Self-Organization and Stability of Dryland Ecosystems in Resisting Desertification

Sijia Zhang

June 2024

Abstract

Desertification is one of the most serious environmental problems we are facing today, and spatial self-organization of dryland vegetation is a key factor in determining it. This paper combines topographical data, optical data, and in situ biomass measurements from two sites in Somalia to generate an extended Klausmeier model for dryland vegetation patterning. Numerical analysis by finite difference method and traveling waves shows that the graph corresponds to physical reality. The analysis of error and stability analysis helps to build the analysis for multistability and find the countermeasures for desertification.

1 Motivation

Today, the Earth encounters many rigorous environmental problems, and one of them is desertification. Vast drylands in semiarid climates face the danger of becoming desert. The main reason behind this is land degradation because of human activities, like overgrazing, deforestation, agricultural activities, etc. To find proper countermeasures, it is crucial to understand the driving mechanisms behind this effect.

2 Background

Self-organization is an important phenomenon in ecology, enabling organisms to cope with harsh environmental conditions and buffering ecosystem degradation. The spatial self-organization of dryland vegetation constitutes one of the most promising indicators of an ecosystem's proximity to desertification.
There are two alternative models to describe spatial self-organization: (1) scale-dependent feedback, and (2) density-dependent aggregation. Scale-dependent feedbacks (SDFs) explain regular patterns observed in ecosystems with a flow-through of resources (such as water) that is concentrated by an ecosystem engineer. Density-dependent aggregation (DDA) explains irregular patterns observed in ecosystems with a mobile abiotic or biotic species that is either in a highly mobile state, depending on the density of a sessile species.

3 Case Study: Dryland Ecosystem

Dryland ecosystem patterns are a well-known example of spatial self-organization using scale-dependent feedbacks. The growth of plants is locally enhanced through the concentration of water by plants (called small-scale positive feedback), leading to a deprivation of water on a larger spatial scale (large-scale negative feedback).
Combining data and in situ biomass measurements from two sites in Somalia, this model is generated by the extended-Klausmeier model.

3.1 Model Description

Multiple reaction-diffusion models of dryland vegetation dynamics include a mechanism in which vegetation acts as an ecosystem engineer, locally increasing the influx of available water. Consider Klausmeier

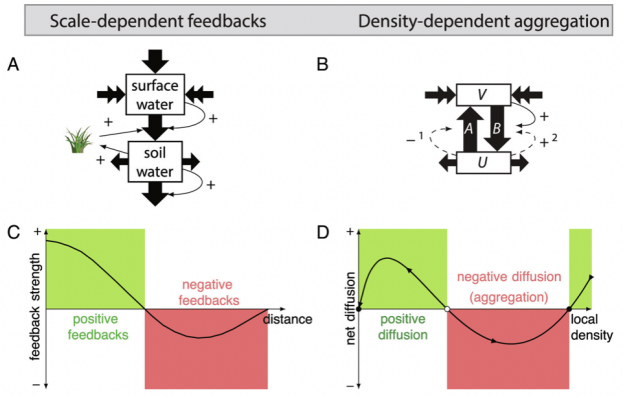

Figure 1: Graphs of SDFs and DDAs, where positive feedback means sufficient water and negative feedback means deprivation of water.

equation, a widely studied reaction-diffusion-advection model of vegetation pattern formation on gently sloped terrain in semiarid ecosystems, as the archetype, and add specific terms in this ecosystem condition, the model is created. This model describes the interaction between water w and plant biomass n.
The model is given by the equations:

$$\begin{cases} \frac{\partial w}{\partial t} = e\frac{\partial^2 w}{\partial x^2} + \frac{\partial (vw)}{\partial x} + a - w - wn^2 \\ \frac{\partial n}{\partial t} = \frac{\partial^2 n}{\partial x^2} - mn + wn^2 \end{cases} \tag{1}$$

The reaction terms model in change in water as a combination of the effect of rainfall (+a), evaporation (-w), and uptake by plants ($-wn^2$). Dispersion by plants is modeled as diffusion and the movement of water is a combined effect of diffusion and advection, which is the exchange of energy, moisture, or momentum as a result of horizontal heterogeneity and it is proportional to the slope parameter v.
The change in plant biomass comes from mortality (-mn) and plant growth ($+wn^2$).

