

Growing Allee Flowers

AmirHosein Sadeghimanesh

2022.09.22

Introduction

Consider a species, an E-coli bacteria, a starfish or a fig tree, that its population growth follows a strong Allee effect model. That means they have a cooperative behavior and the population size need to be above a threshold to not become extinct. At the same time the environment has a limited amount of resources, therefore there is a carrying capacity. After normalizing, we can assume that the carrying capacity of the environment is 1 and let the Allee effect threshold be a parameter, denoted by ' b '. The population is not always distributed homogeneously in the environment, one typical scenario is that the environment is partitioned to several compartments and there is a possibility to move from one to another, migration. Here we focus on the class of population models that consisted of ' n ' compartments and a single species with the strong Allee effect and migration (see [1, 2]). Let ' x_i ' denote the population size of this species in the ' i '-th compartment. If we assume the migration rate for allowed paths from a compartment to another one is fixed and treat it as a parameter denoted by ' a ', then the ODE system of the model is as the following.

$$\dot{x}_i = x_i(1-x_i)(x_i-b) - \left(\sum_{j=1}^n \delta_{i,j}=1\right) x_i + \sum_{j=1}^n \delta_{j,i}=1 x_j, \quad i = 1, \dots, n,$$

Let ' F ' be the set of polynomials in the right hand side of Eq. (1). For a given choice of ' (a, b) ' in ' $\mathbb{R}_{>0}^2$ ', the non-negative solutions to the system of equations made by letting all polynomials in F equal to zero are the steady states of the model. This means that depending to the initial values of ' x_i ' at ' $t=0$ ', after a sufficient amount of time the population sizes in the compartments start converging to the steady states. We are interested in the possible number of steady states and more importantly where in the parameter space the number of steady states can change. There are different tools to find a superset of the boundary curves in the ' $\mathbb{R}_{>0}^2$ ' space of ' (a, b) ' where the number of steady states to Eq. (1) can change. Let us call the union of the curve-boundaries that the number of steady states DO change when the parameter choice cross these boundaries the "minimal discriminant variety" of the system. To see a list of techniques to find sets of curves containing this minimal discriminant variety see [3]. One such a tool is called 'Border Polynomials' (see [4]). The command 'BorderPolynomial' in 'RegularChains' package of Maple computes border polynomials of a parametric system of equations. In below we plot the solution sets to these polynomials which are candidate curves for discriminant variety. In some cases it might be the minimal discriminant variety, and in some it might contain extra curves. But the important thing is that it contains finitely many curves including all parts of the minimal discriminant variety.

In this file we focus on border polynomials for this model when all compartments are connected to each other in both directions. I.e. the connectivity graph is the fully connected simple digraph.

Procedures

First we define a procedure that receives the adjacency matrix of the connectivity graph of compartments of our model and returns the set of set 'F' introduced above. Note that the connectivity graph between compartments is a directed graph without loops and with no multi-edge in our case.

```

1 AdjMatToSSList:= proc(
2     M :: 'Matrix'( { 0, 1 } ),
3     n :: posint := LinearAlgebra-RowDimension( M )
4 ) :: list( polynomial)
5
6     description
7     "This procedure receives a binary matrix, considered as the adjacency matrix associated to a simple digraph
8     , then returns the list of steady state equations of the n-compartment population model with the strong Allee
9     behavior.":
10
11     local
12         i1 :: posint,
13         i2 :: posint,
14         i3 :: posint
15
16     if LinearAlgebra-ColumnDimension( M ) <> n then
17         error "The adjacency matrix is a square matrix, please check your input matrix."
18     end if;
19
20     return( [ seq( cat( x__, i1 ) * ( 1 - cat( x__, i1 ) ) * ( cat( x__, i1 ) - b ) - add( M[ i1, i2 ], i2 in ( { seq

