The Lotka - Volterra asymmetric competition model

Resident population dynamics

$$\frac{d}{dt} n_i = r[x_i]\, n_i \left(1 - \frac{\sum_{j=1}^{k} a[x_i, x_j]\, n_j}{K[x_i]}\right) \qquad (i = 1, \ldots, k)$$

Invader population dynamics

$$\frac{d}{dt} m = r[y]\, m \left(1 - \frac{\sum_{j=1}^{k} a[y, x_j]\, n_j}{k[y]}\right)$$

Invasion fitness

$$\left\langle \frac{d}{dt} \mathrm{Log}[m] \right\rangle = r[y] \left(1 - \frac{\sum_{j=1}^{k} a[y, x_j] \langle n_j \rangle}{k[y]}\right)$$

MONOMORPHIC RESIDENT POPULATION

Monomorphic resident population equilibrium:

```
n[x_] := k[x];
```

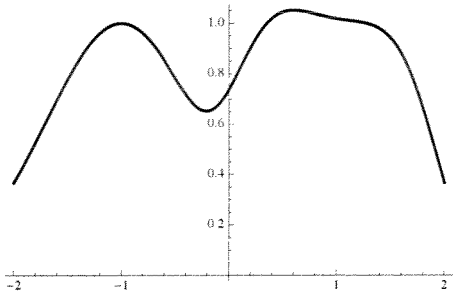
Invasion fitness and its derivatives:

```
s_mo[x_, y_] := r[y] (1 - a[y, x] n[x] / k[y]);
ds_mo[x_] := (∂_y s_mo[x, y]) /. {y → x};
dds_mo[x_] := (∂_{y,y} s_mo[x, y]) /. {y → x};
```

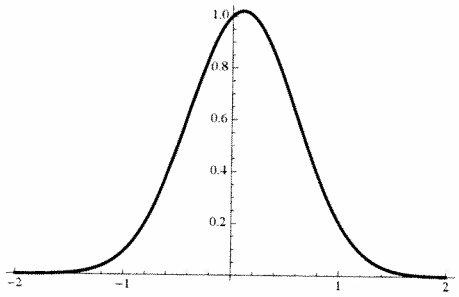
Default parameter values and functions :

```
r[x_] := 1;
k[x_] := Exp[-(x - δ)^4] + Exp[-(x + δ)^2];
δ = 1;
a[x_, y_] := Exp[-α (x - y)^2 - β (x - y)];
α = 2;
β = -0.4;
```

```
Plot[k[x], {x, -2, 2}, PlotStyle -> {Black, Thick}, AxesOrigin → {0, 0}]
```



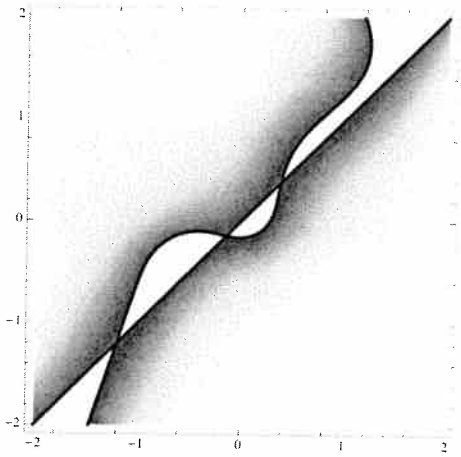
```
Plot[a[x, 0], {x, -2, 2}, PlotStyle -> {Black, Thick}, AxesOrigin → {0, 0}]
```



Pairwise invadability plot (PIP):

```
PIPbnd = ContourPlot[smo[x, y], {x, -2, 2}, {y, -2, 2}, Contours → {0},
   ContourStyle → {Black, Thick}, ContourShading → False, PlotPoints → 100];

PIPint = DensityPlot[If[smo[x, y] > 0, smo[x, y]], {x, -2, 2}, {y, -2, 2},
   PlotPoints → 100]; (* this second plot is just for effect *)
```