4 Numerical Analysis

4.1 Finite Difference Method

Finite-difference method (FDM) is a class of numerical techniques for solving differential equations by approximating derivatives with finite differences. Since the given model is a system of diffusion-like equations, we can apply the FTCS (forward time-centered space) method to solve it numerically. The FTCS method is based on the forward Euler method in time (hence "forward time") and central difference in space (hence "centered space"), giving first-order convergence in time and second-order convergence in space.
Choose small spatial step size $\Delta x = h$ and time step size $\Delta t = k$. The first/second order derivatives can be expressed by:

- first-order time:

$$\frac{\partial W}{\partial t} = \partial_t W = \frac{W_j^{n+1} - W_j^n}{k}, \frac{\partial N}{\partial t} = \partial_t N = \frac{N_j^{n+1} - N_j^n}{k}$$

- first-order space:

$$\frac{\partial W}{\partial x} = \partial_x W = \frac{W_{j+1}^n - W_{j-1}^n}{2k}$$

- second-order space:

$$\frac{\partial W}{\partial xx} = \partial_{xx} W = \frac{\frac{W_{j+1}^n - W_j^n}{h} - \frac{W_j^n - W_{j-1}^n}{h}}{h} = \frac{W_{j+1}^n - 2W_j^n - W_{j-1}^n}{h^2}$$

$$\frac{\partial N}{\partial xx} = \partial_{xx} N = \frac{\frac{N_{j+1}^n - N_j^n}{h} - \frac{N_j^n - N_{j-1}^n}{h}}{h} = \frac{N_{j+1}^n - 2N_j^n - N_{j-1}^n}{h^2}$$

By FTSC finite difference method, equation (1) can be expressed as:

$$\begin{cases} \frac{W_j^{n+1} - W_j^n}{k} = e\frac{W_{j+1}^n - 2W_j^n - W_{j-1}^n}{h^2} + v\frac{W_{j+1}^n - W_{j-1}^n}{2k} + a - W_j^n - W_j^n (N_j^n)^2 \\ \frac{N_j^{n+1} - N_j^n}{k} = \frac{N_{j+1}^n - 2N_j^n - N_{j-1}^n}{h^2} - W_j^n (N_j^n)^2 \end{cases} \tag{2}$$

4.2 Error Analysis

In order to do error analysis, we need to check the numerical scheme $S_1(x,t)$ and $S_2(x,t)$. If our method was exact, $S_1(x,t)$ and $S_2(x,t)$ should be zero. But it's not, so anything left over is an error.

$$\begin{cases} S_1(x,t) = e\frac{W_{j+1}^n - \frac{W_j^{n+1} - W_j^n}{k} - 2W_j^n - W_{j-1}^n}{h^2} + v\frac{W_{j+1}^n - W_{j-1}^n}{2k} + a - W_j^n - W_j^n (N_j^n)^2 \\ S_2(x,t) = \frac{N_{j+1}^n - 2N_j^n - N_{j-1}^n}{h^2} + \frac{N_j^{n+1} - N_j^n}{k} - W_j^n (N_j^n)^2 \end{cases} \tag{3}$$

Apply Taylor expansions to W and N,

$$N_j^n = N(t_n, x_j)$$

$$N_{j+1}^n = N(t_n, x_j + h) = N(t_n, x_j) + \partial_x N h + \frac{1}{2!}\partial_x^2 N h^2 + \frac{1}{3!}\partial_x^3 N h^3 + \frac{1}{4!}\partial_x^4 N h^4 + \mathcal{O}(5)$$

$$N_{j-1}^n = N(t_n, x_j - h) = N(t_n, x_j) + \partial_x N(-h) + \frac{1}{2!}\partial_x^2 N h^2 + \frac{1}{3!}\partial_x^3 N(-h)^3 + \frac{1}{4!}\partial_x^4 N h^4 + \mathcal{O}(5)$$

$$N_j^{n+1} = N(t_n + k, x_j) = N(t_n, x_j) + \partial_t N k + \frac{1}{2!}\partial_t^2 N k^2 + \frac{1}{3!}\partial_t^3 N k^3 + \frac{1}{4!}\partial_t^4 N k^4 + \mathcal{O}(5)$$

