Appendix 2

This document printed 21 June 2022. Made using Mathematica v.12.3

2.1 Starting Assumptions

In[3]:=
```
$Assumptions = (
  disp > 0 &&    (*Dispersal rate*)
   w > 0 &&      (*EPC width*)
   r0 > 0 &&     (*Maximum growth rate (at optima)*)
   v > 0 &&      (*speed of climate change*)
   Element[y, Reals] &&    (*Spatial dimension, moving reference frame*)
   Element[a, Reals] &&    (*Asymetry of Morse EPC*)
   frontWidth > 0 &&    (*Property of population density shape*)
   sigma > 0    (*standard deviation of high-frequency envrionmental variation*)
 )
```

Out[3]= $\text{disp} > 0 \&\& w > 0 \&\& r0 > 0 \&\& v > 0 \&\& y \in \mathbb{R} \&\& a \in \mathbb{R} \&\& \text{frontWidth} > 0 \&\& \text{sigma} > 0$

In[4]:=
```
(* Various rules to simplify expressions later *)
dispRule = Solve[frontWidth == Sqrt[disp]/Sqrt[r0], disp] // Last
frontWidthRule = Solve[frontWidth == Sqrt[disp]/Sqrt[r0], frontWidth] // Last
r0Rule = Solve[frontWidth == Sqrt[disp]/Sqrt[r0], r0] // Last
```

Out[4]= $\{\text{disp} \to \text{frontWidth}^2\, r0\}$

Out[5]= $\left\{\text{frontWidth} \to \sqrt{\frac{\text{disp}}{r0}}\right\}$

Out[6]= $\left\{r0 \to \frac{\text{disp}}{\text{frontWidth}^2}\right\}$

2.2 Defining Core Functions

Start by writing down Schrodinger equation, Where g() is the EPC, f() is the wave function (\psi,) and lam is the overall growth rate (\lambda)

In[7]:=
```
eq = -f[y] * lam + f[y] × g[y] + v f'[y] + disp f''[y] == 0
eq // TraditionalForm
```

Out[7]=
$$-\text{lam}\, f[y] + f[y] \times g[y] + v\, f'[y] + \text{disp}\, f''[y] == 0$$

Out[8]//TraditionalForm=
$$\text{disp}\, f''(y) + v\, f'(y) + f(y)\, g(y) - \text{lam}\, f(y) = 0$$

2.2.1 Morse (Asymmetric) EPC

In[9]:=
```
(* Define Morse Potential Function *)
gMorseRule = {g → Function [y, r0 (1 - 1 / (a^2 w^2) × (1 - Exp[a y]) ^2) ]}
(*  Solve  differential equation (Schodinger with Morse as g) for f in terms of y
 NB is very complex, so not showing here! *)
morseSol = DSolve[eq /. gMorseRule, f, y] // Last;
```

Out[9]=
$$\left\{g \to \text{Function}\left[y, r0 \left(1 - \frac{(1 - \text{Exp}[a\, y])^2}{a^2\, w^2}\right)\right]\right\}$$

In[11]:=
```
(* Add additional condition for localisation of climatic niche
 (otherwise don't get positive growth anywhere and it blows up)*)
```

In[12]:=
```
Reduce[$Assumptions && 1 / (a^2 w^2) > 1, a]
(* Add additional constraints that
 prevent blow ups in corner cases in the solution*)
morseCond = 0 == (Cases[f[y] /. morseSol, LaguerreL[n_, _, _] → n, Infinity, 1] // First)
(*  Solve the condition in terms of lambda  *)
```

Out[12]=
$$y \in \mathbb{R} \&\& w > 0 \&\& v > 0 \&\& \text{sigma} > 0 \&\& r0 > 0 \&\& \text{frontWidth} > 0 \&\& \text{disp} > 0 \&\& \left(-\frac{1}{w} < a < 0 \,||\, 0 < a < \frac{1}{w}\right)$$

Out[13]=
$$0 == \frac{2\sqrt{\text{disp}\, r0} - a^2\, \text{disp}\, w - \sqrt{4\, \text{disp}\, r0 + 4\, a^2\, \text{disp}\, \text{lam}\, w^2 - 4\, a^2\, \text{disp}\, r0\, w^2 + a^2\, v^2\, w^2}}{2\, a^2\, \text{disp}\, w}$$

