e-8; p.nc.dsmax=1.0; p=cont(p,100);
```

Listing 8: `/bwhcont/cmds1.m` continued; swibra to SP_H;

This SP_H branch quickly looses stability and from there, the TW branch can be continued using **swibra** (see listing 9). In this way we can obtain the bifurcation diagram in Fig. 3b of [SUM24] (see figure 2).

```
%% travelling solution branch of both vegetation and herbivore (TW at low P)
% DRIFT bif., hence swibra, not twswibra
p=swibra('bwh3/SPH','bpt1','bwh3/TW_W',0.01); pause;
p.nc.tol=1e-5;p.nc.dsmax=1.0;
p.sw.bifcheck=2; p=cont(p,150);
```

Listing 9: `/bwhcont/cmds1.m` continued; swibra to TW at low P;

Setting, $\alpha = 0.603$, we again get all of the 6 solutions branches as before. However in this case UH solution looses stability to in hopf bifurcation. Using **twswibrax** we switch to the travelling wave branch here (see listing 10). In this way we can generate Fig. 4.

```
%% travelling solution branch from UH branch (TW at high P)
kwnr=1.0;      %guess for spatial wave number
aux.z=[1 -1i]; %auxilary arg for twswibrax, this comb of guesses of z1 and z2 works well
     for hp point on UH
```

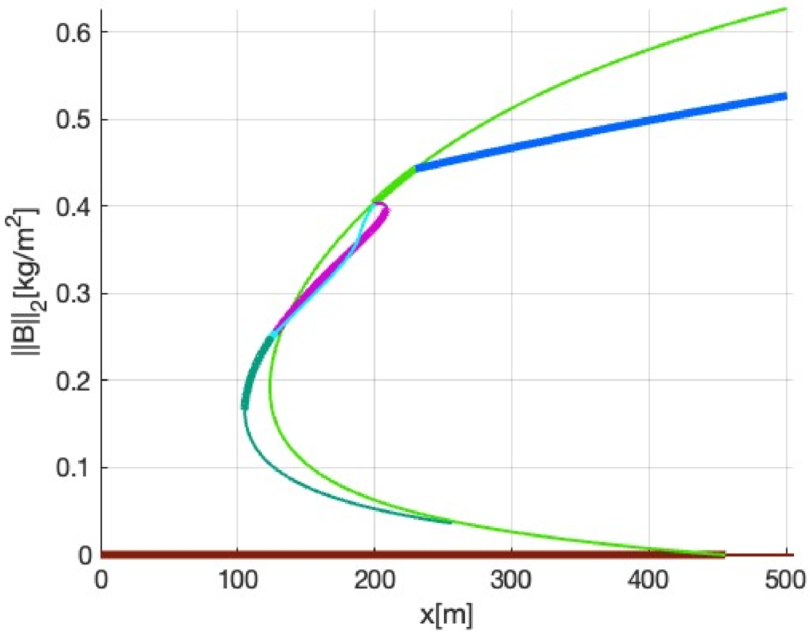

Figure 2: Bifurcation diagram obtained at $\alpha = 0.57$ for set A. (Fig. 3b of [SUM24]).

```
hp='hpt1'; outb='bwh3/TW_H'; %hopf point number;output directory
p=twswibrax('bwh3/UH',hp,20,kwnr,outb,aux); %branch switching from hopf point to
    travelling wave branch
p.sol.ds=0.1; p.nc.dsmax=0.1;
p.u0(1:p.nu)=p.tau(1:p.nu);     %reference profile
p.u0=p.u0'; p.u0x=p.mat.Kx*p.u0; % setting PC
plotsolu(p,p.u0x,1,3,1);
p.u(1:p.nu)=p.u(1:p.nu)+0.01*p.tau(1:p.nu);
p.nc.tol=1e-6; p.nc.nq=1;
p.nc.ilam=[6;20]; p.fuha.qf=@qf; p.sw.qjac=1; p.fuha.qfder=@qjac;
p.sw.bifcheck=2; pause;
p.nc.dsmax=0.2; p=cont(p,570);
par=getaux(p); par(20)          %to check speed
```

Listing 10: `/bwhcont/cmds1.m` continued; twswibrax to the TW branch at high P;

All of the bifurcation diagrams in [SUM24] (Figs. 3, 5, 6, 7, 14, 16) were obtained using the above for different values of parameters. To generate Fig. 16, low values of `p.sol.ds` and `p.nc.dsmin` on the order of 10^{-4} were used, as they were necessary to zoom in on a narrow range of P. For a few customisation advantage we use pyplot (script not provided but available upon request) after extracting the data in a text file (see listing 11).