We can express first/second-order derivatives by Taylor expansions. After simplification, we get:

$$\partial_t N = \partial_t N + \frac{1}{2!}\partial_t^2 N k + \frac{1}{3!}\partial_t^3 N k^2 + \frac{1}{4!}\partial_t^3 N k + \mathcal{O}(5)$$

$$\partial_x W = \frac{1}{2}\partial_x W + \frac{1}{2 \cdot 3!}\partial_x^3 W h^2 + \frac{1}{2 \cdot 5!}\partial_x^5 W h^4 + \mathcal{O}(6)$$

$$\partial_{xx} N = \frac{2}{2!}\partial_x^2 N + \frac{2}{4!}\partial_x^4 N h^2 + \mathcal{O}(6)$$

$$\partial_{xx} W = \frac{2}{2!}\partial_x^2 W + \frac{2}{4!}\partial_x^4 W h^2 + \mathcal{O}(6)$$

Thus, the numerical scheme of $S_1(x,t)$ and $S_2(x,t)$ are:

$$\begin{cases} S_1(x,t) = e(\partial_{xx} W = \frac{2}{2!}\partial_x^2 W + \mathcal{O}(4)) - (\frac{1}{2}\partial_x W + \frac{1}{2 \cdot 3!}\partial_x^3 W h^2 + \mathcal{O}(4)) - mN + WN^2 \sim \alpha_1 k^1 + \beta_1 h^2 \\ S_2(x,t) = \frac{2}{2!}\partial_x^2 N + \mathcal{O}(4) - (\partial_t N + \frac{1}{2!}\partial_t^2 N k + \frac{1}{3!}\partial_t^3 N k^2 + \mathcal{O}(3)) - mN + WN^2 \sim \alpha_2 k^1 + \beta_2 h^2 \end{cases} \tag{4}$$

Thus, the local error of equation (1) after applying FTCS finite difference method is $\mathcal{O}(k^1 + h^2)$, i.e. "first-order in time, second-order in space". Then a crude calculation argues that global error $\approx$ number of steps x error at each step, that is: $(T/h)\mathcal{O}(k^1 + h^2) \sim \mathcal{O}(k^1 + h^1)$, i.e. "first-order in time, first-order in space".

5 Numerical Simulation

5.1 By Finite Difference Method

Write a MATLAB code that applies the finite difference method to the equation.
The graphs follow the nature that water decreases in time since they are absorbed by plants and plants biomass increases because of the nourishment of water.

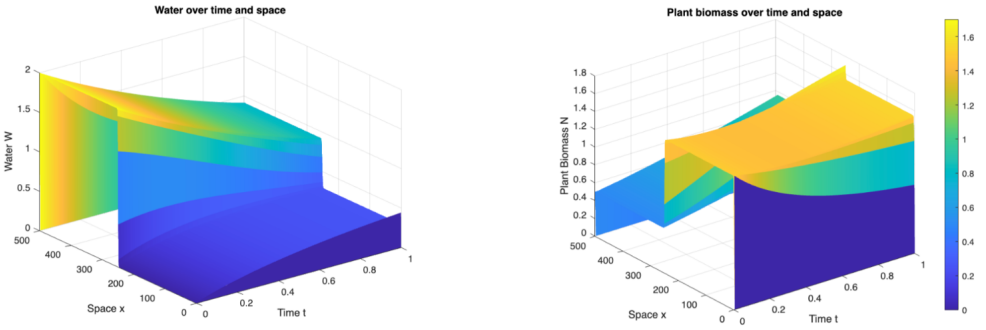

Figure 2: Numerical Analysis of equation (1) by finite difference method.