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

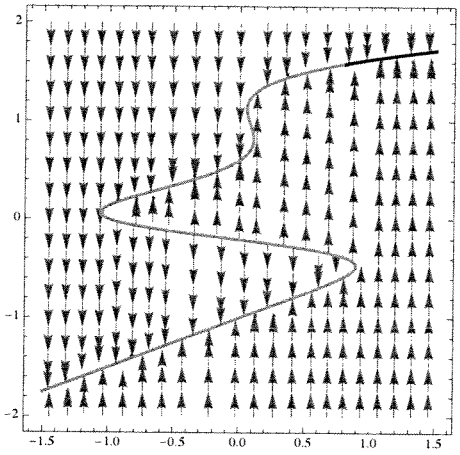


Bifurcation plot singular strategy x versus parameter β:

```
grad = StreamPlot[{0, dsmo[x]}, {β, -1.5, 1.5}, {x, -2, 2},
   StreamStyle → Arrowheads[0.04]];

xES = ContourPlot[If[ddsmo[x] ≤ 0, dsmo[x]], {β, -1.5, 1.5}, {x, -2, 2},
   Contours → {0}, ContourStyle → {Black, Thick}, ContourShading → False,
   PlotPoints → 30];

xNES = ContourPlot[If[ddsmo[x] > 0, dsmo[x]], {β, -1.5, 1.5}, {x, -2, 2},
   Contours → {0}, ContourStyle → {Red, Thick}, ContourShading → False, PlotPoints → 30];
```

Singular strategy x versus asymmetry parameter β:

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



Canonical equation:

Mutation rate and standard deviation of the mutation step distribution:

```
μ[x_] := 1;
σ[x_] := 0.05;
```

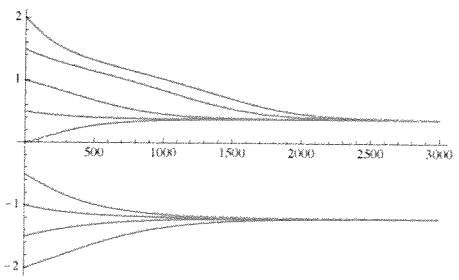
Deterministic drift:

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

```
(* this is the right hand side of the canonical equation *)
```

Deterministic orbit (Euler method):

```
x0 = Table[x, {x, -2, 2, 0.5}]; (* starting points *)
t0 = 0;(* start time *)
t∞ = 3000;(* stop time *)
Δt = 5;(* 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 driftmo[x];
   t = t + Δt;
  ];
 ];

CEplot = ListPlot[data, PlotStyle → {Red, PointSize[0.001]}, Joined → False,
  AxesOrigin → {0, 0}, PlotRange → All]
```



Stochastic differential equation (Ito):

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

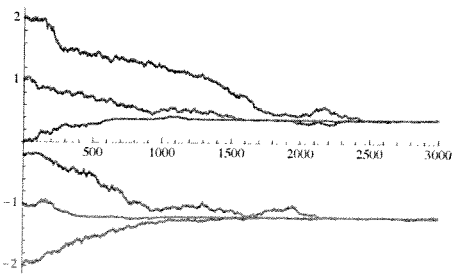
$$\theta 3[x\_] := 2\,\sigma[x]^3\,\sqrt{\frac{2}{\pi}};$$

Diffusion coefficient:

$$\text{diff}_{mo}[x\_] := \frac{1}{2}\,\mu[x]\,\theta 3[x]\,n[x]\,\text{Abs}[\text{ds}_{mo}[x]];$$

Single stochastic orbit per starting point (Euler metod):

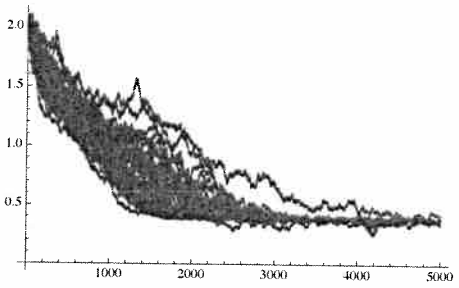
```
x0 = {-2, -1, -0.2, 0, 1, 2}; (* starting points *)
t0 = 0; (* start time *)
t∞ = 3000; (* stop time *)
Δt = 1; (* integration time step *)

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 driftmo[x] + z √(Δt diffmo[x]);
   t = t + Δt;
  ];
 ];

SDEplot = ListPlot[data, PlotStyle → PointSize[0.001], Joined → False,
  AxesOrigin → {0, 0}]
```

