# Prey – predator coevolution

$$\frac{d}{dt}m_i = r[x_i]\, m_i \left(1 - \frac{\sum_{j=1}^{k} a[x_i, x_j]\, m_j}{K[x_i]}\right) - m_i \sum_{j=1}^{l} b[x_i, y_j]\, n_j \quad (i = 1, \ldots, k) \quad \text{(prey)}$$

$$\frac{d}{dt}n_i = n_i \sum_{j=1}^{k} c[x_j, y_i]\, b[x_j, y_i]\, m_j - d[y_i]\, n_i \qquad (i = 1, \ldots, l) \ \text{(predator)}$$

## MONOMORPHIC PREY AND PREDATOR POPULATIONS

- **Resident dynamic s**

- **m = prey pop dens; n = pred pop dens; x = log body size prey; y = log body size pred**

```
dLogm[m_, n_] := r[x] (1 - m a[x, x] / K[x]) - n b[x, y];
dLogn[m_, n_] := m b[x, y] c[x, y] - d[y];
```

## Population equilibrium (= time averages // explain!)

```
Solve[{dLogm[m, n] == 0, dLogn[m, n] == 0}, {m, n}]
```

$$\left\{\left\{n \to \frac{r[x]}{b[x, y]} - \frac{a[x, x]\, d[y]\, r[x]}{b[x, y]^2\, c[x, y]\, K[x]},\ m \to \frac{d[y]}{b[x, y]\, c[x, y]}\right\}\right\}$$

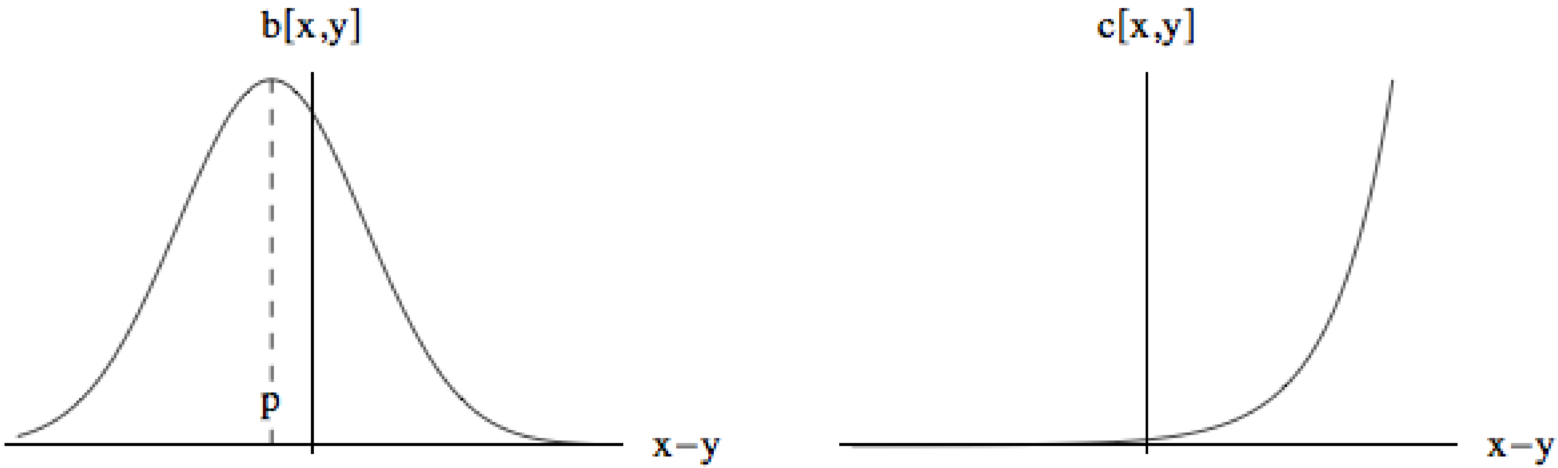
$$m[\{x\_, y\_\}] := \frac{d[y]}{b[x, y]\, c[x, y]};$$

$$n[\{x\_, y\_\}] := \frac{r[x]}{b[x, y]} - \frac{d[y]\, r[x]\, a[x, x]}{K[x]\, b[x, y]^2\, c[x, y]};$$

## Parameter values and functions

```
r[x_] := 1;
K[x_] := e^(-x^2);
a[x1_, x2_] := e^(-α (x1-x2)^2); α = 0.5;
b[x_, y_] := β1 e^(-β2 (-p+x-y)^2); β1 = 100; β2 = 0.2; p = Log[0.5];(* attack rate *)
c[x_, y_] := γ e^(x-y); γ = 0.2;(* conversion coefficient *)
d[y_] := δ1 e^(y (-δ2)); δ1 = 1; δ2 = 1;
```

```
Row[{Show[Plot[b[x, 0], {x, -5, 5}, AxesOrigin → {0, 0}, PlotStyle → Black],
    Graphics[{Dashed, Line[{{p, 0}, {p, b[p, 0]}}]}],
    Graphics[Text["p", {p, 12}, Background → White]], Ticks → None, AxesLabel → {"x-y", "b[x,y]"},
   Show[Plot[c[x, 0], {x, -5, 5}, AxesOrigin → {0, 0}, PlotStyle → Black], Ticks → None, AxesLabel →
```




## ■ Coexistence

```
xmin = -1.5; xmax = 2.5; ymin = -3.5; ymax = 6;
```

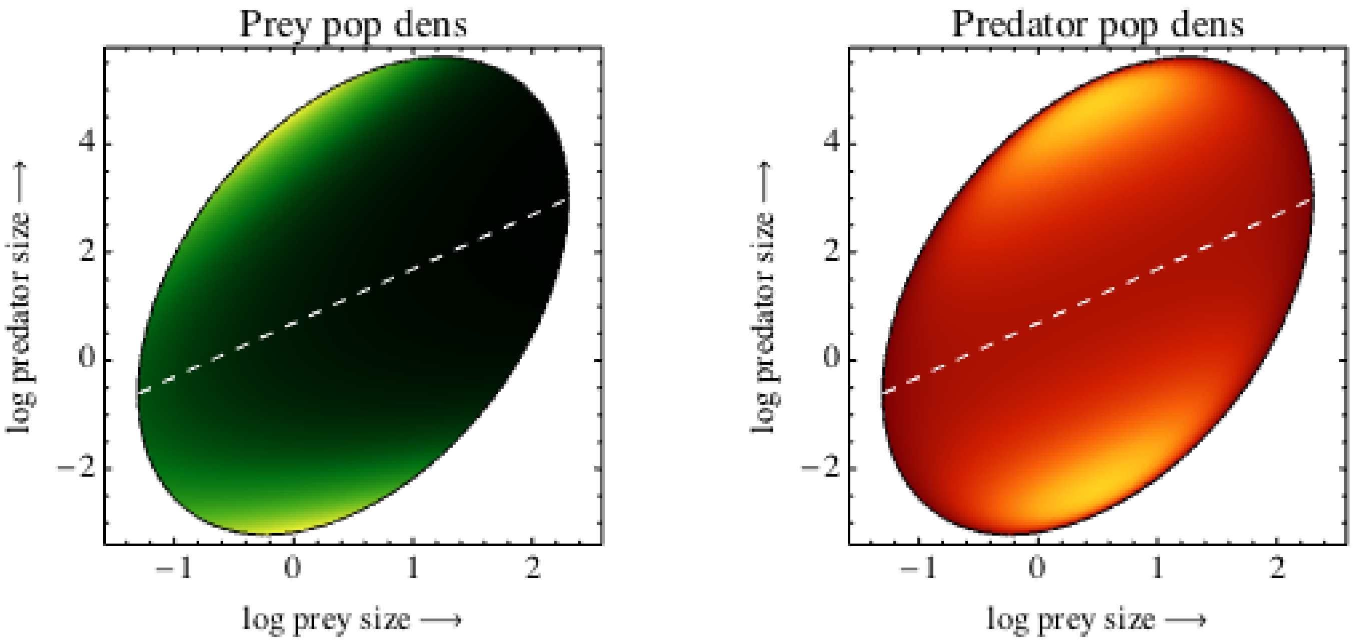
```
coexBnd = ContourPlot[If[m[{x, y}] > 0 && n[{x, y}] > 0, 1, -1], {x, xmin, xmax}, {y, ymin, ymax}, Co
    PlotPoints → 60];
```

```
preyDens = DensityPlot[If[m[{x, y}] > 0 && n[{x, y}] > 0, m[{x, y}], k[x]], {x, xmin, xmax}, {y, ymin
   PlotPoints → 60];
```

```
predDens = DensityPlot[If[m[{x, y}] > 0 && n[{x, y}] > 0, n[{x, y}]], {x, xmin, xmax}, {y, ymin, ymax
```

```
optRatio = Plot[x - p, {x, xmin, xmax}, PlotStyle → {White, Dashed}];
```

```
Row[{Show[preyDens, coexBnd, optRatio, FrameLabel → {"log prey size ⟶", "log predator size ⟶"
  "        ", Show[predDens, coexBnd, optRatio, FrameLabel → {"log prey size ⟶", "log predator si
```



# Invasion fitness and derivatives

## Prey

```
sPrey_mo[{x_, y_}, xMut_] := r[xMut] (1 - a[xMut, x] m[{x, y}] / K[xMut]) - b[xMut, y] n[{x, y}];
dsPrey_mo[{x_, y_}] := ∂_xMut sPrey_mo[{x, y}, xMut] /. {xMut → x};
ddsPrey_mo[{x_, y_}] := ∂_{xMut,xMut} sPrey_mo[{x, y}, xMut] /. {xMut → x};
```

## Predator

```
sPred_mo[{x_, y_}, yMut_] := b[x, yMut] c[x, yMut] m[{x, y}] - d[yMut];
dsPred_mo[{x_, y_}] := ∂_yMut sPred_mo[{x, y}, yMut] /. {yMut → y};
ddsPred_mo[{x_, y_}] := ∂_{yMut,yMut} sPred_mo[{x, y}, yMut] /. {yMut → y};
```

# Isoclines plot

## Prey

- *solid = uninvadable (ES); dashed = invadable (NES)*

```
preyES = ContourPlot[If[n[{x, y}] > 0 && ddsPreymo[{x, y}] ≤ 0, dsPreymo[{x, y}]], {x, xmin, xmax}, {
    ContourStyle → {Darker[Green], Thick}, PlotPoints → 30];
preyNES = ContourPlot[If[n[{x, y}] > 0 && ddsPreymo[{x, y}] > 0, dsPreymo[{x, y}]], {x, xmin, xmax},
    ContourStyle → {Darker[Green], Thick, Dashed}, PlotPoints → 30];
```

## Predator

- *solid = uninvadable (ES); dashed = invadable (NES)*

```
predES = ContourPlot[If[n[{x, y}] > 0 && ddsPredmo[{x, y}] ≤ 0, dsPredmo[{x, y}]], {x, xmin, xmax}, {
    ContourStyle → {Red, Thick}, PlotPoints → 30];
predNES = ContourPlot[If[n[{x, y}] > 0 && ddsPredmo[{x, y}] > 0, dsPredmo[{x, y}]], {x, xmin, xmax},
    ContourStyle → {Red, Thick, Dashed}, PlotPoints → 30];
```