5.2 By Traveling Waves

A traveling wave is a solution of a Partial Differential Equation (PDE) that propagates with a constant speed while maintaining its shape in space. Traveling waves are a common phenomenon in biology, evidenced by the fact that Chapter 13 of the authoritative work "Mathematical Biology" by Murray is dedicated to biological waves. In the context of the multistability of ecosystems model, we can express graphs of water and plant biomass with the help of traveling waves.
Define a variable $z = x+ct$, where c is the traveling speed of the wave W and N. Thus, we can express W(x,t) and N(x,t) by W(z) and N(z). Define U(z) as an array with length $2N + 1$, where $U(z) = [W(z), N(z), c]$.
Apply FTCS scheme to equation (1). We can create $D1$ and $D2$ that approximate the first and second-order derivatives. Use $sparse$ to store only the nonzero elements of the matrix, together with their indices in order to reduce computation time by eliminating operations on zero elements.
By applying the finite difference method to equation (1), we can express (1) as:

$$\begin{cases} E_W : cD_1W - eD_2W - vD_1W - a + W + WN^2 = 0 \\ E_N : cD_1N - D_2N + mN - WN^2 = 0 \end{cases} \tag{5}$$

Since two equations E_W, E_N are not sufficient to solve for three unknown variablesW, N, c. We are going to introduce another equation E_C:

$$E_C : \begin{bmatrix} D_1W \\ D_2N \end{bmatrix}^T \cdot \left(\begin{bmatrix} W \\ N \end{bmatrix} - \begin{bmatrix} W_0 \\ N_0 \end{bmatrix} \right) = 0 \tag{6}$$

Introduce a new Matlab command "fsolve". fsolve uses the idea of Newton's method and attempts to solve a system of equations by minimizing the sum of squares of the components.
Solves a problem specified by F(x) = 0 for x, where F(x) is a function that returns a vector value and x is a vector or a matrix. Acting as a nonlinear system solver, fsolve can be used in the form:

$$x = fsolve(fun, x_0, options)$$

In order to make the code run faster, we introduce Jacobian to reduce computations. Jacobian can be expressed as:

$$Jacobian = \begin{pmatrix} D_WE_W & D_NE_W & D_CE_W \\ D_WE_N & D_NE_N & D_CE_N \\ D_WE_C & D_NE_C & D_CE_C \end{pmatrix}$$

where D_W, D_N, D_C represents derivatives over W, N, C, respectively.
Plug in equations (5) and (6), Jacobian is a $2N + 1$ by $2N + 1$ matrix:

$$Jacobian = \begin{pmatrix} cD_1 - eD_2 - vD_1 + 1 + spdigs(n^2, 0, Nx, Nx) & 2spdiags(WN, 0, Nx, Nx) & D_1W \\ -spdigs(n^2, 0, Nx, Nx) & cD_1 - D_2 - 2spdigs(WN, 0, Nx, NX) & D_1N \\ (D_1W)^T + (W - W_0)^T D_1 & (D_1N)^T + (N - N_0)^T D_1 & 0 \end{pmatrix}$$

where *spdiags* is a Matlab command used to extract nonzero diagonals and create sparse band and diagonal matrices.

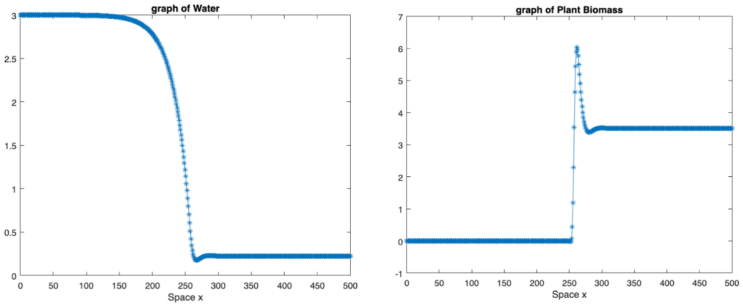

Figure 3: Graphs of water and plant biomass based on parameters $e = 50, v = 4, a = 3.0, m = 0.45$ and initial conditions $W_0 = 0.5 - 0.5 * tanh((x - L/2)), N_0 = 0.5 + 0.5 * tanh((x - L/2)), c_0 = 1$.