```

Now we define a procedure that receives a positive integer, n , and returns the adjacency matrix of the fully connected simple digraph on n nodes.

```

1 AdjMatFullyConnected= proc(
2     n :: posint
3     ) :: 'Matrix( {0, 1} ):
4
5     description
6     "For any given positive integer, n, this procedure returns the adjacency matrix of the fully connected simple graph on n nodes."
7
8     local
9         i :: posint,
10        j :: posint
11
12     return( Matrix( [ seq( [ seq( 1, j = 1 .. i-1 ), 0, seq( 1, j = i+1 .. n ) ], i = 1 .. n ) ] ) ):

```

Combining the two former procedures we have get the following one.

```

1 SSPlistFullyConnected= proc(
2     n :: posint
3     ) :: list( polynom):
4
5     description
6     "The list of steady state polynomials of the fully connected population model with the strong Allee effect and
7     rtments."
8
9     return( AdjMatToSSPlist( AdjMatFullyConnected(n) ) );

```

Finally a procedure that receives the set of polynomials 'F' and returns the border polynomials that we are looking for.

```

1 SSPlistToBPlist= proc(
2     n :: posint,
3     F :: list( polynom)
4     ) :: list( polynom):
5
6     description
7     "This procedure receives a positive integer, n, and a list of polynomials supposedly in the variables x_i,
8     n, and two parameters a and b, and then returns the list of border polynomials of the parametric system of
9     created by the input list of polynomials."
10
11     local
12         i :: posint,
13         ring,
14         BPrax
15
16     ring := RegularChains-PolynomialRing [ seq( cat( x_, i ), i = 1 .. n ), a, b ] :
17     BPrax := RegularChains-ParametricSystemToolsBorderPolynomial( F, [ seq( cat( x_, i ), i = 1 .. n ), a, b ], [
18     ], [], 2, ring );
19
20     return( BPrax );

```

Now combining all above procedures together.

```

1 BPlistFullyConnected= proc(
2     n :: posint
3     ) :: list( polynom):
4
5     description
6     "This procedure returns the list of border polynomials of the population model with the strong Allee effect
7     rtments with fully connected connectivity graph."
8
9     return( SSPlistToBPlist( n, SSPlistFullyConnected(n) ) );

```

Allee Flowers

Note that in case $n = 1$, there is no migration and we practically have a 1-dimensional parameter space.

```

> B1 := BPlistFullyConnected( 1 );
      B1 := [a, b, b - 1]

```

(1)

Note that the curve $a = 0$ is doing nothing other than defining the border of the parameter region $(a, b) \in \mathbb{R}_{>0}^2$. But $b = 0$ other than being the border of the region has the property that the

number of steady states on this curve is 2 while for b any neighborhood of the parameter points on this curve intersected with $\mathbb{R}_{>0}^2$ the number of steady states is 3. A similar behavior when $b = 1$. The border polynomials in this case give us three lines. Things become more interesting for $n > 1$.

Let us compute and save the border polynomials for $n = 2, 3, 4, 5, 6$ which Maple finishes them on our computer without raising an error. For $n = 2$, the computation finishes in less than a second and for $n = 6$, it takes around 118 seconds. Computations performed on Windows 10, Intel(R) Core(TM) i7-10850H CPU @ 2.70 GHz 2.71 GHz, x64-based processor, 64.0 GB (RAM).

```
> B2 := BplistFullyConnected( 2 ):
> B3 := BplistFullyConnected( 3 ):
> B4 := BplistFullyConnected( 4 ):
> B5 := BplistFullyConnected( 5 ):
> B6 := BplistFullyConnected( 6 ):
```

To make the worksheet look better, we showed the polynomials in the following section that can get closed.

