The Lotka - Volterra cannibalism time budget model

Resident population dynamics

$$\frac{d}{dt} R = r\,R\left(1 - \frac{R}{K}\right) - \alpha\,R \sum_{j=1}^{l} (1 - x_j)\,n_j$$

$$\frac{d}{dt} n_i = \epsilon\,\alpha\,R\,(1 - x_i)\,n_i - \delta\,n_i + \gamma\,\beta[x_i]\,x_i\,n_i \left(\sum_{j=1}^{l} (1 - x_j)\,n_j\right) - (1 - x_i)\,n_i \left(\sum_{j=1}^{l} \beta[x_j]\,x_j\,n_j\right)$$

$(i = 1,\ \ldots,\ l)$

Invader population dynamics

$$\frac{d}{dt} \mathrm{Log}[m] = \epsilon\,\alpha\,R\,(1 - y) - \delta + \gamma\,\beta[y]\,y\,\left(\sum_{j=1}^{l} (1 - x_j)\,n_j\right) - (1 - y)\left(\sum_{j=1}^{l} \beta[x_j]\,x_j\,n_j\right)$$

$$= \ E_1\,(1 - y) - \delta + E_2\,\beta[y]\,y - E_3\,(1 - y)$$

where $E_1 = \epsilon\,\alpha\,R$ and $E_2 = \gamma \sum_{j=1}^{l} (1 - x_j)\,n_j$ and $E_3 = \sum_{j=1}^{l} \beta[x_j]\,x_j\,n_j)$

Invasion fitness

$$s_{\underline{E}}(y) = \langle E_1 \rangle\,(1 - y) - \delta + \langle E_2 \rangle\,\beta[y]\,y - \langle E_3 \rangle\,(1 - y)$$

MONOMORPHIC RESIDENT POPULATION

Monomorphic resident population dynamics:
(R = resource density; n = resident population density)

```
dLogR = r - r R / k - α (1 - x) n;

dLogn = ϵ α (1 - x) R - δ + γ β[x] x (1 - x) n - (1 - x) β[x] x n;
```

Monomorphic resident population equilibrium:

```
Solve[{0 == r - r R / k - α (1 - x) n, 0 == ϵ α (1 - x) R - δ + γ β[x] x (1 - x) n - (1 - x) β[x] x n},
  {R, n}]
```

$$\left\{\left\{R \to -\frac{k\,\alpha\,\delta + k\,r\,x\,\beta[x] - k\,r\,x\,\gamma\,\beta[x]}{-k\,\alpha^2\,\epsilon + k\,x\,\alpha^2\,\epsilon - r\,x\,\beta[x] + r\,x\,\gamma\,\beta[x]},\ n \to -\frac{r\,\delta - k\,r\,\alpha\,\epsilon + k\,r\,x\,\alpha\,\epsilon}{(-1 + x)\,\left(-k\,\alpha^2\,\epsilon + k\,x\,\alpha^2\,\epsilon - r\,x\,\beta[x] + r\,x\,\gamma\,\beta[x]\right)}\right\}\right\}$$

```
R[x_] := - k (α δ - r x (-1 + γ) β[x]) / (k (-1 + x) α^2 ϵ + r x (-1 + γ) β[x]);

n[x_] := - r (δ + k (-1 + x) α ϵ) / ((-1 + x) (k (-1 + x) α^2 ϵ + r x (-1 + γ) β[x]));
```

Invasion fitness and its derivatives:

```
smo[x_, y_] := ϵ α (1 - y) R[x] - δ + γ β[y] y (1 - x) n[x] - (1 - y) β[x] x n[x];

dsmo[x_] := (∂y smo[x, y]) /. {y → x};
ddsmo[x_] := (∂y ∂y smo[x, y]) /. {y → x};
```

Default parameter values and functions:

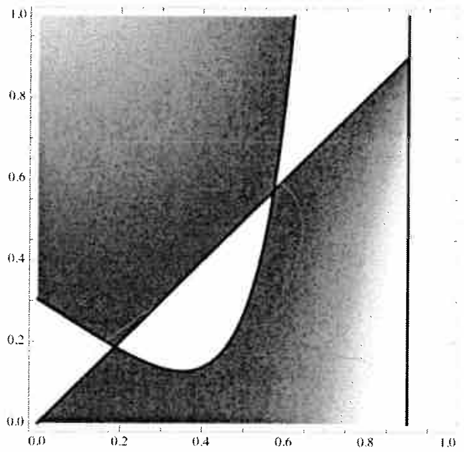
```
α = 1; γ = 0.2; δ = 0.1; ϵ = 0.1; r = 1; k = 10;

β0 = 0.1; β1 = 1.5 ; p = 1;

β[x_] := β0 + β1 x^p;
```

Pairwise invadability plot (PIP):

```
PIPbnd = ContourPlot[If[n[x] > 0, s_mo[x, y]], {x, 0, 1}, {y, 0, 1}, Contours → {0},
   ContourStyle → {Black, Thick}, ContourShading → False, PlotPoints → 100];

PIPint = DensityPlot[If[s_mo[x, y] > 0 && n[x] > 0, s_mo[x, y]], {x, 0, 1}, {y, 0, 1},
   PlotPoints → 50];

nPos = ContourPlot[n[x], {x, 0, 1}, {y, 0, 1}, Contours → {0}, ContourStyle → {Black, Thick},
   ContourShading → False, PlotPoints → 10];
```

Pairwise invadability plot (PIP):

```
Show[PIPint, PIPbnd, nPos]
```

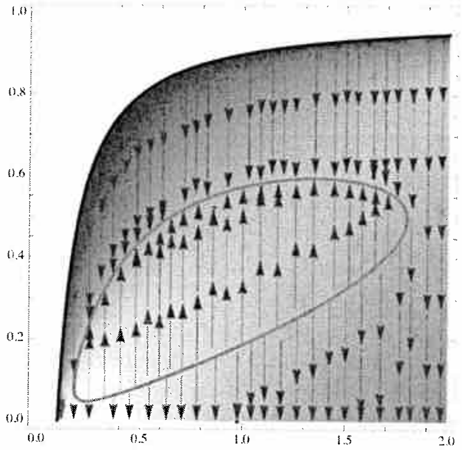


Bifurcation plot singular strategy x versus parameter α:

```
mag = DensityPlot[If[n[x] > 0, n[x]], {α, 0, 2}, {x, 0, 1}];

grad = StreamPlot[If[n[x] > 0, {0, ds_mo[x]}, {0, 0}], {α, 0, 2}, {x, 0, 1},
   StreamStyle → Arrowheads[0.04]];

xES = ContourPlot[If[n[x] > 0 && dds_mo[x] ≤ 0, ds_mo[x]], {α, 0, 2}, {x, 0, 1},
   Contours → {0}, ContourStyle → {Green, Thick}, ContourShading → False, PlotPoints → 60];

xNES = ContourPlot[If[n[x] > 0 && dds_mo[x] > 0, ds_mo[x]], {α, 0, 2}, {x, 0, 1},
   Contours → {0}, ContourStyle → {Red, Thick}, ContourShading → False, PlotPoints → 60];

nPos = ContourPlot[n[x], {α, 0, 2}, {x, 0, 1}, Contours → {0}, ContourStyle → {Black, Thick},
   ContourShading → False, PlotPoints → 60];
```

Bifurcation plot x versus α:

```
Show[mag, grad, xES, xNES, nPos]
```



Critical function analysis:

Reset:

```
Clear[α, β, γ, δ, ϵ, r, k];
```

Critical functions:

```
dsmo[x] == 0;
Simplify[%]
```

$$\frac{r\,(-\gamma\,\delta+k\,\alpha\,\gamma\,\epsilon+x\,(-\delta+\gamma\,\delta-k\,\alpha\,\gamma\,\epsilon))\,\beta[x]+(-1+x)\,\left(k\,\alpha^2\,\delta\,\epsilon+r\,x\,\gamma\,(\delta+k\,(-1+x)\,\alpha\,\epsilon)\,\beta'[x]\right)}{(-1+x)\,\left(k\,(-1+x)\,\alpha^2\,\epsilon+r\,x\,(-1+\gamma)\,\beta[x]\right)}==0$$

```
DSolve[dsmo[x] == 0, β[x], x];
Simplify[%]
```

