1. > # 3 Species with Variable Parms: Basins of Attraction (BOA)

BOA **(1)**

2 > *restart* : *with*(*plots*) : *with*(*StringTools*) :
with(*DEtools*) : *with*(*ColorTools*) : *with*(*plots*) : *with*(*geom3d*) : *with*(*plottools*) :
with(*FileTools*) : *with*(*Optimization*) : *with*(*VectorCalculus*) : *with*(*LinearAlgebra*) :
with(*geom3d*) :

3 > *FormatTime*("%I:%M-%p----%d-%b-%Y"); *currentdir*( ); *printlevel* := 1 :

"09:43-AM----20-May-2020"

"C:\Users\nn\Documents\2 research\2017 summer non const parms\current" **(2)**

4. > ###### TOP ###########

5. > **#Analytic Solution 2D**

6. > **#Analytic Solution 3D**

7. > **#Parameters**

8. > **#s=10**

9. > **#Jacobian**

10. > **#Parameters**

11. > *unassign*('s') :

12. > $e1 := \frac{d}{dt} U(t) = -u1(s) \cdot U(t) + u2(s) \cdot V(t) \cdot U(t) \cdot \left(1 - \frac{U(t)}{ku}\right)$:

13. > $e2 := \frac{d}{dt} V(t) = -v1 \cdot V(t) + v2 \cdot V(t) \cdot U(t) \cdot \left(1 - \frac{V(t)}{kv}\right) - v3 \cdot A(t) \cdot V(t)$:

14. > $e2b := \frac{d}{dt} V(t) = -v1 \cdot V(t) + v2 \cdot V(t) \cdot U(t) \cdot \left(1 - \frac{V(t)}{kv}\right)$:

15. > $e3 := \frac{d}{dt} A(t) = -a1(s) \cdot A(t) + a2(s) \cdot V(t) \cdot A(t)$:

16. > ########### *3D system*

17. > $Udot3d(s) := simplify\left(\frac{rhs(e1)}{U(t)}\right)$:

18. > $Vdot3d(s) := simplify(rhs(e2))$:

19. > $Adot3d(s) := simplify\left(rhs\left(\frac{e3}{A(t)}\right)\right)$:

20. > $Adot3D(s) := subs(\{V(t) = V\}, Adot3d(s))$:

21. > $Udot3D(s) := subs(\{V(t) = V, U(t) = U\}, Udot3d(s))$:

22. > $Vdot3D(s) := subs(\{V(t) = V, U(t) = U\}, Vdot3d(s))$:

23. > ### *solve without time dependence for analytic solution*

24. > $Vs3D := s \rightarrow solve(Adot3D(s) = 0, V)$:

25. > $Us3D := s \rightarrow simplify(subs(V(t) = Vs3D(s), solve(Udot3d(s) = 0, U(t))))$:

26. > $As1 := s \rightarrow solve(subs(A(t) = A, Vdot3D(s)) = 0, A)$:

27. > $As2 := s \rightarrow subs(U = Us3D(s), As1(s))$:

28. > $As3D := s \rightarrow simplify(subs(V = Vs3D(s), As2(s)))$:

> $Astar := s \rightarrow As3D(s) : Vstar := s \rightarrow Vs3D(s) : Ustar := s \rightarrow Us3D(s) :$

> $unassign('ku','kv','v1','v2','v3','u1','u2','a1','a2','s')$

> $Vdot2D := s \rightarrow -v1 + v2 \cdot U \cdot \left(1 - \frac{V}{kv}\right) :;$

> $Udot2D := s \rightarrow -u1(s) + u2(s) \cdot V \left(1 - \frac{U}{ku}\right) :$