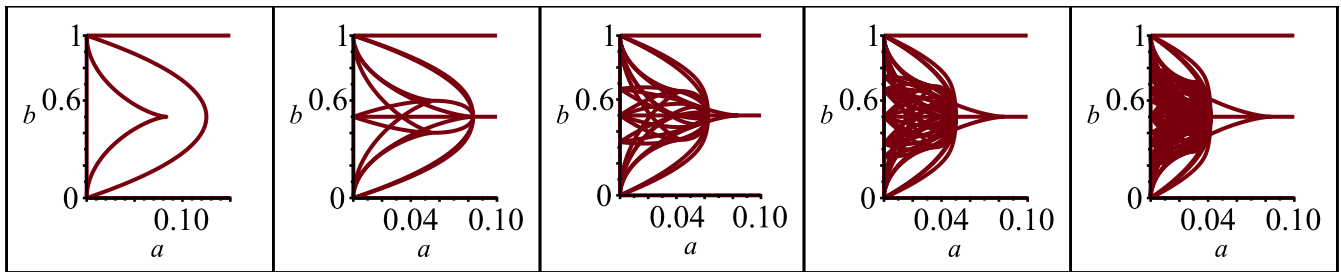
► Border Polynomial Lists

Just in case you want to see the error message of Maple when we try $n = 7$. It takes the expected amount of time. When we increase n by 1 from 2 to 6, the computation time increased by a factor a bit more than 10. For $n = 7$, it takes 1826 seconds on our computer which is what we expected, but when it finishes, it shows an error message "Error, (in Tools:-Regroup) too many levels of recursion".

```
> # st := time[real]():
# B7 := BplistFullyConnected( 7 ):
# time[real]() - st;
Error, (in Tools:-Regroup) too many levels of recursion
1826.2060000000
```

(2)

```
> Now let us plot the curves defined by the border polynomials.
> # PA for "Plot Array"
PA1 := Array( 1 .. 5 ):
PA1[1] := plots:-implicitplot( B2, a = 0 .. 0.15, b = 0 .. 1,
size = [ 400, 400 ] ):
PA1[2] := plots:-implicitplot( B3, a = 0 .. 0.1, b = 0 .. 1, size
= [ 400, 400 ] ):
PA1[3] := plots:-implicitplot( B4, a = 0 .. 0.1, b = 0 .. 1, size
= [ 400, 400 ] ):
PA1[4] := plots:-implicitplot( B5, a = 0 .. 0.1, b = 0 .. 1, size
= [ 400, 400 ] ):
PA1[5] := plots:-implicitplot( B6, a = 0 .. 0.1, b = 0 .. 1, size
= [ 400, 400 ] ):
plots:-display( PA1 );
```



These are what we call Allee flowers :)

Now just for the sake of our artistic hunger, let's play a bit around with these curves. In the following closed section (to keep the worksheet prettier), I just plot each single polynomial involved in Border polynomial lists to partition them to two sets of the flower part and the leaves part. The partitioning here is based on my visual imagination.

► Partitioning Border Polynomial Sets to two parts.

Now is time to have some cute plots :)

► Two parted Allee flowers?