From the graphs, we can tell that the trend of water and plant biomass is basically in line with the reality that with plenty of water, plant biomass grows, and with the growth of plant biomass, the amount of water decreases. The trend of graphs also meets the output we got from the finite difference method.

5.3 Stability Analysis

To analyze the stability of the given equations, we generate the eigenvalues of the Jacobian matrix. We don't need c value here, thus, we generate a new matrix called "Jacobian-main".

$$Jacobian{-}main = \begin{pmatrix} cD_1 - eD_2 - vD_1 + 1 + spdigs(n^2, 0, Nx, Nx) & 2spdiags(WN, 0, Nx, Nx) \\ -spdigs(n^2, 0, Nx, Nx) & cD_1 - D_2 - 2spdigs(WN, 0, Nx, NX) \end{pmatrix}$$

If all eigenvalues have a negative real part, i.e. located on the right side of the graph, then we conclude that equation (1) is stable under the given conditions. If the eigenvalue(s) have a positive real part, i.e. located on the left side of the graph, then we conclude that equation (1) is unstable under the given conditions.

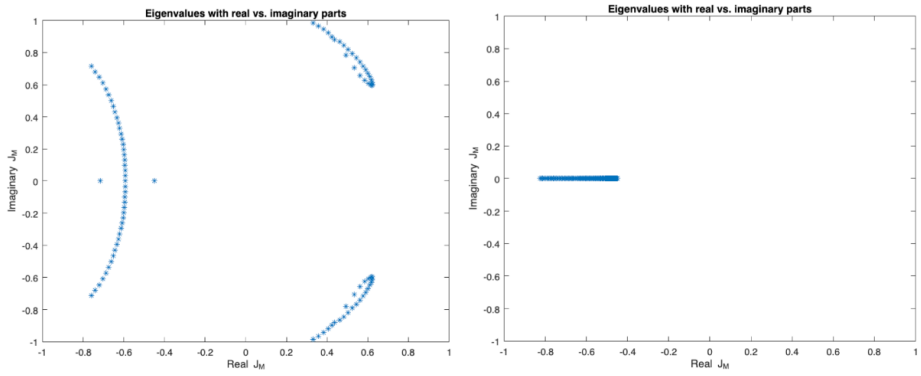

Figure 4: These two graphs are graphed by real and imaginary parts of eigenvalues with initial conditions $W_0 = 0.5 - 0.5 * tanh((x - L/2)), N_0 = 0.5 + 0.5 * tanh((x - L/2)), c_0 = 1$. The left graph with parameters $e = 500, v = 4, a = 3.0, m = 0.45$. With eigenvalues on the right side of the graph, equation (1) under these parameters is unstable. The right graph with parameters $e = 5, v = 1, a = 0.2, m = 0.45$. With no eigenvalues on the right side of the graph, equation (1) under these parameters is stable.

6 Result

The paper showed the Busse balloon slices for the extended-Klausmeier model for two different values of the rainfall parameter a. Busse balloon is defined as the set in (slope v, wave number k) space where patterns with that wave number are dynamically stable for that parameter combination. Wave number is the magnitude of the wave vector, calculated by: $k = \frac{1}{\lambda}$. If the combination (v,k) lies inside the Busse balloon (the banded pattern solution), then the solution is stable.

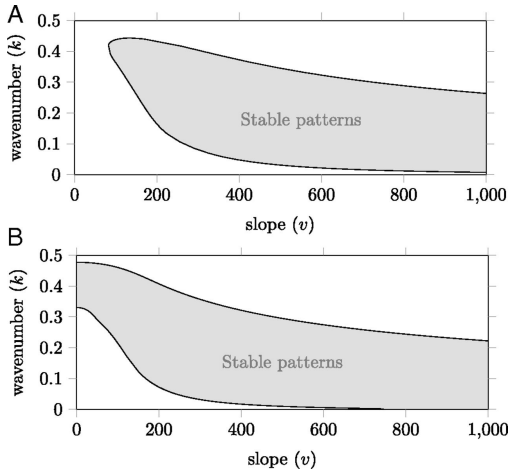

Figure 5: The shape of a Busse balloon can change between models and between parameter values. This is illustrated in A and B, which were computed for different values. Model parameters used are e = 500, m = 0.45, and a = 3.0 (A), a = 2.5 (B).