```
%% solution branches
npt=3; nski=20; P=zeros(npt,3);         %array size depends on the number of points in
    the dir
for i=1:npt;
    pt=['bwh/BS/pt' mat2str((i-1)*nski)]; tmp=load(pt);tmpp=tmp.p;       %change input
    directory accordingly

    P(i,1)=tmpp.u(tmpp.nu + 6);

    np=p.nu/p.nc.neq;n=p.nu;

    P(i,2)=(tmpp.u(1:np)'*(p.mat.M(1:np,1:np)*(tmpp.u(1:np))))/tmpp.vol;
    P(i,2)=sqrt(P(i,2));

    P(i,3)=tmpp.sol.ineg;
```

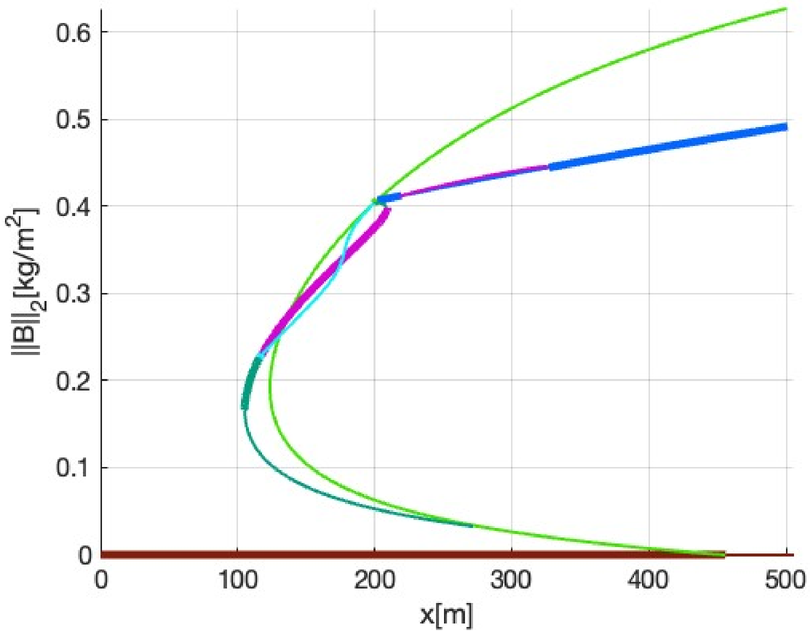

Figure 3: Bifurcation diagram obtained at $\alpha = 0.603$ for set A. (Fig. 3c of [SUM24]).

```
end
ptr=fopen('BS.txt','w');                %change output directory accordingly
fprintf(ptr,'%e\t%e\t%e\n',P');
fclose(ptr);
```

Listing 11: `/bwhcont/cmdsdatext.m` continued; bifurcation diagram data extraction;

Similarly the the solution profiles in Fig. 3, 5, 6, 7 are also obtained (see listing 12).

```
%% solution profile data
n = p.nu/3;
p=loadp('bwhcont/BS','pt198');
B = p.u(1:n);
W = p.u(n+1:2*n);
H = p.u(2*n+1:3*n);
ptr=fopen('bwhcont/pt198.txt','w');
fprintf(ptr,'%f\t%f\t%f\n',[B W H]');
fclose(ptr);
```

Listing 12: `/bwhcont/cmdsdatext.m` continued; solution profile data extraction;

Figures 10.b, c were obtained by changing `p.nc.ilam` corresponding to the number of position of D_{HB} with other corresponding parameters and changing the output component in `p.plot.bpcmp` to observe speed (see listing 13). Figure 15 is also obtained simply by changing `p.plot.bpcmp` for the TW branches for the corresponding value of the parameters.

```
%% continuation wrt DHB
p=loadp('bwh/TW','pt137','bwh/speed_DHB');
p.sol.ds=10.0;p.nc.ilam=[17,20];
p=resetc(p);p.nc.lammax=6000;pause;
p.nc.tol=1e-4;
p.sw.bifcheck=0; p.nc.dsmax=10.0;
p.plot.bpcmp=23;p=cont(p,3); pause;
p.nc.nq=1;  p.fuha.qf=@qf1;
p.fuha.qfder=@qf1der; p.sw.qjac=1; % switch on PC
p.sw.bifcheck=2;p=cont(p,1000);
```

Listing 13: `/bwhcont/cmds1.m` continued; speed vs D_{HB} diagrams;

As before, their data were extracted and plotted in pyplot.

3 Phase diagrams : branch point continuation

The phase diagrams in Figs. 9.a and 10.a in [SUM24] were mostly obtained using branch point continuation method in `pde2path` (see listings 14, 15, 16 for branch, fold and hopf point continuation respectively.). We advise caution for these continuations whenever `H` is non-zero or branch points are nearby as the `p.nc.ntol` needs to adjusted during continuation. Whenever there are convergence error the boundaries between these stable solutions are completed manually from the bifurcation diagram. We also advise that the boundary of the phase diagram should be checked always using direct numerical simulation whether above and below the boundary the desired solutions are obtained.

The listing 14 shows that `pde2path` commands for branch point continuation for branch point on UV which generates UH that runs for few continuation steps (see figure ??).