2.2.2 Harmonic EPC

In[14]:=
```
(* Harmonic EPC for baseline calculations of shift*)
(*Follows same procedures*)
gHarmRule = {g → Function [y, r0 (1 - y^2 / (w^2))]} (* Harmonic Potential *)
harmSol = DSolve[eq /. gHarmRule, f, y] // Last;
harmCond = 0 == (Cases[f[y] /. harmSol, HermiteH[n_, _] → n, Infinity, 1] // First);
harmSigRule = Solve[harmCond, lam] // Last // FullSimplify;
harmSolClean = {f → Function @@
      {y, f[y] /. harmSol /. harmSigRule /. dispRule /. c1 → 1 /. c2 → 0 // FullSimplify}};
```

Out[14]=
$$\left\{g \to \text{Function}\left[y, r0 \left(1 - \frac{y^2}{w^2}\right)\right]\right\}$$

2.3 Response 1: Population Invasion Fitness & Critical Speed of Climate Change

2.3 Response 1: Population Invasion Fitness & Critical Speed of Climate Change

In[19]:=
```
morseSigRule = Solve[morseCond, lam] // Last // FullSimplify ;
Style[%, Background → LightBlue]
```

Out[20]=
$$\left\{\text{lam} \to \frac{a^2\,\text{disp}}{4} + r0 - \frac{v^2}{4\,\text{disp}} - \frac{\sqrt{\text{disp}\,r0}}{w} \;\text{if}\; \text{disp} < \frac{4\,r0}{a^4\,w^2}\right\}$$

In[21]:=
```
(*under the assumption that: *)
lam / r0 /. morseSigRule /. dispRule // FullSimplify // Expand
```

Out[21]=
$$1 + \frac{a^2\,\text{frontWidth}^2}{4} - \frac{v^2}{4\,\text{frontWidth}^2\,r0^2} - \frac{\text{frontWidth}}{w} \;\text{if}\; a^4\,\text{frontWidth}\,w < 2\,a^2$$

In[22]:=
```
(* Create a neater solution for future use that builds in these assumptions*)
morseSolClean = {f → Function @@ {y,
      Assuming[-2 + a^2 frontWidth w < 0,

       f[y] /. morseSol /. morseSigRule /. dispRule /. c1 → 0 /. c2 → 1 // FullSimplify]
     }}
```

Out[22]=
$$\left\{f \to \text{Function}\left[y, e^{-\frac{2\,e^{a\,y}\,\text{frontWidth}\,r0 + a\,\left(-2\,\text{frontWidth}\,r0 + a\,\left(a\,\text{frontWidth}^2\,r0 + v\right)\,w\right)\,y}{2\,a^2\,\text{frontWidth}^2\,r0\,w}}\right]\right\}$$

2.3.1 Critical Speed of Climate Change

In[23]:=
```
(*Finding  critical climate change speed: (largest permissible v value) *)
vBlowRule =
  Assuming[a < 0, Solve[a (a frontWidth^2 r0 + v) w == 2 frontWidth r0, v]] /. morseSigRule /.
    frontWidthRule // Last // FullSimplify;
Style[vBlowRule, Background → LightBlue]

(*This needs to have lambda_0 slotted into it,
solved from above, so it matches expression in main text *)
```

Out[24]=
$$\left\{v \to -a\,\text{disp} + \frac{2\,\sqrt{\text{disp}\,r0}}{a\,w} \;\text{if}\; \sqrt{\frac{\text{disp}}{r0}} > \frac{2}{a^2\,w}\right\}$$

2.3.2 Impact of Weather (high frequency variation in growth rate)

In[25]:=
```
(* Convolution of Morse performance curve with Gaussian weather
 variability gives another, modified Morse performance curve: *)
gMorseWeather =
 Integrate[(g[x] /. gMorseRule) * PDF[NormalDistribution[0, sigma]][y - x],
  {x, -Infinity, Infinity}, GenerateConditions → False]
```

Out[25]=
$$\frac{r0\left(-1 - e^{2\,a\,\left(a\,\text{sigma}^2 + y\right)} + 2\,e^{\frac{a^2\,\text{sigma}^2}{2} + a\,y} + a^2\,w^2\right)}{a^2\,w^2}$$

