LEGI–Amp–nz Model

- **Init**

```
<< Graphics`Graphics`
Off[General::"spell"]
Off[General::"spell1"]

SetDirectory["/home/abergman/dicty/figs"];
$TextStyle = {FontFamily → "Helvetica", FontWeight -> "Bold", FontSize → 14};
$FormatType = TraditionalForm;

Unprotect[Mean, StandardDeviation];
Mean[{}] := 0
StandardDeviation[{}] := 0
Protect[Mean, StandardDeviation];
```

The Model

$$\text{RangeI}[a\_, b\_, i\_] := \text{If}\left[i == 1, a, \text{Range}\left[a, b, \frac{b-a}{i-1}\right]\right]$$

```
ampTot = 1;
rTot = 1;
Protect[ampTot, rTot];
```

$$\text{sys} := \Big\{$$

$$a'[t] == \tau a\,(kl * L - kd * a[t] + kc),$$

$$i'[t] == \tau i\,(kl * L - kd * i[t] + kc),$$

$$r'[t] == \tau r\,(a[t] * (rTot - r[t]) - i[t] * r[t]),$$

$$amp'[t] == \tau m \left(\frac{r[t]\,(ampTot - amp[t])}{Km + (ampTot - amp[t])} - \frac{I2 * amp[t]}{Km + amp[t]}\right)$$

$$\Big\};$$

```
params = {
```

```
    L → 10^2. * UnitStep[t],
    τa → 0.1, τi → 0.01, τr → 1,
    kl → 0.3, kd → 2, kc → 0.1,
    τm → .3,
    Km → 10^-2.,
    I2 → 0.7
   };
ic[p___] := {
    icA | icI → kc/kd /. Flatten[{{p}, params}],
    icR → 1 / 2.,
    icAmp[I2_] :→ Block[{ss},
      If[NumericQ[I2] && I2 == 0.5, 0.5,
       ss = 1/(2 (-1 + 2 I2))
           (-ampTot + 2 ampTot I2 + Km + 2 I2 Km + # √(-4 ampTot (-1 + 2 I2) Km + (ampTot - 2 ampTot I2 - Km - 2 I2 Km)^2)) & /@ {-1, 1};
       ss = ss /. Flatten[{{p}, params}];
       Which[
        0 ≤ ss[[1]] ≤ ampTot, ss[[1]],
        0 ≤ ss[[2]] ≤ ampTot, ss[[2]],
        True, Message[icAmp::"outOfRange", ss[[1]], ss[[2]], ampTot]; Exit[]]
       ]
      ]
   };
Protect[sys, params, ic];
icAmp::"outOfRange" = "Initial condition for amp[0]={`1`,`2`} outside feasible region [0,`3`]";
rdt = x_[t] → x;
t0 = 0;
tend = 200;
runsim[p___] := First@NDSolve[Join[sys /. {p} /. params, {a[t0] == icA, i[t0] == icI, r[t0] == icR, amp[t0] == icAmp[I2]} /. ic[p]],
    {a, i, r, amp}, {t, t0, tend}, MaxSteps → ∞, MaxStepSize → 1]

i2n = 200;
i2range = RangeI[0.55, .9, i2n];
```

```
popsim[p___] := Array[amp[#][t] &, i2n] /. First@NDSolve[Join[
      sys[[{1, 2, 3}]] /. Flatten[{{p}, params}],
      Array[amp[#]'[t] == τm (r[t] (ampTot - amp[#][t])/(Km + (ampTot - amp[#][t])) - i2range[[#]] amp[#][t]/(Km + amp[#][t])) &, i2n] /. Flatten[{{p}, params}],
      Array[amp[#][t0] == icAmp[i2range[[#]]] &, i2n],
      {a[t0] == icA, i[t0] == icI, r[t0] == icR}] /. ic[p],
    Join[{a, i, r}, Array[amp, i2n]], {t, t0, tend}, MaxSteps → ∞, MaxStepSize → 1]

responderThres = 0.2;
PickResponders[sims_, duration_: 100] := Pick[sims, responderThres < Max[# /. t → Range[0, duration, .1]] & /@ sims];

Protect[rdt, t0, i2n, i2range, responderThres];
```

Initial Condition for R and amp

```
Solve[Thread[sys[[{1, 2, 3}, 2]] == 0] /. rdt, r]
```

$$\left\{\left\{r \to \frac{a}{a+i}\right\}\right\}$$

```
{r, amp} /. Solve[Thread[sys[[All, 2]] == 0] /. rdt, {r, amp}, {a, i}] /. params
```

$$\left\{\left\{\frac{1}{2}, 1.03587\right\}, \left\{\frac{1}{2}, 0.0241344\right\}\right\}$$

```
Unprotect[ampTot]
ampTot
```

{ampTot}

1

```
Block[{ampTot},
 Print@Solve[Thread[sys[[All, 2]] == 0] /. rdt, {r, amp}, {a, i}]
]
```

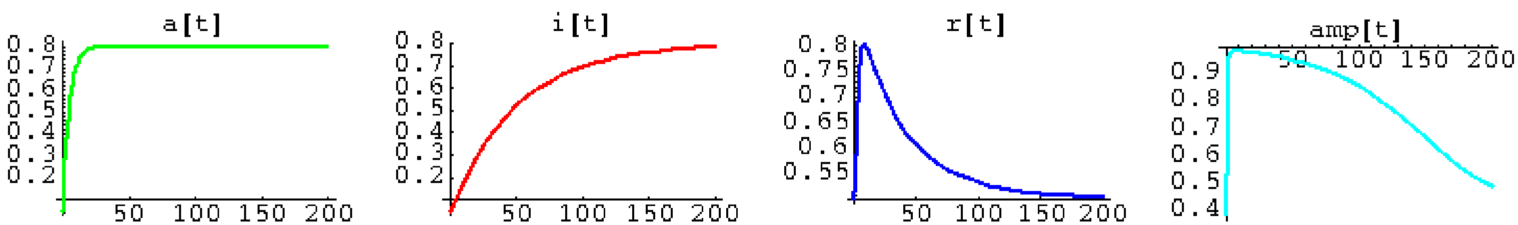
$$\left\{\left\{r\to\frac{1}{2},\ amp\to\frac{1}{2\,(-1+2\,I2)}\left(-ampTot+2\,ampTot\,I2+Km+2\,I2\,Km-\sqrt{-4\,ampTot\,(-1+2\,I2)\,Km+(ampTot-2\,ampTot\,I2-Km-2\,I2\,Km)^{2}}\right)\right\},\right.$$
$$\left.\left\{r\to\frac{1}{2},\ amp\to\frac{1}{2\,(-1+2\,I2)}\left(-ampTot+2\,ampTot\,I2+Km+2\,I2\,Km+\sqrt{-4\,ampTot\,(-1+2\,I2)\,Km+(ampTot-2\,ampTot\,I2-Km-2\,I2\,Km)^{2}}\right)\right\}\right\}$$

I2 Distribution

```
sys /. params
```

$$\left\{a'[t] == 0.1\,(0.1-2\,a[t]+30.\,\mathrm{UnitStep}[t]),\ i'[t] == 0.01\,(0.1-2\,i[t]+30.\,\mathrm{UnitStep}[t]),\right.$$
$$\left.r'[t] == a[t]\,(1-r[t])-i[t]\,r[t],\ amp'[t] == 0.3\left(-\frac{0.7\,amp[t]}{0.01+amp[t]}+\frac{(1-amp[t])\,r[t]}{1.01-amp[t]}\right)\right\}$$

```
sim = runsim[L → 5, I2 → 0.505, τm → 10];

g = DisplayTogetherArray[
   Plot[#1 /. sim, {t, t0, tend}, PlotLabel → ToString[#1], PlotRange → All, PlotStyle → {Thickness[.02], #2}] &~
    MapThread~{{a[t], i[t], r[t], amp[t]}, {Green, Red, Blue, Cyan}}];
```




```
dat = {a[t], i[t], r[t]} /. sim /. t → Range[t0, tend, .1];
Export["Fig4.a-i-r.csv", Transpose[dat]]
Export["Fig4.a-i-r.gif", g]
```

Fig4.a-i-r.csv

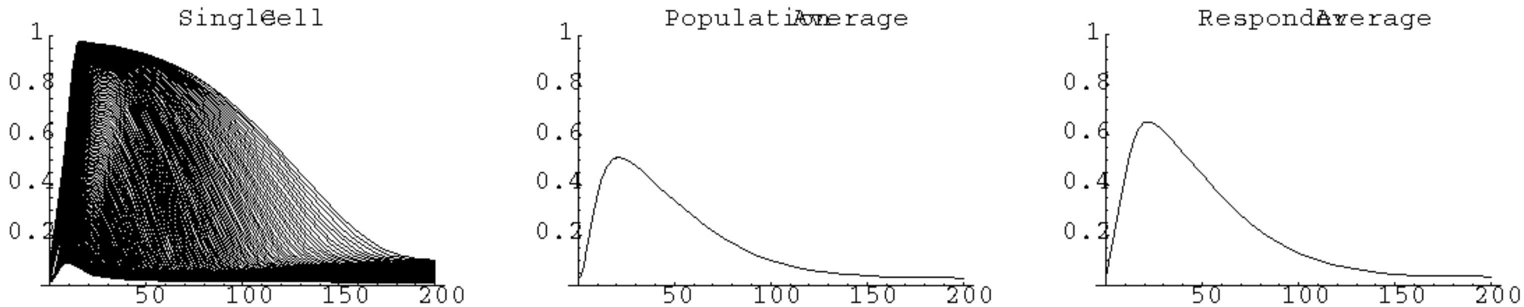
Fig4.a-i-r.gif

```
i2range = Range[0.55, .9, .01];
```

```
Set::wrsym : Symbol i2range is Protected. More…
```

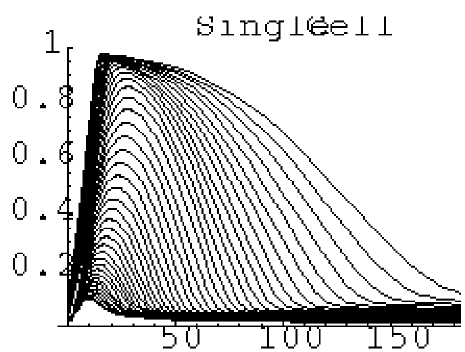
```
sims = amp[t] /. runsim[L → 10^-8, I2 → #] & /@ i2range;
rsims = PickResponders[sims];
```

```
DisplayTogetherArray[
  Plot[Evaluate@sims, {t, t0, tend}, PlotRange → {0, 1}, PlotLabel → "Single Cell"],
  Plot[Evaluate@Mean@sims, {t, t0, tend}, PlotRange → {0, 1}, PlotLabel -> "Population Average"],
  Plot[Evaluate@Mean[rsims], {t, t0, tend}, PlotRange → {0, 1}, PlotLabel -> "Responder Average"]];
```




```
tsims = Take[sims, {1, -1, 5}];
```

```
g = Plot[Evaluate@tsims, {t, t0, tend}, PlotRange → {0, 1}, PlotLabel → "Single Cell"];
```

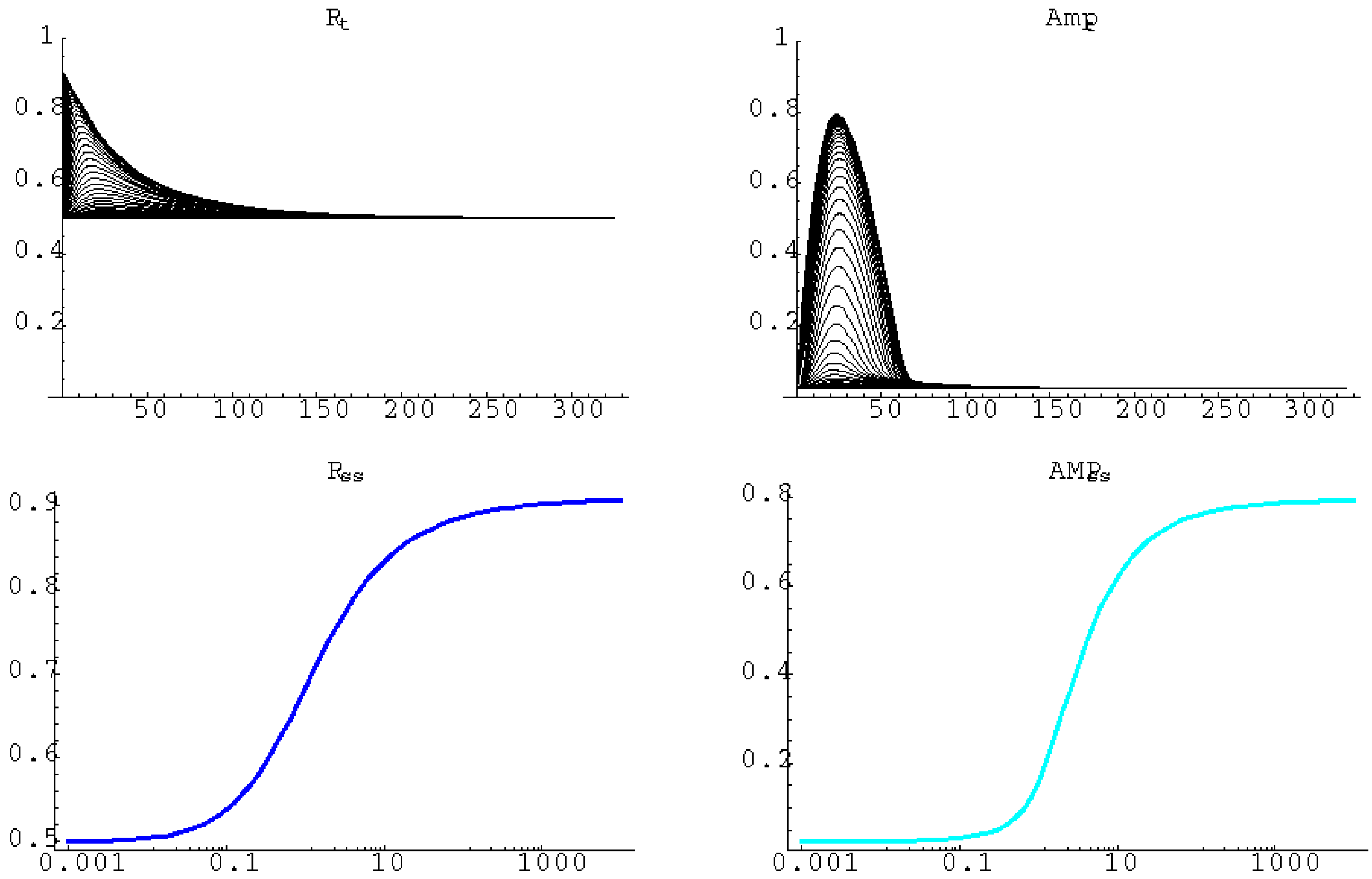



```
dat = tsims /. t → Range[t0, tend, .1];
Export["Fig4.amps.csv", Transpose[dat]];
Export["Fig4.amps.gif", g];
```