```
%% For branch point continuation to find variation in the position of the turing
    bifurcation point on UV
p=bpcontini('bwh1/UV','bpt3',13,'bwh1/Turing_UV'); %branch point continuation w.r.t
    parameter 13
plotsol(p); mclf(2); pause  %solution profile;to clear fig 2;
p.nc.dlammax=0.0001;p.nc.lammax=1.0;p.sw.bifcheck=0;
p.sw.foldcheck=0; %switch of fold
p.sol.ds=-0.01;
p.nc.tol=1e-7; p.nc.del=0.001; p.nc.njthreshsp=1e3;
p.sw.spjac=0; %for fuha.spjac
p.nc.dsmax=0.01; p.nc.dsmin=0.00001;
p.file.smod=1; p=cont(p,10);
```

Listing 14: `/bwhcont/cmds2.m` continued; branch point continuation;

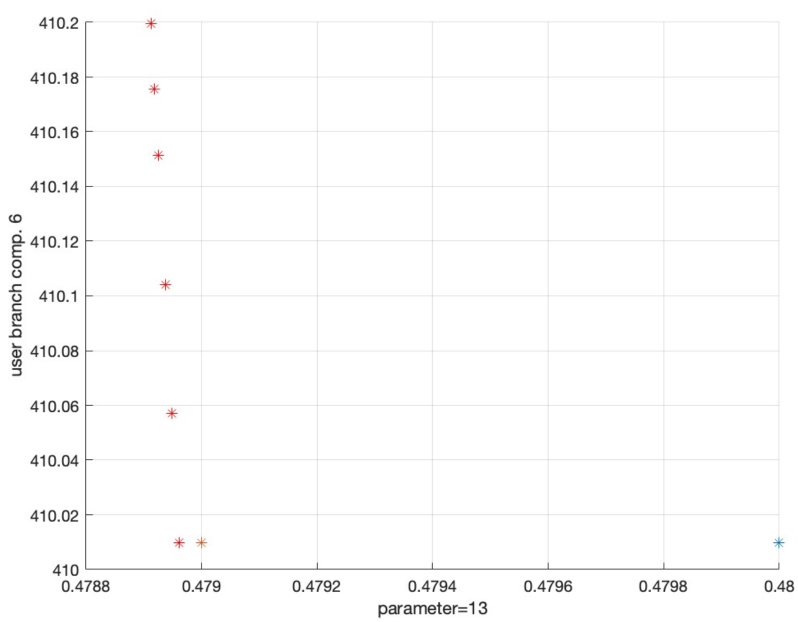

Figure 4: Branch point continuation.

Both the fold point continuation and hopf continuations (see listings 15, 16) runs for few steps initially however we get convergence errors so therefore we mostly use manual evaluation of the bifurcation diagrams and obtain the phase boundaries.

```
%% for fold point continuation
```

```
p=spcontini('bwh3/SP0','fpt2',13,'bwh3/SP0_fold_point'); %fold point cont.
huclean(p);
plotsol(p); pause
p.nc.dlammax=10; p.nc.lammax=1.5;
p.nc.del=1e-2;p.nc.njthresh=1e-2; p.nc.njthreshsp=1e5;
p.sol.ds=0.0001; p.plot.bpcmp=p.nc.ilam(2);
p.nc.tol=1e-5; p.sw.spjac=0; %for fuha.spjac
p.file.smod=1; p.sw.bifcheck=0; p.sw.foldcheck=0;
p.sw.verb=2; p.nc.dsmax=0.001; p=cont(p,50);
```

Listing 15: `/bwhcont/cmds2.m` continued; fold point continuation;

```
%% Hopf point continuation
p=hpcontini('bwh3/UH','hpt1',13,'bwh3/UH_hopf_point'); %hopf point cont
huclean(p); plotsol(p); p.nc.dlammax=0.0001;
p.nc.lammax=0.608;pause
p.plot.bpcmp=p.nc.ilam(2);
```

Listing 16: `/bwhcont/cmds2.m` continued; hopf point continuation;

References

[SUM24] J. Singha, H. Uecker, and E. Meron. Traveling vegetation–herbivore waves may sustain ecosystems threatened by droughts and population growth, 2024. Preprint.

[Uec21] Hannes Uecker. Numerical continuation and bifurcation in nonlinear pdes. *SIAM*, 2021.

[Uec25] Hannes Uecker. pde2path - a matlab package for continuation and bifurcation in system of pdes, v3.1. 2025.