2.3.3 How effective values of key parameters are impacted by weather

In[26]:=
```
(* Because doing it directly takes too long,
so we break it up into a series expansion *)
gWSol1 =
  Solve[And @@ Table[SeriesCoefficient[
         Series[(g[y] /. gMorseRule /. {r0 → r02, w → w2, y → y - y2}) - gMorseWeather,
                                                    {y, 0, 2}],
                                              n] == 0,
                        {n, 1, 1}],
         {r02}] // FullSimplify // Last;

gWSol2 = Assuming[Element[y2, Reals],
                  Solve[And @@ Table[SeriesCoefficient[
           Series[(g[y] /. gMorseRule /. {r0 → r02, w → w2, y → y - y2}) - gMorseWeather,
                                                          {y, 0, 2}],
                                        n] == 0,
                                        {n, 2, 2}] /. gWSol1, {y2}]] // Last;

gWSol3 = Assuming[Element[y2, Reals] && w2 > 0 && a > 0,
      Solve[And @@ Table[SeriesCoefficient[Series[
                 (g[y] /. gMorseRule /. {r0 → r02, w → w2, y → y - y2}) - gMorseWeather,
                                                    {y, 0, 2}],
                          n] == 0,
                          {n, 0, 0}] /. gWSol1 /. gWSol2 /. w2 → Sqrt[w4] // FullSimplify,
            {w4}]] // Last // FullSimplify;
```

In[29]:=
```
morseWeatherRule = {r0 → r02,
        frontWidth → Sqrt[r0 / r02] * frontWidth,
        y → y - y2,
        w → Abs[w2]
       } /. gWSol1 /. gWSol2 /. w2 → Sqrt[w4] /. gWSol3 // FullSimplify
```

Out[29]=
$$\left\{r0 \to \frac{r0\left(-1+e^{-a^2\,sigma^2}+a^2\,w^2\right)}{a^2\,w^2},\right.$$
$$frontWidth \to e^{\frac{a^2\,sigma^2}{2}}\,frontWidth\,w\sqrt{\frac{1}{1+e^{a^2\,sigma^2}\left(-1+a^2\,w^2\right)}}\,Abs[a],$$
$$\left.y \to \frac{3\,a\,sigma^2}{2}+y,\ w \to \sqrt{Abs\left[\frac{1+e^{a^2\,sigma^2}\left(-1+a^2\,w^2\right)}{a^2}\right]}\right\}$$

2.3.4 How weather affects population fitness:

In[30]:=
```
(* Find series of the weather rule in terms of sigma up to 4th order terms  *)
Series[r0 /. morseWeatherRule, {sigma, 0, 4}]
```

Out[30]=
$$r0-\frac{r0\,sigma^2}{w^2}+\frac{a^2\,r0\,sigma^4}{2\,w^2}+O[sigma]^5$$

In[31]:=
```
(* Check consistency of calculation: *)
morseWeatherRule /. sigma → 0 // FullSimplify
```

Out[31]=
$$\{r0 \to r0,\ frontWidth \to frontWidth,\ y \to y,\ w \to w\}$$

In[32]:=
```
(*Use to define a new overall expression for growth rate that includes sigma*)
(*For simplicity need to include variety of assumptions*)
morseSigWeatherRule = Assuming[a ≠ 0,
  (morseSigRule /. disp < 4 r0 / (a^4 w^2) → True /. morseWeatherRule // FullSimplify) /.
  Sign[1 + ℯ^(a^2 sigma^2) (-1 + a^2 w^2)] → 1] // FullSimplify
```

Out[32]=
$$\left\{lam \to \frac{a^2\, disp}{4} + r0 - \frac{v^2}{4\, disp} - \sqrt{\frac{disp\, \mathrm{e}^{-a^2\, sigma^2}\, r0}{w^2} + \frac{\left(-1 + \mathrm{e}^{-a^2\, sigma^2}\right) r0}{a^2\, w^2}}\right\}$$

In[33]:=
```
(* Does weather increase or decrease lam? *)
weatherDirRule =
  D[(lam /. morseSigWeatherRule), {sigma, 2}] /. sigma → 0 // FullSimplify;
Style[weatherDirRule, Background → LightBlue]
```

Out[34]=
$$\frac{-2\, r0 + a^2 \sqrt{disp\, r0}\, w}{w^2}$$

In[35]:=
```
(* Taking a series expansion in terms of a, (two second order),
it is clear that the sign of 'a' doesn't matter*)
Series[(-1 + ℯ^(-a^2 sigma^2)) r0 / (a^2 w^2), {a, 0, 2}]
```

Out[35]=
$$-\frac{r0\, sigma^2}{w^2} + \frac{r0\, sigma^4\, a^2}{2\, w^2} + O[a]^3$$

In[36]:=
```
(*As 'a' goes to zero, this becomes very simple*)
Limit[(-1 + ℯ^(-a^2 sigma^2)) r0 / (a^2 w^2), a → 0]
```

Limit: Warning: Assumptions that involve the limit variable are ignored.