$$\left\{\left\{\beta[x]\to\frac{(\delta+k\,(-1+x)\,\alpha\,\epsilon)^{-1/\gamma}\left(r\,(-1+x)^{\frac{1}{\gamma}}\,(-1+\gamma)\,\delta\,C[1]-k\,(-1+x)\,\alpha\,\epsilon\left(\alpha\,(\delta+k\,(-1+x)\,\alpha\,\epsilon)^{\frac{1}{\gamma}}-r\,(-1+x)^{\frac{1}{\gamma}}\,(-1+\gamma)\,C[1]\right)\right)}{r\,x\,(-1+\gamma)}\right\}\right\}$$

```
βcrit[x_] :=
```

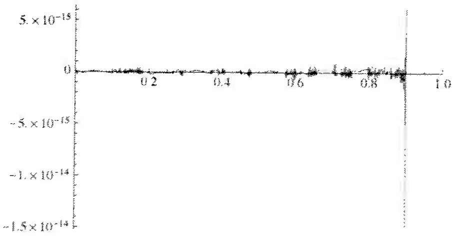
$$\frac{1}{r\,x\,(-1+\gamma)}\,(\delta+k\,(-1+x)\,\alpha\,\epsilon)^{-1/\gamma}\left(r\,(-1+x)^{\frac{1}{\gamma}}\,(-1+\gamma)\,\delta\,C[1]-k\,(-1+x)\,\alpha\,\epsilon\left(\alpha\,(\delta+k\,(-1+x)\,\alpha\,\epsilon)^{\frac{1}{\gamma}}-r\,(-1+x)^{\frac{1}{\gamma}}\,(-1+\gamma)\,C[1]\right)\right);$$

Default parameter values:

```
α = 1; γ = 0.2; δ = 0.1; ϵ = 0.1; r = 1; k = 10;
```

Test critical functions:

```
β[x_] := βcrit[x];

Plot[If[(n[x] /. {C[1] → -1}) > 0, dsmo[x] /. {C[1] → -1}], {x, 0, 1},
 PlotRange → {{0, 1}, All}]
```



Plot critical functions:

```
Limit[βcrit[x], x → 0]
```

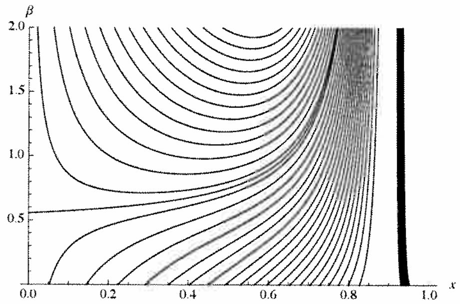
```
DirectedInfinity[(-1. + 0. i) - (1.21933 + 0. i) C[1]]
```

```
Solve[(-1.` + 0.` i) - (1.2193263222069808` + 0.` i) C[1] == 0, C[1]]
```

```
{{C[1] → -0.820125 + 0. i}}
```

```
crit = Plot[βcrit[x] /. {C[1] → Join[{-0.8201249999999999`}, Table[-i / 25, {i, 1, 50}]]},
   {x, 0, 1}, AxesOrigin → {0, 0}, PlotRange → {0, 2}, AxesLabel → {x, β},
   PlotStyle → {Black}];
```

```
Show[crit]
```

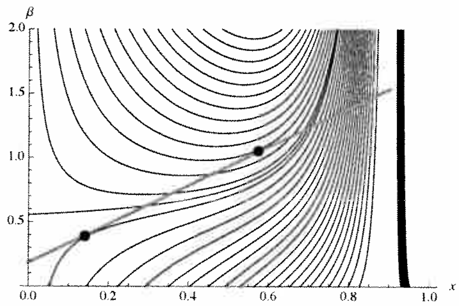


Superimpose β on the critical functions:

Case as we had in the PIP above:

```
β[x_] := β0 + β1 x^p; β0 = 0.185; β1 = 1.5; p = 1;

βNES = Plot[If[dds_mo[x] > 0 && n[x] > 0, β[x]], {x, 0, 1}, PlotStyle → {Red, Thick}];

βES = Plot[If[dds_mo[x] ≤ 0 && n[x] > 0, β[x]], {x, 0, 1}, PlotStyle → {Black, Thick}];
```