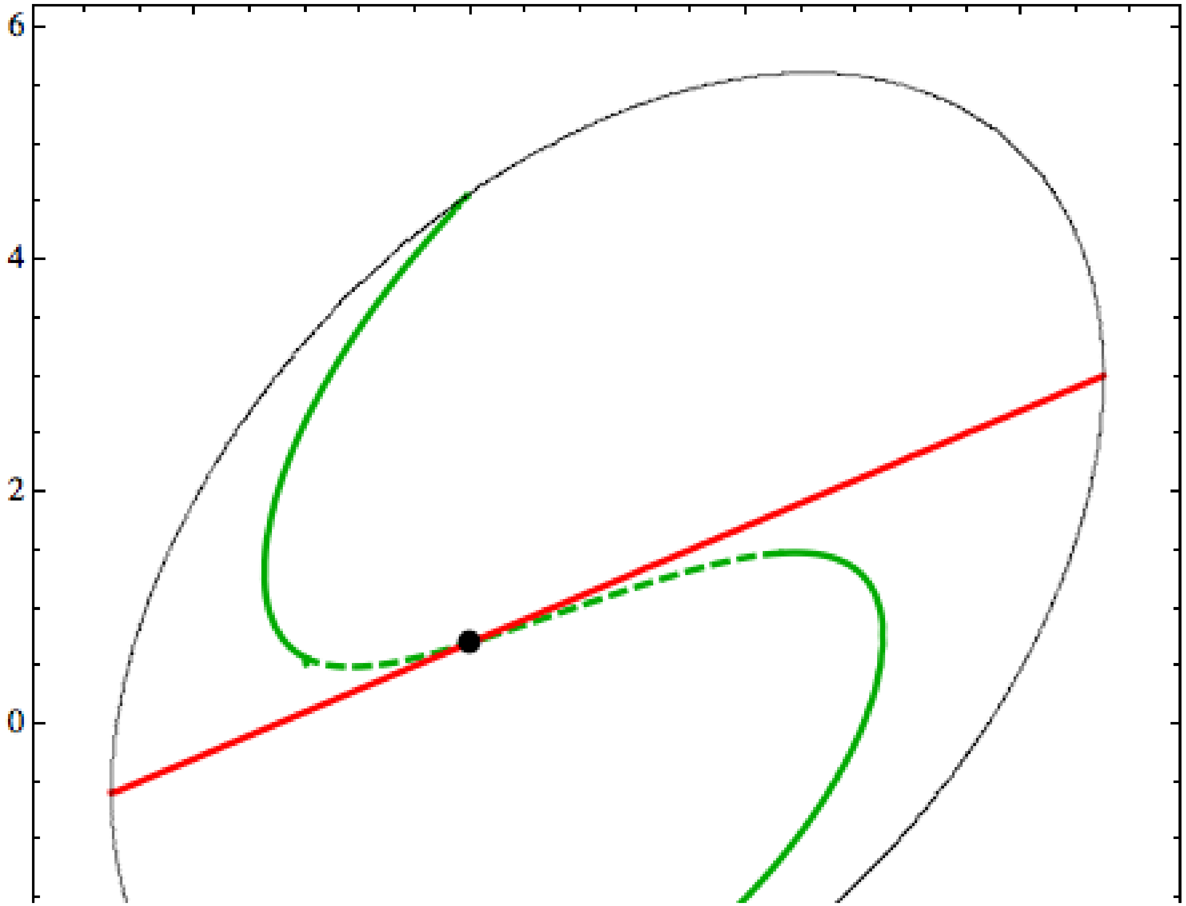
## Singularity

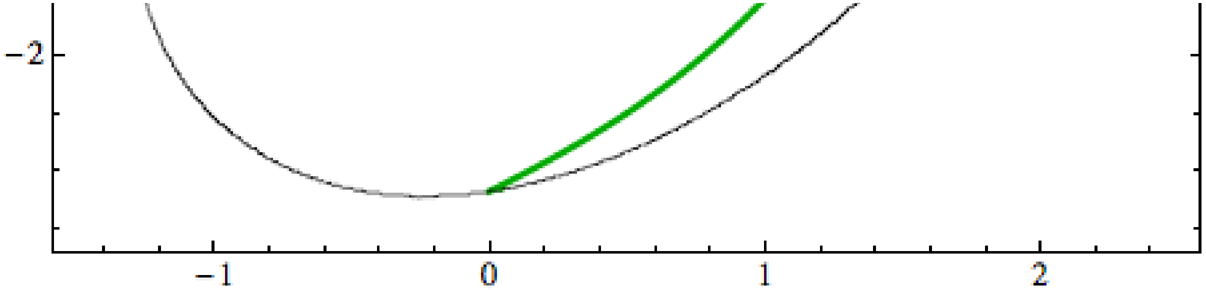
```
singPoint = {x, y} /. FindRoot[{dsPreymo[{x, y}] == 0, dsPredmo[{x, y}] == 0}, {x, p}, {y, 0}];
sngPnt = Graphics[{PointSize → Large, Point[singPoint]}];
```

## ■ Isocline plot

- ***green = prey isocline; red = predator isocline; solid = uninvadable (ES); dashed = invadable (NES)***

```
Show[preyES, preyNES, predES, predNES, coexBnd, sngPnt]
```





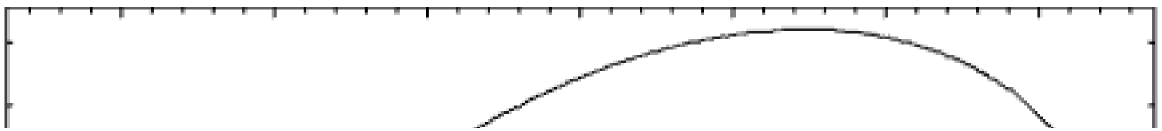

- **Stability**

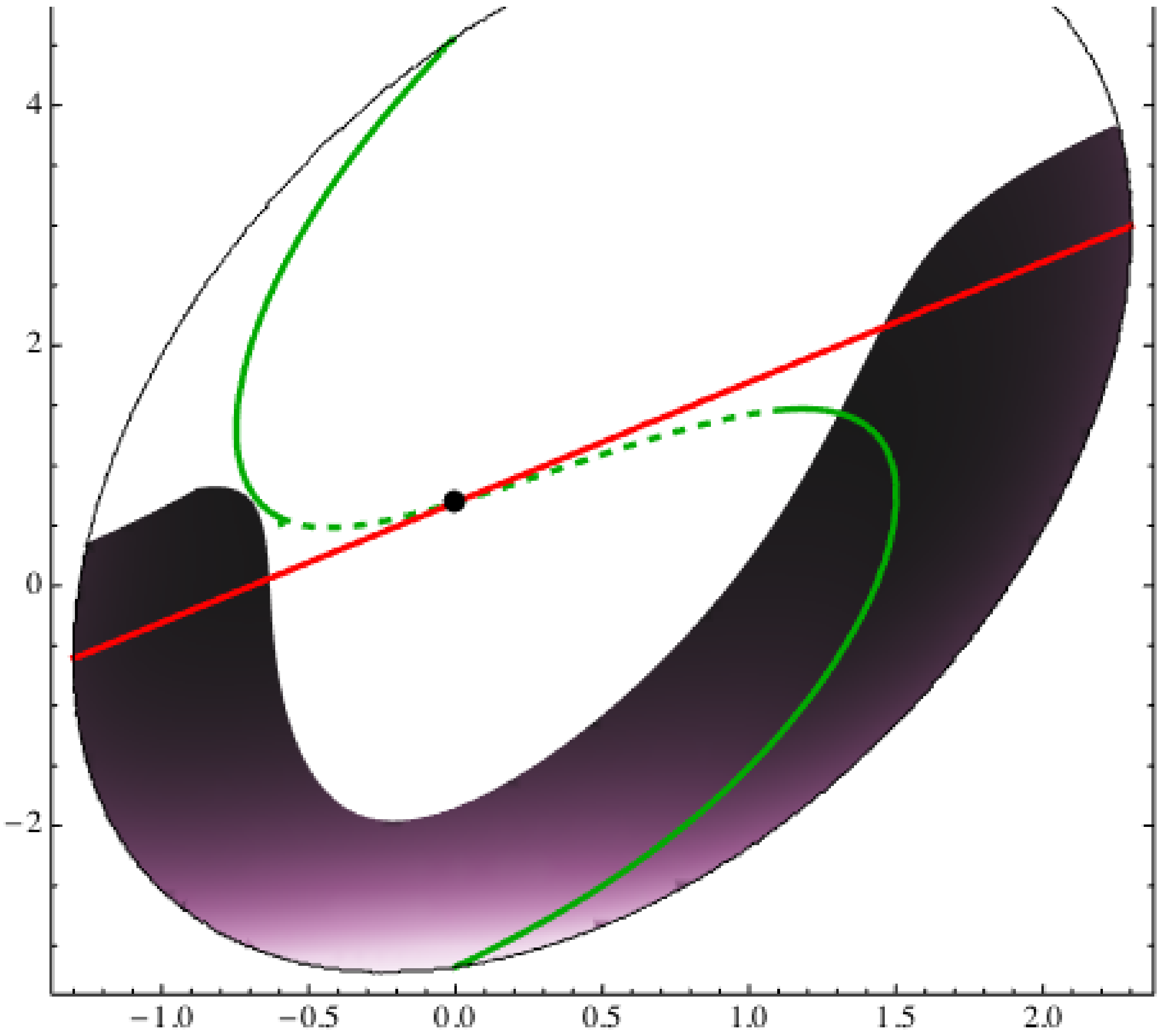
- **Jacobi matrix** {{a11, a12}, {a21, a22}}

```
a11[{x_, y_}] := ∂X2,X2 sPreymo[{x, y}, X2] + ∂X1,X2 sPreymo[{X1, y}, X2] /. {X1 → x, X2 → x};
a12[{x_, y_}] := ∂X,Y sPreymo[{x, Y}, X] /. {X → x, Y → y};
a21[{x_, y_}] := ∂X,Y sPredmo[{X, y}, Y] /. {X → x, Y → y};
a22[{x_, y_}] := ∂Y2,Y2 sPredmo[{x, y}, Y2] + ∂Y1,Y2 sPredmo[{x, Y1}, Y2] /. {Y1 → y, Y2 → y};
```

- **Total stability**

```
totStab = DensityPlot[If[n[{x, y}] > 0 && a11[{x, y}] < 0 ∧ a22[{x, y}] < 0 ∧ a11[{x, y}] a22[{x, y}] :
    Abs[dsPredmo[{x, y}]] + Abs[dsPreymo[{x, y}]]], {x, xmin, xmax}, {y, ymin, ymax}, PlotRange → A

Show[totStab, preyES, preyNES, predES, predNES, coexBnd, sngPnt]
```

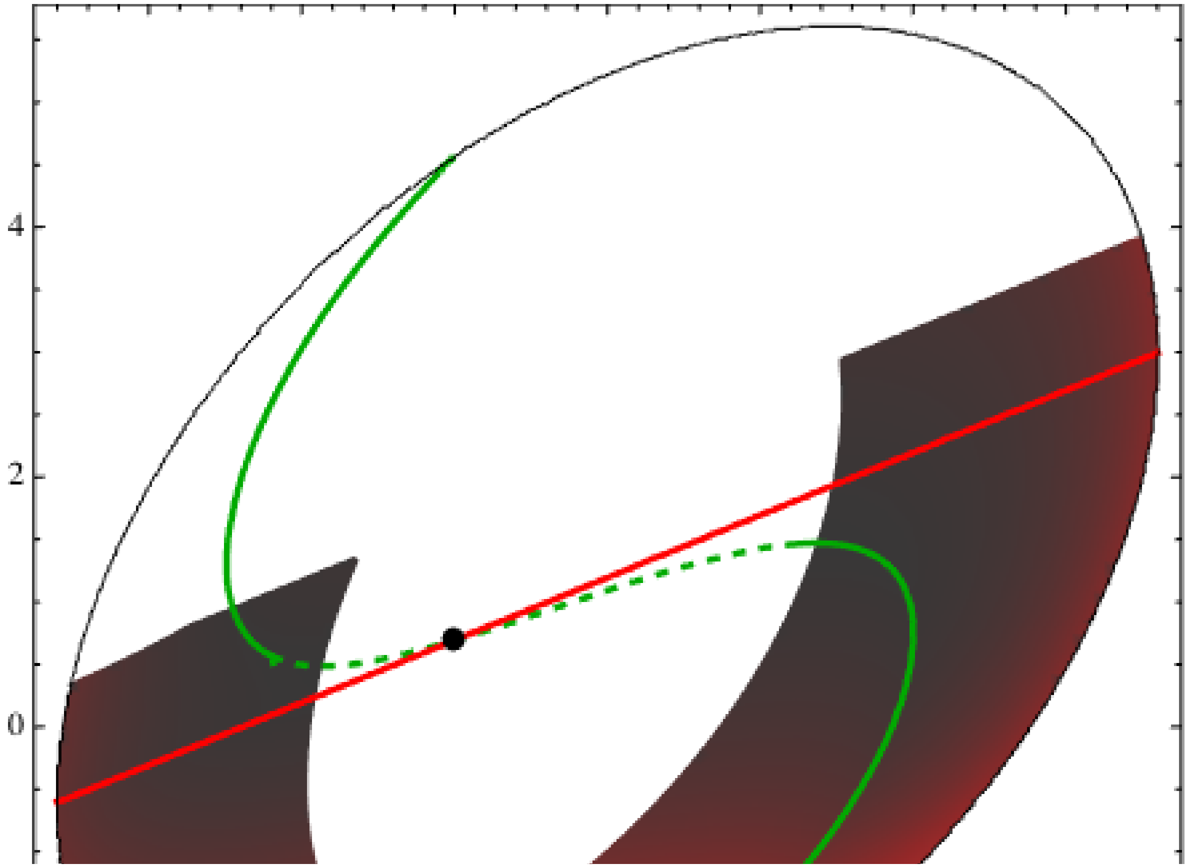


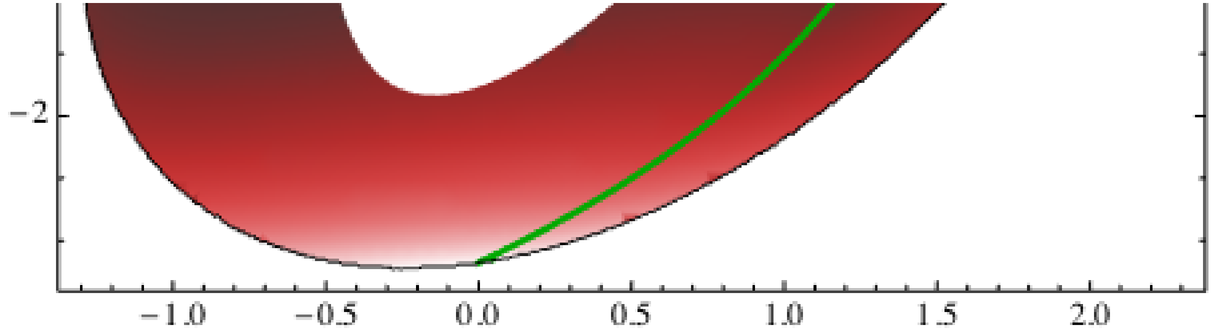



- ***Conclusion: the singular point is NOT ABSOLUTELY STABLE.***

- **Strong**

```
strStab = DensityPlot[If[n[{x, y}] > 0 ⋀ a11[{x, y}] < 0 ⋀ a22[{x, y}] < 0 ⋀ a11[{x, y}] a22[{x, y}] >
    {x, xmin, xmax}, {y, ymin, ymax}, PlotRange → All, ColorFunction → "CherryTones", PlotPoints → :

Show[strStab, preyES, preyNES, predES, predNES, coexBnd, sngPnt]
```