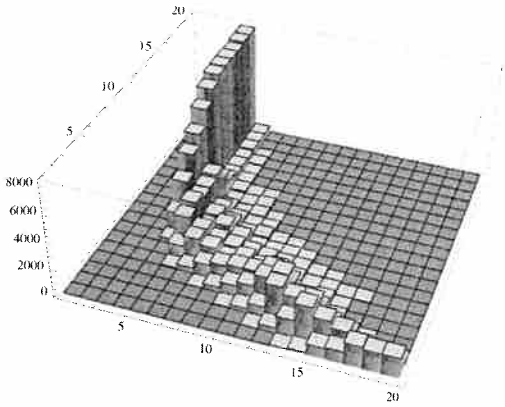


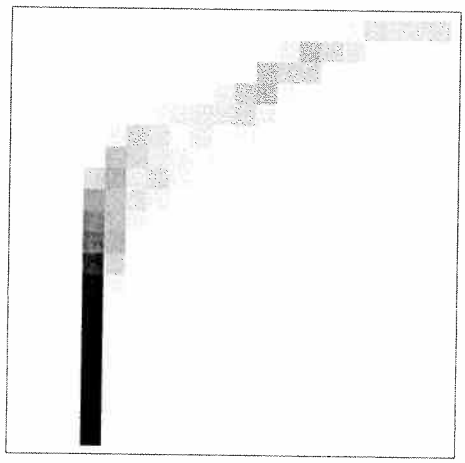
Multiple stochastic orbits for a single starting point (Euler metod):

```
x0 = 2.0;(* starting point orbit *)
t0 = 0;(* start time *)
t∞ = 5000;(* stop time *)
Δt = 1;(* integration time step *)
SampleSize = 100;(* number of orbits from the same starting point *)

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

```
ListPlot[data, Joined → False, PlotStyle → PointSize[0.001]]

counts = BinCounts[data, {t0, t∞, 250 Δt}, {0, 2, 0.1}];

ListPlot3D[counts, InterpolationOrder → 0, Filling → Axis, Mesh → None, PlotRange → All]

ArrayPlot[counts]
```







DIMORPHIC RESIDENT POPULATION

Reset:

```
Clear[r, k, a, α, β];
```

Dimorphic resident population equilibrium:

```
Solve[{0 == 1 - (n1 + a[x1, x2] n2)/k[x1], 0 == 1 - (a[x2, x1] n1 + n2)/k[x2]}, {n1, n2}];
Simplify[%]
```

$$\left\{\left\{n1 \to \frac{k[x1] - a[x1, x2]\, k[x2]}{1 - a[x1, x2]\, a[x2, x1]},\ n2 \to \frac{a[x2, x1]\, k[x1] - k[x2]}{-1 + a[x1, x2]\, a[x2, x1]}\right\}\right\}$$

```
n1[x1_, x2_] := (k[x1] - a[x1, x2] k[x2]) / (1 - a[x1, x2] a[x2, x1]);

n2[x1_, x2_] := (k[x2] - a[x2, x1] k[x1]) / (1 - a[x1, x2] a[x2, x1]);
```

Dimorphic invasion fitness and derivatives

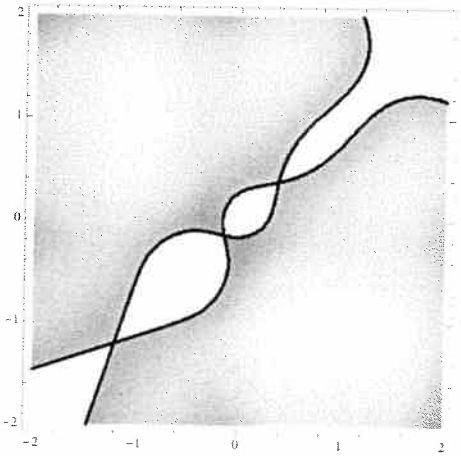
```
sdi[x1_, x2_, y_] := r[y] (1 - (a[y, x1] n1[x1, x2] + a[y, x2] n2[x1, x2]) / k[y]);