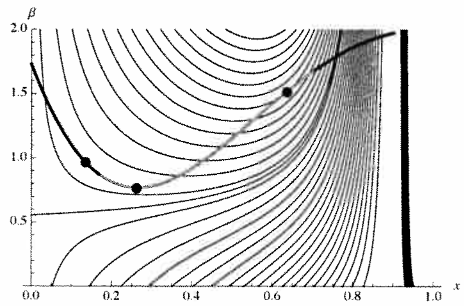
```
Show[crit, βNES, βES]
```



Case with three singular strategies:

```
β[x_] := 2 - 9 (0.03 + x)^p (1 - x)^q; p = 1; q = 2.5;

βNES = Plot[If[dds_mo[x] > 0 && n[x] > 0, β[x]], {x, 0, 1}, PlotStyle → {Red, Thick}];

βES = Plot[If[dds_mo[x] ≤ 0 && n[x] > 0, β[x]], {x, 0, 1}, PlotStyle → {Black, Thick}];
```

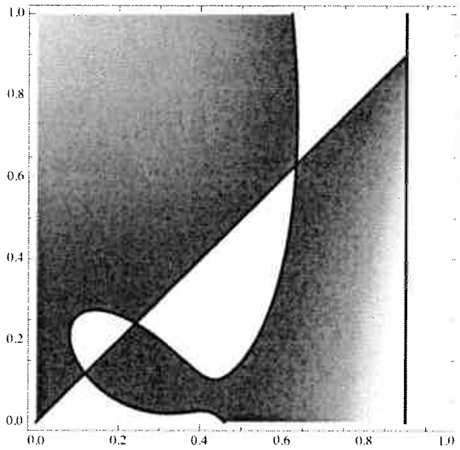
```
Show[crit, βNES, βES]
```



Pairwise invadability plot for the latter case:

```
PIPbnd = ContourPlot[If[n[x] > 0, smo[x, y]], {x, 0, 1}, {y, 0, 1}, Contours → {0},
   ContourStyle → {Black, Thick}, ContourShading → False, PlotPoints → 100];

PIPint = DensityPlot[If[smo[x, y] > 0 && n[x] > 0, smo[x, y]], {x, 0, 1}, {y, 0, 1},
   PlotPoints → 50];

nPos = ContourPlot[n[x], {x, 0, 1}, {y, 0, 1}, Contours → {0}, ContourStyle → {Black, Thick},
   ContourShading → False, PlotPoints → 10];
```

```
Show[PIPint, PIPbnd, nPos]
```



Canonical equation:

Mutation rate:

```
μ[x_] := 1;
```

Standard deviation of the mutation step distribution:

```
σ[x_] := 0.04 x (1 - x);
```

Deterministic drift:

$$\text{drift}_{mo}[x\_] := \frac{1}{2}\,\mu[x]\,\sigma[x]^2\,n[x]\,ds_{mo}[x];$$

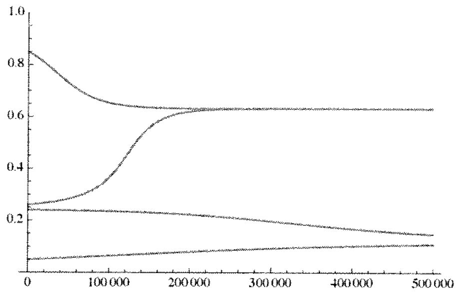
Deterministic orbit (Euler method):

```
x0 = {0.05, 0.24, 0.26, 0.85}; (* starting points *)
t0 = 0; (* start time *)
t∞ = 500000; (* stop time *)
Δt = 100; (* integration time step *)

noofStarts = Length[x0];

data = {};
For[i = 1, i ≤ noofStarts, i ++,
  x = x0[[i]];
  t = t0;
  While[t ≤ t∞ && n[x] > 0,
   data = Join[data, {{t, x}}];
   x = x + Δt drift_mo[x];
   t = t + Δt;
  ];
 ];
```