> ### 2D equations ##########

> $eqV2D := s \rightarrow Vdot2D(s) = 0 :$

> $eqU2D := s \rightarrow Udot2D(s) = 0 :$

> $UVsol := s \rightarrow solve([eqV2D(s), eqU2D(s)], [U, V]) :$

> ###Parms Depend on S with logistic curve

> $a1 := s \rightarrow \frac{La1}{1 + \mathrm{e}^{-ma1\,(s - ca1)}} :$

> $a2 := s \rightarrow La2 - \frac{La2}{1 + \mathrm{e}^{-ma2\,(s - ca2)}} :$

> $u1 := s \rightarrow \frac{Lu1}{1 + \mathrm{e}^{-mu1\,(s - cu1)}} :$

> $u2 := s \rightarrow Lu2 - \frac{Lu2}{1 + \mathrm{e}^{-mu2\,(s - cu2)}} :$

> $ku := 10000 : kv := 30000 : \ v1 := 0.001 : v2 := 0.03 : v3 := 0.1 :$

> $La1 := 0.2 : \quad ma1 := 0.4 : \quad ca1 := 10 :$

> $La2 := 0.00003 : \quad ma2 := 0.6 : \quad ca2 := 8.845 :$

> $Lu1 := 0.2 : \quad mu1 := 1 : \quad cu1 := 12.944 :$

> $Lu2 := 0.02 : \quad mu2 := 1 : \quad cu2 := 10 :$

> $a1(10); a2(10); evalf(u1(10)); evalf(u2(10))$

$$0.1000000000$$
$$0.00001000098123$$
$$0.01000417112$$
$$0.01000000000 \qquad (3)$$

> ##\
##############

> $L := [1, 2, 7, 10, 12]; Outlist := [\ ] : print$ ("s", "origin", "A,U,V 2 D soln", "A,U,V 3D solution") :

$$L := [1, 2, 7, 10, 12] \qquad (4)$$

> **for** s **in** L **do**

> ########################LOOP on Security

> $dmax := 1 : \ Tend := 300 :$

> $Vmax := 30000 : Umax := 12000 : Amax := 1500 : Vstep := 5000 : Ustep := 3000 : Astep := 500 :$

> $point(origin, [0, 0, 0]) :$

> $Ustar2 := evalf\left(rhs\left(UVsol(s)_{2,\,1}\right)\right) :$

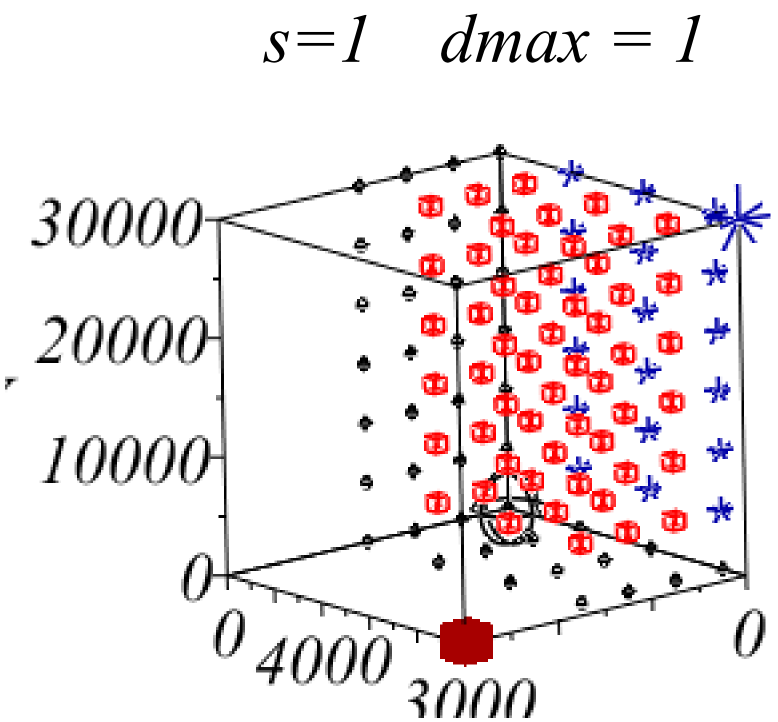
> $Vstar2 := evalf\left(rhs\left(UVsol(s)_{2,\,2}\right)\right) :$

> $point(Spoint3, As3D(s), Us3D(s), Vs3D(s)) : Astar3 := evalf(As3D(s)) :$

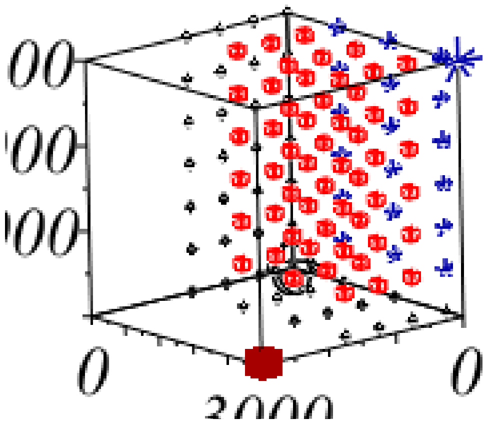
> $point(Spoint2, 0, Ustar2, Vstar2) :$

> $C3 := coordinates(Spoint3); C2 := coordinates(Spoint2); C0 := coordinates(origin);$