Out[36]=
$$-\frac{r0\, sigma^2}{w^2}$$

2.4 Response 2 - Lags

We analyze two measures of lag behind climate velocity : 1) the location of the peak density, and 2) the center of mass of the population.

2.4.1: Friction Coefficient

In[37]:=
```
(*Solve for location of peak of population
 density in terms of y (the moving window) *)
(*Harmoinc EPC*)
harmShift = y /. Solve[D[f[y] /. harmSolClean, y] == 0, y] // Last // FullSimplify

(*Morse EPC*)
morseShiftRaw = y /. Solve[D[f[y] /. morseSolClean, y] == 0, y] // Last // FullSimplify
morseShift =
  y - (morseShiftRaw /. v → 0) /. Solve[D[f[y] /. morseSolClean, y] == 0, y] // Last //
   FullSimplify;
(*Simplify solution to case where speed of climate change (v) is small*)
Series[D[morseShift, v] /. v → 0, {a, 0, 2}] /. frontWidthRule // FullSimplify
```

Out[37]=
$$-\frac{v\,w}{2\,\text{frontWidth}\,r0}$$

Out[38]=
$$\frac{\text{Log}\left[1-\frac{a\left(a\,\text{frontWidth}^2\,r0+v\right)w}{2\,\text{frontWidth}\,r0}\right]}{a}\ \text{if}\ \boxed{\textit{condition}}$$

Out[40]=
$$\frac{w}{-2\sqrt{\text{disp}\,r0}+a^2\,\text{disp}\,w}\ \text{if}\ \boxed{\textit{condition}}$$

In[41]:=
```
(* Effect of weather on "friction" constant: *)
(* Notable effect at second order only if sigma = O(Sqrt[frontWidth * w]) *)
Assuming[a > 0, Series[D[morseShift // FullSimplify, v] /. v → 0, {a, 0, 2}] //
  FullSimplify] (* without 'weather'*)
Assuming[e^(a^2 sigma^2) (-1 + a^2 w^2) > -1,
    Series[D[morseShift /. morseWeatherRule // FullSimplify, v] /. v → 0, {a, 0, 2}] //
     FullSimplify] /. frontWidthRule // FullSimplify;
Style[%, Background → LightBlue]
```

Out[41]=
$$-\frac{w}{2\,(\text{frontWidth}\,r0)}-\frac{w^2\,a^2}{4\,r0}+O[a]^3$$

Out[43]=
$$-\frac{w}{2\sqrt{\text{disp}\,r0}}-\frac{\left(w\left(\sqrt{\text{disp}\,r0}\ \text{sigma}^2+\text{disp}\,w\right)\right)a^2}{4\,(\text{disp}\,r0)}+O[a]^3\ \text{if}\ a^2<e^{\frac{a^2\,\text{sigma}^2}{2}}\sqrt{\frac{\text{disp}}{r0}}\ w<2$$

2.4.2: Centre of mass motion:

In[44]:=
```
(*  First define the total area under the curve, to standardise with *)
morseNorm = Integrate[f[x] /. morseSolClean,
    {x, -Infinity, Infinity}, GenerateConditions → False] // FullSimplify;


(* Define the moment generating function  *)
(*not sure about the sudden jump to using'x'?*)
morseMGF = Integrate[Exp[t x] × f[x] /. morseSolClean, {x, -Infinity, Infinity},
                    GenerateConditions → False] / morseNorm // FullSimplify;

(*  Make an expression for changes in  MGF through time,
then solve for the start point  *)
morseCentreMass = D[morseMGF, t] /. t → 0 // FullSimplify;
```