Another result is that each ecosystem state also has a specific biomass and a specific pattern migration speed. This migration speed can be calculated by the traveling waves method.
These two graphs indicate that the biomass (per unit area) is positively correlated with both the wave number k of the pattern and the slope parameter v, meaning that the steeper slopes and higher wave numbers (lower wavelengths), the shorter time that it takes for water to reach vegetation bands (reduce water losses during the transportation process).

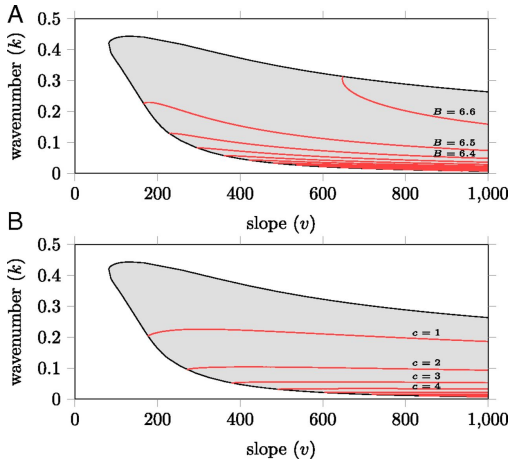

Figure 6: Busse balloon slices for the extended-Klausmeier model that include contours for the total biomass (per area) B (A) and the migration speed c (B). Biomass (per area) is positively correlated with both wavenumber k and slope v; the migration speed is negatively correlated with the wavenumber k.

7 Prospect

By numerically analyzing equation (1), we can conclude that the trend of water and plant biomass is basically proportional: sufficient water provides sufficient nutrients for plant biomass growth, and plant biomass growth reduces water content in ecosystems.
There are still a few questions that we can move further:

1. Why is there a peak occurring in the graph of plant biomass when numerically analyzing equation (1) with traveling waves? Is it because of the specific choice of parameters and initial conditions or a general pattern of the plant biomass?
2. Is there are any other forms of traveling waves, like sech-like functions?
3. How to analyze multistability with the help of stability?

8 MATLAB Code

8.1 Finite Difference Method

```
    % Parameters (you should define these based on your specific problem)
e = 0.5; % Diffusivity coefficient for the first equation
v = 1; % Slope parameter
a = 1;
m = 0.3;
time_end = 1; % End time
L = 500; % Spatial domain length
Nx = 500; % Number of spatial points
Nt = 200; % Number of time steps
dx = L / (Nx - 1); % Spatial step size
dt = time_end / (Nt - 1); % Time step size
x = linspace(0, L, Nx) % Spatial points
time = linspace(0, time_end, Nt); % Time points

% Initial conditions
W = zeros(Nx, Nt); % Initialize w
N= zeros(Nx, Nt); % Initialize n

% Boundary conditions (example: Dirichlet boundary conditions)
W(1, :) = 0; % Left boundary condition for w
W(end, :) = 0; % Right boundary condition for w
N(1, :) = 0; % Left boundary condition for n
N(end, :) = 0; % Right boundary condition for n

% Initial conditions (example: arbitrary initial condition)
W(:, 1) = 1+tanh(x - L/2); % Non-negative initial condition for w
N(:, 1) = 1-0.5*tanh(x - L/2); % Non-negative initial condition for n


% Time-stepping loop
for k = 1:Nt-1
    for h = 2:Nx-1
        % Finite Difference scheme for w equation
        d2Wdx2 = (W(h+1, k) - 2*W(h, k) + W(h-1, k)) / dx^2;
        dWdx = (W(h+1, k) - W(h-1, k)) / (2*dx);
        W(h, k+1) = W(h, k) + dt * (e*d2Wdx2 + v*dWdx + a - W(h, k) - W(h, k)*N(h, k)^2);

        % Finite Difference scheme for n equation
```

```
        d2Ndx2 = (N(h+1, k) - 2*N(h, k) + N(h-1, k)) / dx^2;
        N(h, k+1) = N(h, k) + dt * (d2Ndx2 - m*N(h, k) + W(h, k)*N(h, k)^2);
    end
end

for t = 1:Nt
    plot(x,transpose(W(:,t)));
    title('Water over time')
    xlabel('Space x');
    ylabel('Water W');
    axis([0 500 0 1]);
    pause(0.000001)
end

for t = 1:Nt
    plot(x,transpose(N(:,t)));
    title('Plant biomass over time')
    xlabel('Space x');
    ylabel('Plant Biomass N');
    axis([0 500 0 2]);
    pause(0.000001)
end

% Visualize the results
figure;
surf(time, x, W);
title('Water over time and space');
xlabel('Time t');
ylabel('Space x');
zlabel('Water W')
shading flat;

surf(time, x, N);
title('Plant biomass over time and space');
xlabel('Time t');
ylabel('Space x');
zlabel ('Plant Biomass N')
shading flat;
colorbar;
```