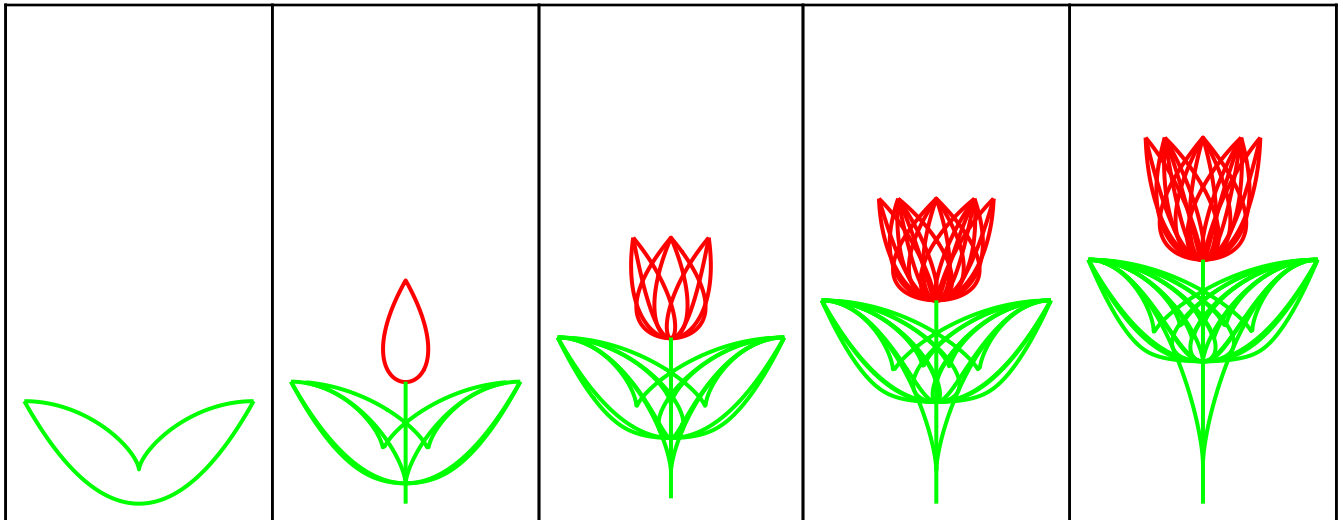
Here is the outcome after a vertical rescaling of the plots.

```
> P2v2 := plots:-display( P1, view = [ 0 .. 1, -0.3 .. 0.3 ], axes
= none, size = [ 400, 800 ] ):
> P3v2 := plots:-display( P5, P4, view = [ 0 .. 1, -0.2 .. 0.2 ],
axes = none, size = [ 400, 800 ] ):
> P4v2 := plots:-display( P9, P8, view = [ 0 .. 1, -0.15 .. 0.15 ],
axes = none, size = [ 400, 800 ] ):
> P5v2 := plots:-display( P13, P12, view = [ 0 .. 1, -0.12 .. 0.12
], axes = none, size = [ 400, 800 ] ):
> P6v2 := plots:-display( P17, P16, view = [ 0 .. 1, -0.1 .. 0.1 ],
axes = none, size = [ 400, 800 ] ):
> P2v3 := plots:-display( plottools:-translate( P2v2, 0, -0.17 ),
view = [ 0 .. 1, -0.3 .. 0.3 ], axes = none, size = [ 400, 800 ]
):
> P3v3 := plots:-display( plottools:-translate( P3v2, 0, -0.1 ),
view = [ 0 .. 1, -0.2 .. 0.2 ], axes = none, size = [ 400, 800 ]
):
> P4v3 := plots:-display( plottools:-translate( P4v2, 0, -0.05 ),
view = [ 0 .. 1, -0.15 .. 0.15 ], axes = none, size = [ 400, 800
] ):
> P5v3 := plots:-display( plottools:-translate( P5v2, 0, -0.02 ),
view = [ 0 .. 1, -0.12 .. 0.12 ], axes = none, size = [ 400, 800
] ):
```

```

> P6v3 := plots:-display( plottools:-translate( P6v2, 0, 0 ), view
= [ 0 .. 1, -0.1 .. 0.1 ], axes = none, size = [ 400, 800 ] );
> PA12 := Array( 1 .. 5 ):
PA12[1] := P2v3:
PA12[2] := P3v3:
PA12[3] := P4v3:
PA12[4] := P5v3:
PA12[5] := P6v3:
> plots:-display( PA12 );

```



Or you might try to bring out a lotus flower from Allee flowers, that's another possibility :)

References:

- [1] Gergely Röst and AmirHosein Sadeghimanesh. *Exotic bifurcations in three connected populations with Allee-effect*, International Journal of Bifurcation and Chaos, Volume 31, Issue 13, article no. 2150202, 2021, DOI: [10.1142/S0218127421502023](https://doi.org/10.1142/S0218127421502023).
- [2] Gergely Röst and AmirHosein Sadeghimanesh, *Unidirectional migration of populations with Allee effect*, 2021, bioRxiv: [10.1101/2021.06.24.449708](https://doi.org/10.1101/2021.06.24.449708).
- [3] AmirHosein Sadeghimanesh and Matthew England. *Resultant Tools for Parametric Polynomial Systems with Application to Population Models*, 2022, arXiv: [2201.13189](https://arxiv.org/abs/2201.13189).
- [4] Lu Yang and Bican Xia. *Real Solution Classification for Parametric Semi-Algebraic Systems*, In: Algorithmic Algebra and Logic - Proceedings of the A3L, pages 281 - 289, 2005.