Varying Input

```
tend = 325;
stiminput = 10^Range[-3., 4, .1];
sims = runsim[L → #, I2 → 0.7005] & /@ stiminput;
datmax = Max[r[Range[0, tend, .1]] /. #] & /@ sims;
```

```
g = DisplayTogetherArray[{{
    DisplayTogether[Plot[r[t] /. sims // Evaluate, {t, 0, tend}], PlotRange → {0, 1}, PlotLabel -> "Rt"],
    DisplayTogether[Plot[amp[t] /. sims // Evaluate, {t, 0, tend}], PlotRange → {0, 1}, PlotLabel -> "Ampt"]}, {
    LogLinearListPlot[{stiminput, datmax}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Blue}, PlotLabel -> "Rss"],
    LogLinearListPlot[{stiminput, Max[amp[Range[0, tend, .1]] /. #] & /@ sims}ᵀ,
     PlotJoined → True, PlotStyle → {Thickness[.01], Cyan}, PlotRange → All, PlotLabel -> "AMPss"]}}];
```



```
stiminput
```

```
{}
```

```
Export["with-kc.gif", g]
```

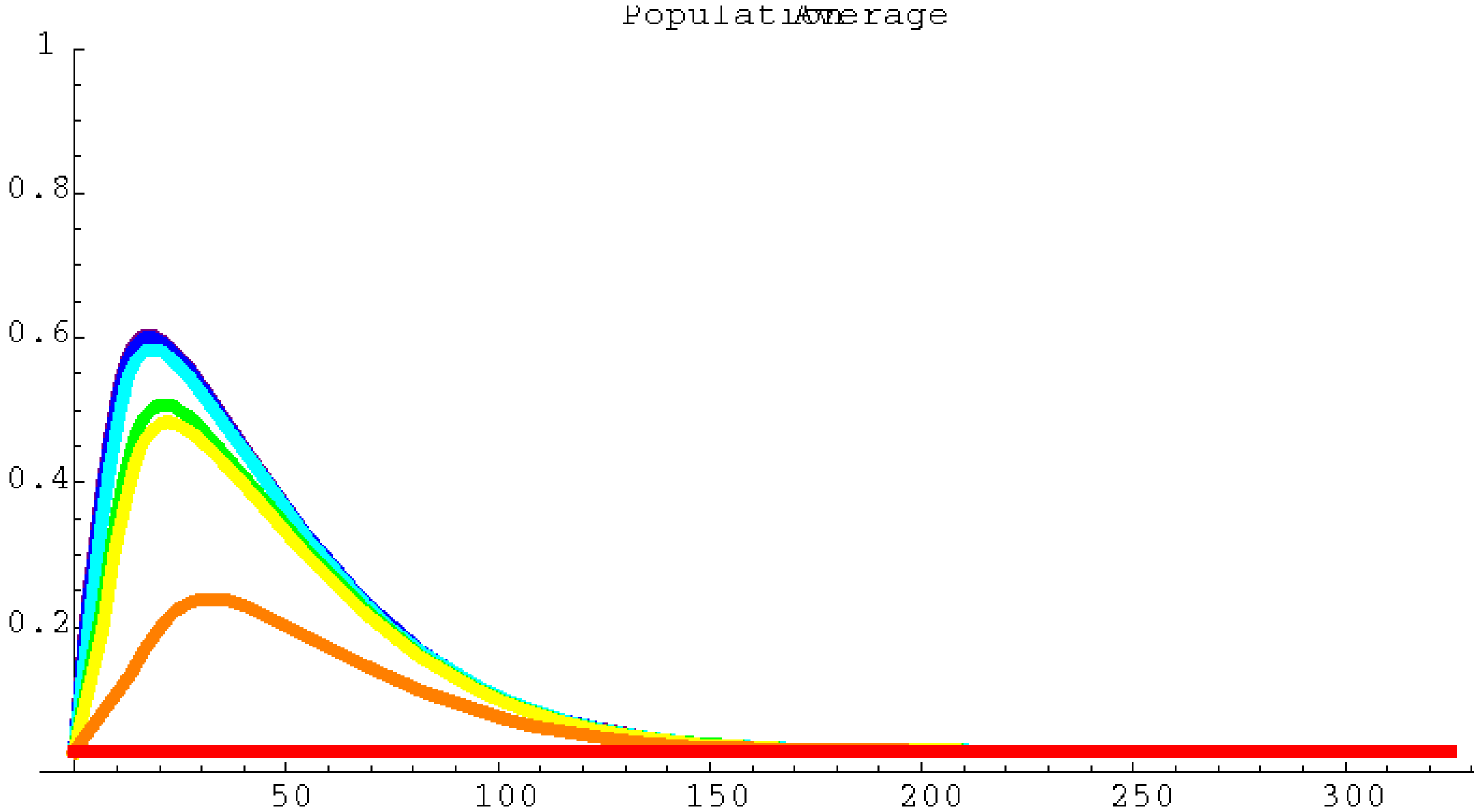
```
with-kc.gif
```

```
stiminput = 10^Range[-12., -5, 1];
```

```
colors = Reverse@{Red, Orange, Yellow, Green, Cyan, Blue, Purple};
stiminput = Reverse@{10^-12, 10^-9, 7 * 10^-9, 10 * 10^-9, 100 * 10^-9, 10^-6, 10 * 10^-6} * 10^9;
len = Length[stiminput];
```

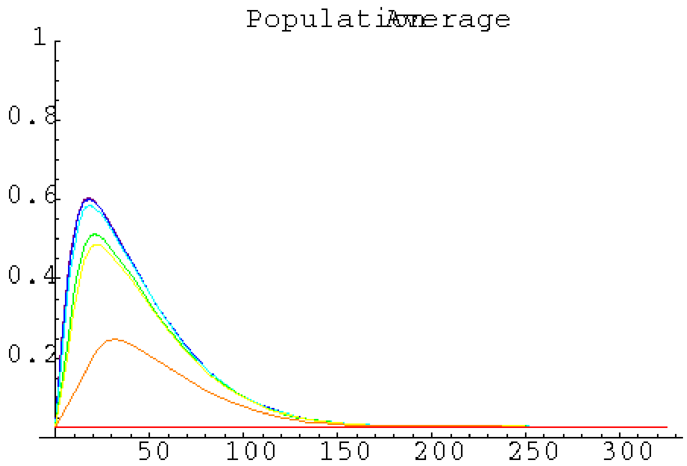
```
sims = popsim[L → #] & /@ stiminput;
rsims = PickResponders /@ sims;
msims = Mean /@ sims;
```

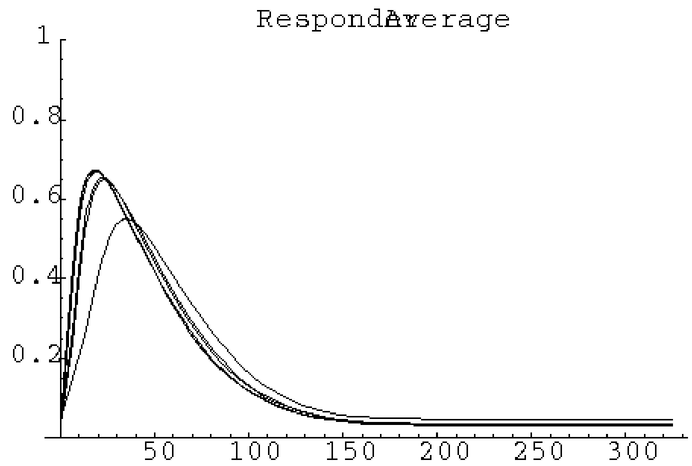
```
g = DisplayTogether[Plot[msims[[#]], {t, t0, tend}, PlotRange → {0, 1},
      PlotLabel -> "Population Average", PlotStyle → {colors[[#]], Thickness[0.01]}, AspectRatio → 0.5] & ~ Array ~ len];
```




```
ti = Range[t0, tend, .5];
dat = Prepend[msims, t] /. t → ti // Transpose;
Export["temporal-dose-response-population.gif", g];
Export["temporal-dose-response-population.csv", dat];
```

```
DisplayTogetherArray[
  DisplayTogether[Plot[Evaluate@Mean[sims[[#]]], {t, t0, tend},
      PlotRange → {0, 1}, PlotLabel -> "Population Average", PlotStyle → colors[[#]]] &~Array~len],
  DisplayTogether[Plot[Evaluate@Mean[#], {t, t0, tend}, PlotRange → {0, 1}, PlotLabel -> "Responder Average"] & /@ rsims]];
```





```
Block[{TickFontSize = 15, YLabelFontSize = 25, tend = 200},
 DisplayTogetherArray[Function[L0,
      sim = runsim[L → 10^L0 * UnitStep[t]];
      Plot[{r[t], amp[t]} /. sim // Evaluate, {t, t0, tend},
       PlotRange → {All, {0, 1}}, Epilog → {Red, Thickness[0.01], Line[{{0, I2}, {100, I2}} /. params]},
       PlotStyle → Thread[{{Blue, Cyan}, Thickness[0.02]}], PlotLabel → "Input: " <> ToString[10.^L0]
      ]] /@ Range[-12, -7] // Partition[#, 3] &];
]
```

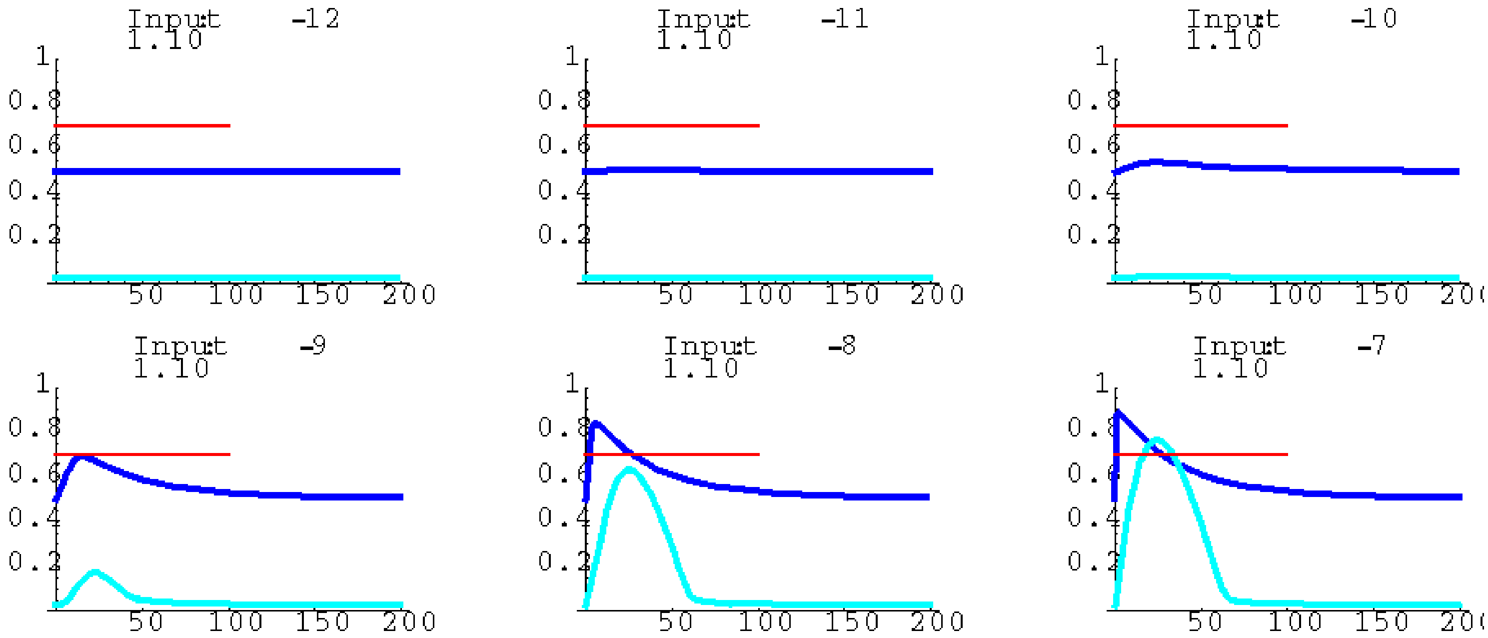
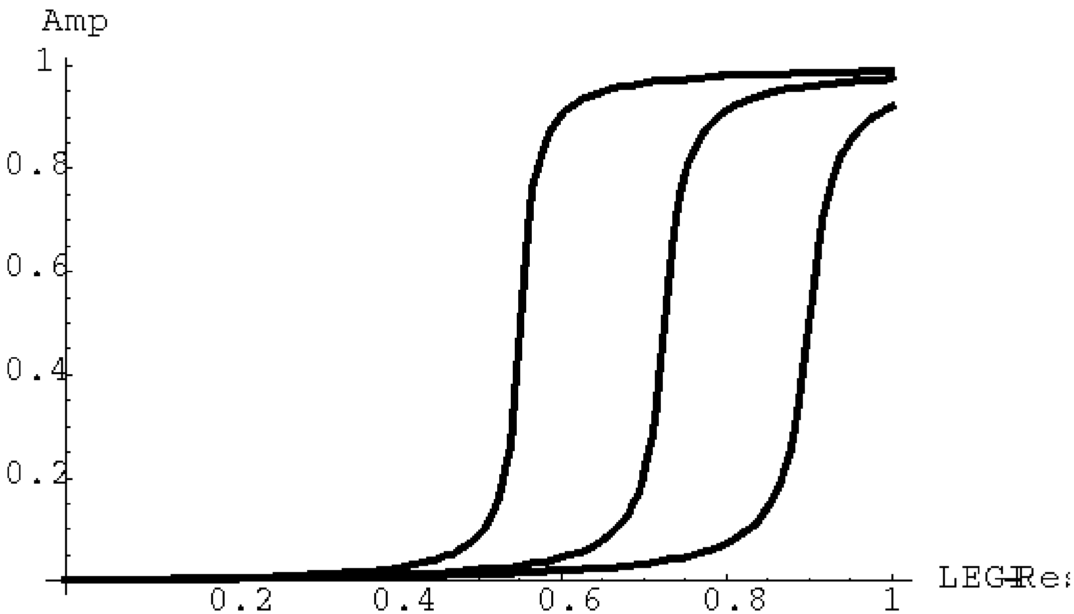


Fig 4b. LEGI-Response vs Amp

```
ssAmp = amp /. Solve[sys[[-1, 2]] == 0 /. rdt, amp][[1]]
```

$$\frac{1}{2\,(\text{I2}-r)}\left(\text{I2}+\text{I2}\,\text{Km}-r+\text{Km}\,r-\sqrt{-4\,\text{Km}\,(\text{I2}-r)\,r+(-\text{I2}-\text{I2}\,\text{Km}+r-\text{Km}\,r)^2}\right)$$