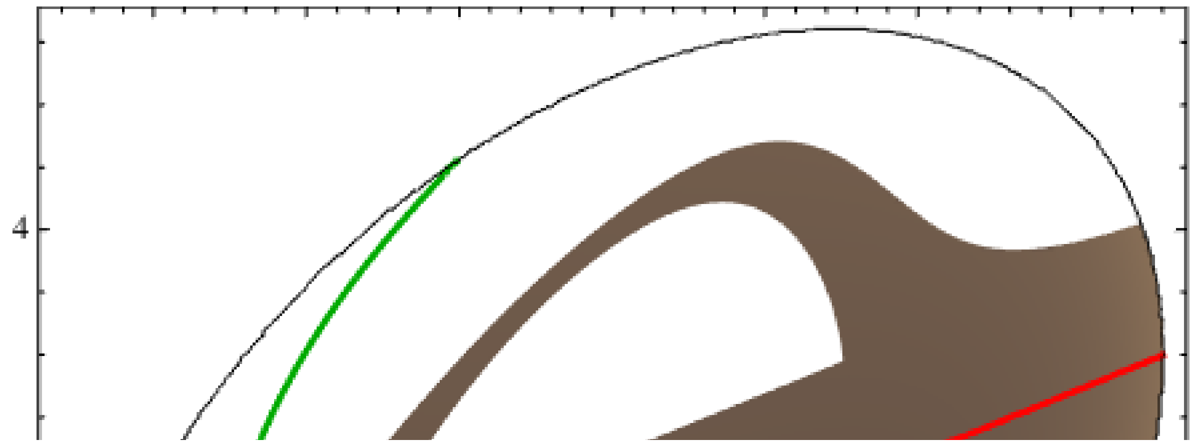


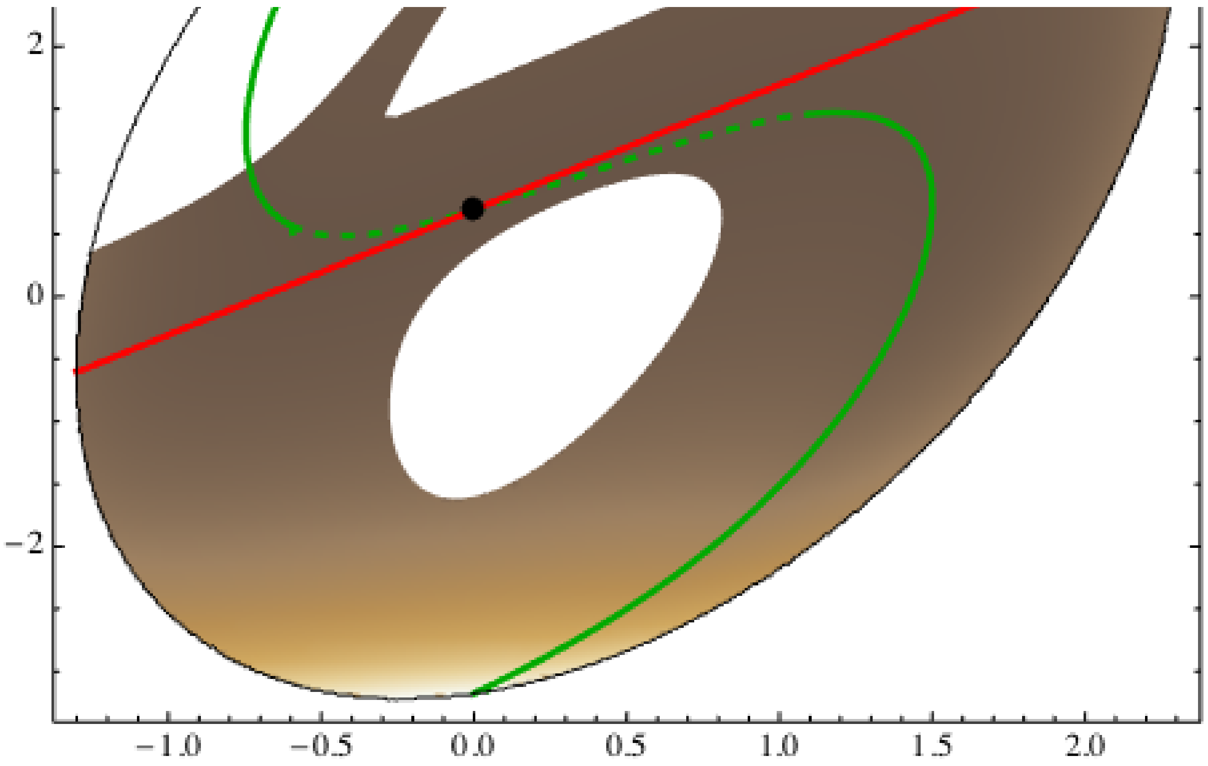



- ***Conclusion: the singular point is NOT STRONGLY STABLE.***

- **stability Weak**

```
wkStab = DensityPlot[If[n[{x, y}] > 0 ∧ (a11[{x, y}] < 0 ∨ a22[{x, y}] < 0) ∧ a11[{x, y}] a22[{x, y}]
    {x, xmin, xmax}, {y, ymin, ymax}, PlotRange → All, ColorFunction → "CoffeeTones", PlotPoints →

Show[wkStab, preyES, preyNES, predES, predNES, coexBnd, sngPnt]
```






- *Conclusion : the singular point is WEAKLY STABLE*

## Canonical equation

- **Mutation probability per birth event** ($\mu$) **and mutation variance** ($\sigma$**2**)

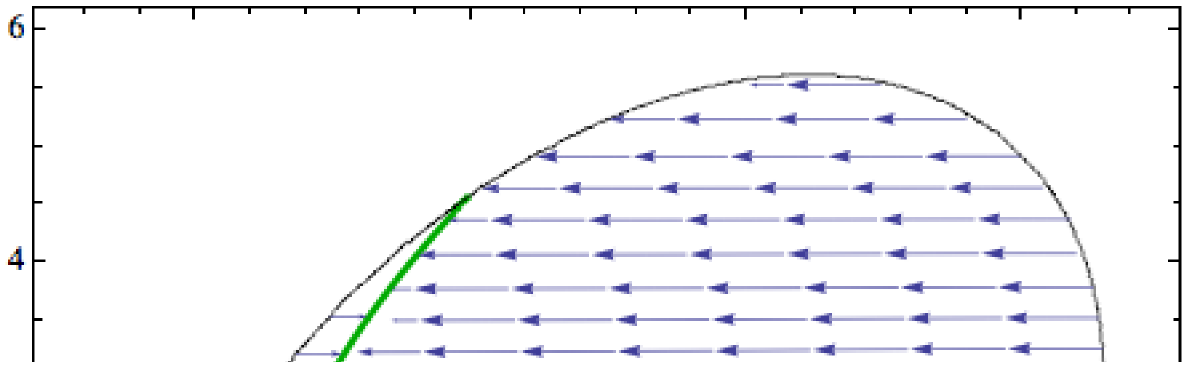
```
μPrey = 10^-6; σ2Prey = 10^-6;
μPred = 10^-6; σ2Pred = 10^-6;

μPrey = 1; σ2Prey = 1;
μPred = 1; σ2Pred = 1;
(* for the shape of the orbits only the relative values matter *)
```

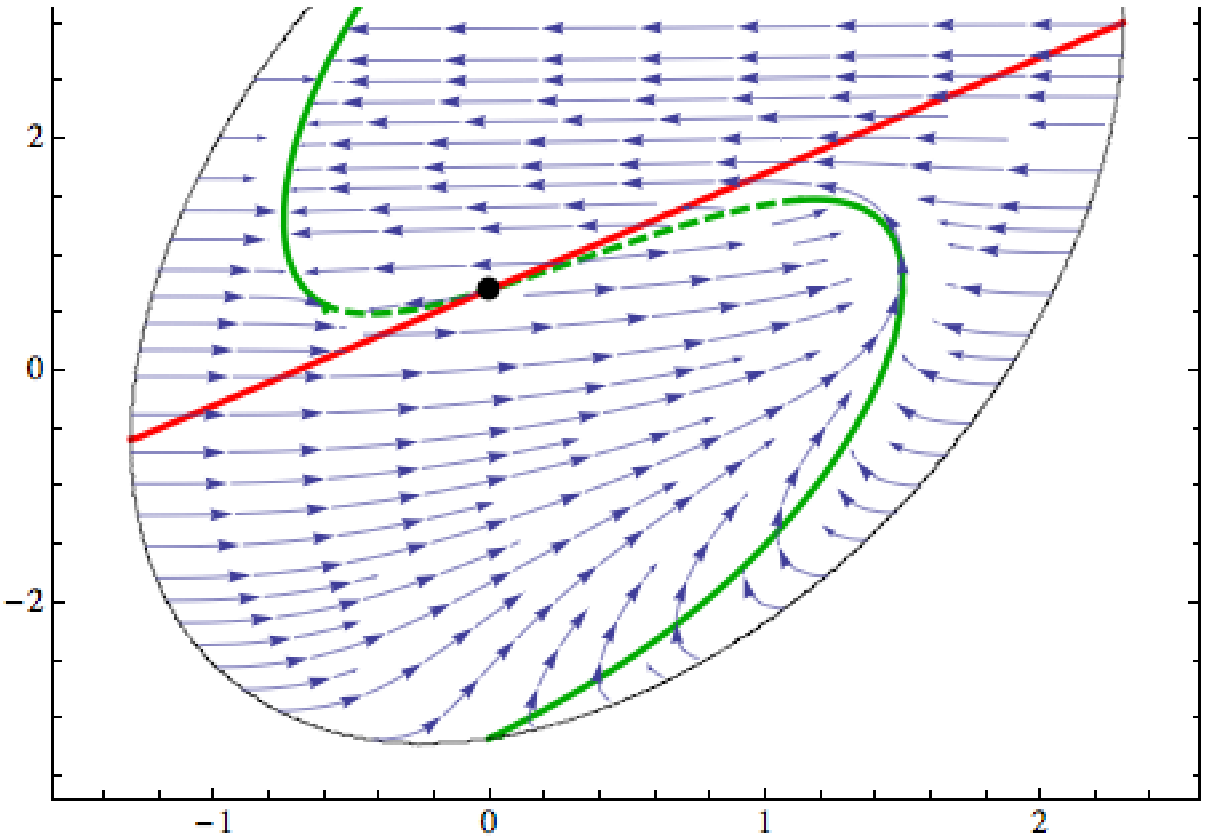
- **Deterministic drift**

```
drift_mo[{x_, y_}] := {1/2 μPrey σ2Prey m[{x, y}] dsPrey_mo[{x, y}], 1/2 μPred σ2Pred n[{x, y}] dsPred_mo[{
```

- **CE plot Stream**

```
CEstream = StreamPlot[If[n[{x, y}] > 0, drift_mo[{x, y}], {0, 0}], {x, xmin, xmax}, {y, ymin, ymax}, !

Show[preyES, preyNES, predES, predNES, coexBnd, CEstream, sngPnt]
```