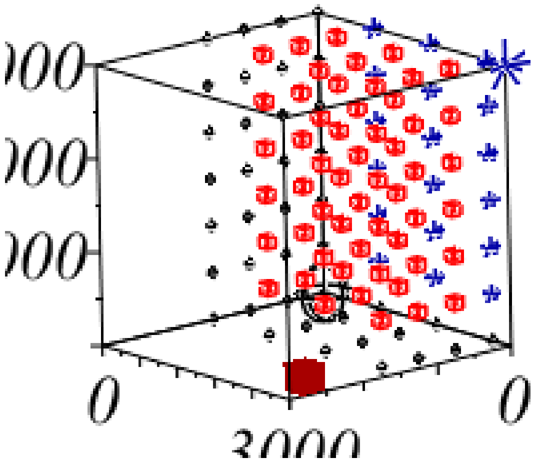
```
> ### Create output list of CP ##############################
> print(s, C0, C2, C3);
> #Outlist:=[op(Outlist) + [s, C0, C2, C3]]
> Vdelta := (Vmax/Vstep + 1) : Adelta := (Amax/Astep + 1) :
> ####### need to ensure that Spoints have the same ordering as
> for Ustart from 0 by Ustep to Umax do                ## loop on starting points
> for Astart  from 0 by Astep to Amax do
> for Vstart  from 0 by Vstep to Vmax do
>           i := Vstart/Vstep + Astart/Astep · Vdelta + Ustart/Ustep · Adelta · Vdelta :
> if Vstart · Ustart = 0 then Senda := 0; Sendv := 0; Sendu := 0;
        ## for starting point, calculate solution
> print ("line  74")

> else
>           soln := dsolve( {e1, e2, e3, V(0) = Vstart, U(0) = Ustart, A(0) = Astart},
        maxfun = 0, numeric) :
>           Send := soln(Tend) :
>           Senda := rhs(Send[2]) : Sendu := rhs(Send[3]) : Sendv := rhs(Send[4]) :
> end if;
> ###print(Senda, Sendu, Sendv)
        ## select symbol for solution at the starting point
>           point(Spoint, Senda, Sendu, Sendv) :
>          color := white;
>          d0 := distance(origin, Spoint);
>          d2 := distance(Spoint2, Spoint) :
>          d3 := distance(Spoint3, Spoint) :
>           if d2 ≤ dmax then c :='green' end if:
>           if d3 ≤ dmax then c :='red'  end if:
>           if d0 ≤ dmax then c :='black'  end if:
>           ColorArray[i] := (Astart, Ustart, Vstart, c, d0, d3, d2) :
> end do; end do; end do;
> ########### print assignments######
        ######################
> LB := [ ] : LR := [ ] : LG := [ ] : cb := 0 : cr := 0 : cg := 0 : s3 := s
> for Ustart from 0 by Ustep to Umax do                ##loop thru starting points
> for Astart  from 0  by Astep to Amax  do
> for Vstart  from 0 by Vstep to Vmax do
>  i := Vstart/Vstep + Astart/Astep · Vdelta + Ustart/Ustep · Adelta · Vdelta :
> ############## create array for the symbols
> if ColorArray[i][4] ='black'  then cb := cb + 1 :  LB := [op(LB),
        [(ColorArray[i][1], ColorArray[i][2], ColorArray[i][3])]]; end if;
> if ColorArray[i][4] ='green'  then cg := cg + 1 :  LG := [op(LG),
        [(ColorArray[i][1], ColorArray[i][2], ColorArray[i][3])]]; end if;
> if ColorArray[i][4] ='red'     then cg := cg + 1 :  LR := [op(LR),
        [(ColorArray[i][1], ColorArray[i][2], ColorArray[i][3])]]; end if;
> end do;end do;end do;
> unassign('color') :        ############ create plots
> p0 := pointplot3d([(0, 0, 0)], view = [0..10, 0..10, 0..10], color = black, symbolsize
        = 100, symbol = circle) :
> p3 := pointplot3d([coordinates(Spoint3)], labels = ["A", "U", "V"], view = [0
        ..12000, 0..10000, 0..30000], color = "DarkRed", symbolsize = 50, symbol
        = solidbox) :
> p2 := pointplot3d([coordinates(Spoint2)], labels = ["A", "U", "V"], color
        = "DarkBlue", symbolsize = 200, symbol = asterisk) :
> pR := pointplot3d(LR, color = red, symbol = box, symbolsize = 20) :
> pB := pointplot3d(LB, color = [black], symbol = circle, symbolsize = 15) :
> pG := pointplot3d(LG, color = "DarkBlue", symbol = asterisk, symbolsize = 80) :
> graph||s := display(pR, pG, pB, p3, p2, p0, labels = ['A','U','V'], orientation
        = [45, 75, -5], labelfont = [Roman, italic, 20], font = [Roman, italic, 20], title
        = [typeset(cat("s=", s), cat( "    dmax = ", dmax))], view = [0..3000, 0
        ..10000, 0..30000])
> end do:
   1, [0, 0, 0], [0, 9999.999978, 29999.90000], [2982.097442, 9999.996369, 178.9147545]
   2, [0, 0, 0], [0, 9999.999941, 29999.90000], [2973.447837, 9999.993342, 265.4018394]
   7, [0, 0, 0], [0, 9999.990850, 29999.90000], [2794.626543, 9999.866320, 2053.260975]
  10, [0, 0, 0], [0, 9999.666527, 29999.90000], [1999.888000, 9998.999480, 9999.018866]
  12, [0, 0, 0], [0, 9992.167586, 29999.89992], [-513.594406, 9993.313133, 35139.28063]     (5)
> for s in L do graph||s end do;
```