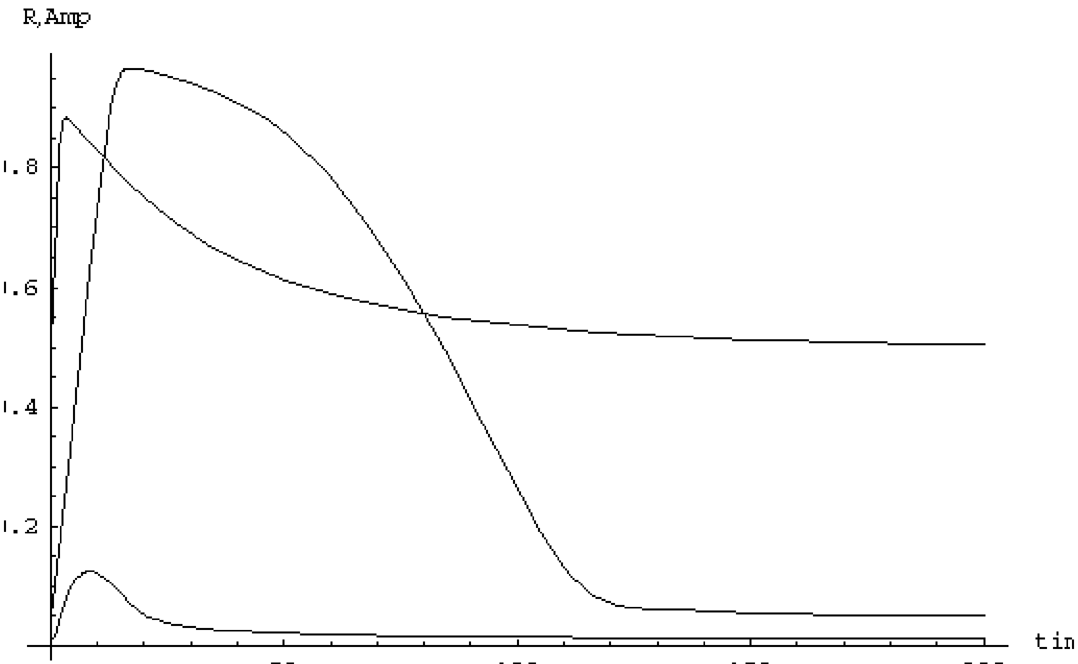
```
temp = {.55, (.9 + .55)/2, .9};
g = DisplayTogether[Plot[ssAmp /. I2 → # /. params // Evaluate, {r, 0, 1}, PlotStyle → Thickness[0.01]] & /@ temp,
   AxesLabel → {"LEGI-Response", "Amp"}];
```




```
ti = Range[0.0001, 1, .001];
dat = ssAmp /. I2 → # /. params /. r → ti & /@ temp;
PrependTo[dat, ti];

file = "ss-legi-r-vs-amp";
Export[file <> ".gif", g];
Export[file <> ".csv", dat⊤];
```

Fig 4c. Amp at two I2 levels

```
tend = 200;
sims = runsim[I2 → #] & /@ {0.6, 0.9};

g = Plot[{r[t], amp[t]} /. sims // Evaluate, {t, t0, tend}, AxesLabel → {"time (sec)", "R,Amp"}]
```




```
- Graphics -

Export["fig4c.gif", g]

fig4c.gif

timeIdx = Range[t0, tend, .1];

dat = Flatten[{t, r[t], amp[t]} /. sims] /. t → timeIdx;
```

```
Dimensions[dat]
```

{6, 2001}

```
Export["fig4c.csv", datᵀ]
```

fig4c.csv

Varying I2

```
Block[{TickFontSize = 15, tend = 200},
 DisplayTogetherArray[Function[i2,
     sim = runsim[I2 → i2, L → 10^-10];
     Plot[{r[t], amp[t]} /. sim // Evaluate,
      {t, t0, tend}, PlotRange → {All, {0, 1}}, PlotStyle → Thread[{{Blue, Cyan}, Thickness[0.02]}],
      PlotLabel -> "I2: " <> ToString[i2], Epilog → {Red, Thickness[0.01], Line[{{0, i2}, {100, i2}}]}]
    ] /@ Range[.501, .6, .01] // Partition[#, 3] &];
]
```

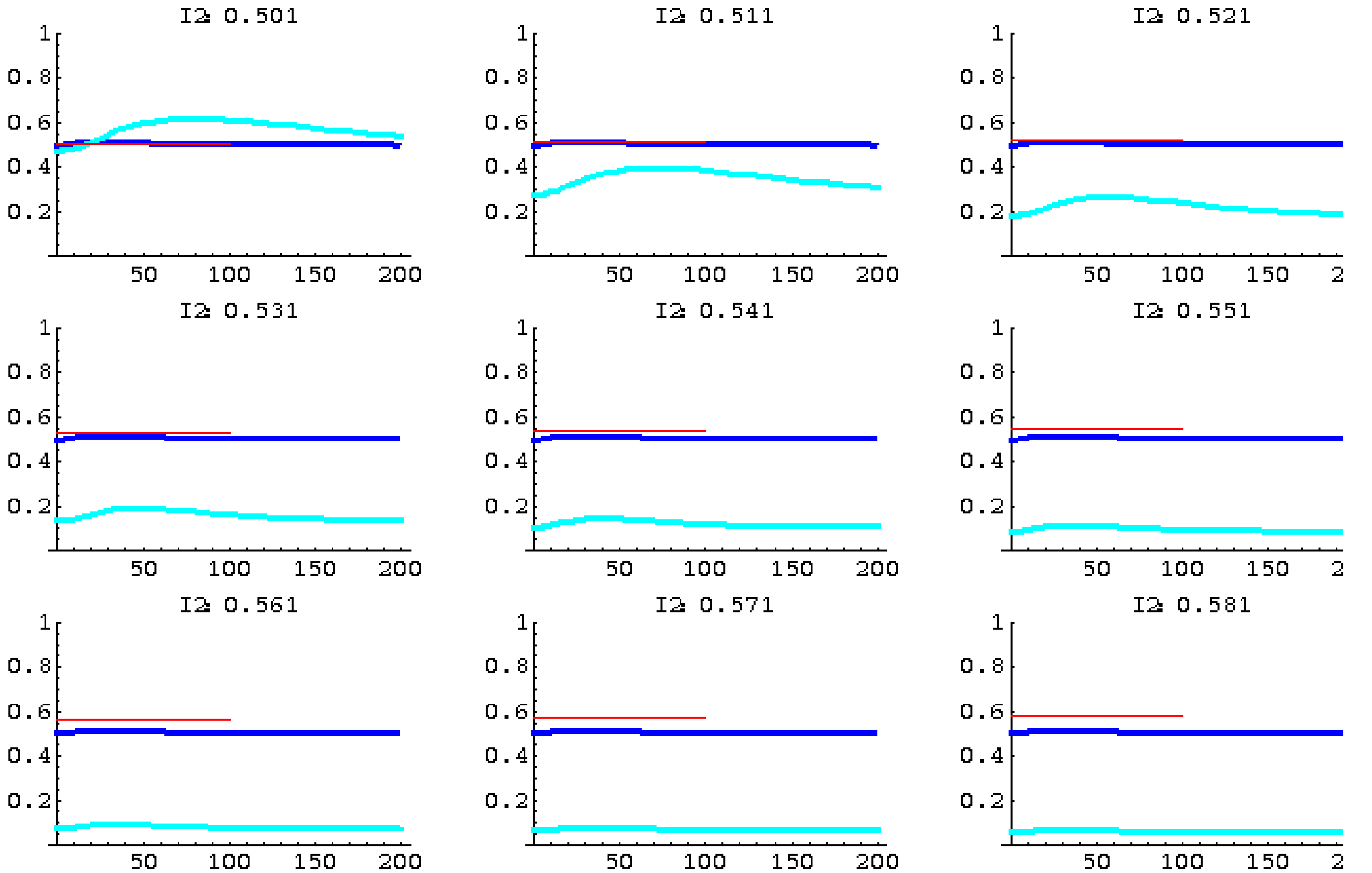


Fig 4d. Average Peak Amp for Whole pop, responders, non-responders

```
stiminput = 10^RangeI[-12., -5, 200];
```

```
tend = 200;
sims = popsim[L → #] & /@ stiminput;
Dimensions[sims]
```

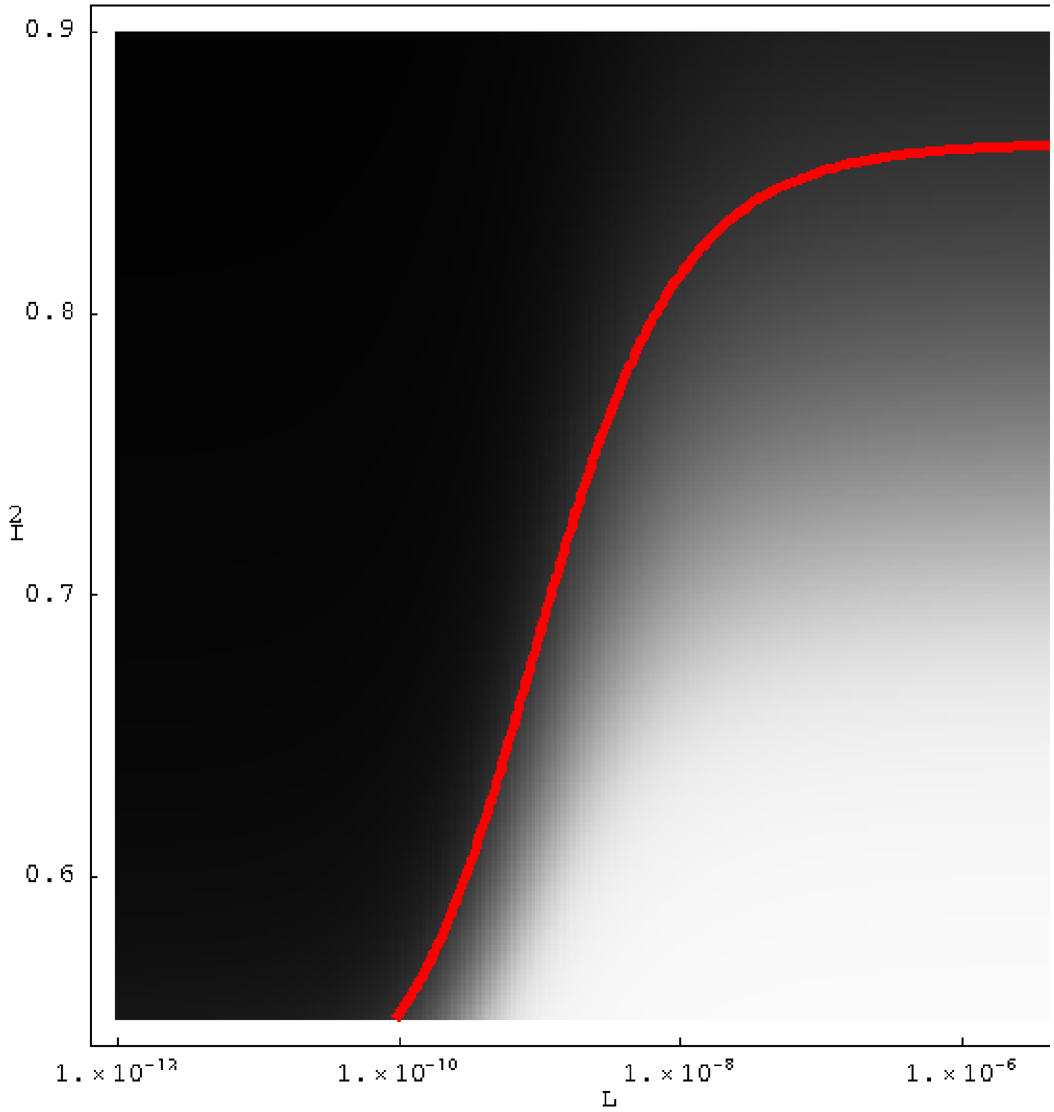
```
{200, 200}
```

```
peak = Map[Max[# /. t -> Range[0, tend, .1]] &, sims, {2}];
```

```
TickInterpolation = Function[list, Module[{x1, x2, y1, y2},
    x1 = list[[1]];
    x2 = list[[-1]];
    y1 = 0.5;
    y2 = Length[list] - 0.5;
    Function[x, Evaluate@ (x y1 - x2 y1 - x y2 + x1 y2)/(x1 - x2)]]]];
```

```
xTI = TickInterpolation[Log[10, stiminput]];
yTI = TickInterpolation[i2range];
```

```
g = DisplayTogether[
   ListDensityPlot[peakᵀ, FrameLabel → {"L", "I2"}, Frame → True, Mesh → False, PlotRange → {All, All, {0, 1}},
    FrameTicks → {{xTI[#], 10.^#} & /@ Range[-12, -5, 2], {yTI[#], #} & /@ Range[.6, 1, .1], None, None}],
   ListContourPlot[peakᵀ, ContourShading → None, Contours → {responderThres}, ContourStyle → {Red, Thickness[.01]}]]];
```




```
Export["fig4d.density.gif", g];
```

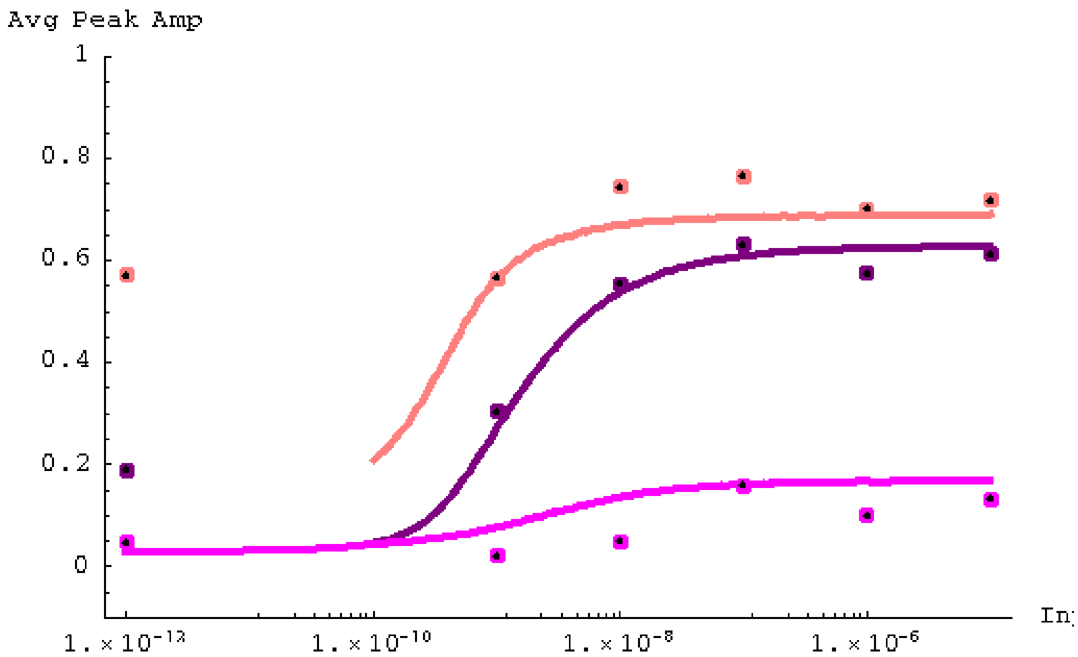
```
Export["fig4d.density.csv", peak];

expConc = 10^{-12, -9, -8, -7, -6, -5};
expPop = {1.1563, 1.2083, 1.3220, 1.3573, 1.3317, 1.3489};
expRes = {1.3300, 1.3270, 1.4090, 1.4180, 1.3890, 1.3970};
expNR = {1.0920, 1.0797, 1.0927, 1.1420, 1.1160, 1.1300};

avgResponder = Mean[Select[#, # > responderThres &]] & /@ peak /. Mean[{}] → Null;
avgNonResponder = Mean[Select[#, # < responderThres &]] & /@ peak;
avgPopulation = Mean /@ peak;

g = DisplayTogether[
   LogLinearListPlot[{stiminput, avgPopulation}ᵀ, PlotJoined → True,
    PlotStyle → {Thickness[.01], Purple}, PlotRange → {-0.1, 1}, AxesLabel → {"Input L", "Avg Peak Amp"}],
   LogLinearListPlot[{stiminput, avgResponder}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Pink}],
   LogLinearListPlot[{stiminput, avgNonResponder}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Magenta}],
   Block[{o = 1.07, s = 2.2}, {
     LogLinearListPlot[{expConc, (expRes - o) s}ᵀ, PlotStyle → #] & /@ {{Pink, PointSize[.02]}, {PointSize[.01]}},
     LogLinearListPlot[{expConc, (expNR - o) s}ᵀ, PlotStyle → #] & /@ {{Magenta, PointSize[.02]}, {PointSize[.01]}},
     LogLinearListPlot[{expConc, (expPop - o) s}ᵀ, PlotStyle → #] & /@ {{Purple, PointSize[.02]}, {PointSize[.01]}}
    }]
  ];
```

ScaledListPlot::sptn : Coordinate $\{1.\times10^{-12}, \mathrm{Null}\}$ is not a pair of numeric values.