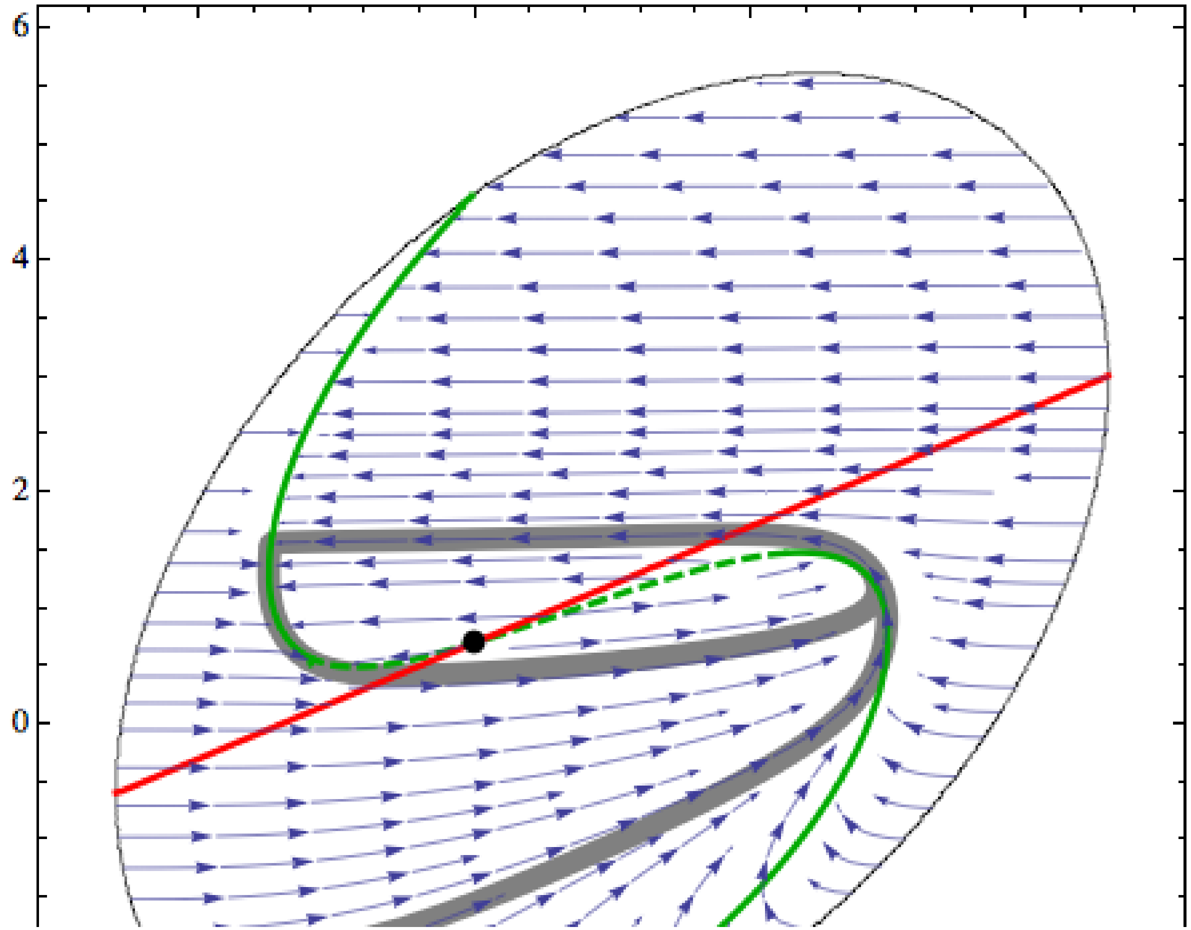


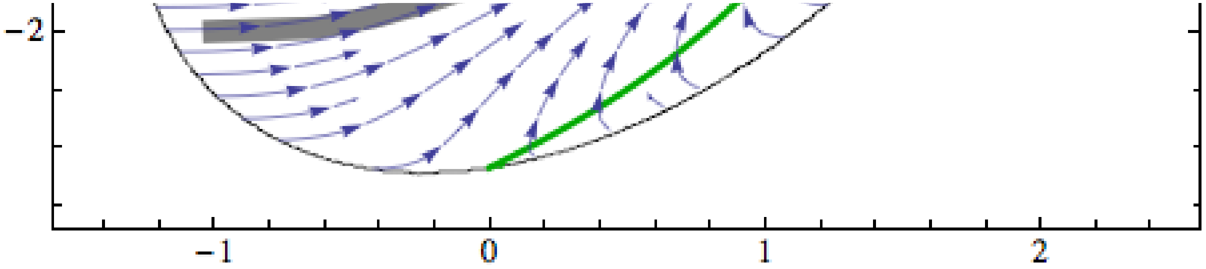

■ **Deterministic orbit (Euler method)**

```
v0 = {-1, -2}; t0 = 0; t∞ = 20 000; Δt = 2; data = {};
v = v0; t = t0;
While[t ≤ t∞ && n[v] > 0, data = Join[data, {Append[v, t]}]; v = Δt drift_mo[v] + v; t = t + Δt;];
```

- **... projected in the (x, y) - plane**

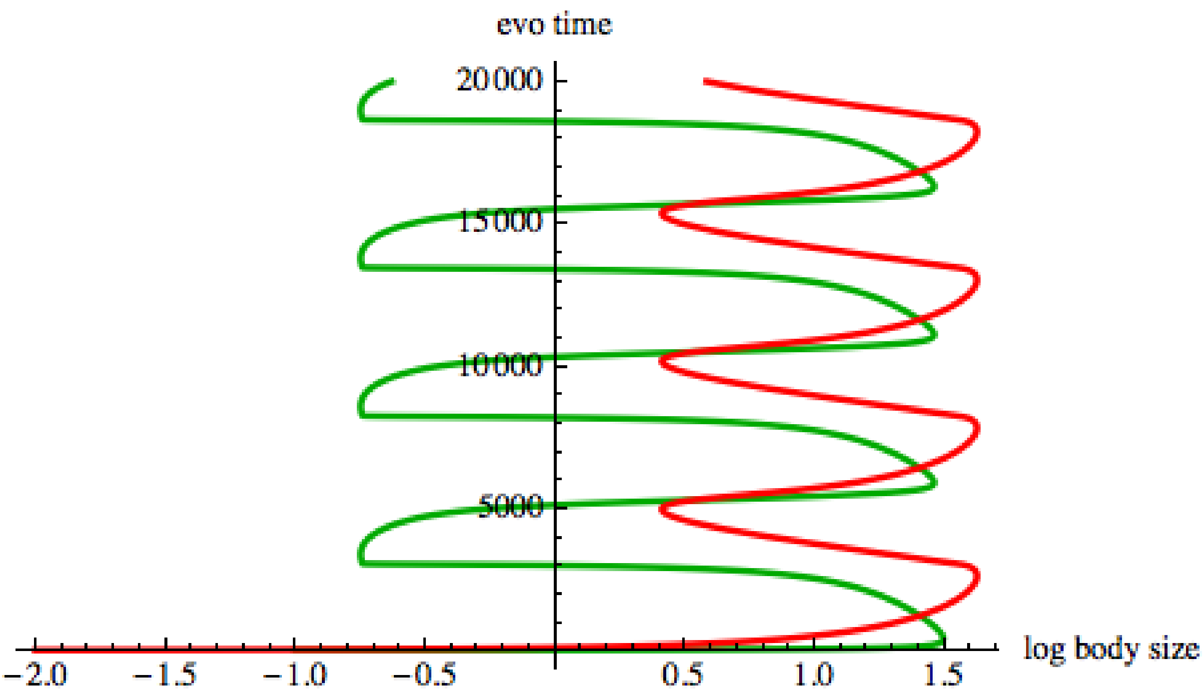
```
CEorbit = ListPlot[data[[All, {1, 2}]], PlotStyle → {Gray, Thickness[.02]}, Joined → True];

Show[coexBnd, CEorbit, preyES, preyNES, predES, predNES, CEstream, sngPnt]
```






- ... as evolutionary tree

```
xtree = ListPlot[data[[All, {1, 3}]], PlotStyle → {Darker[Green], Thick}, Joined → True];
ytree = ListPlot[data[[All, {2, 3}]], PlotStyle → {Red, Thick}, Joined → True];
Show[xtree, ytree, AxesLabel → {"log body size", "evo time"}]
```




- **Different parameter set**

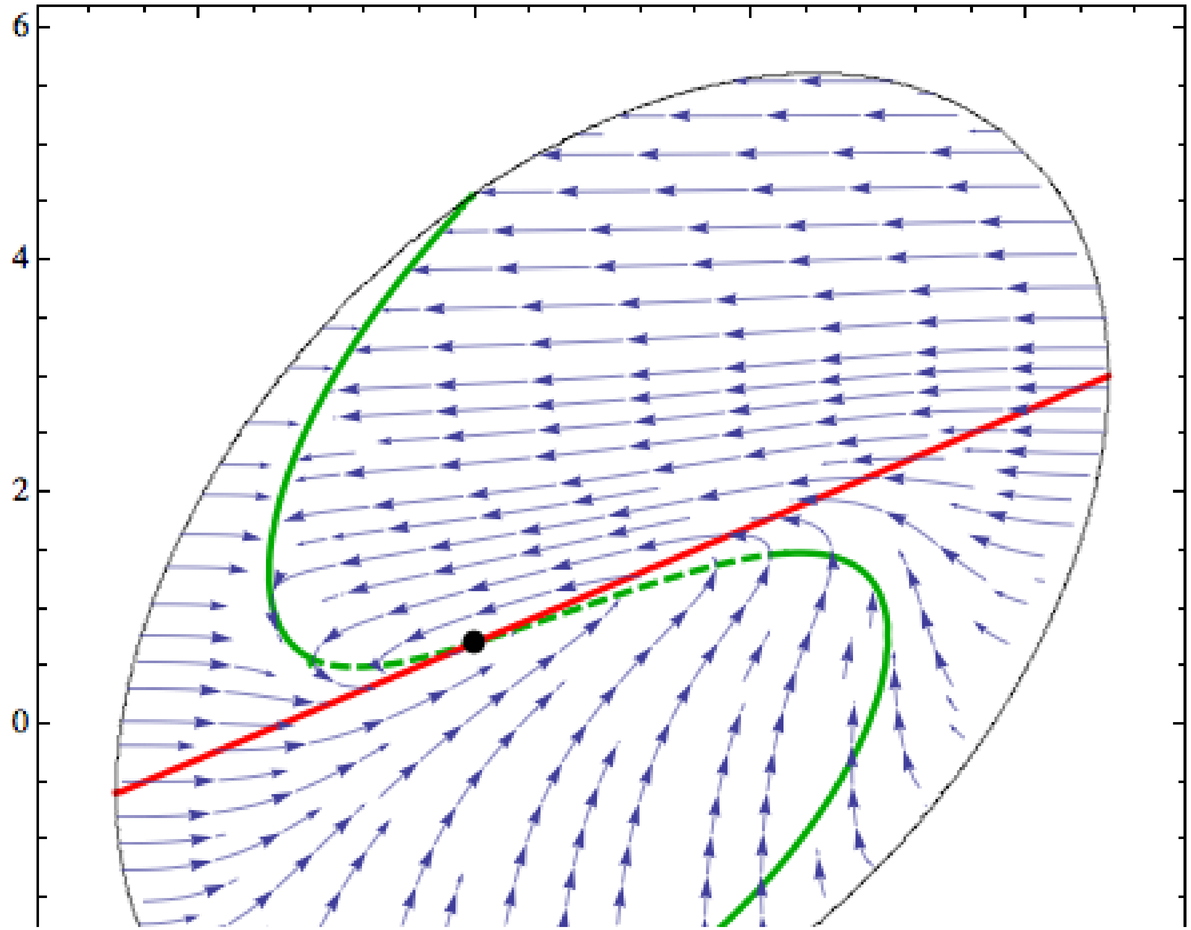
---

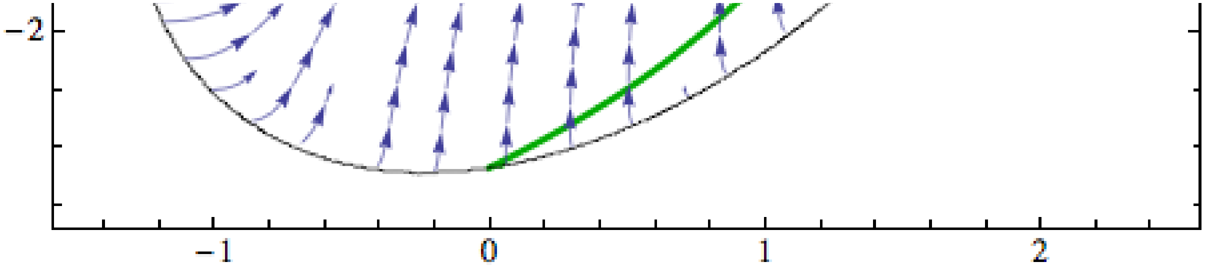
```
μPrey = 1;
σ2Prey = 0.1;
μPred = 1;
σ2Pred = 1;
```

---

## ▪ CE plot Stream

```
CEstream = StreamPlot[If[n[{x, y}] > 0, drift_mo[{x, y}], {0, 0}], {x, xmin, xmax}, {y, ymin, ymax}, 

Show[preyES, preyNES, predES, predNES, coexBnd, CEstream, sngPnt]
```