Deterministic orbits:

```
CEorbit = ListPlot[data, PlotStyle → {Red, PointSize[0.001]}, Joined → False,
   AxesOrigin → {0, 0}, PlotRange → {0, 1}];
```

```
Show[CEorbit]
```



Sochastic differential equation (Ito):

Third absolute moment of the mutation step distribution, which is assumed to be Gaussian:

$$\theta[x\_] := 2\,\sigma[x]^3 \sqrt{\frac{2}{\pi}};$$

Diffusion coefficient:

$$\text{diff}_{mo}[x\_] := \frac{1}{2}\,\mu[x]\,\theta[x]\,n[x]\,\text{Abs}[ds_{mo}[x]];$$

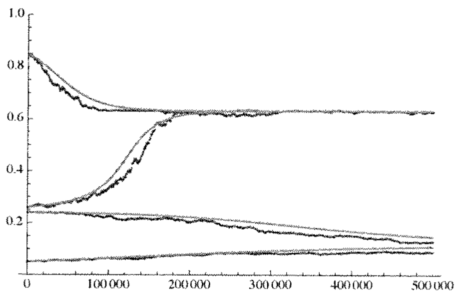
Single stochastic orbit per starting point (Euler metod):

```
x0 = {0.05, 0.24, 0.26, 0.85}; (* starting points *)
t0 = 0; (* start time *)
t∞ = 500000; (* stop time *)
Δt = 500; (* integration time step *)

noofStarts = Length[x0];

data = {};
For[i = 1, i ≤ noofStarts, i++,
  x = x0[[i]];
  t = t0;
  While[t ≤ t∞,
   data = Join[data, {{t, x}}];
   z = RandomReal[NormalDistribution[0, 1]];
   x = x + Δt drift_mo[x] + z Sqrt[Δt diff_mo[x]];
   t = t + Δt;
  ];
 ];
```

Stochastic orbits:

```
SDEorbit = ListPlot[data, PlotStyle → PointSize[0.005], Joined → False,
   AxesOrigin → {0, 0}, PlotRange → {0, 1}];
```

```
Show[SDEorbit, CEorbit]
```




```
data
```

```
{{0, 0.05}, {0, 0.24}, {0, 0.26}, {0, 0.85}}
```

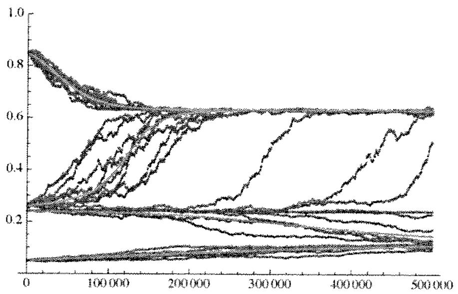
Multiple stochastic orbits for each starting point (Euler metod):

```
x0 = {0.05, 0.24, 0.26, 0.85}; (* starting points *)
t0 = 0; (* start time *)
t∞ = 500000; (* stop time *)
Δt = 500; (* integration time step *)

noofStarts = Length[x0];
sampleSize = 10;

data = {};
For[i = 1, i ≤ noofStarts, i ++,
  For[j = 1, j ≤ sampleSize, j ++,
    x = x0[[i]];
    t = t0;
    While[t ≤ t∞,
     data = Join[data, {{t, x}}];
     z = RandomReal[NormalDistribution[0, 1]];
     x = x + Δt drift_mo[x] + z Sqrt[Δt diff_mo[x]] ;
     t = t + Δt;
    ];
  ];
];
```

Stochastic orbits (blue) and deterministic orbits (red):

```
SDEorbit = ListPlot[data, PlotStyle → PointSize[0.005], Joined → False,
   AxesOrigin → {0, 0}, PlotRange → {0, 1}];
```

```
Show[SDEorbit, CEorbit]
```