ScaledListPlot::sptn : Coordinate $\{1.08437\times10^{-12}, \mathrm{Null}\}$ is not a pair of numeric values.

ScaledListPlot::sptn : Coordinate $\{1.17585\times10^{-12}, \mathrm{Null}\}$ is not a pair of numeric values.

General::stop : Further output of ScaledListPlot::sptn will be suppressed during this calculation. More…




```
Export["fig4d.gif", g]
```

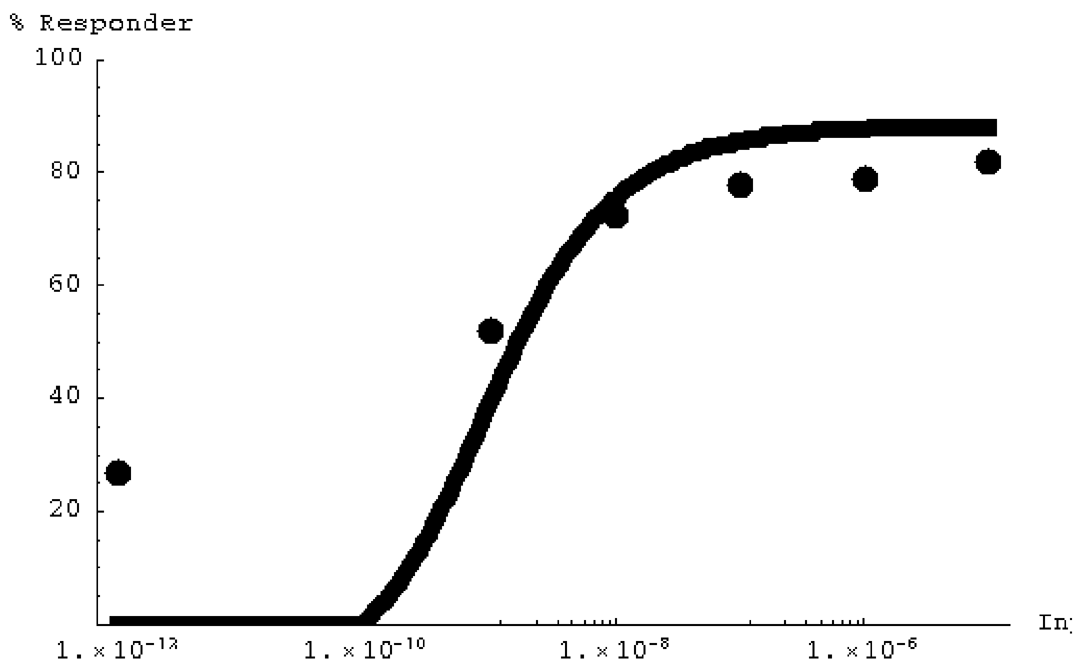
```
fig4d.gif
```

```
Export["fig4d.csv", {stiminput, avgResponder, avgNonResponder, avgPopulation}ᵀ]
```

```
fig4d.csv
```

```
rsims = PickResponders /@ sims;
dat = 100 * Length[rsims[[#]]] / Length[sims[[#]]] &~Array~Length[sims];
```

```
expPerResp = {27, 52, 72.5, 78, 79, 82};
DisplayTogether[
  LogLinearListPlot[{stiminput, dat}ᵀ, PlotJoined → True,
   PlotRange → {0, 100}, AxesLabel → {"Input L", "% Responder"}, PlotStyle → Thickness[0.02]],
  LogLinearListPlot[{expConc, expPerResp}ᵀ, PlotStyle → PointSize[0.03]]
 ];
```



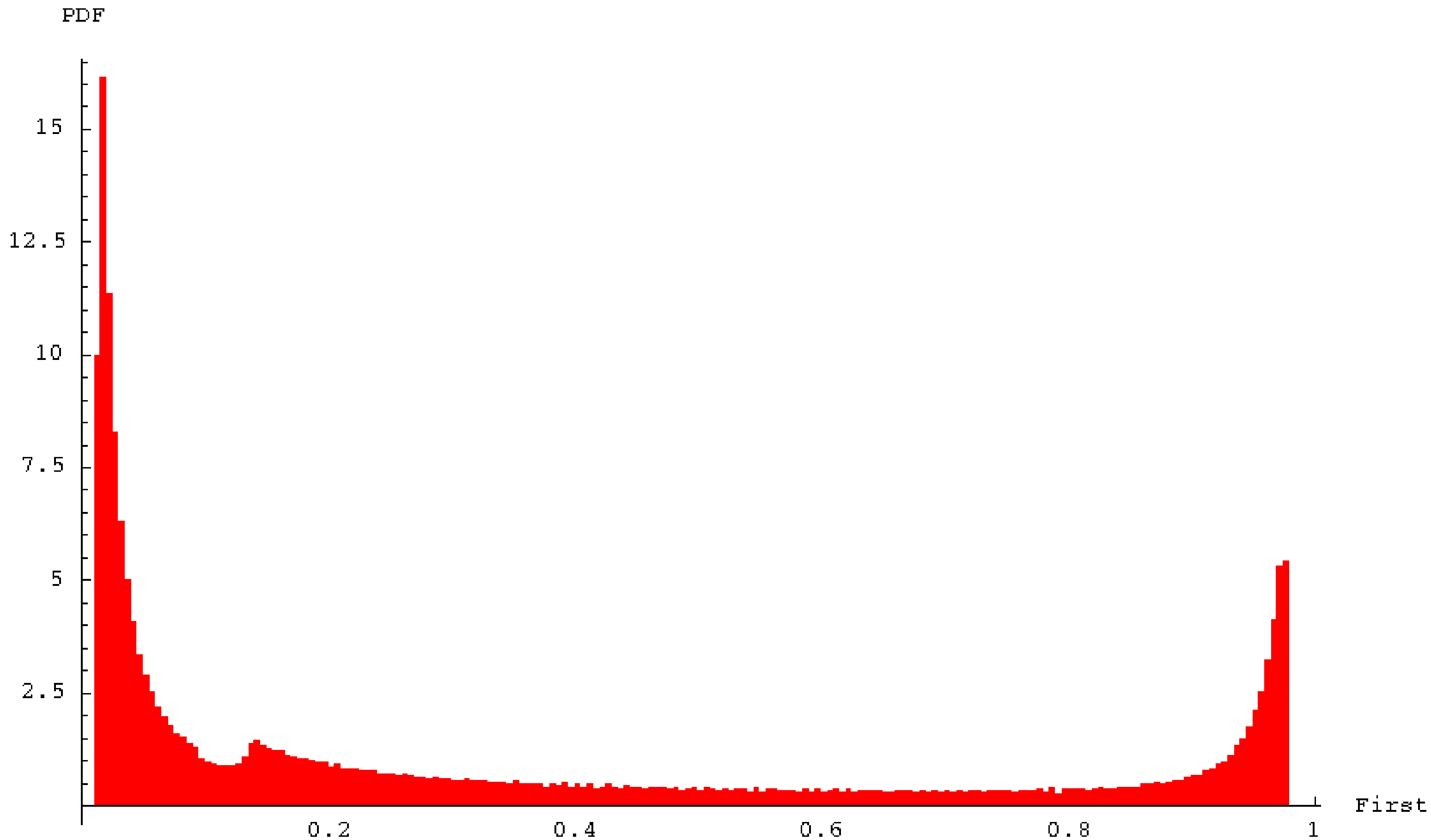
■ Bimodal Distribution

```
Dimensions[peak]
```

{200, 200}

```
SetOptions[Histogram, BarEdges → False, HistogramScale → 1, HistogramRange → {0, 1}, AxesLabel → {"First Peak", "PDF"}];
```

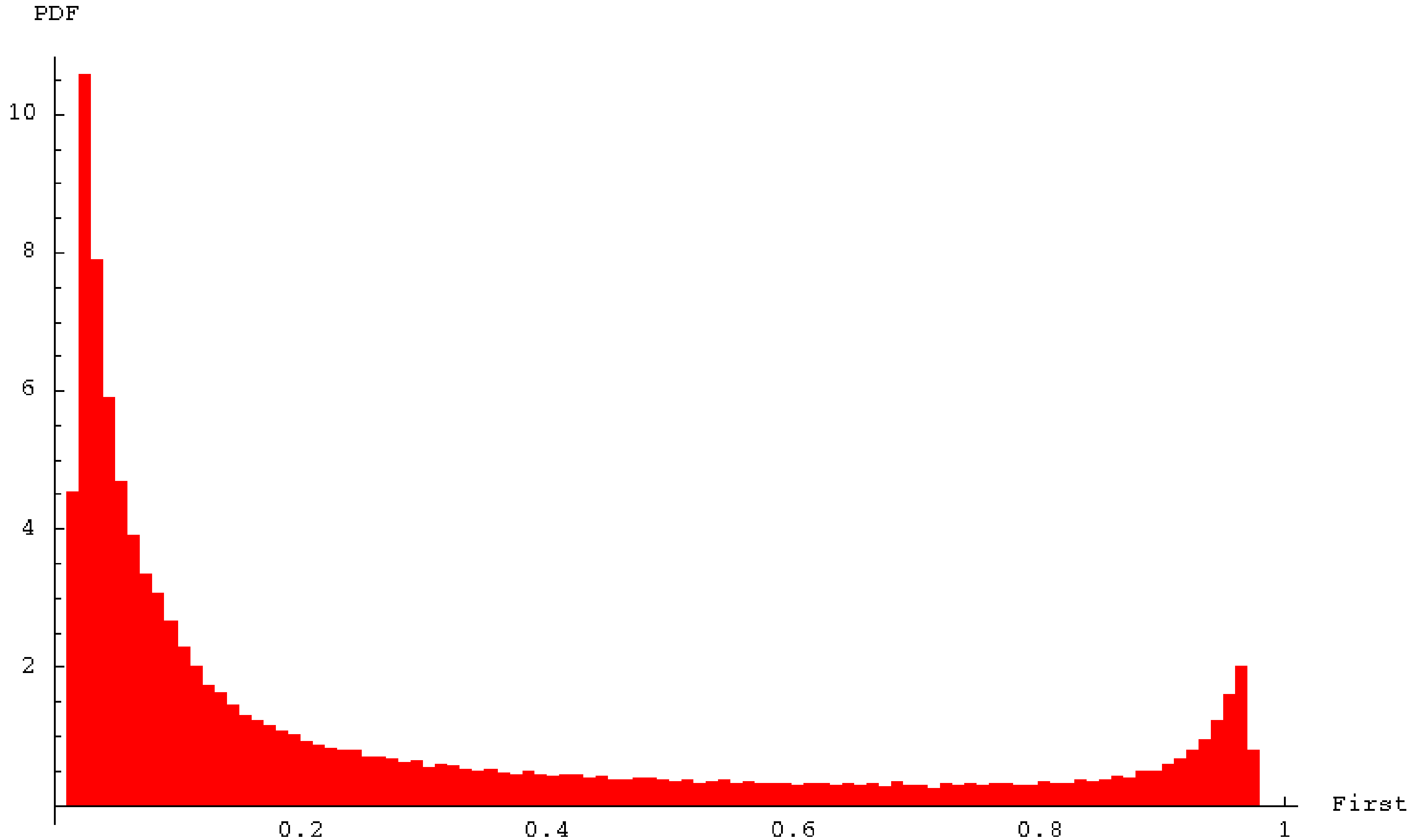
```
Histogram[Flatten[peak]];
```




```
Export["Hist.Entire.Range.csv", Flatten[peak]];
```

dat = Pick[peak, stiminput, _? (10^{-10} ≤ # ≤ 10^{-8} &)];

```
Histogram[Flatten[dat]];
```




```
Export["Hist.Dynamic.Range.csv", Flatten[peak]];
```

- **Supplemental Figure 1a**

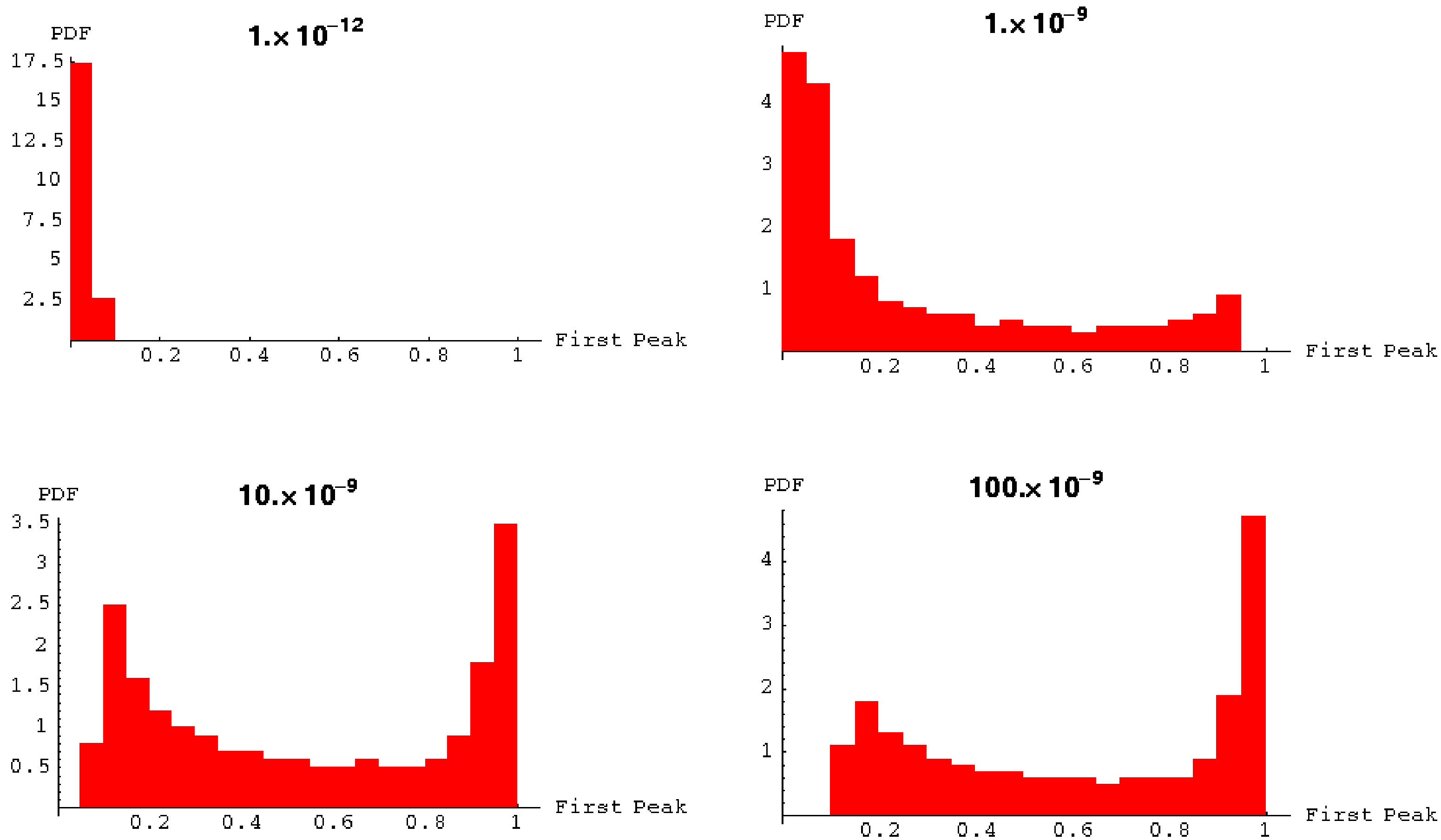
```
stiminput = 10^{-12, -9, -8, -7, -6, -5};
```