The formula for the position of the centre of mass looks singular for v, a == 0, but actually it's smooth (technically, it can be "analytically continued" to $a==0$ and $v==0$). Based on the symmetry of the problem, centre of mass (in the co-moving reference system) is given by $\text{centre of mass} = c_1 a + c_2 v + c_3 a^3 + c_4 a^2 v + c_5 a v^2 + c_6 v^3$ + terms of order 5 and higher, with some constants c_1, ..., c_6.
However, since we are only interested in the lag in time for small v, which can be computed using only the even powers: as $D[\text{centre of mass},v] (at v=0) = c2 + c4 a^2$ + terms of order 4 and higher. Since $c2$ is the same as for the harmonic potential, we take if from there.
To get $c4$, we first compute $(1/2) * d^2 (\text{centre of mass}) / da^2 = 3 c_3 a + c_4 v $+ terms of order 2 and higher (and calls this "morseCentreMassEffect"), then takes the limit $a ->0$ (which remove the c_3 term and all higher order terms containing a) and then takes the derivative with respect to v at $v = 0$ (which converts $c_4 v$ to c_4 and drops the remaining higher order terms). The result is c_4.
Finally, the lag $c_2 + c_4 a^2$ is assembled from the isolated terms.

In[47]:=
```
morseCentreMassEffect = (1 / 2) D[morseCentreMass, {a, 2}] // FullSimplify;
morseCentreMassCorrection =
 a^2 D[Limit[morseCentreMassEffect, a → 0, Direction → "FromAbove"] // FullSimplify,
    v] /. v → 0
harmLag = D[y /. Solve[D[f[y] /. harmSolClean, y] == 0, y] // Last, v];

morseCentreMassLag = harmLag + morseCentreMassCorrection // FullSimplify;

(*Present in a neater form*)
morseCentreMassLag /. frontWidthRule // FullSimplify // Apart

(*with some corner case-avoiding assumptions,
 find power series expansion in terms of a to second order,
and present using raw parameters*)
morseCentreMassLag2 =
 Assuming[1 + ℯ^(a^2 sigma^2) (-1 + a^2 w^2) > 0,
   (Series[morseCentreMassLag /. morseWeatherRule // FullSimplify, {a, 0, 2}] //
     FullSimplify)] /. frontWidthRule
```

Limit: Warning: Assumptions that involve the limit variable are ignored.

Out[48]=
$$-\frac{a^2 w^2}{2 r0}$$

Out[51]=
$$-\frac{\sqrt{\text{disp}\, r0}\, w}{2\, \text{disp}\, r0} - \frac{a^2 w^2}{2 r0}$$

Out[52]=
$$-\frac{w}{2\left(\sqrt{\frac{\text{disp}}{r0}}\, r0\right)} - \frac{\left(w\left(\text{sigma}^2 + 2\sqrt{\frac{\text{disp}}{r0}}\, w\right)\right) a^2}{4\left(\sqrt{\frac{\text{disp}}{r0}}\, r0\right)} + O[a]^3$$

2.5 Response 3: Local of Maximum Sensitivity

In[53]:=
```
(* Calculation of location (in terms of y) of maximum sensitivity  *)
sensitivity =
  (f[y] /. morseSolClean /. v → -v) * (f[y] /. morseSolClean) // FullSimplify;
ySenseRule = Solve[D[sensitivity, y] == 0, y] // FullSimplify // Last;

Style[ySenseRule, Background → LightBlue]
```

Out[55]=
$$\left\{y \to \frac{\text{Log}\left[1 - \frac{1}{2} a^2\, \text{frontWidth}\, w\right]}{a} \text{ if } w < \frac{2}{a^2\, \text{frontWidth}}\right\}$$

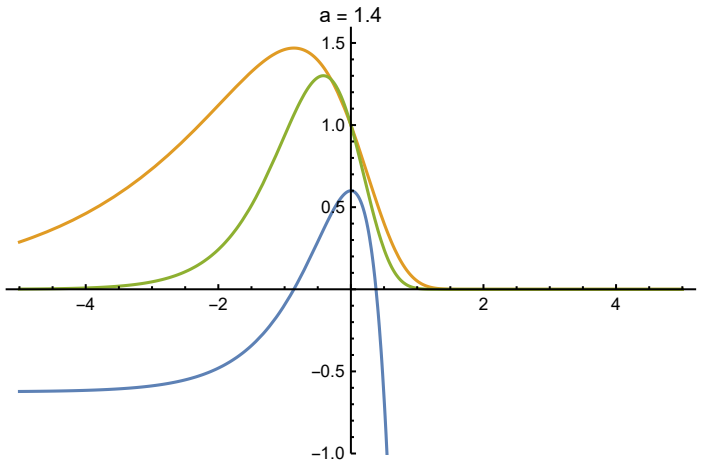
In[56]:=
```
(*Calculate f at y =0, to standardise plotting height*)
f0 = f[y] /. morseSolClean /. y → 0
```