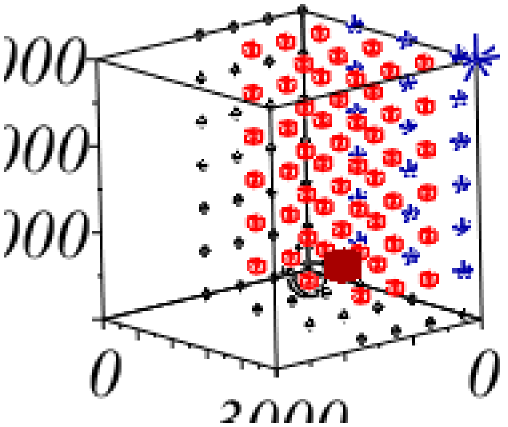
s=2   dmax = 1

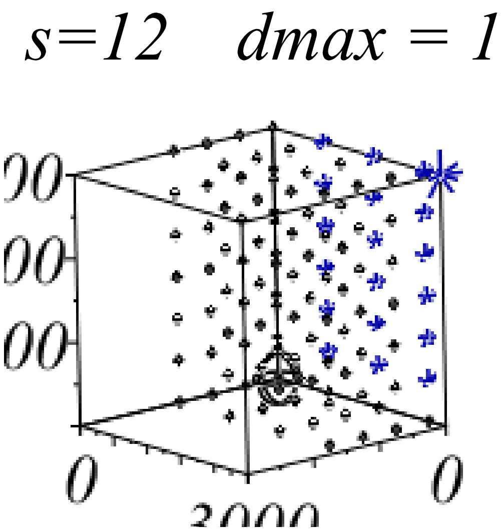


s=7   dmax = 1

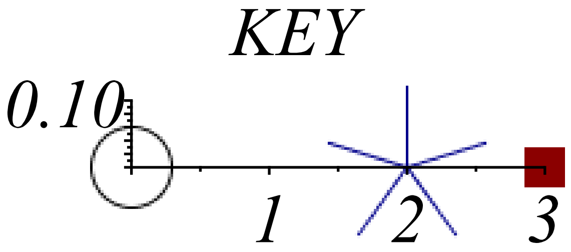


s=10   dmax = 1





```
108. > #save graph1,graph2,graph7,graph10,graph12,"graphs.mw"
109. > ############### KEY
          ####################################################
          3
110. > unassign('color') :
111. > p0 := pointplot([(0, 0)], color = black, symbolsize = 100, symbol = circle, view
          = [0 ..3, 0 ..0.1]) :
112. > p3 := pointplot([(3, 0)], color = "DarkRed", symbolsize = 50, symbol = solidbox,
          view = [0 ..3, 0 ..0.1]) :
113. > p2 := pointplot([(2, 0)], color = "DarkBlue", symbolsize = 200, symbol
          = asterisk) :
114. > display(p0, p2, p3, labelfont = [Roman, italic, 20], font = [Roman, italic, 20], title
          = ['KEY'], caption = typeset(" Extinction               2 D       3 D"))
```



Extinction

2 D 3 D