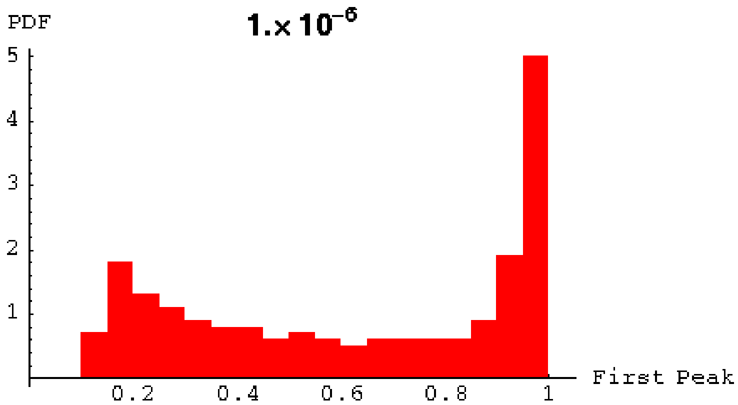
```
tend = 200;
sims = popsim[L → #] & /@ stiminput;
Dimensions[sims]
```

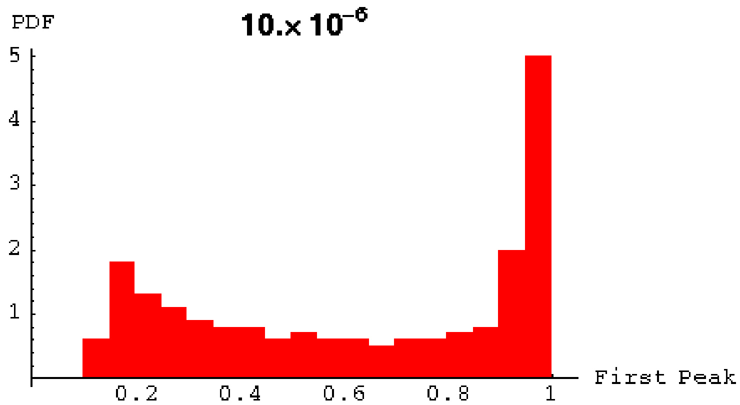
```
{6, 200}
```

```
peak = Map[Max[# /. t -> Range[0, tend, .1]] &, sims, {2}];
```

```
DisplayTogetherArray[MapThread[
    Histogram[#1, PlotLabel → StyleForm[EngineeringForm[N@#2], "Subtitle"]] &, {peak, stiminput}] // Partition[#, 2] &];
```





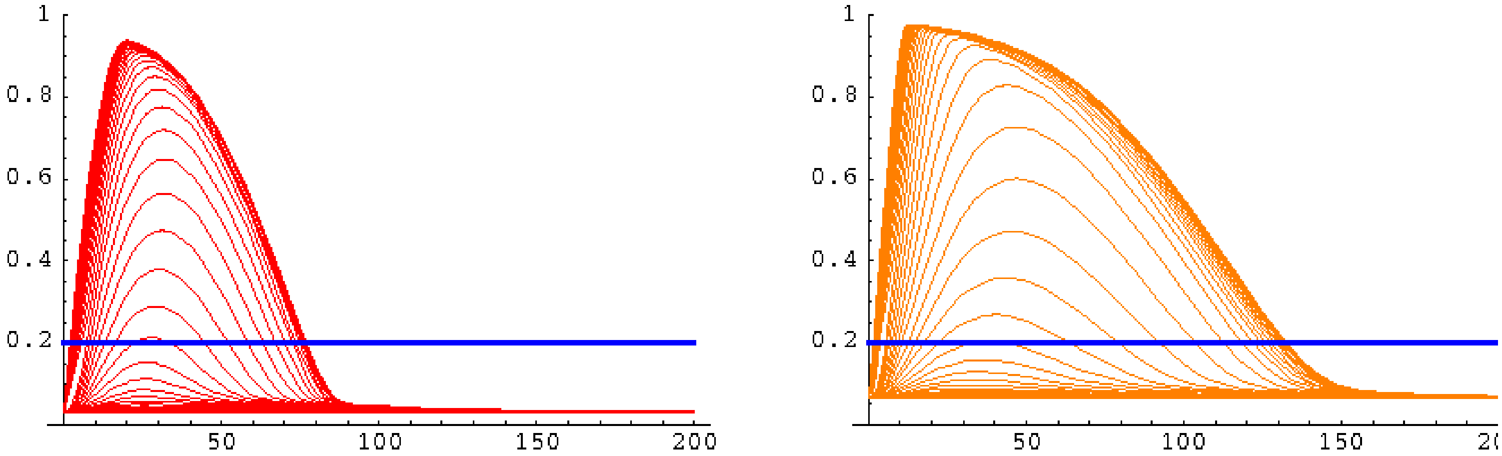



```
Export["Hist.Single.Conc.csv", Transpose[peak]];
```

Fig 4e. Adaptation times vs Dose

```
tend = 200;
stiminput = 10^RangeI[-12, -5, 61];

sims1 = amp[t] /. runsim[I2 → 0.65, L → #] & /@ stiminput // Flatten;
sims2 = amp[t] /. runsim[I2 → 0.57, L → #] & /@ stiminput // Flatten;
```

```
g = DisplayTogetherArray[Plot[#1 // Evaluate, {t, 0, tend}, PlotRange → {0, 1},
      Epilog → {Blue, Thickness[.01], Line[{{0, responderThres}, {200, responderThres}}]},
      PlotStyle → #2] & ~ MapThread ~ {{sims1, sims2}, {Red, Orange}}];
```

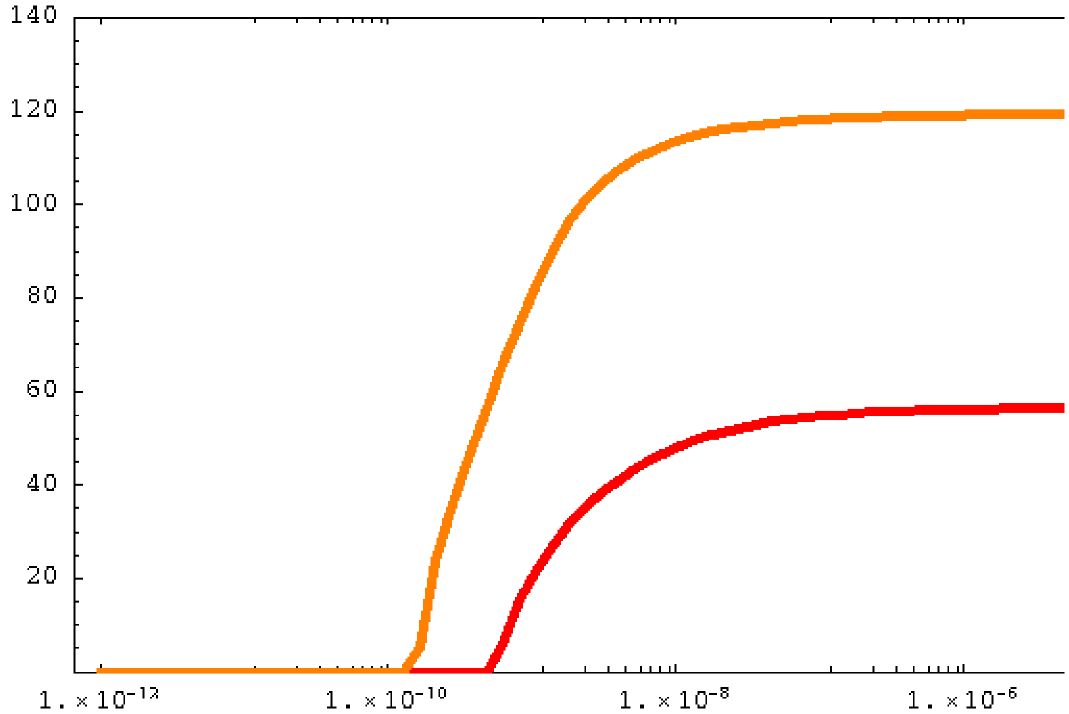



```
Export["fig4e.amps.gif", g]
```

fig4e.amps.gif

```
dt = .1;
ti = Range[t0, tend, dt];
respTime1 = UnitStep[sims1 - responderThres /. t → ti];
respTime2 = UnitStep[sims2 - responderThres /. t → ti];

peakTime1 = Ordering[# /. t → ti, -1] & /@ sims1 // Flatten;
peakTime2 = Ordering[# /. t → ti, -1] & /@ sims2 // Flatten;

dat1 = Function[{p, r}, dt * Total@Take[r, {p, -1}]] ~ MapThread ~ {peakTime1, respTime1};
dat2 = Function[{p, r}, dt * Total@Take[r, {p, -1}]] ~ MapThread ~ {peakTime2, respTime2};
```

```
g = DisplayTogether[
   LogLinearListPlot[{stiminput, dat1}ᵀ, PlotJoined → True,
    PlotStyle → {Red, Thickness[0.01]}, PlotRange → {0, 140}, Frame → True, Axes → False],
   LogLinearListPlot[{stiminput, dat2}ᵀ, PlotJoined → True, PlotStyle → {Orange, Thickness[0.01]}]];
```




```
Export["fig4e.gif", g]
```

fig4e.gif

```
dat = {N@stiminput, dat1, dat2}ᵀ;
```

```
Export["fig4e.csv", dat]
```

fig4e.csv

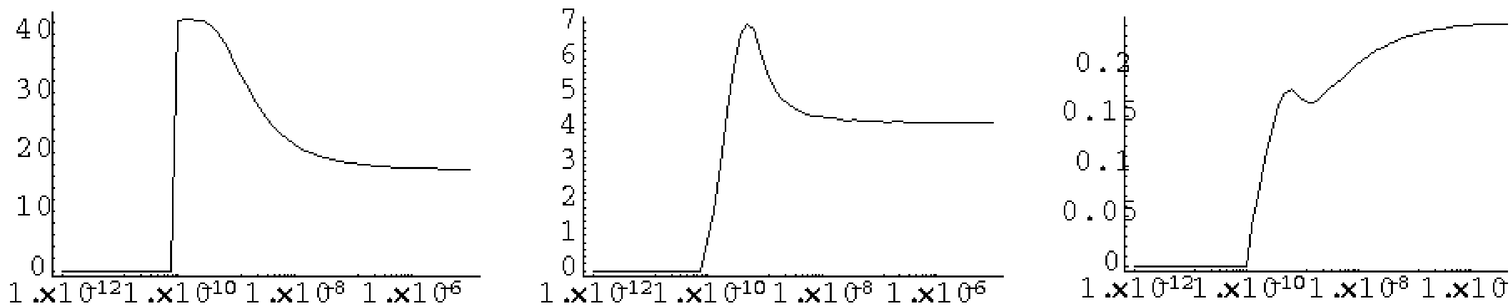
Fig4: Time to peak, Adaptation time

```
tend = 300;
stiminput = 10.^RangeI[-12, -5, 61];
sims = popsim[L → #] & /@ stiminput;
rsims = PickResponders /@ sims;
dt = 0.1;
ti = Range[t0, tend, dt];
peaktime = Function[sim,
    ti[[First@Ordering[# /. t -> ti, -1]]] & /@ sim // Flatten
   ] /@ rsims;
meanpeaktime = Mean /@ peaktime;
stdpeaktime = StandardDeviation /@ peaktime;
```

StandardDeviation::shlen : The argument {42.2} should have at least two elements. More…

```
covPeakTime = Function[{m, s}, If[s == 0 || Not@NumberQ[s], 0, s / m]] ~MapThread~ {meanpeaktime, stdpeaktime};
DisplayTogetherArray[LogLinearListPlot[{stiminput, #}ᵀ, PlotJoined → True] & /@ {meanpeaktime, stdpeaktime, covPeakTime}];
```

ScaledListPlot::sptn : Coordinate {9.62351×10^{-11}, StandardDeviation[{42.2}]} is not a pair of numeric values.




```
rsims[[18]]
```

{InterpolatingFunction[{{0., 300.}}, <>][t]}

```
adapttime = Function[sim, Module[{p, r, s},
      s = sim /. t → ti;
      If[sim == {}, Return[0]];
      r = UnitStep[# - responderThres] & /@ s;
      p = Ordering[#, -1] & /@ s // Flatten;
      dt * Total@Take[#1, {#2, -1}] &~MapThread~{r, p}
     ]
   ] /@ rsims;
```

```
Dimensions[adapttime]
```

```
{61}
```

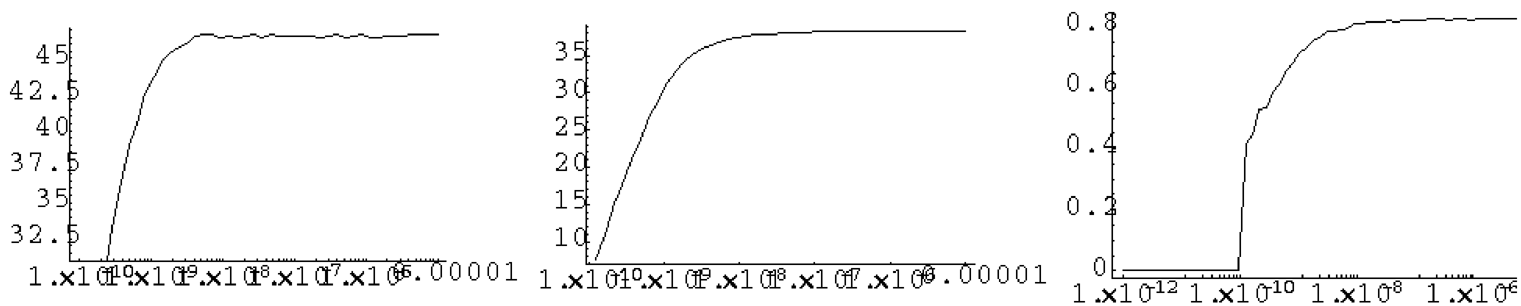
```
meanadapttime = Mean /@ adapttime;
stdadapttime = StandardDeviation /@ adapttime;
```

```
StandardDeviation::shlen : The argument {7.7} should have at least two elements. More…
```