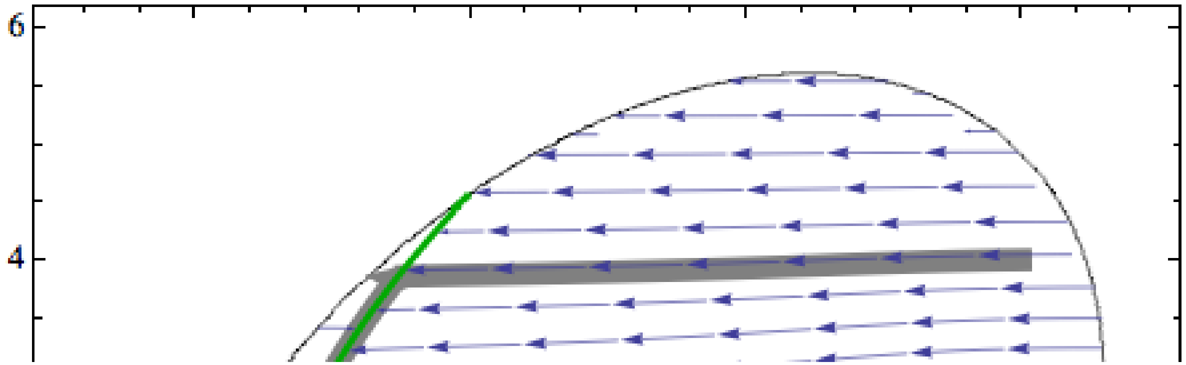


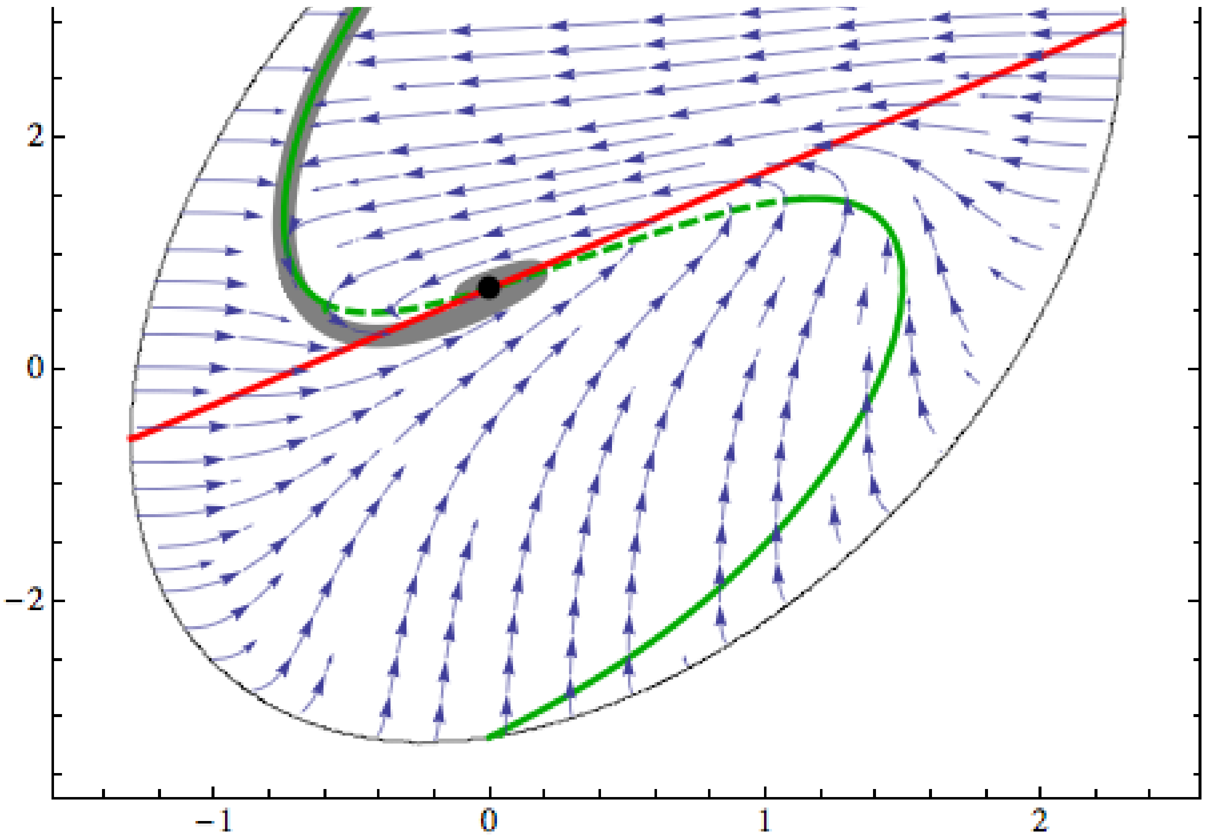

■ **Deterministic orbit (Euler method)**

```
v0 = {2, 4}; t0 = 0; t∞ = 60000; Δt = 10; data = {};
v = v0; t = t0;
While[t ≤ t∞ && n[v] > 0, data = Join[data, {Append[v, t]}]; v = Δt drift_mo[v] + v; t = t + Δt;];
```

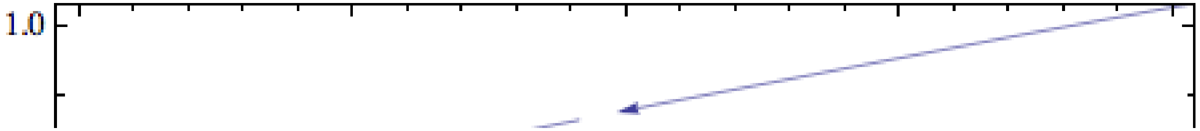
■ **... projected in the (x, y) - plane**

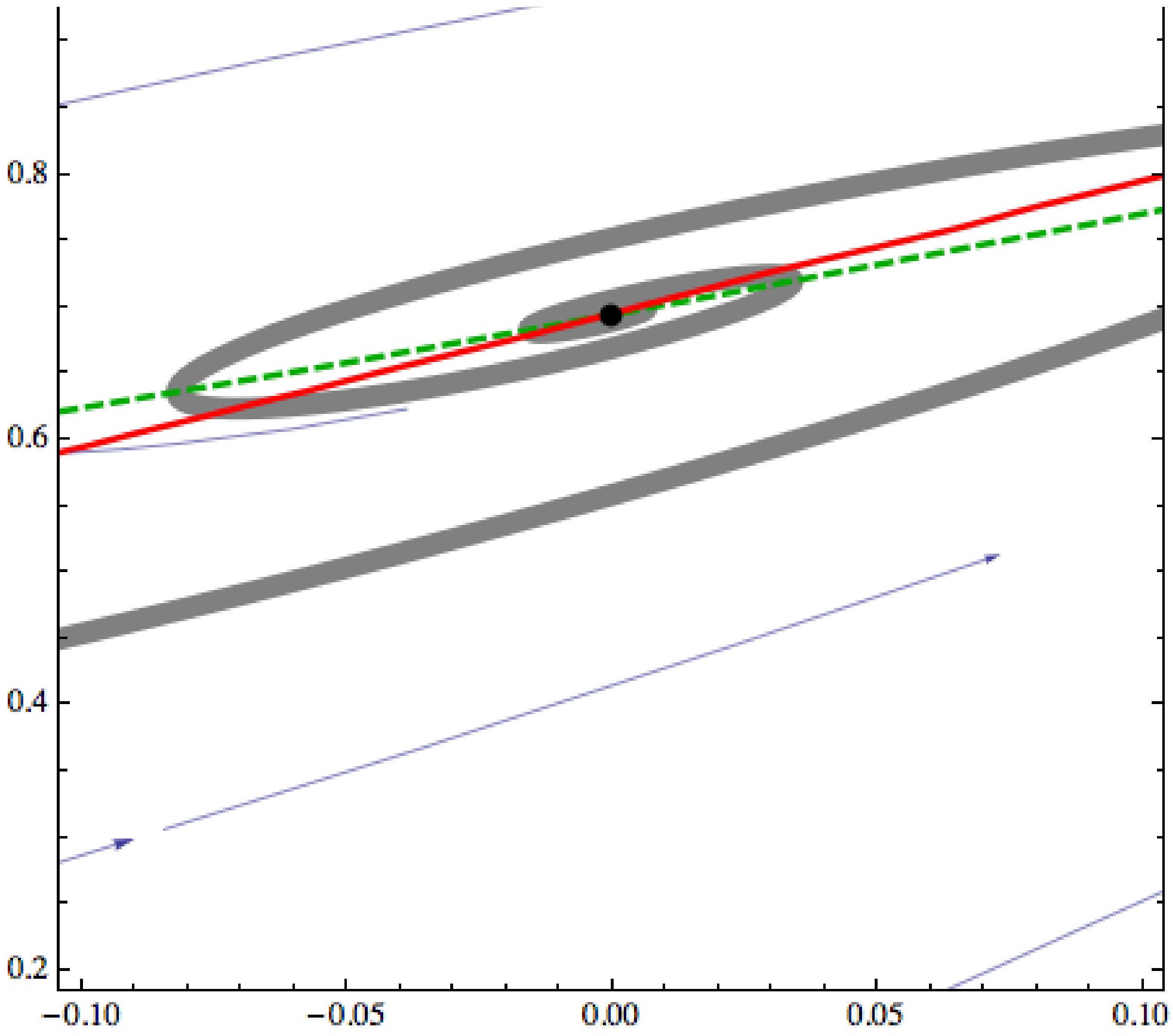
```
CEorbit = ListPlot[data[[All, {1, 2}]], PlotStyle → {Gray, Thickness[.02]}, Joined → True];

Show[coexBnd, CEorbit, preyES, preyNES, predES, predNES, CEstream, sngPnt]
```





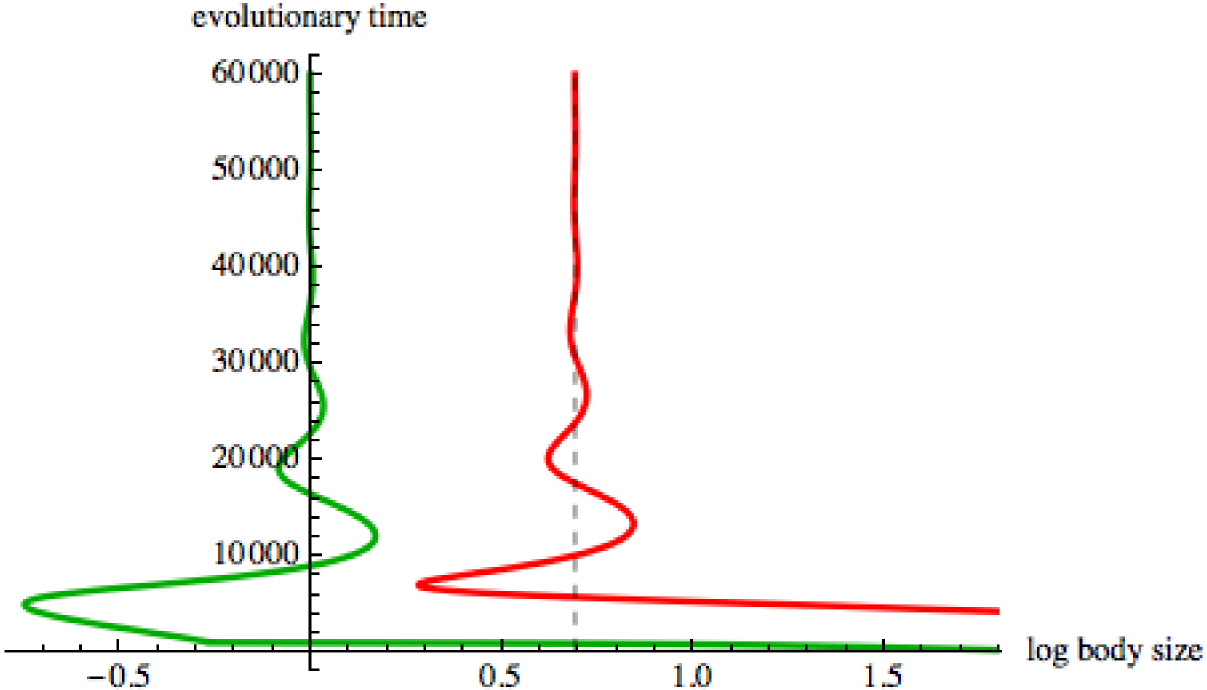

```
CEorbit = ListPlot[data⟦All, {1, 2}⟧, PlotStyle → {Gray, Thickness[.02]}, Joined → True];

Show[coexBnd, CEorbit, preyES, preyNES, predES, predNES, CEstream, sngPnt, PlotRange → {{-.1, .1}
```