x1dsdi[x1_, x2_] := ∂y sdi[x1, x2, y] /. {y → x1};

x1ddsdi[x1_, x2_] := ∂y,y sdi[x1, x2, y] /. {y → x1};

x2dsdi[x1_, x2_] := ∂y sdi[x1, x2, y] /. {y → x2};

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

Default parameter values and functions :

```
r[x_] := 1;
k[x_] := Exp[-(x - δ)^4] + Exp[-(x + δ)^2];
δ = 1;
a[x_, y_] := Exp[-α (x - y)^2 - β (x - y)];
α = 2;
β = -0.4;
```

Coexistence plot:

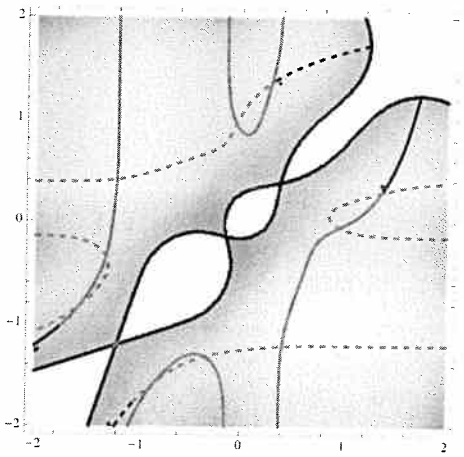
```
coexBnd = ContourPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0, 1, -1], {x1, -2, 2},
   {x2, -2, 2}, Contours → {0}, ContourStyle → {Black, Thick}, ContourShading → False,
   PlotPoints → 50];

coexInt = DensityPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0, n1[x1, x2] + n2[x1, x2]],
   {x1, -2, 2}, {x2, -2, 2}, PlotPoints → 50];

Show[coexInt, coexBnd]
```



Isocline plot

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

```
x1isoES = ContourPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0 && x1ddsdi[x1, x2] ≤ 0,
     x1dsdi[x1, x2]], {x1, -2, 2}, {x2, -2, 2}, Contours → {0}, ContourShading → False,
   ContourStyle → {Black, Thick}];

x1isoNES = ContourPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0 && x1ddsdi[x1, x2] > 0,
     x1dsdi[x1, x2]], {x1, -2, 2}, {x2, -2, 2}, Contours → {0}, ContourShading → False,
   ContourStyle → {Red, Thick}];

x2isoES = ContourPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0 && x2ddsdi[x1, x2] ≤ 0,
     x2dsdi[x1, x2]], {x1, -2, 2}, {x2, -2, 2}, Contours → {0}, ContourShading → False,
   ContourStyle → {Black, Thick, Dashed}];

x2isoNES = ContourPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0 && x2ddsdi[x1, x2] > 0,
     x2dsdi[x1, x2]], {x1, -2, 2}, {x2, -2, 2}, Contours → {0}, ContourShading → False,
   ContourStyle → {Red, Thick, Dashed}];

Show[coexInt, coexBnd, x1isoES, x1isoNES, x2isoES, x2isoNES]
```



Canonical equation:

Mutation rate and standard deviation of the mutation step distribution:

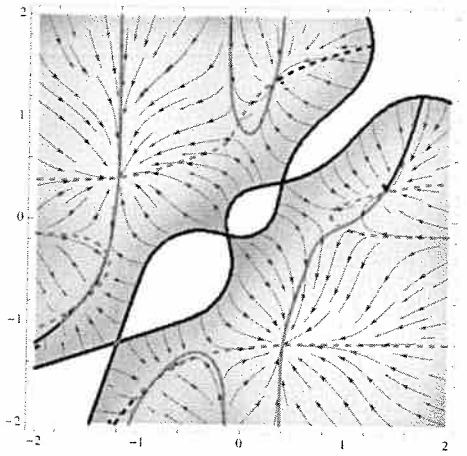
```
μ[x_] := 1;
σ[x_] := 0.05;
```

Deterministic drift:

$$\text{drift}_{di}[\{x1\_, x2\_\}] := \left\{\frac{1}{2}\,\mu[x1]\,\sigma[x1]^2\,n1[x1, x2]\,x1ds_{di}[x1, x2],\ \frac{1}{2}\,\mu[x2]\,\sigma[x2]^2\,n2[x1, x2]\,x2ds_{di}[x1, x2]\right\};$$

Stream plot:

```
CEstream = StreamPlot[If[n1[x1, x2] > 0 && n2[x1, x2] > 0, drift_di[{x1, x2}], {0, 0}],
    {x1, -2, 2}, {x2, -2, 2}];

Show[coexInt, coexBnd, x1isoES, x1isoNES, x2isoES, x2isoNES, CEstream]
```



Stochastic differential equation (Ito)

$\theta_3[x]$ = third absolute moment of the mutation step distribution

(assumed to be Gaussian with mean zero and standard variation $\sigma[x]$)

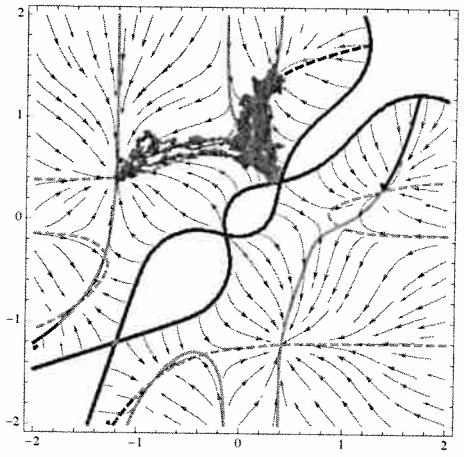
```
θ3[x_] = 2 σ[x]^3 √(2 / Pi);
```

Diffusion coefficient

```
diff_di[{x1_, x2_}] :=
  {1/2 μ[x1] θ3[x1] n1[x1, x2] Abs[x1ds_di[x1, x2]],
   1/2 μ[x2] θ3[x2] n2[x1, x2] Abs[x2ds_di[x1, x2]]};
```

Stochastic orbits (Euler metod)

```
x0 = {0.35, 0.41};(* starting point orbit *)
t0 = 0;(* start time *)
t∞ = 5000;(* stop time *)
Δt = 5;(* integration time step *)
SampleSize = 50;(* number of orbits from the same starting point *)

data = {};
For[i = 1, i ≤ SampleSize, i++,
  x = x0;
  t = t0;
  While[t ≤ t∞,
   data = Join[data, {x}];
   z = RandomReal[NormalDistribution[0, 1], 2];
   x = x + Δt drift_di[x] + z √(Δt diff_di[x]);
   t = t + Δt;
  ];
 ];

SDEplot = ListPlot[data, Joined → False, PlotStyle → PointSize[0.001]];

Show[coexBnd, x1isoES, x1isoNES, x2isoES, x2isoNES, CEstream, SDEplot]
```



This is how stochasticity matters : the CE gives convergence to the uppermost dimorphic singularity; the SDE show that about 1 in 10 of the orbits converges to the middle singularity.