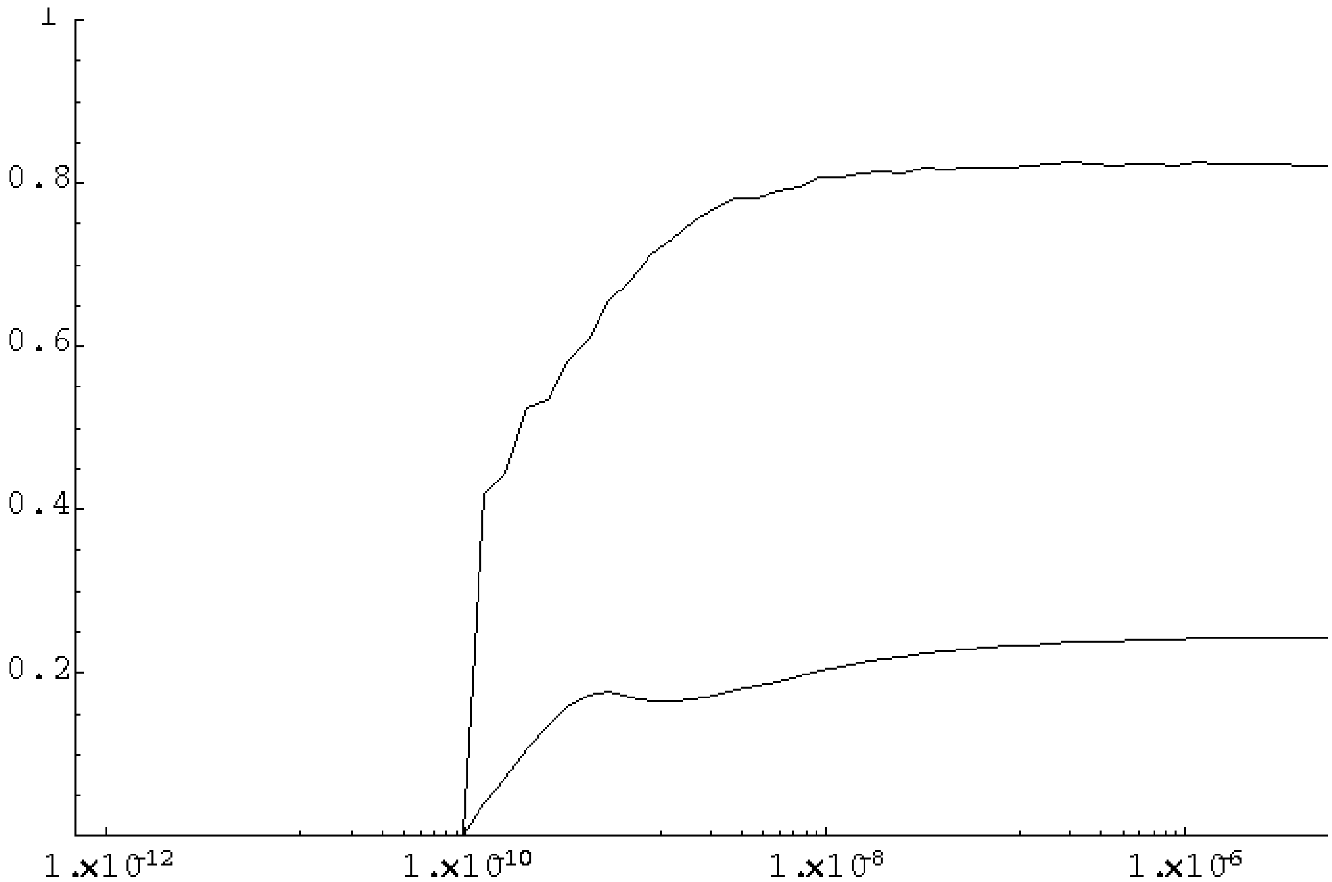
```
covAdaptTime = Function[{m, s}, If[s == 0 || Not@NumberQ[s], 0, s / m]]~MapThread~{meanadapttime, stdadapttime};
```

```
DisplayTogetherArray[LogLinearListPlot[{stiminput, #}ᵀ, PlotJoined → True] & /@ {meanadapttime, stdadapttime, covAdaptTime}];
```

```
ScaledListPlot::sptn : Coordinate {1.×10^-12, Mean[Return[0]]} is not a pair of numeric values.

ScaledListPlot::sptn : Coordinate {1.30818×10^-12, Mean[Return[0]]} is not a pair of numeric values.

ScaledListPlot::sptn : Coordinate {1.71133×10^-12, Mean[Return[0]]} is not a pair of numeric values.

General::stop : Further output of ScaledListPlot::sptn will be suppressed during this calculation. More…
```



```
g = DisplayTogether[LogLinearListPlot[{stiminput, #}ᵀ, PlotJoined → True, PlotRange → {0, 1}] & /@ {covPeakTime, covAdaptTime}]
```




```
- Graphics -
```

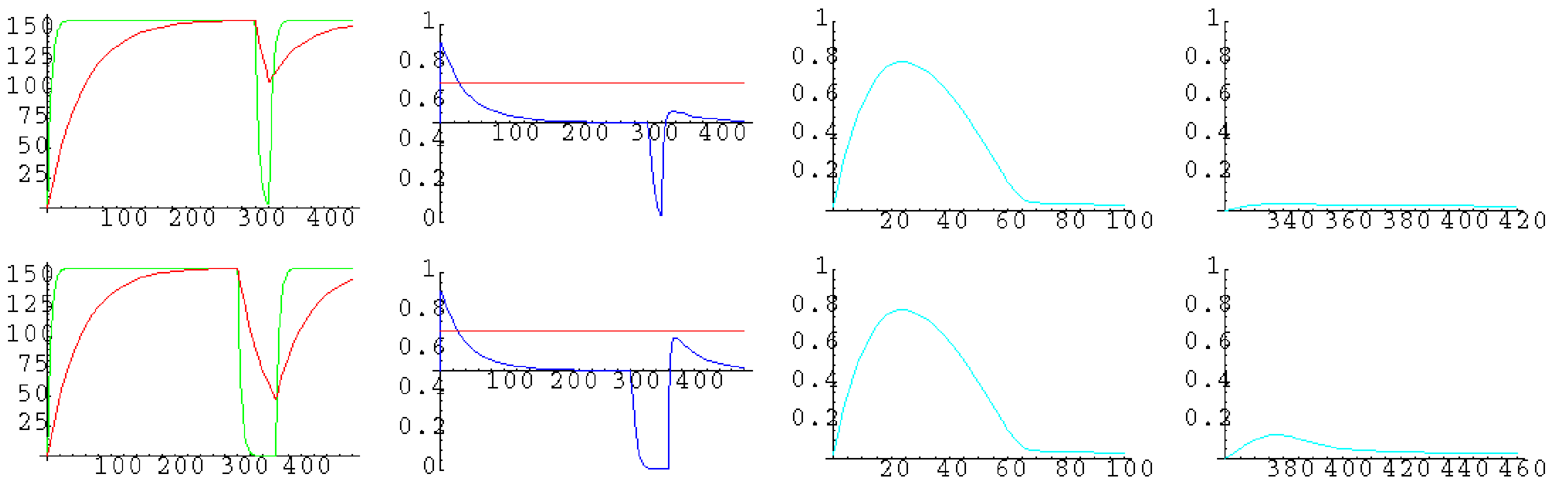
```
dat = {stiminput, covPeakTime, covAdaptTime} // Transpose;
Export["fig4-cv.peak.adapt.csv", dat]
Export["fig4-cv.peak.adapt.gif", g]
```

```
fig4-cv.peak.adapt.csv
```

```
fig4-cv.peak.adapt.gif
```

Varying wash time, single cell

```
Block[{TickFontSize = 15},
 Function[w,
    wash = w * 60;
    tend = 5 * 60 + wash + 2 * 60;
    sim = runsim[L → 10^-6 (UnitStep[t, 5 * 60 - t] + UnitStep[t - (wash + 5 * 60)])];
    DisplayTogetherArray[
     Plot[{a[t], i[t]} /. sim // Evaluate, {t, t0, tend}, PlotRange → All, PlotStyle → {Green, Red}],
     Plot[{r[t], I2 /. params} /. sim // Evaluate,
      {t, t0, tend}, PlotRange → {All, {0, 1}}, AxesOrigin → {0, 0.5}, PlotStyle → {Blue, Red}],
     Plot[amp[t] /. sim, {t, 0, 100}, PlotRange → {0, 1}, PlotStyle → {Cyan}],
     Plot[amp[t] /. sim, {t, wash + 5 * 60, wash + 5 * 60 + 100}, PlotRange → {0, 1}, PlotStyle → {Cyan}]
    ]] /@ {.33, 1, 2, 4};
]
```



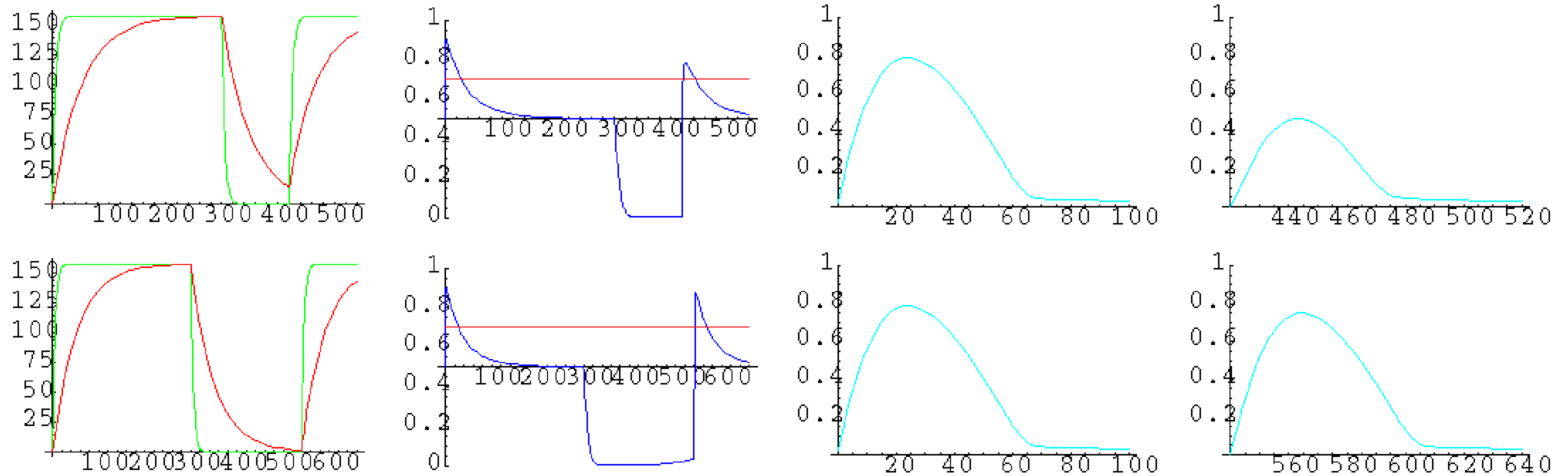
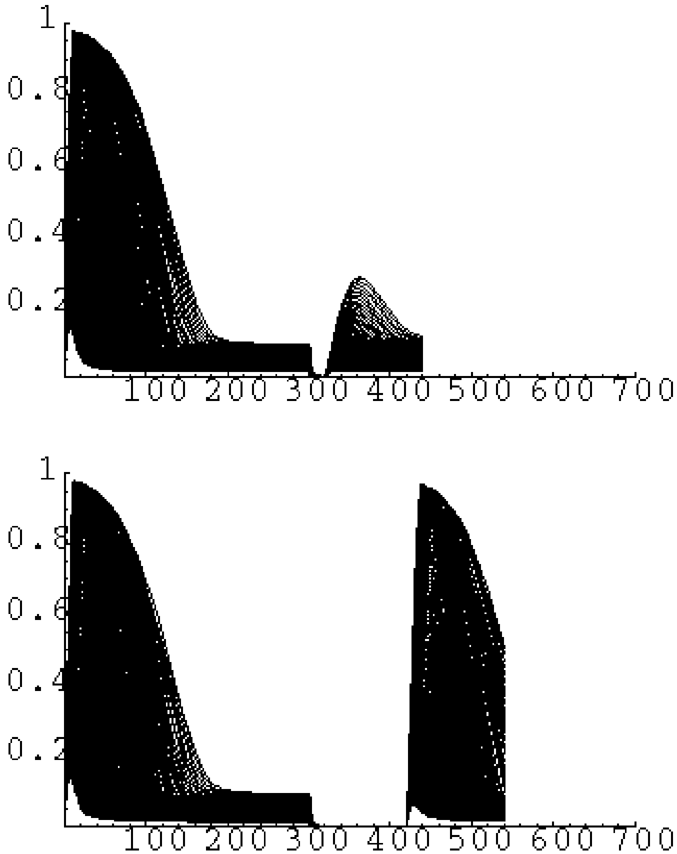


Fig 5a: Stim, variable wash, stim

```
washtimes = {20 / 60, 1, 2, 4};

SWS = Function[w,
   wash = w * 60;
   tend = 5 * 60 + wash + 2 * 60;
   {tend, popsim[L → 10^-6 (UnitStep[t, 5 * 60 - t] + UnitStep[t - (wash + 5 * 60)])]}
  ];

{endTimes, sims} = Transpose[SWS /@ washtimes];
```

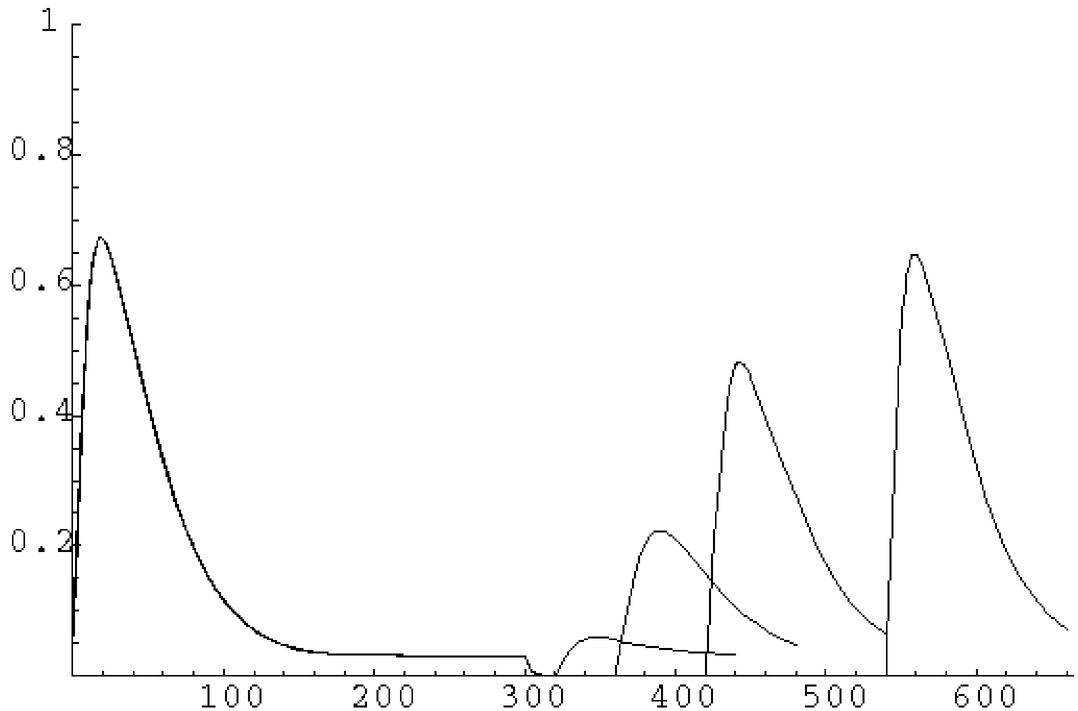
```
DisplayTogetherArray[
  Plot[sims[[#]] // Evaluate, {t, t0, endTimes[[#]]}, PlotRange → {{0, 700}, {0, 1}}] & /@ Range[4] // Partition[#, 2] &];
```