DIMORPHIC RESIDENT POPULATION

Reset:

```
Clear[α, β, γ, δ, ε, r, k, x];
```

Dimorphic resident population equilibrium:

```
equu =
  {0 == r - r R / k - α ((1 - x1) n1 + (1 - x2) n2),
   0 == ϵ α (1 - x1) R - δ + γ β[x1] x1 ((1 - x1) n1 + (1 - x2) n2) - (1 - x1) (β[x1] x1 n1 + β[x2] x2 n2),
   0 == ϵ α (1 - x2) R - δ + γ β[x2] x2 ((1 - x1) n1 + (1 - x2) n2) -
     (1 - x2) (β[x1] x1 n1 + β[x2] x2 n2)};
var = {R, n1, n2};
Solve[equ, var];
Simplify[%]
```

```
{{R → (k ((x1 - x2) α δ + r x1 (-1 + x2) γ β[x1] - r (-1 + x1) x2 γ β[x2])) /
       (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2])),
  n1 → (k (x1 - x2) (-1 + x2) α^2 δ ϵ + r x1 (-1 + x2) γ (δ + k (-1 + x2) α ϵ) β[x1] +
      r x2 (γ (δ - k α ϵ) + x2 (δ - γ δ + k α γ ϵ) - x1 (δ + k (-1 + x2) α γ ϵ)) β[x2]) / (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2])^2),
  n2 → (r x1 (-x2 δ + γ δ - k α γ ϵ + k x2 α γ ϵ + x1 (δ - γ δ - k (-1 + x2) α γ ϵ)) β[x1] -
      (-1 + x1) (k (x1 - x2) α^2 δ ϵ - r x2 γ (δ + k (-1 + x1) α ϵ) β[x2])) / (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2])^2)}}
```

```
R[{x1_, x2_}] :=
  (k ((x1 - x2) α δ + r x1 (-1 + x2) γ β[x1] - r (-1 + x1) x2 γ β[x2])) /
   (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2]));

n1[{x1_, x2_}] :=
  (k (x1 - x2) (-1 + x2) α^2 δ ϵ + r x1 (-1 + x2) γ (δ + k (-1 + x2) α ϵ) β[x1] +
     r x2 (γ (δ - k α ϵ) + x2 (δ - γ δ + k α γ ϵ) - x1 (δ + k (-1 + x2) α γ ϵ)) β[x2]) /
   (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2])^2);

n2[{x1_, x2_}] :=
  (r x1 (-x2 δ + γ δ - k α γ ϵ + k x2 α γ ϵ + x1 (δ - γ δ - k (-1 + x2) α γ ϵ)) β[x1] -
     (-1 + x1) (k (x1 - x2) α^2 δ ϵ - r x2 γ (δ + k (-1 + x1) α ϵ) β[x2])) /
   (r γ (x1 (-1 + x2) β[x1] - (-1 + x1) x2 β[x2])^2);
```

Dimorphic invasion fitness and derivatives:

```
sdi[{x1_, x2_}, y_] :=
  ϵ α (1 - y) R[{x1,x2}] - δ + γ β[y] y ((1 - x1) n1[{x1, x2}] + (1 - x2) n2[{x1, x2}]) -
   (1 - y) (β[x1] x1 n1[{x1, x2}] + β[x2] x2 n2[{x1, x2}]);

x1dsdi[{x1_, x2_}] := (∂y sdi[{x1, x2}, y]) /. {y → x1};
x2dsdi[{x1_, x2_}] := (∂y sdi[{x1, x2}, y]) /. {y → x2};
x1ddsdi[{x1_, x2_}] := (∂y ∂y sdi[{x1, x2}, y]) /. {y → x1};
x2ddsdi[{x1_, x2_}] := (∂y ∂y sdi[{x1, x2}, y]) /. {y → x2};
```

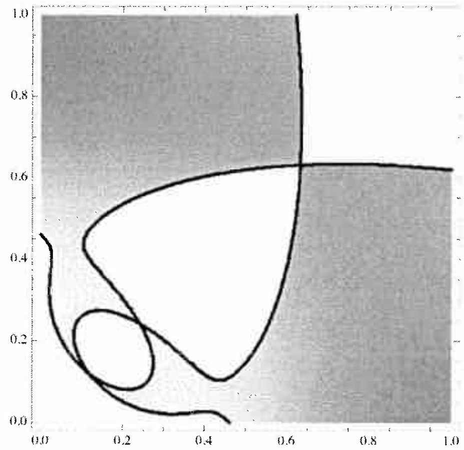
Default parameter values and functions :

```
α = 1; γ = 0.2; δ = 0.1; ϵ = 0.1; r = 1; k = 10;
p = 1; q = 2.5;
β[x_] := 2 - 9 (0.03 + x)^p (1 - x)^q;
```