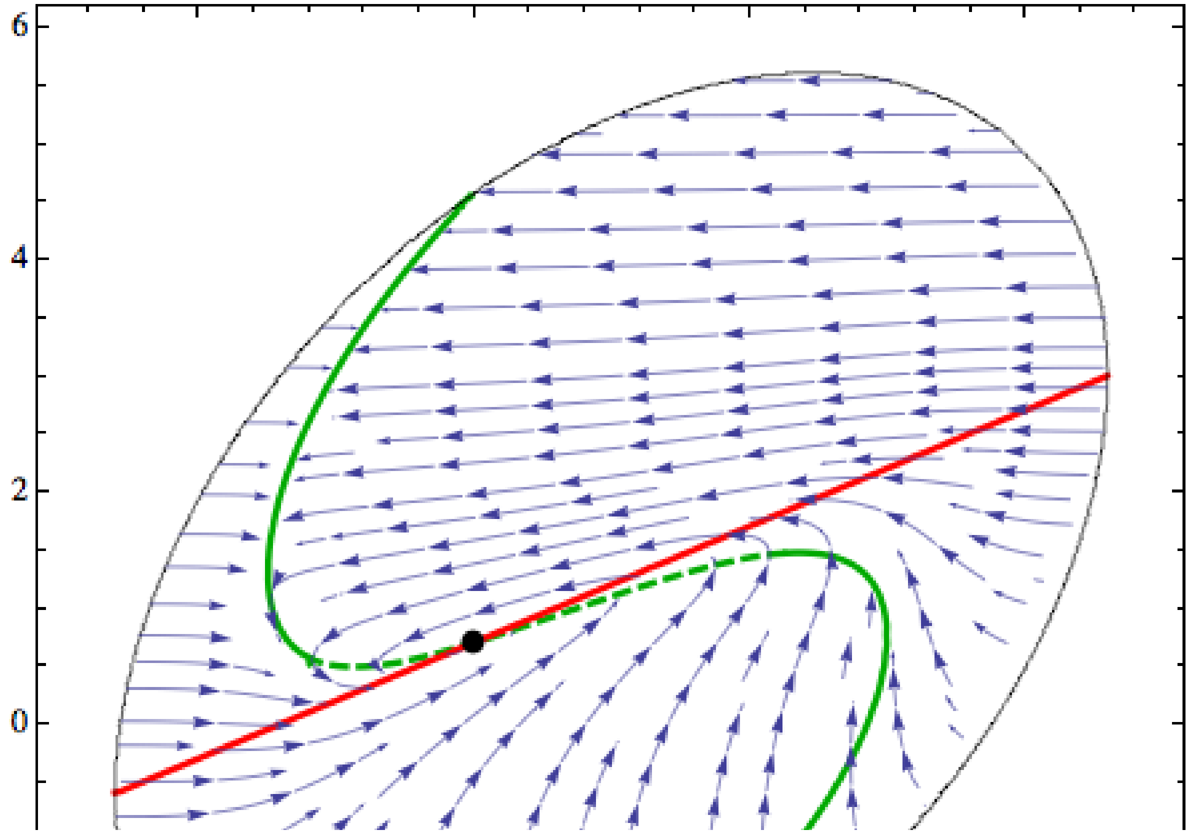

- **... as evolutionary tree**

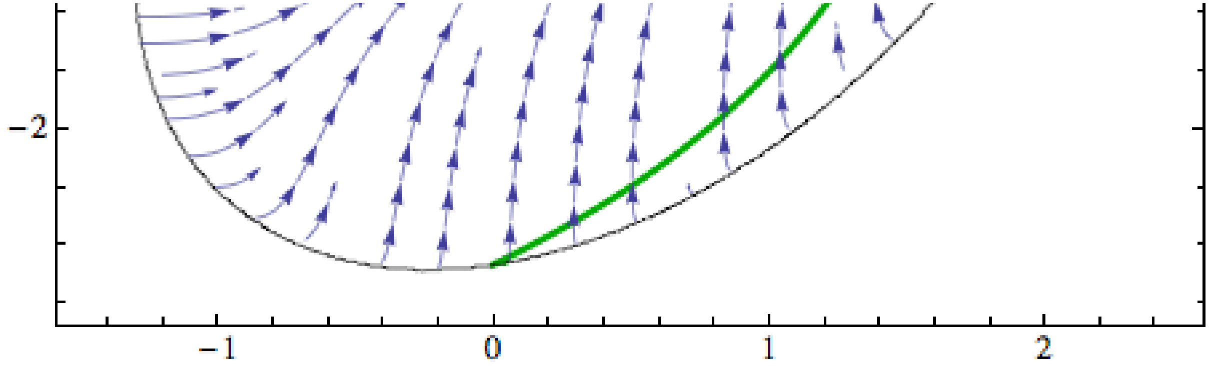
```
xtree = ListPlot[data⟦All, {1, 3}⟧, PlotStyle → {Darker[Green], Thick}, Joined → True];
ytree = ListPlot[data⟦All, {2, 3}⟧, PlotStyle → {Red, Thick}, Joined → True];
pLevel = Graphics[{Dashed, Line[{{-p, 0}, {-p, 60 000}}]}];

Show[xtree, ytree, pLevel, AxesLabel → {"log body size", "evolutionary time"}]
```



# Evolutionary branching in the prey?

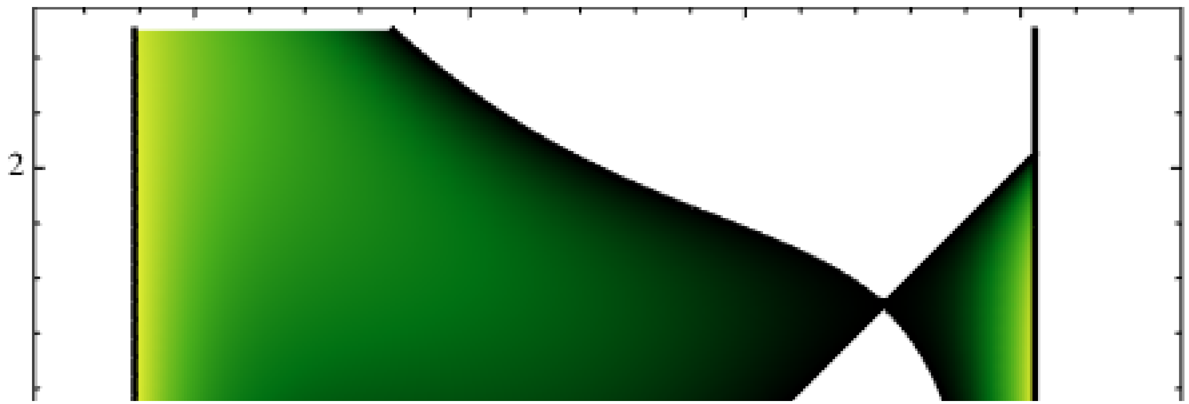
- **Continuation with last data set;**
  **Notice that the prey - isocline near the intersection is not evolutionarily stable (dash**

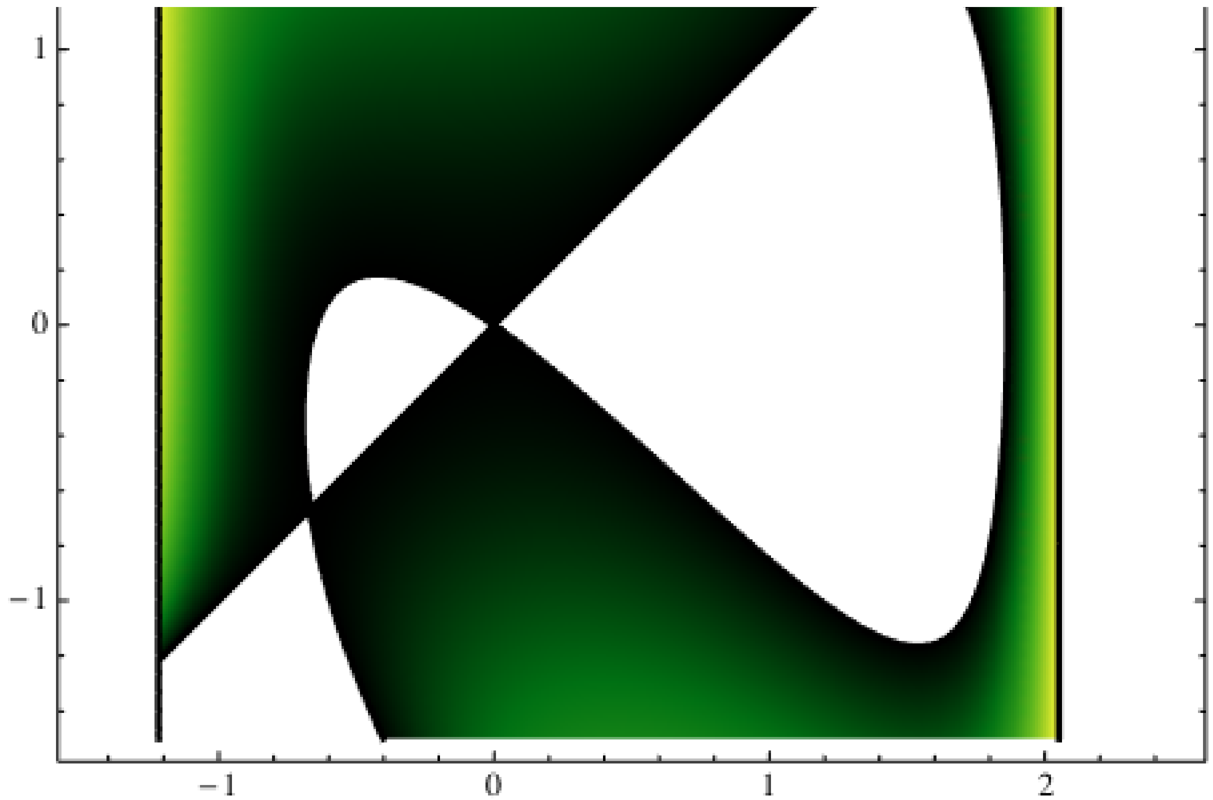





- **PIP for predator trait fixed at its singular value**

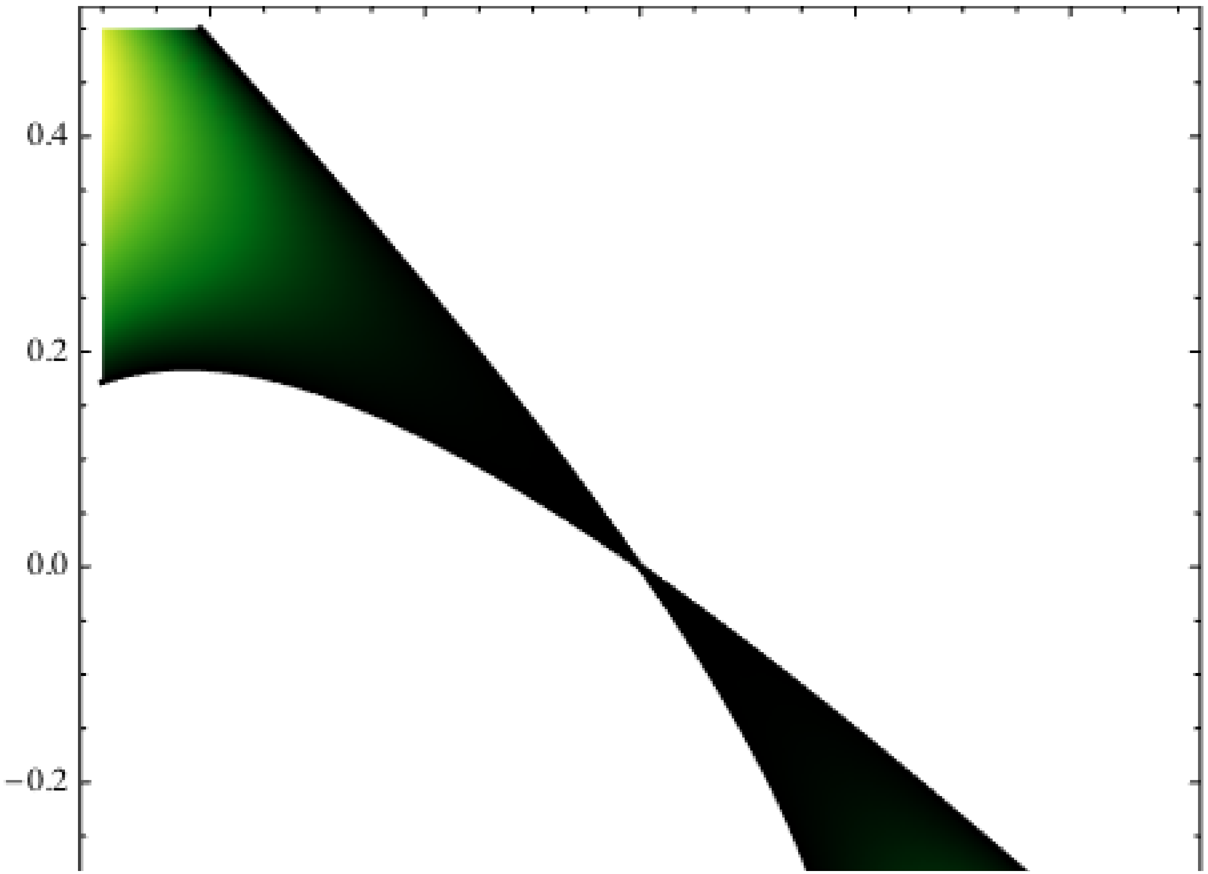
```
PIPbnd = ContourPlot[If[n[{x, -p}] > 0, sPreymo[{x, -p}, X]], {x, xmin, xmax}, {X, xmin, xmax}, Cont
    ContourShading → False, PlotPoints → 60];
nPos = ContourPlot[n[{x, -p}], {x, xmin, xmax}, {X, xmin, xmax}, Contours → {0}, ContourStyle → {Bl
PIPint = DensityPlot[If[n[{x, -p}] > 0 && sPreymo[{x, -p}, X] > 0, sPreymo[{x, -p}, X]], {x, xmin, xm
Show[PIPint, PIPbnd, nPos]
```

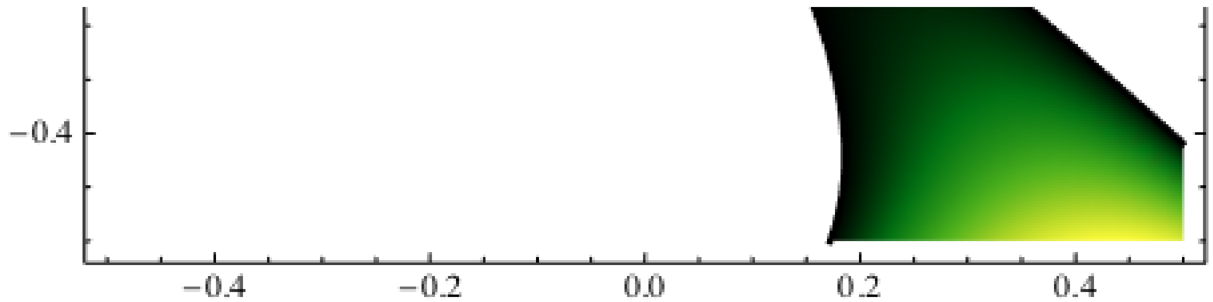





- **MIP near singular point**

```
MIPbnd = ContourPlot[If[Abs[x - X] > 0.001, sPreymo[{x, -p}, X] sPreymo[{X, -p}, x]], {x, -0.5, 0.5}
   ContourShading → False, PlotPoints → 60];
MIPint = DensityPlot[If[sPreymo[{x, -p}, X] > 0 ∧ sPreymo[{X, -p}, x] > 0, sPreymo[{x, -p}, X] sPreym
   ColorFunction → "AvocadoColors", PlotPoints → 60];
Show[MIPint, MIPbnd]
```






- Conclusion: evolutionary branching might be possible, but since the cone of mutual invadabil the dimorphic orbit might not stay inside it. This we have to investigate further.