- **Responder Average**

```
rsims = PickResponders /@ sims;
mrsims = Mean /@ rsims;
```

```
DisplayTogether[
  Plot[mrsims[[#]] // Evaluate, {t, t0, endTimes[[#]]}, PlotRange → {{0, 700}, {0, 1}}] & /@ Range[4] // Partition[#, 2] &];
```

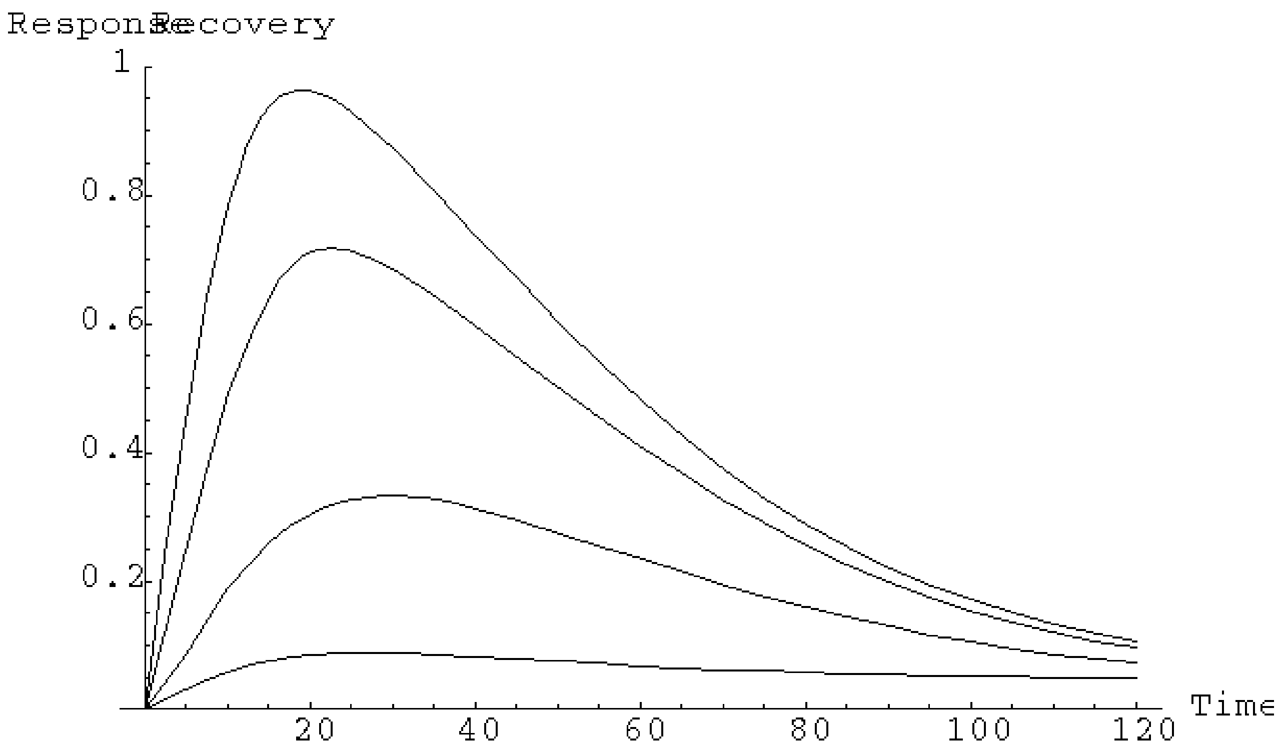



```
maxResponse = Max[mrsims[[1]] /. t → Range[0, 100, .1]]
```

0.67231

$$\text{dat} = \frac{\text{mrsims[[\#]] /. t} \to \text{s} + (\text{endTimes[[\#]]} - 2 * 60)}{\text{maxResponse}} \text{ \& /@ Range[4];}$$

```
g = DisplayTogether[Plot[dat[[#]] // Evaluate, {s, 0, 2 * 60}, PlotRange → {0, 1}] & /@ Range[4],
   AxesLabel → {"Time (sec)", "Response Recovery"}];
```




```
ti = Range[0, 120, .1];
Export["fig5-SWS.response-recovery.gif", g];
Export["fig5-SWS.response-recovery.csv", Transpose[Prepend[dat, s] /. s → ti]];
```

- **Finer wash time discretization**

```
washtimes = Range[.2, 10, .1];
```

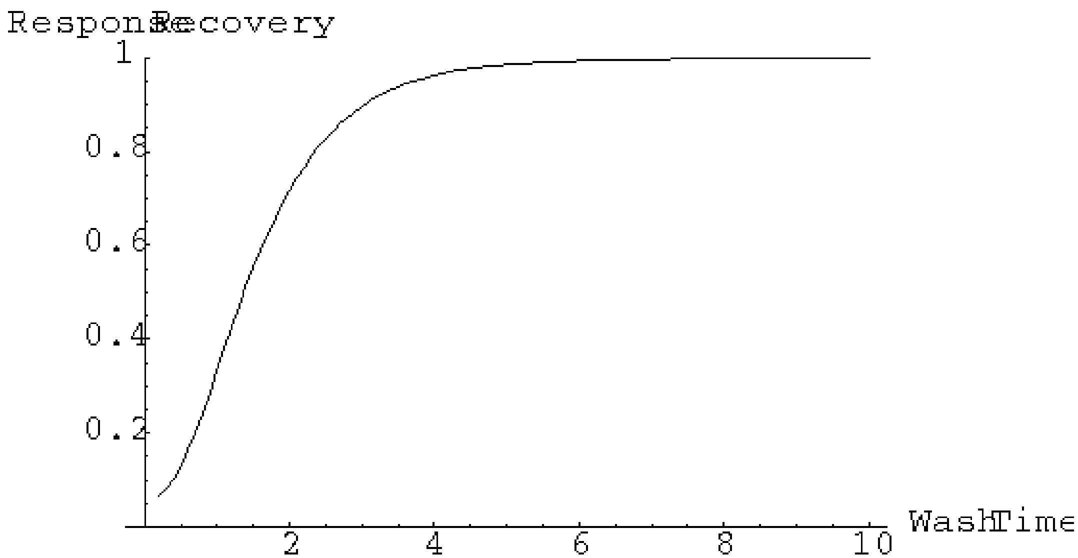
```
{endTimes, sims} = Transpose[SWS /@ washtimes];
```

```
rsims = PickResponders /@ sims;
```

```
dat = Function[s,
    1/maxResponse Max[Mean[rsims[[s]] /. t → Range[endTimes[[s]] - 120, endTimes[[s]], .1]]]
  ] ~Array~ Length[sims]
```

{0.0658512, 0.0820028, 0.103281, 0.130122, 0.16247, 0.199816, 0.2413, 0.28583, 0.332177, 0.379068, 0.425314,
0.469985, 0.512523, 0.552672, 0.590346, 0.625549, 0.658326, 0.688751, 0.71691, 0.742904, 0.766836, 0.788813,
0.808949, 0.827349, 0.844128, 0.85939, 0.873245, 0.885795, 0.897138, 0.907379, 0.916605, 0.924903, 0.932359,
0.939048, 0.945042, 0.950413, 0.955216, 0.959511, 0.963348, 0.966776, 0.969832, 0.972566, 0.975004, 0.977178,
0.979119, 0.980856, 0.982406, 0.983792, 0.985032, 0.986143, 0.987143, 0.988042, 0.988853, 0.989586, 0.99025,
0.990854, 0.991406, 0.991912, 0.992378, 0.992809, 0.99321, 0.993585, 0.993939, 0.994273, 0.99459, 0.994892,
0.995181, 0.995459, 0.995726, 0.995984, 0.996232, 0.996472, 0.996704, 0.996927, 0.997142, 0.997348, 0.997545,
0.997734, 0.997914, 0.998085, 0.998246, 0.998398, 0.99854, 0.998672, 0.998795, 0.998908, 0.999013, 0.999109,
0.999197, 0.999278, 0.999351, 0.999418, 0.999478, 0.999533, 0.999582, 0.999627, 0.999666, 0.999702, 0.999735}

```
g = ListPlot[{washtimes, dat}ᵀ, PlotJoined → True, AxesLabel → {"Wash Time (min)", "Response Recovery"}, PlotRange → {0, 1}];
```




```
Export["fig5a-SWS.gif", g]
```

fig5a-SWS.gif

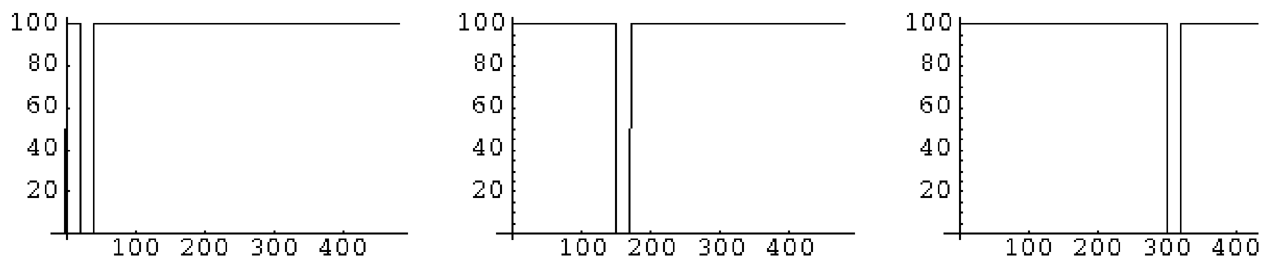
```
Export["fig5a-SWS.csv", {washtimes, dat}ᵀ]
```

fig5a-SWS.csv

Fig 5b: Variable Stim, wash, Stim

```
tend = 8 * 60;
washtime = 20;
VarStim = Function[{stim},
   100 (UnitStep[t, stim - t] + UnitStep[t - (stim + washtime)])
  ];
```

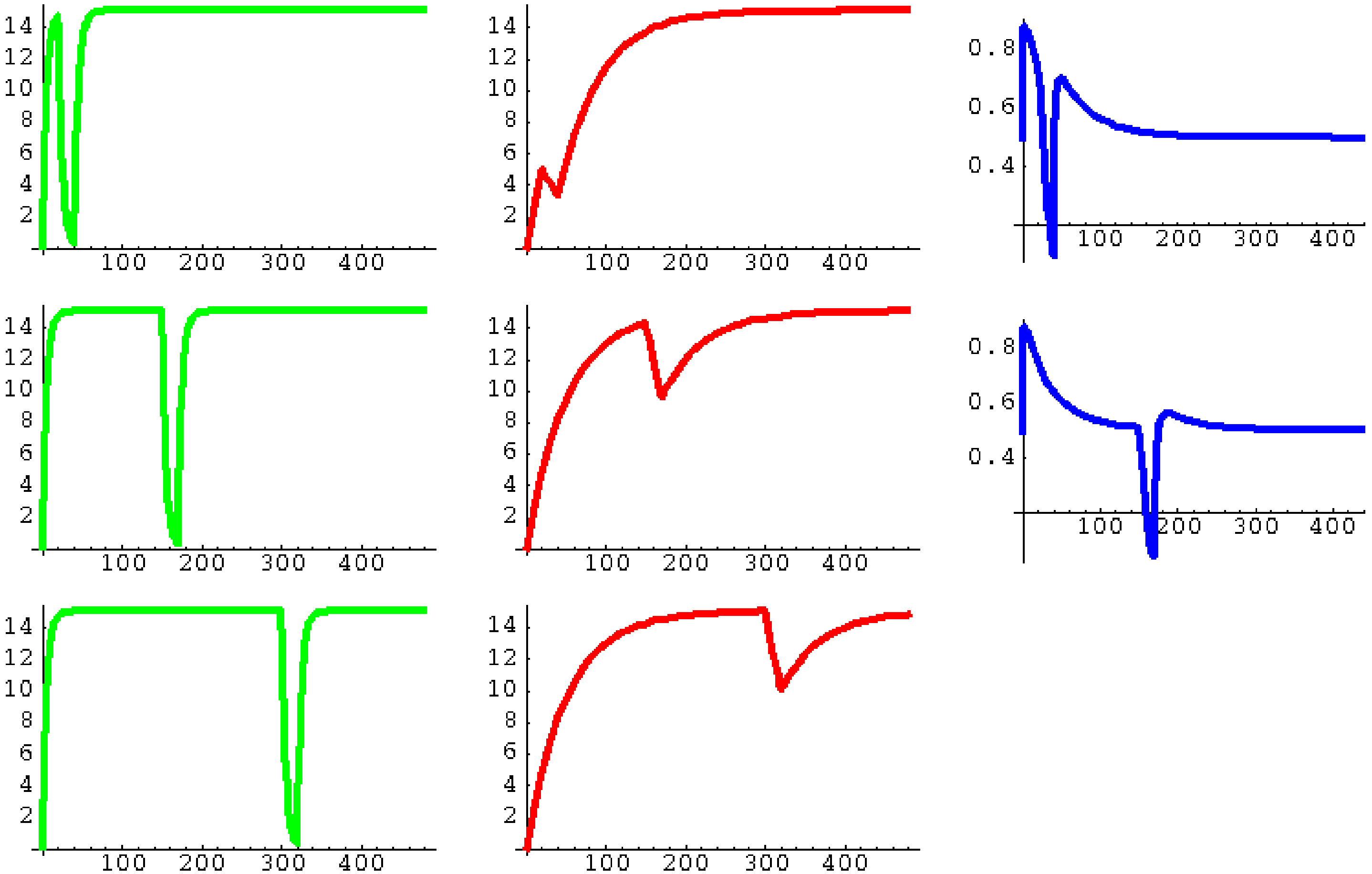
```
stimtimes = {20, 2.5 * 60, 5 * 60};
DisplayTogetherArray[Plot[VarStim[#], {t, -10, tend}] & /@ stimtimes];
```




```
sim
```

```
{{a → InterpolatingFunction[{{0., 480.}}, <>], i → InterpolatingFunction[{{0., 480.}}, <>],
  r → InterpolatingFunction[{{0., 480.}}, <>], amp → InterpolatingFunction[{{0., 480.}}, <>]},
 {a → InterpolatingFunction[{{0., 480.}}, <>], i → InterpolatingFunction[{{0., 480.}}, <>],
  r → InterpolatingFunction[{{0., 480.}}, <>], amp → InterpolatingFunction[{{0., 480.}}, <>]},
 {a → InterpolatingFunction[{{0., 480.}}, <>], i → InterpolatingFunction[{{0., 480.}}, <>],
  r → InterpolatingFunction[{{0., 480.}}, <>], amp → InterpolatingFunction[{{0., 480.}}, <>]}}
```

```
sim = runsim[L → VarStim[#]] & /@ stimtimes;
g = Function[s, DisplayTogetherArray[
     Plot[#1 /. s, {t, t0, tend}, PlotRange → All, PlotStyle → {Thickness[.02], #2}] &~
      MapThread~{{a[t], i[t], r[t]}, {Green, Red, Blue}}]] /@ sim;
```