Out[56]= $e^{-\frac{1}{a^2\ \text{frontWidth}\ w}}$

```
Plot[{ g[y] / 10 /. gMorseRule,  (* Plot out Growth function [Blue] *)
       f[y] / f0 /. morseSolClean,
     (* Plot out psi function, standardised to equal 1 at y=0 [Orange ]*)
     sensitivity / f0^2} /.        (* Plot out Sensitivity, also scaled [Green]*)
    {a → 1.4, w → 0.5, frontWidth → 0.9, v → 4, r0 → 6} // Evaluate,
 {y, -5, 5},  (* Define conditions to plot*)
     PlotRange → {-1, All}, (* vertical axis range *)
    PlotLabel → "a = 1.4",
     PlotLegends → Placed[LineLegend[ColorData[97, "ColorList"][[1 ;; 3]],
     {"EPC", "Density", "Sensitivity"}], {0.85, 0.8}]]

Plot[{ g[y] / 10 /. gMorseRule,
       f[y] / f0 /. morseSolClean,
     sensitivity / f0^2} /.
    {a → -1.4, w → 0.5, frontWidth → 0.9, v → 4, r0 → 6} // Evaluate, {y, -5, 5},
     PlotRange → {-1, All},
    PlotLabel → "a = -1.4",
     PlotLegends → Placed[LineLegend[ColorData[97, "ColorList"][[1 ;; 3]],
     {"EPC", "Density", "Sensitivity"}], {0.85, 0.8}]]
```

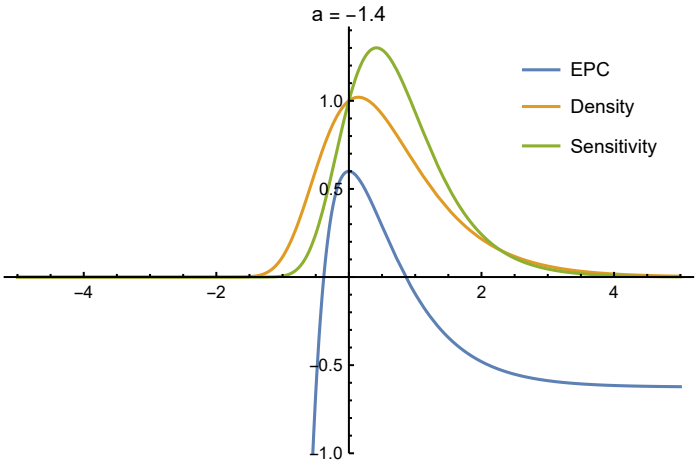
Out[57]=



General: Exp[−1235.37] is too small to represent as a normalized machine number; precision may be lost.

General: Exp[−2474.85] is too small to represent as a normalized machine number; precision may be lost.

Out[58]=



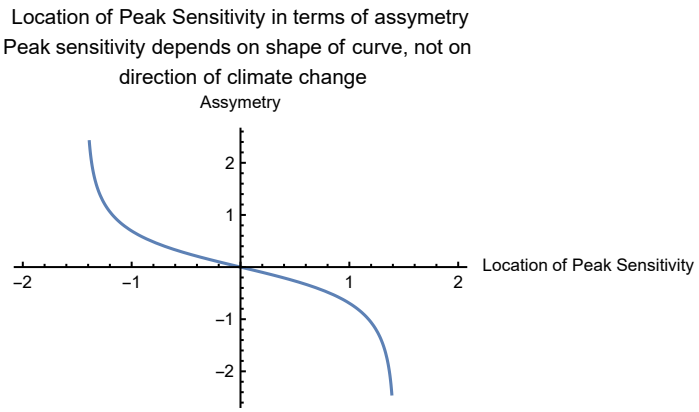
In[59]:=
```
(* Range of permissible a values *)
Solve[a (a frontWidth^2 r0 + v) w == 2 frontWidth r0, {a}] // FullSimplify
```

Out[59]=

$$\left\{\left\{a \to -\frac{v + \sqrt{v^2 + \frac{8\,\text{frontWidth}^3\,r0^2}{w}}}{2\,\text{frontWidth}^2\,r0}\right\}, \left\{a \to \frac{-v + \sqrt{v^2 + \frac{8\,\text{frontWidth}^3\,r0^2}{w}}}{2\,\text{frontWidth}^2\,r0}\right\}\right\}$$