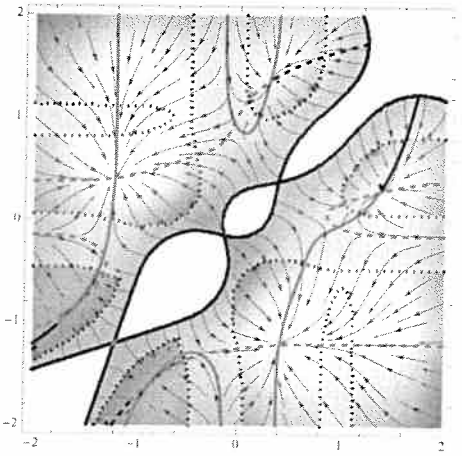
STABILITY OF DIMORPHIC SINGULAR POINTS

Jacobi matrix for stability calculations

```
a11[x1_, x2_] := (∂Y,Y sdi[x1, x2, Y] + ∂X1,Y sdi[X1, x2, Y]) /. {X1 → x1, Y → x1};

a12[x1_, x2_] := (∂X2,Y sdi[x1, X2, Y]) /. {X2 → x2, Y → x1};

a21[x1_, x2_] := (∂X1,Y sdi[X1, x2, Y]) /. {X1 → x1, Y → x2};

a22[x1_, x2_] := (∂Y,Y sdi[x1, x2, Y] + ∂X2,Y sdi[x1, X2, Y]) /. {X2 → x2, Y → x2};
```

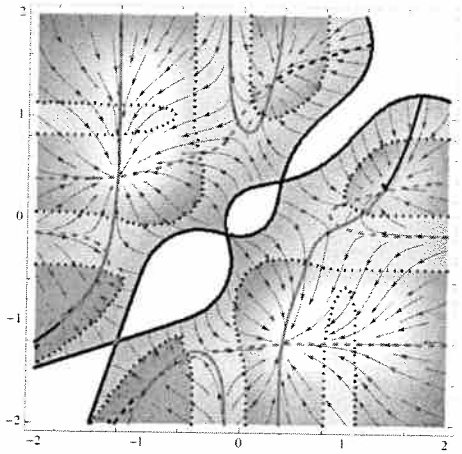
Total stability:

```
test[x1_, x2_] := n1[x1, x2] > 0 && n2[x1, x2] > 0 &&
    a11[x1, x2] < 0 && a22[x1, x2] < 0 &&
    a11[x1, x2] a22[x1, x2] > Abs[a12[x1, x2] a21[x1, x2]];

totBnd = ContourPlot[If[test[x1, x2], 1, -1], {x1, -2, 2}, {x2, -2, 2},
    Contours → {0}, ContourStyle → {Thick, Dotted}, ContourShading → False,
    PlotPoints → 30];

totInt = DensityPlot[If[test[x1, x2], n1[x1, x2] + n2[x1, x2]], {x1, -2, 2},
    {x2, -2, 2}, ColorFunction → "FuchsiaTones"];

Show[coexInt, coexBnd, totInt, totBnd, x1isoES, x1isoNES, x2isoES, x2isoNES,
 CEstream]
```



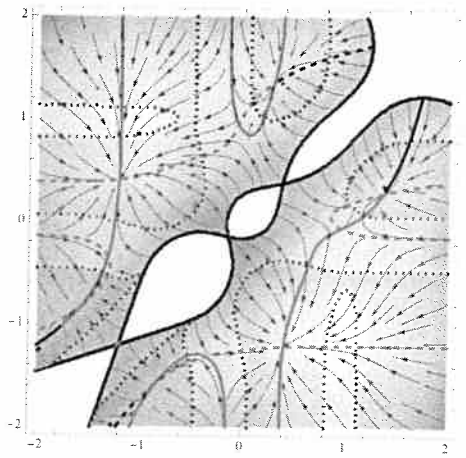
Strong stability:

```
test[x1_, x2_] := n1[x1, x2] > 0 && n2[x1, x2] > 0 &&
    a11[x1, x2] < 0 && a22[x1, x2] < 0 && a11[x1, x2] a22[x1, x2] > a12[x1, x2] a21[x1, x2];

strBnd = ContourPlot[If[test[x1, x2], 1, -1], {x1, -2, 2}, {x2, -2, 2},
    Contours → {0}, ContourStyle → {Thick, Dotted}, ContourShading → False,
    PlotPoints → 30];

strInt = DensityPlot[If[test[x1, x2], n1[x1, x2] + n2[x1, x2]], {x1, -2, 2},
    {x2, -2, 2}, ColorFunction → "CherryTones"];

Show[coexInt, coexBnd, strInt, strBnd, x1isoES, x1isoNES, x2isoES, x2isoNES,
 CEstream]
```