8.2 Traveling Waves

```
% Define time-space
T = 10; Nt = 1000;
dt = T/(Nt-1);
tspan = 0:dt:T;
L = 200; Nx = 500;
dx = L/(Nx-1);

% Discretize space
x = (transpose(0:(Nx-1))) * dx;

% Define Initial Condition
W0 = @(x) 0.5-0.5*tanh((x-L/2)); % Non-negative initial condition for w
```

```
N0 = @(x) 0.5+0.5*tanh((x-L/2)); % Non-negative initial condition for n
c0 = 1;
U0 = [W0(x); N0(x); c0];

% Define parameters
e = 50; para.e = e;  % any positive value
v = 4; para.v = v; % slope parameter
a = 3.0; para.a = a;
m = 0.45; para.m = m;

% Define D2
D2 = zeros(Nx,Nx);
D2(1,1) = -2; D2(1,2) = 2;
    for i = 2:Nx-1
        D2(i,i) = -2;
        D2(i,i-1) = 1;
        D2(i,i+1) = 1;
    end
D2(Nx,Nx) = -2; D2(Nx,Nx-1) = 2;
D2 = D2/dx^2;
D2 = sparse(D2);

% Define D1
D1 = zeros(Nx,Nx);
D1(1,2) = 0;
for i = 2:Nx-1
        D1(i,i) = 0;
        D1(i,i-1) = -1;
        D1(i,i+1) = 1;
end
D1(Nx,Nx-1) = 0;
D1 = D1/(2*dx);
D1 = sparse(D1);


format long
options = optimset('display', 'iter', 'Algorithm','levenberg-marquardt');
% output the solution
WN = fsolve(@(U) WN_function(U, U0, D1, D2, Nx, para), U0, options);

Obtain the solution
% define solution of F and S, and the speed c seperately
W_solution = WN(1:Nx);
N_solution = WN(Nx+1:2*Nx);
c_solution = WN(2*Nx+1)
Plot the graph
% Plot W
W_solution = WN(1:Nx);
plot(x, W_solution, '-*')
title('graph of Water')
xlabel('Space x')
ylabel('Water W')
% Plot N
N_solution = WN(Nx+1:2*Nx);
```

```
plot(x, N_solution, '-*')
title('graph of Plant Biomass')
xlabel('Space x')
ylabel('Plant Biomass N')
% W vs N
plot(WN(1:Nx), WN(Nx+1:2*Nx), '-*')
title('relationship between Water v.s. Plant Biomass')
Evaluate Jacobian by the solution
% define the "main" Jacobian J_M as a 2Nx2N matrix
J_M = zeros(2*Nx, 2*Nx);
    % row 1st half: water
    J_M(1:Nx, 1:Nx) = c_solution*D1 - para.e*D2 - para.v*D1 + spdiags(1+N_solution.*N_solution, 0, Nx, N
    J_M(1:Nx, Nx+1:2*Nx) = 2*spdiags(W_solution.*N_solution, 0, Nx, Nx);
    % row 2nd half: savanna
    J_M(Nx+1:2*Nx, 1:Nx) = -spdiags(N_solution.*N_solution, 0, Nx, Nx);
    J_M(Nx+1:2*Nx, Nx+1:2*Nx) = c_solution*D1 - D2 +spdiags(para.m-2*W_solution.*N_solution, 0, Nx, Nx)
% compute 100 eigenvalues closet to 0
lambda = eigs(-J_M, 100, 0);
% seperate real and imaginary parts of eigenvalues
lambda_real = real(lambda)
lambda_imag = imag(lambda)

for i = 1:1:100
    plot(lambda_real(:,:), lambda_imag(:,:), '*');
    title('Eigenvalues with real vs. imaginary parts');
    xlabel('Real J_M')
    ylabel('Imaginary J_M')
    axis([-1 1 -1 1]);
end
%%
function [WN_function, Jacobian] = WN_function(U, U0, D1, D2, Nx, para)
    W = U(1:Nx);
    N = U(Nx+1:2*Nx);
    c = U(2*Nx+1);
    W0 = U0(1:Nx);
    N0 = U0(Nx+1:2*Nx);
    % Construct functions
    WN_function = [c*D1*W - para.e*D2*W - para.v*D1*W - para.a + W + W.*N.*N;
                   c*D1*N - D2*N + para.m*N - W.*N.*2;
                   transpose([D1*W;D1*N])*([W;N]-[W0;N0])];
     % Construct Jacobian
    Jacobian = zeros(2*Nx+1,2*Nx+1);
        Jacobian(1:Nx, 1:Nx) = c*D1 - para.e*D2 - para.v*D1 + spdiags(1+N.*N, 0, Nx, Nx);
        Jacobian(1:Nx, Nx+1:2*Nx) = 2*spdiags(W.*N, 0, Nx, Nx);
        Jacobian(1:Nx, 2*Nx+1) = D1*W;

        Jacobian(Nx+1:2*Nx, 1:Nx) = -spdiags(N.*N, 0, Nx, Nx);
        Jacobian(Nx+1:2*Nx, Nx+1:2*Nx) = c*D1 - D2 +spdiags(para.m-2*W.*N, 0, Nx, Nx);
        Jacobian(Nx+1:2*Nx, 2*Nx+1) = D1*N;

        Jacobian(2*Nx+1, 1:Nx) = transpose(D1*W) + transpose(W-W0)*D1;
        Jacobian(2*Nx+1, Nx+1:2*Nx) = transpose(D1*N) + transpose(N-N0)*D1;
end
```