In[60]:=
```
Plot[y /. ySenseRule /. frontWidth → 1 /. w → 1 // Evaluate,
 {a, -2, 2}, AxesLabel → {"Location of Peak Sensitivity", "Assymetry"},
     PlotLabel → "Location of Peak Sensitivity in terms of assymetry\nPeak
   sensitivity depends on shape of curve, not on \n direction of climate change" ]
```

Out[60]=

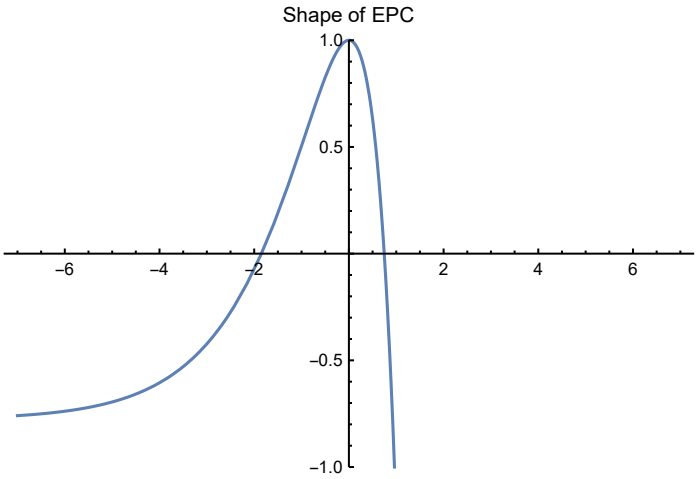


2.6 Example Plots

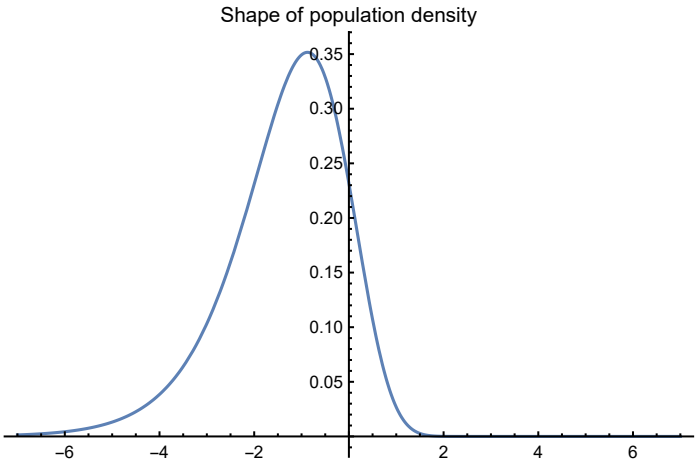
In[61]:=
```
num = Join[dispRule, {w → 1, r0 → 1, a → 3 / 4, v → 0.5, frontWidth → 0.6}];
morseSigRule //. num
(*Repeatedly put in the parameters to find what lambda is with these*)
Plot[g[y] /. gMorseRule //. num, {y, -7, 7},
      PlotRange → {-r0, r0} //. num // Evaluate,
      PlotLabel → "Shape of EPC"]
Plot[f[y] / Abs[morseNorm] /. morseSolClean //. num // Evaluate, {y, -7, 7},
     PlotRange → All,
     PlotLabel → "Shape of population density"]
```

Out[62]= {lam → 0.277014}

Out[63]=



Out[64]=



In[65]:=
```
(* Evaluation through time*)
(* Defining  movement equation explicitly,then put into moving reference frame *)
```

In[66]:=
```
eqPLOT = (D[f[x - x0[t], t], t] == disp * D[f[x - x0[t], t], {x, 2}] +
      g[x - x0[t]] × f[x - x0[t], t] - f[x - x0[t], t]^2) /. x → y + x0[t];
L = 13;
tMin = -20;
tMax = 25;
num = {disp → 0.2,
      v0 → 0.45,  (*rate of climate change*)
     x0 → Function[t, t * v0 * HeavisideTheta[t]] ,  (*Effectively this defines the
    changing envrionment*)(*v →Function[t,v0*HeavisideTheta[t]],*)
   g → Function [y, (1 - (1 - Exp[y])^2)] };
    (* Implicitly using r0 = 1, a = 1, w = 1,
i.e. : g→Function [y, r0(1-1/(a^2 w^2)(1-Exp[a y])^2)]     *)
(*      g→ Function[{y},2*Exp[-y^2/2]-1]}; Gaussian Growth function*)



sol = NDSolve[eqPLOT &&
             f[y, tMin] == 0.2 * (1 - (1 - Exp[y])^2) &&
       (*Define starting distribution   (1-(1-Exp[y])^2)  *)
             f[-L, t] == 0 &&
             f[L, t] == 0 //. num // FullSimplify,
            f,  (*Solve for F*)
    {y, -L, L},  (*Range of y to consider*)
    {t, tMin, tMax}] //  (*Range of times to consider*)
   Last;
```

NDSolve: Warning: boundary and initial conditions are inconsistent.