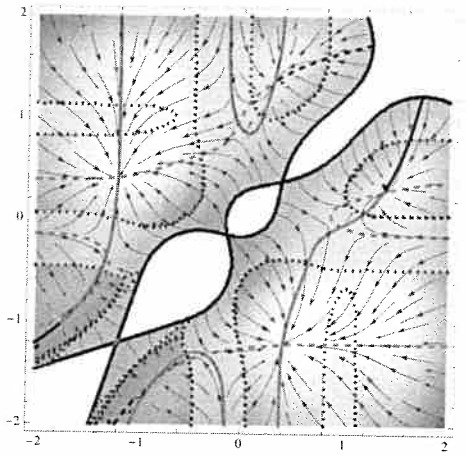
Weak stability:

```
test[x1_, x2_] := n1[x1, x2] > 0 && n2[x1, x2] > 0 &&
    μ σ^2 n1[x1, x2] a11[x1, x2] + μ σ^2 n2[x1, x2] a22[x1, x2] < 0 &&
    a11[x1, x2] a22[x1, x2] > a12[x1, x2] a21[x1, x2];

wkBnd = ContourPlot[If[test[x1, x2], 1, -1], {x1, -2, 2}, {x2, -2, 2},
    Contours → {0}, ContourStyle → {Thick, Dotted}, ContourShading → False,
    PlotPoints → 30];

wkInt = DensityPlot[If[test[x1, x2], n1[x1, x2] + n2[x1, x2]], {x1, -2, 2},
    {x2, -2, 2}, ColorFunction → "SiennaTones"];

Show[coexInt, coexBnd, wkInt, wkBnd, x1isoES, x1isoNES, x2isoES, x2isoNES,
 CEstream]
```



Total/strong/weak stability in one plot:

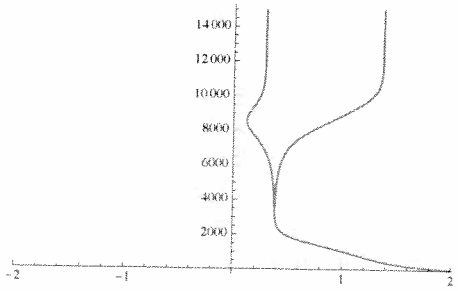
```
Show[coexInt, coexBnd, wkInt, wkBnd, strInt, strBnd, absInt, absBnd, x1isoES,
 x1isoNES, x2isoES, x2isoNES, CEstream]
```



EVOLUTIONARY TREE

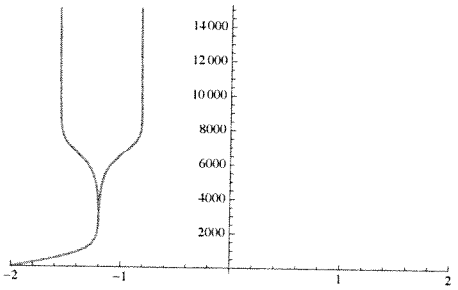
Deterministic approximation of the evolutionary tree using the canonical equation (CE).

```
x0 = 2; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 15 000; (* stop time *)
Δt = 5; (* integration time step *)

data = {};

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

ListPlot[data, PlotStyle → {Red, PointSize[0.005]}, Joined → False,
 AxesOrigin → {0, 0}, PlotRange → {{-2, 2}, All}]
```



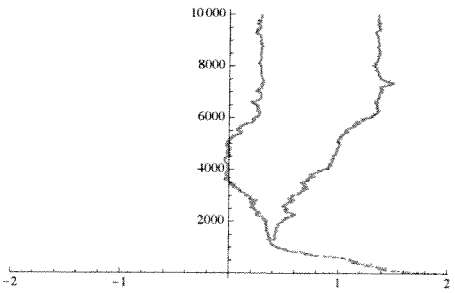
Deterministic approximation of the evolutionary tree using the canonical equation (CE) with a starting point.

```
x0 = -2; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 15 000; (* stop time *)
Δt = 5; (* 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;
 ];

ListPlot[data, PlotStyle → {Red, PointSize[0.005]}, Joined → False,
 AxesOrigin → {0, 0}, PlotRange → {{-2, 2}, All}]
```