# DIMORPHIC PREY POPULATION AND MONOMORPHIC PREDATOR POPULATION

## Reset

```
Clear[r, K, a, b, c, d];
```

## Population equations

## m1,m2 = prey pop dens; nn = pred pop dens; x1,x2 = prey log body size; y = pred log body size

```
dLogm1[m1_, m2_, nn_] := r[x1] (1 - (m1 a[x1, x1] + m2 a[x1, x2])/K[x1]) - nn b[x1, y];
dLogm2[m1_, m2_, nn_] := r[x2] (1 - (m1 a[x2, x1] + m2 a[x2, x2])/K[x2]) - nn b[x2, y];
dLognn[m1_, m2_, nn_] := m1 b[x1, y] c[x1, y] + m2 b[x2, y] c[x2, y] - d[y];
```

## Population equilibrium

```
Solve[{dLogm1[m1, m2, nn] == 0, dLogm2[m1, m2, nn] == 0, dLognn[m1, m2, nn] == 0}, {m1, m2, nn}] // Sim
```

```
{{nn ->
   (((a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] + a[x1, x2] (a[x2, x1] d[y] - b[x
      r[x1] r[x2]) / (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] 
      b[x1, y] (a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] r[x2]),
  m1 -> (-a[x1, x2] b[x2, y] d[y] K[x2] r[x1] + K[x1] (b[x2, y]^2 c[x2, y] K[x2] r[x1] + a[x2, x2] b[x1, y
    (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] K[x2] r[x1] + b[x
  m2 -> (a[x1, x1] b[x2, y] d[y] K[x2] r[x1] + b[x1, y] K[x1] (-b[x2, y] c[x1, y] K[x2] r[x1] + (-a[x2,
    (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] K[x2] r[x1] +
      b[x1, y] (a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] r[x2])}}
```

```
m1[{x1_, x2_, y_}] := (-a[x1, x2] b[x2, y] d[y] K[x2] r[x1] + K[x1] (b[x2, y]^2 c[x2, y] K[x2] r[x1] + a
    (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] K[x2] r[x1] +
      b[x1, y] (a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] r[x2]);
m2[{x1_, x2_, y_}] := (a[x1, x1] b[x2, y] d[y] K[x2] r[x1] + b[x1, y] K[x1] (-b[x2, y] c[x1, y] K[x2] r
    (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] K[x2] r[x1] +
      b[x1, y] (a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] r[x2]);
nn[{x1_, x2_, y_}] :=
  (((a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] + a[x1, x2] (a[x2, x1] d[y] - b[x
     r[x1] r[x2]) / (-a[x1, x2] b[x1, y] b[x2, y] c[x1, y] K[x2] r[x1] + a[x1, x1] b[x2, y]^2 c[x2, y] K
     b[x1, y] (a[x2, x2] b[x1, y] c[x1, y] - a[x2, x1] b[x2, y] c[x2, y]) K[x1] r[x2]);
```

## ■ Parameter values and functions

```
r[x_] := 1;
K[x_] := e^(-x^2);
a[x1_, x2_] := e^(-α (x1-x2)^2); α = 0.5;
b[x_, y_] := β1 e^(-β2 (x-y-p)^2); β1 = 100; β2 = 0.2; p = Log[0.5]; (* attack rate *)
c[x_, y_] := γ e^(x-y); γ = 0.2; (* conversion coefficient *)
d[y_] := δ1 e^(-δ2 y); δ1 = 1; δ2 = 1;
```

## ▪ Parameter set

```
μPrey = 1;
σ2Prey = 0.1;
μPred = 1;
σ2Pred = 1;
```

## ▪ Invasion fitness, gradient and curvature

### ▪ Prey

$$\text{sPrey}_{\text{di}}[\{x1\_, x2\_, y\_\}, xMut\_] := r[xMut]\left(1 - \frac{a[xMut, x1]\ m1[\{x1, x2, y\}] + a[xMut, x2]\ m2[\{x1, x2, y}{K[xMut]}\right.$$

$$\text{d1sPrey}_{\text{di}}[\{x1\_, x2\_, y\_\}] := \partial_{\text{xMut}}\text{sPrey}_{\text{di}}[\{x1, x2, y\}, \text{xMut}] /. \{\text{xMut} \to x1\};$$
$$\text{d2sPrey}_{\text{di}}[\{x1\_, x2\_, y\_\}] := \partial_{\text{xMut}}\text{sPrey}_{\text{di}}[\{x1, x2, y\}, \text{xMut}] /. \{\text{xMut} \to x2\};$$

$$\text{dd1sPrey}_{\text{di}}[\{x1\_, x2\_, y\_\}] := \partial_{\text{xMut},\text{xMut}}\text{sPrey}_{\text{di}}[\{x1, x2, y\}, \text{xMut}] /. \{\text{xMut} \to x1\};$$
$$\text{dd2sPrey}_{\text{di}}[\{x1\_, x2\_, y\_\}] := \partial_{\text{xMut},\text{xMut}}\text{sPrey}_{\text{di}}[\{x1, x2, y\}, \text{xMut}] /. \{\text{xMut} \to x2\};$$

- **Predator invasion fitness and its derivatives:**

```
sPred_di[{x1_, x2_, y_}, yMut_] := b[x1, yMut] c[x1, yMut] m1[{x1, x2, y}] + b[x2, yMut] c[x2, yMut] m2

dsPred_di[{x1_, x2_, y_}] := ∂_yMut sPred_di[{x1, x2, y}, yMut] /. {yMut → y};

ddsPred_di[{x1_, x2_, y_}] := ∂_yMut,yMut sPred_di[{x1, x2, y}, yMut] /. {yMut → y};
```

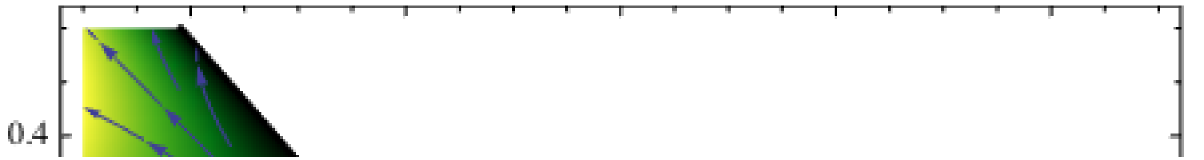
- **Canonical equation**

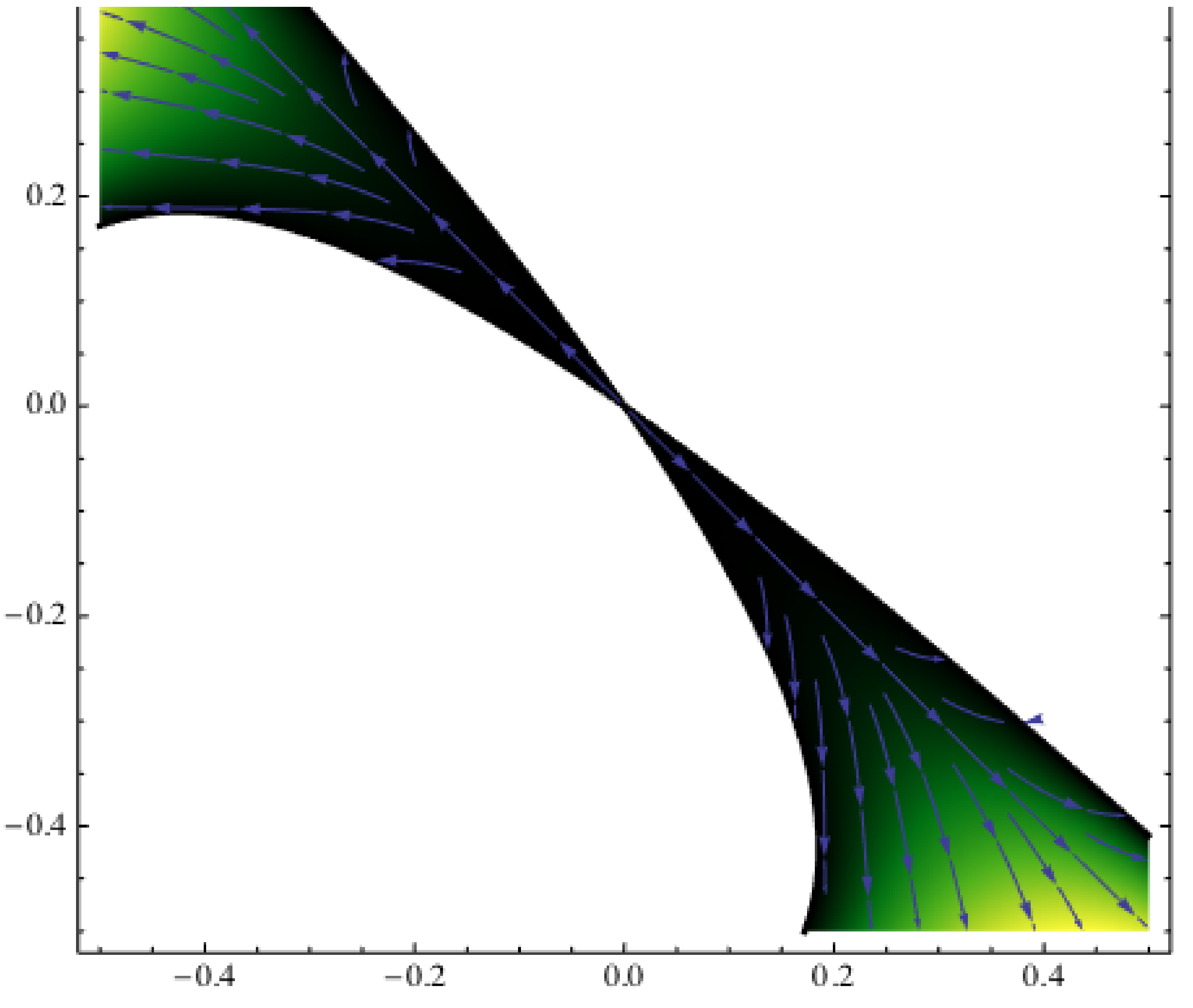
- **Deterministic drift**

```
drift_di[{x1_, x2_, y_}] := {1/2 μPrey σ2Prey m1[{x1, x2, y}] d1sPrey_di[{x1, x2, y}], 1/2 μPrey σ2Prey m2
   1/2 μPred σ2Pred nn[{x1, x2, y}] dsPred_di[{x1, x2, y}]};
```

- **Stream plot (note below that the boundaries of the mutual invadability cone are attracting, ar**

```
CEstream = StreamPlot[If[m1[{x1, x2, -p}] > 0 ∧ m2[{x1, x2, -p}] > 0 ∧ Abs[x1 - x2] > 0.01, drift_di[{

Show[MIPint, MIPbnd, CEstream]
```