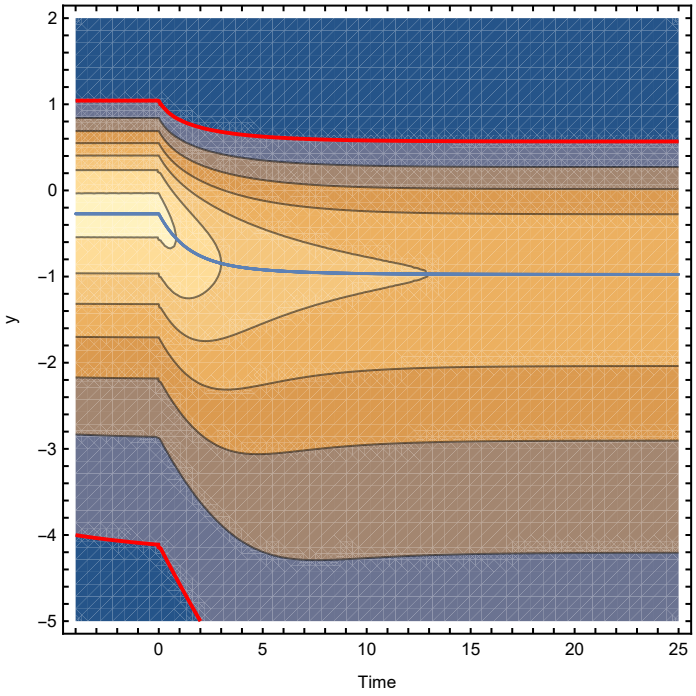
In[72]:=
```
pl1 = ContourPlot[10^-2 + f[x, t] /. sol, {t, 0.2 tMin, tMax},
                  {x, -5, 2},
                 PlotRange → {0, All}, PlotPoints → 50, FrameLabel → {"Time", "y"}];

(*Add boundary for where population is above a threshold*)
pl1r = ContourPlot[10^-2 + f[x, t] /. sol,
                  {t, 0.2 tMin, tMax},
                  {x, -5, 2},
                   PlotRange → {0, All}, PlotPoints → 50, Contours → {0.1},
                   ContourStyle → {{Red, Thick}}, ContourShading → None];

(*Location of peak of distribution*)
pl3 = ListPlot[Table[{t, y /. FindMaximum[f[y, t] /. sol,
                        {y, -0.1 L, 0.1 L}][[2]]},
                        {t, 0.2 tMin, tMax, 0.01}]];

(*Combine all the parts of the plot together*)
Show[pl1, pl3, pl1r]
```

Out[75]=



In[76]:=
```
(* Same plot, now in spatially fixed reference frame *)
(* Define f for fixed coordinate system *)
fFixed[x_, t_] = If[Abs[x - x0[t]] < L, 10^-2 + f[x - x0[t], t], 0] //. num;

(* Contour plot of f *)
pl1 = ContourPlot[fFixed[x, t] /. sol // Evaluate, {t, 0.2 tMin, tMax},
                  {x, -5, 2},
                 PlotRange → {0, All}, PlotPoints → 50, FrameLabel → {"Time", "x"}];

(*Add boundary for where population is above a threshold*)
pl1r = ContourPlot[10^-2 + fFixed[x, t] /. sol // Evaluate,
                  {t, 0.2 tMin, tMax},
                  {x, -5, 2},
                   PlotRange → {0, All}, PlotPoints → 50, Contours → {0.1},
                   ContourStyle → {{Red, Thick}}, ContourShading → None];

(*Location of peak of distribution*)
pl3 = ListPlot[Table[{t, (x0[t] //. num) + y /. FindMaximum[f[y, t] /. sol,
                       {y, -0.1 L, 0.1 L}][[2]]},
                       {t, 0.2 tMin, tMax, 0.01}]];

(*Combine all the parts of the plot together*)
Show[pl1, pl3, pl1r]
```

Out[80]=