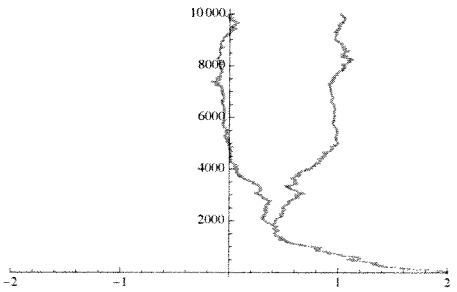
SDE (Ito) approximation of the evolutionary tree (with convergence to the more likely dimorphic singularity).

```
x0 = 2; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 10 000; (* stop time *)
Δt = 5; (* 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}}];
  z = RandomReal[NormalDistribution[0, 1]];
  x = x + Δt drift_mo[x] + z √(Δt diff_mo[x]) ;
  t = t + Δt;
 ];

(* dimorphic part of the deterministic tree *)
x = {x - 0.5 σ[x], x + 0.5 σ[x]};
While[t ≤ t∞,
  data = Join[data, {{x[[1]], t}, {x[[2]], t}}];
  z = RandomReal[NormalDistribution[0, 1], 2];
  x = x + Δt drift_di[x] + z √(Δt diff_di[x]) ;
  t = t + Δt;
 ];

ListPlot[data, PlotStyle → {Red, PointSize[0.001]}, Joined → False,
 AxesOrigin → {0, 0}, PlotRange → {{-2, 2}, All}]
```



SDE (Ito) approximation of the evolutionary tree (with convergence to the less likely dimorphic singularity).

```
x0 = 2; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 10 000; (* stop time *)
Δt = 5; (* integration time step *)

data = {};

(* monomorphic part of the deterministic tree;
notice the extra stopping condition *)
x = x0;
t = t0;
While[t ≤ t∞ && dsmo[x - 0.01 σ[x]] dsmo[x + 0.01 σ[x]] > 0,
  data = Join[data, {{x, t}}];
  z = RandomReal[NormalDistribution[0, 1]];
  x = x + Δt driftmo[x] + z √(Δt diffmo[x]) ;
  t = t + Δt;
 ];

(* dimorphic part of the deterministic tree *)
x = {x - 0.5 σ[x], x + 0.5 σ[x]};
While[t ≤ t∞,
  data = Join[data, {{x[[1]], t}, {x[[2]], t}}];
  z = RandomReal[NormalDistribution[0, 1], 2];
  x = x + Δt driftdi[x] + z √(Δt diffdi[x]) ;
  t = t + Δt;
 ];

ListPlot[data, PlotStyle → {Red, PointSize[0.001]}, Joined → False,
 AxesOrigin → {0, 0}, PlotRange → {{-2, 2}, All}]
```



SDE (Ito) approximation of the evolutionary tree (with a different starting point).

```
x0 = -2; (* starting point of tree *)
t0 = 0; (* start time *)
t∞ = 10 000; (* stop time *)
Δt = 1; (* 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}}];
  z = RandomReal[NormalDistribution[0, 1]];
  x = x + Δt drift_mo[x] + z √(Δt diff_mo[x]) ;
  t = t + Δt;
 ];

(* dimorphic part of the deterministic tree *)
x = {x - 0.3 σ[x], x + 0.3 σ[x]};
While[t ≤ t∞,
  data = Join[data, {{x[[1]], t}, {x[[2]], t}}];
  z = RandomReal[NormalDistribution[0, 1], 2];
  x = x + Δt drift_di[x] + z √(Δt diff_di[x]) ;
  t = t + Δt;
 ];

ListPlot[data, PlotStyle → {Red, PointSize[0.001]}, Joined → False,
 AxesOrigin → {0, 0}, PlotRange → {{-2, 2}, All}]
```