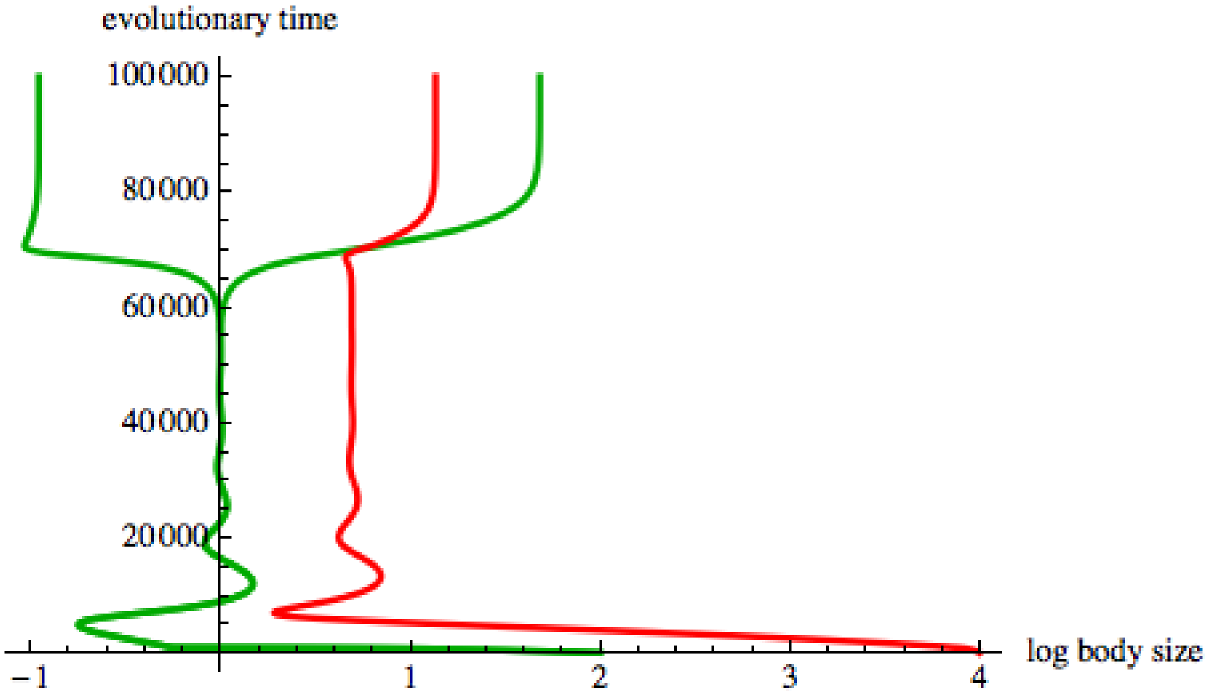



- **Deterministic evolutionary tree**

```
(* monomorphic prey *)
v0 = {2, 4}; t0 = 0; t∞ = 100 000; Δt = 10;
x1data = {}; x2data = {}; ydata = {}; x1stab = {}; x2stab = {}; ystab = {};
v = v0; t = t0;
While[t ≤ t∞ ⋀ n[v] > 0 ⋀ Abs[v⟦1⟧] > 10^-6 ⋀ Abs[v⟦2⟧ + p] > 10^-6,
  x1data = Join[x1data, {{v⟦1⟧, t}}];
  x2data = Join[x2data, {{v⟦1⟧, t}}];
  ydata = Join[ydata, {{v⟦2⟧, t}}];
  x1stab = Join[x1stab, {{ddsPrey_mo[v], t}}];
  x2stab = Join[x2stab, {{ddsPrey_mo[v], t}}];
  ystab = Join[ystab, {{ddsPred_mo[v], t}}];
  v = v + Δt drift_mo[v];
  t = t + Δt];

(* dimorphic prey *)
v = {v⟦1⟧ - 0.01 Sqrt[σ2Prey], v⟦1⟧ + 0.01 Sqrt[σ2Prey], v⟦2⟧};

While[t ≤ t∞ && m1[v] > 0 && m2[v] > 0 && nn[v] > 0,
  x1data = Join[x1data, {{v⟦1⟧, t}}];
  x2data = Join[x2data, {{v⟦2⟧, t}}];
  ydata = Join[ydata, {{v⟦3⟧, t}}];
  x1stab = Join[x1stab, {{dd1sPrey_di[v], t}}];
  x2stab = Join[x2stab, {{dd2sPrey_di[v], t}}];
  ystab = Join[ystab, {{ddsPred_di[v], t}}];
  v = Δt drift_di[v] + v;
```

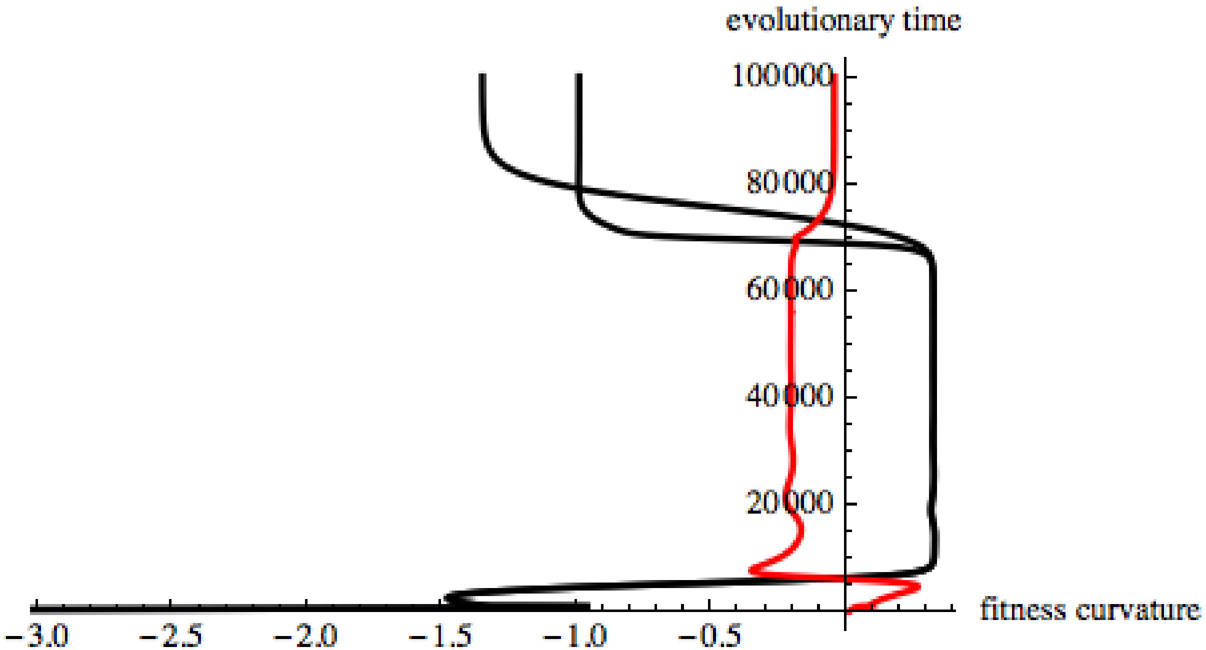
- **Trait values**

```
x1tree = ListPlot[x1data, PlotStyle → {Darker[Green], Thick}, Joined → True];
x2tree = ListPlot[x2data, PlotStyle → {Darker[Green], Thick}, Joined → True];
ytree = ListPlot[ydata, PlotStyle → {Red, Thick}, Joined → True];
Show[x1tree, x2tree, ytree, AxesLabel → {"log body size", "evolutionary time"}]
```



- **Uninvadability test**

```
x1stree = ListPlot[x1stab, PlotStyle → {Black, Thick}, Joined → True];
x2stree = ListPlot[x2stab, PlotStyle → {Black, Thick}, Joined → True];
ystree = ListPlot[ystab, PlotStyle → {Red, Thick}, Joined → True];

Show[x1stree, x2stree, ystree, AxesLabel → {"fitness curvature", "evolutionary time"}]
```




- ***Conclusion: after branching in the prey species, the population evoves to an evolutionarily mutual invadability cone is not forward invariant (see above) stochastic orbits may readily mutation step sizes, however, there is a positive probability of not leaving the cone, an sc branching the population will in the neighborhood of the singular point, branching will occu***

***branching should be exponentially distributed and may be very long. This phenomenon has beer prey + 1 pred) reached in the end is uninvadable, because the curvature of the fitness lands above).***

## Stability of trimorphism

### Singular point

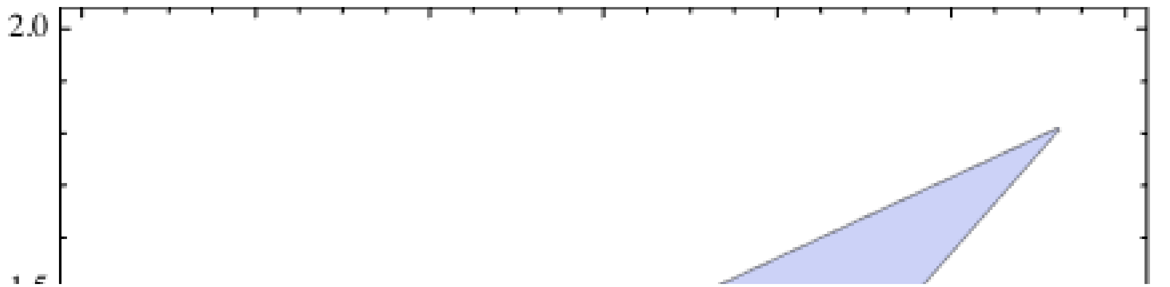
```
x1Sing = x1data⟦-1, 1⟧;
x2Sing = x2data⟦-1, 1⟧;
ySing = ydata⟦-1, 1⟧;
```

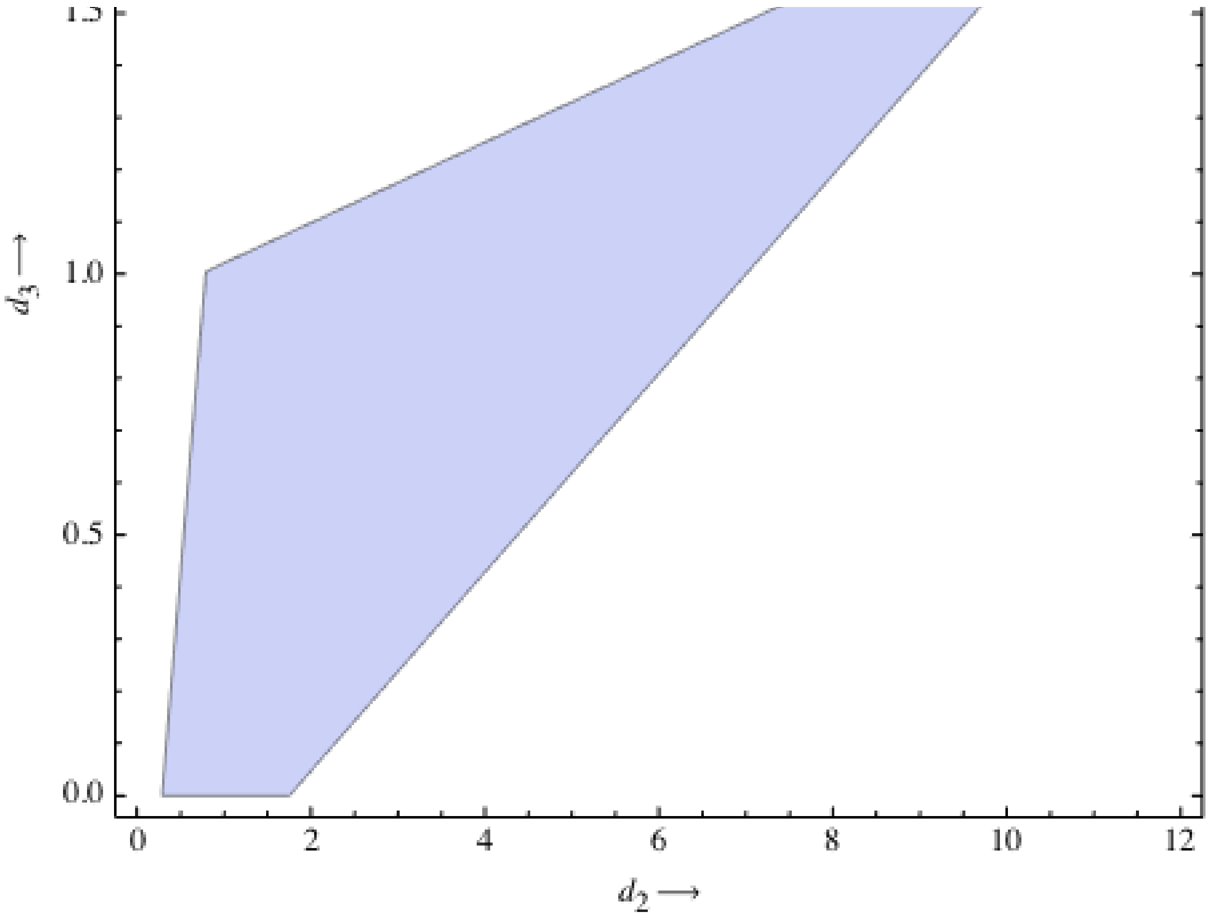
### Jacobi matrix {{a11, a12, a13}, {a21, a22, a23}, {a31, a32, a33}}

```
A = D[{d1sPrey_di[{x1, x2, y}], d2sPrey_di[{x1, x2, y}], dsPred_di[{x1, x2, y}]}, {{x1, x2, y}}] /. {x1
```

### Total stability test

```
d1 = 1;(* just a matter of scaling *)

RegionPlot[ d1 A⟦1, 1⟧ + d2 A⟦1, 2⟧ + d3 A⟦1, 3⟧ < 0 ⋀ d1 A⟦2, 1⟧ + d2 A⟦2, 2⟧ + d3 A⟦2, 3⟧ < 0 ⋀ d1 A⟦3, 1⟧
 PlotPoints → 100, FrameLabel → {"d2⟶", "d3⟶"}]
```






▫ ***Conclusion: The trimorphic singular point is totally stable (and hence also strongly and wea***