Coexistence plot:

```
coexBnd = ContourPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0, 1, -1], {x1, 0, 1},
   {x2, 0, 1}, Contours → {0}, ContourStyle → {Black, Thick}, ContourShading → False,
   PlotPoints → 60];

coexInt = DensityPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0, n1[{x1, x2}] + n2[{x1, x2}]],
   {x1, 0, 1}, {x2, 0, 1}, PlotPoints → 60];
```

```
Show[coexInt, coexBnd]
```



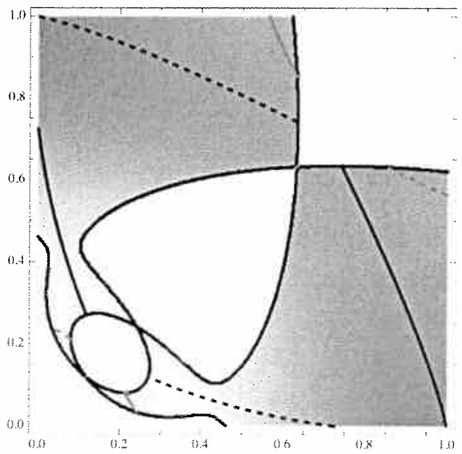
Isocline plot:

(solid = x1-isocline; dashed = x2-isocline; black = evolutionarily stable; red = not evolutionarily stable)

```
iso1ES = ContourPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0 && x1ddsdi[{x1, x2}] ≤ 0,
    x1dsdi[{x1, x2}]], {x1, 0, 1}, {x2, 0, 1}, Contours → {0}, ContourShading → False,
   ContourStyle → {Black, Thick}, PlotPoints → 30];

iso1NES = ContourPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0 && x1ddsdi[{x1, x2}] > 0,
    x1dsdi[{x1, x2}]], {x1, 0, 1}, {x2, 0, 1}, Contours → {0}, ContourShading → False,
   ContourStyle → {Red, Thick}, PlotPoints → 30];

iso2ES = ContourPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0 && x2ddsdi[{x1, x2}] ≤ 0,
    x2dsdi[{x1, x2}]], {x1, 0, 1}, {x2, 0, 1}, Contours → {0}, ContourShading → False,
   ContourStyle → {Black, Thick, Dashed}, PlotPoints → 30];

iso2NES = ContourPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0 && x2ddsdi[{x1, x2}] > 0,
    x2dsdi[{x1, x2}]], {x1, 0, 1}, {x2, 0, 1}, Contours → {0}, ContourShading → False,
   ContourStyle → {Red, Thick, Dashed}, PlotPoints → 30];
```

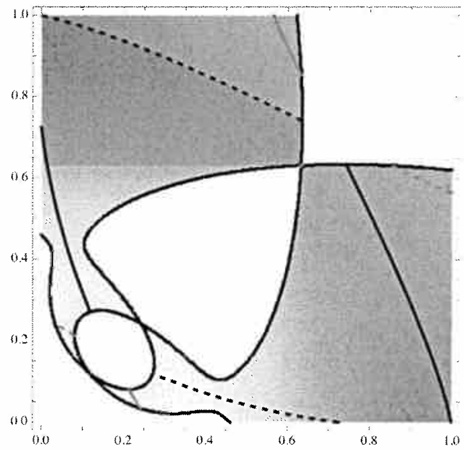
Isoclines (solid = x1-isocline; dashed = x2-isocline; black = ES; red = NES):

```
Show[coexInt, coexBnd, iso1ES, iso1NES, iso2ES, iso2NES]
```



Differential inclusion:

Reachability set:



Canonical equation:

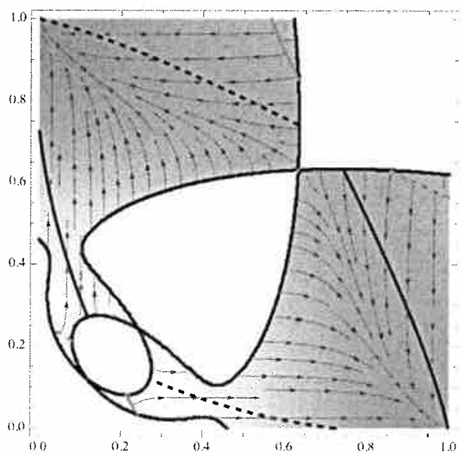
Deterministic drift:

```
drift_di[{x1_, x2_}] :=
  {1/2 μ[x1] σ[x1]^2 n1[{x1, x2}] x1ds_di[{x1, x2}],
   1/2 μ[x2] σ[x2]^2 n2[{x1, x2}] x2ds_di[{x1, x2}]};
```

Stream plot:

```
CEstream = StreamPlot[If[n1[{x1, x2}] > 0 && n2[{x1, x2}] > 0, drift_di[{x1, x2}], {0, 0}],
    {x1, 0, 1}, {x2, 0, 1}];
```

```
Show[coexInt, coexBnd, iso1ES, iso1NES, iso2ES, iso2NES, CEstream]
```



Stochastic orbits:

Diffusion coefficient:

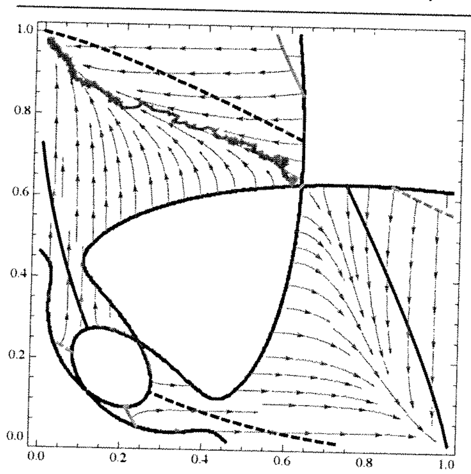
```
diff_di[{x1_, x2_}] :=
  {1/2 μ[x1] θ[x1] n1[{x1, x2}] Abs[x1ds_di[{x1, x2}]],
   1/2 μ[x2] θ[x2] n2[{x1, x2}] Abs[x2ds_di[{x1, x2}]]};
```

Single stochastic orbit (Euler metod):

```
x0 = {0.615, 0.645}; (* starting point *)
t0 = 0; (* start time *)
t∞ = 15 000 000; (* stop time *)
Δt = 5000; (* integration time step *)

data = {};
x = x0;
t = t0;
While[t ≤ t∞,
  data = Join[data, {Append[x, t]}];
  z = RandomReal[NormalDistribution[0, 1], 2];
  x = x + Δt drift_di[x] + z Sqrt[Δt diff_di[x]];
  t = t + Δt;
];
```

```
SDEorbit = ListPlot[data[[All, {1, 2}]], PlotStyle → {Thick}, Joined → True];
```

```
Show[coexBnd, iso1ES, iso1NES, iso2ES, iso2NES, CEstream, SDEorbit]
```



EVOLUTIONARY TREE

Deterministic approximation of the evolutionary tree using the canonical equation (CE).

```
x0 = 0.26; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 5 000 000; (* stop time *)
Δt = 500; (* integration time step *)

data = {};

(* monomorphic part of the deterministic tree; notice the extra stopping condition *)
x = x0;
t = t0;
While[t ≤ t∞ && ds_mo[x - 0.01 σ[x]] ds_mo[x + 0.01 σ[x]] > 0,
  data = Join[data, {{x, t}}];
  x = x + Δt drift_mo[x];
  t = t + Δt;
];

(* dimorphic part of the deterministic tree *)
x = {x - 0.1 σ[x], x + 0.1 σ[x]};
While[t ≤ t∞,
  data = Join[data, {{x[[1]], t}, {x[[2]], t}}];
  x = x + Δt drift_di[x];
  t = t + Δt;
];
```

```
CEtree = ListPlot[data, PlotStyle → {Red, PointSize[0.005]}, Joined → False,
   AxesOrigin → {0, 0}, PlotRange → {{0, 1}, All}];
```

```
Show[CEtree]
```