9 Reference

Bastiaansen et al. *Multistability of model and real dryland ecosystems through spatial self-organization.* PNAS.October 15, 2018. https://www.pnas.org/doi/full/10.1073/pnas.1804771115.
Bastiaansen et al. *The effect of climate change on the resilience of ecosystems with adaptive spatial pattern formation.* Wiley Online Library. January 7, 2020. https://onlinelibrary.wiley.com/doi/full/10.1111/ele.13449.
Carter, Paul and Doelman, Arjen. *Traveling Stripes in the Klausmeier Model of Vegetation Pattern Formation.* Society of Industrial and Applied Mathematics. https://epubs.siam.org/doi/10.1137/18M1196996.
Carvalho, Keegan. *Desertification: Causes, Effects, And Solutions.* EARTH.ORG. August 3rd, 2022. https://earth.org/what-is-desertification/
Jonkhout, C.J.H. *Traveling wave solutions of reaction-diffusion equations in population dynamics.* July 27, 2016. https://www.universiteitleiden.nl/binaries/content/assets/science/mi/scripties/jonkhout.pdf
Rietkerk et al. *Evasion of tipping in complex systems through spatial pattern formation.* Science. October 8 2021. https://www.science.org/doi/full/10.1126/science.abj0359.
FTCS Scheme. Wikipedia. https://en.wikipedia.org/wiki/FTCS_scheme.
What is the Major Causes of Desertification? Ministry of the Government. https://www.env.go.jp/en/nature/desert/download/p2.pdf