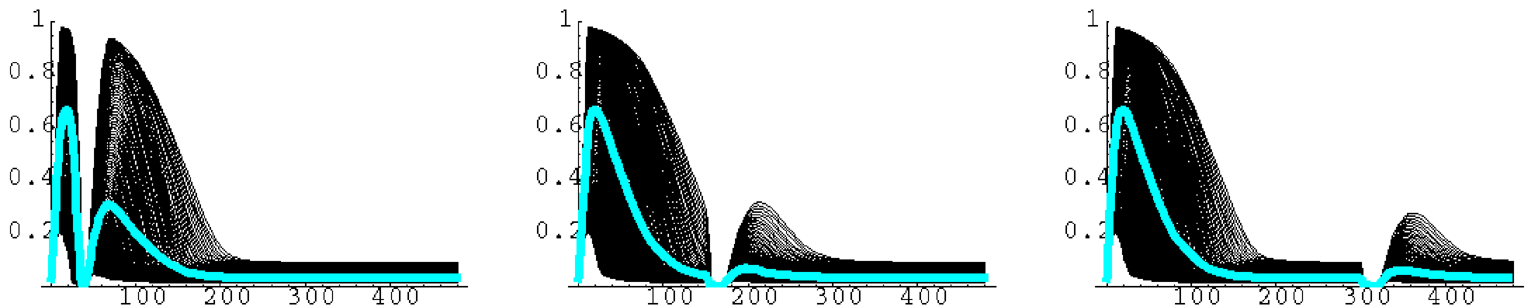



```
ti = Range[t0, tend, .5];

Function[s,
  dat = {t, a[t], i[t], r[t]} /. sim[[s]] /. t → ti;
  Export["fig5b-vws." <> ToString[s] <> ".gif", g[[s]]];
  Export["fig5b-vws." <> ToString[s] <> ".csv", datᵀ];
 ] /@ Range[3]
```

```
{Null, Null, Null}
```

```
sims = popsim[L → VarStim[#]] & /@ stimtimes;
rsims = PickResponders /@ sims;
```

```
DisplayTogetherArray[
 g = DisplayTogether[Plot[# // Evaluate, {t, t0, tend}, PlotRange → {0, 1}],
     Plot[Mean[#] // Evaluate, {t, t0, tend}, PlotStyle → {Thickness[.02], Cyan}]] & /@ rsims
]
```




```
Function[s,
  dat = Prepend[rsims[[s]] /. t → ti, ti];
  Export["fig5b-vws.amp." <> ToString[s] <> ".gif", g[[s]]];
  Export["fig5b-vws.amp." <> ToString[s] <> ".csv", dat^T];

  dat = Mean[rsims[[s]] /. t → ti];
  dat = {ti, dat};
  Export["fig5b-vws.amp.mean." <> ToString[s] <> ".csv", dat^T]
 ] /@ Range[3]
```

```
{fig5b-vws.amp.mean.1.csv, fig5b-vws.amp.mean.2.csv, fig5b-vws.amp.mean.3.csv}
```

```
Array[Function[i,
  Count[Function[r, Max[r /. t → Range[stimtimes[[i]], stimtimes[[i]] + 100, .1]]] /@ rsims[[i]],
   _?(# > responderThres &)]],
 Length[stimtimes]]
```

```
{156, 13, 9}
```

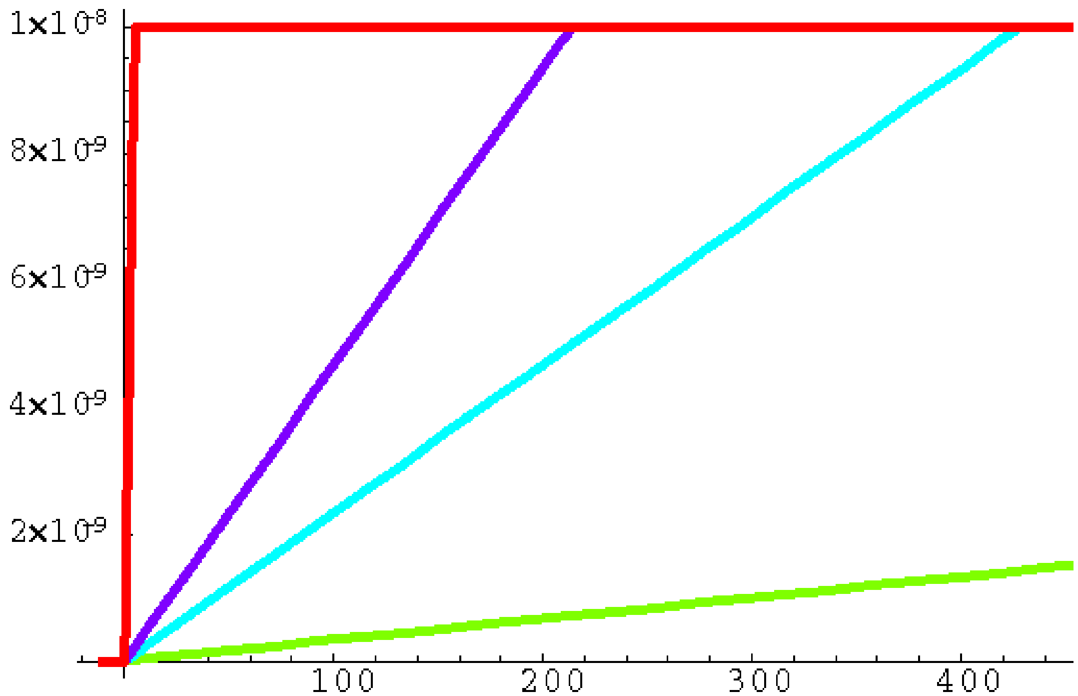
```
Length /@ rsims
```

```
{171, 171, 171}
```

```
stimtimes
{20, 150., 300}
```

Fig 5c: Varying Ramp Speed

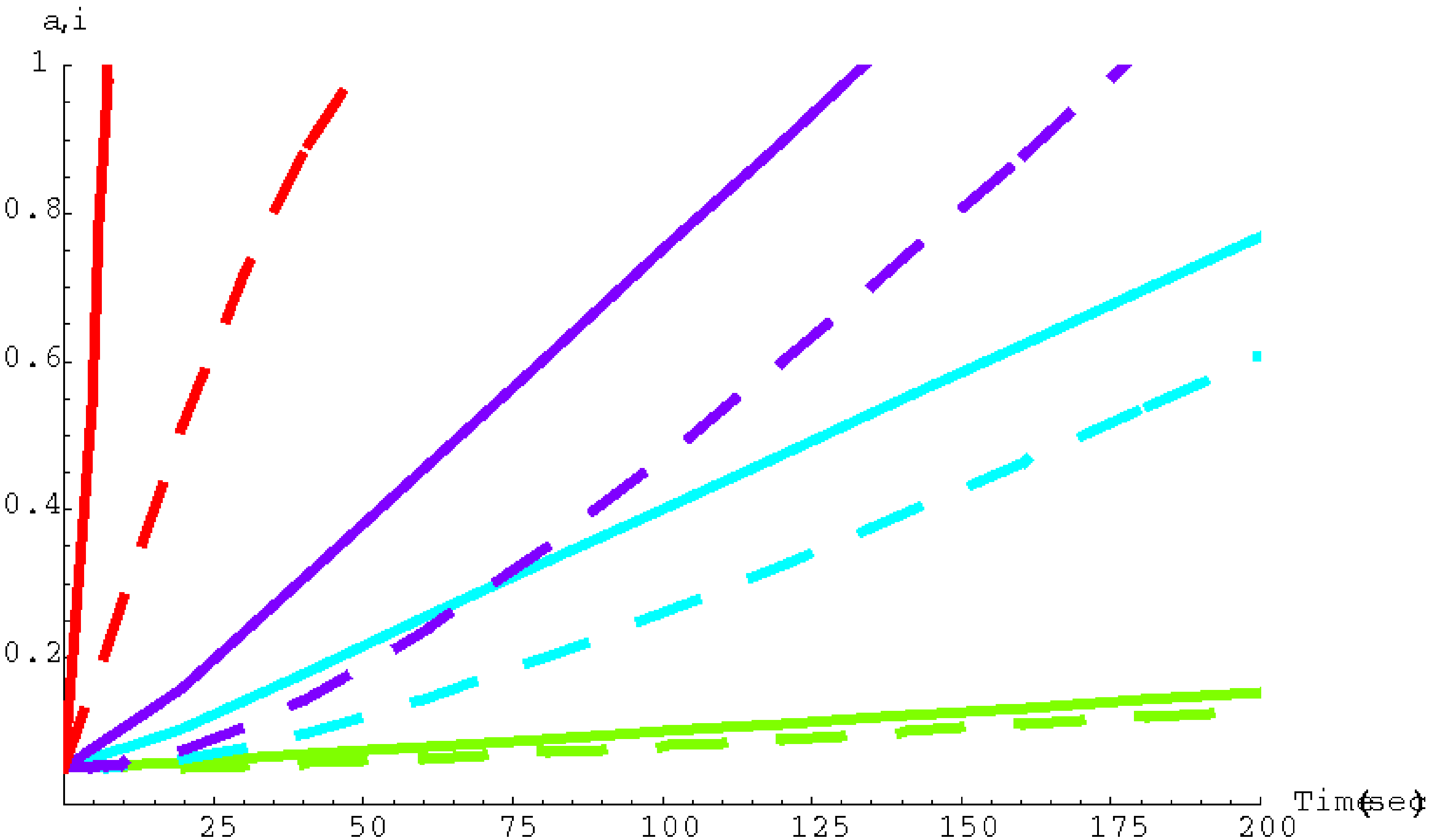
```
tend = 8 * 60;
tq = 60 * 2;
maxC = 10. * 10^-9;
rampSpeeds = {0.2, 1.4, 2.8, 120} * 10^-9 / 60;
inputs = L -> Clip[# * t, {0, maxC}] & /@ rampSpeeds;
rampSpeeds = rampSpeeds * 60 * 10^9;
rampSpeeds = rampSpeeds * (60 * 10^9);
pstyle = {Thickness[.01], Hue[# / Length[inputs]]} &;
g = DisplayTogether[
   Plot[Evaluate@inputs[[#, 2]], {t, -10, tend}, PlotRange → All, PlotStyle → pstyle[#]] &~Array~Length[inputs]];
```



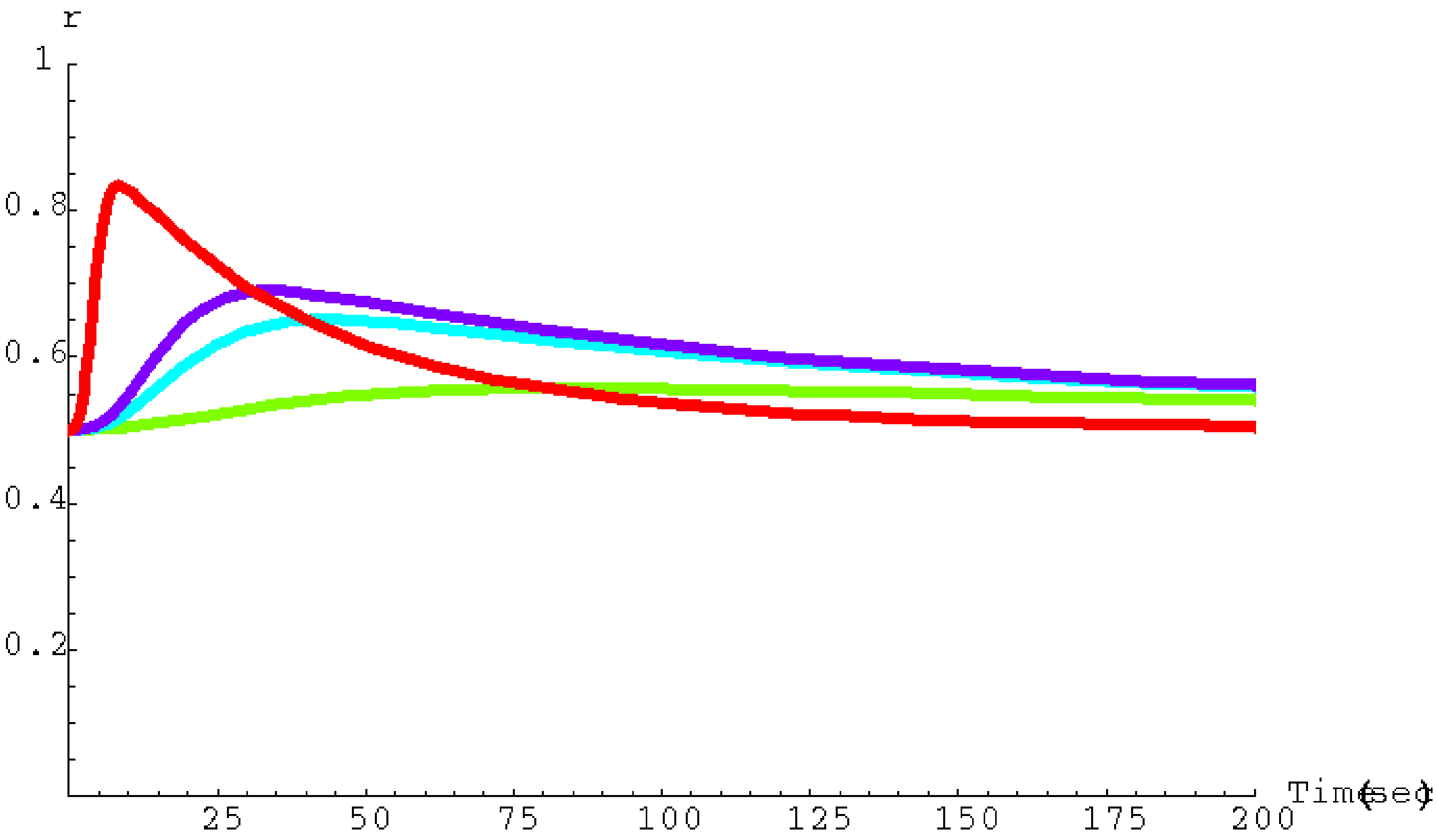

```
ti = Range[t0, tend];
```

```
dat = L /. {inputs}ᵀ /. t → ti;
Export["fig5.fiveRamps.csv", datᵀ];
Export["fig5.fiveRamps.gif", g];
```

```
sol = runsim[#] & /@ inputs;
DisplayTogether[
  Table[{
    Plot[a[t] /. sol[[s]] // Evaluate, {t, t0, tend}, PlotRange → {{t0, 200}, {0, 1}}, PlotStyle → pstyle[s]],
    Plot[i[t] /. sol[[s]] // Evaluate, {t, t0, tend}, PlotRange → {{t0, 200}, {0, 1}},
     PlotStyle → Append[pstyle[s], Dashing[{.05, .05}]]]}, {s, Length[inputs]}], AxesLabel → {"Time(sec)", "a,i"}];
```

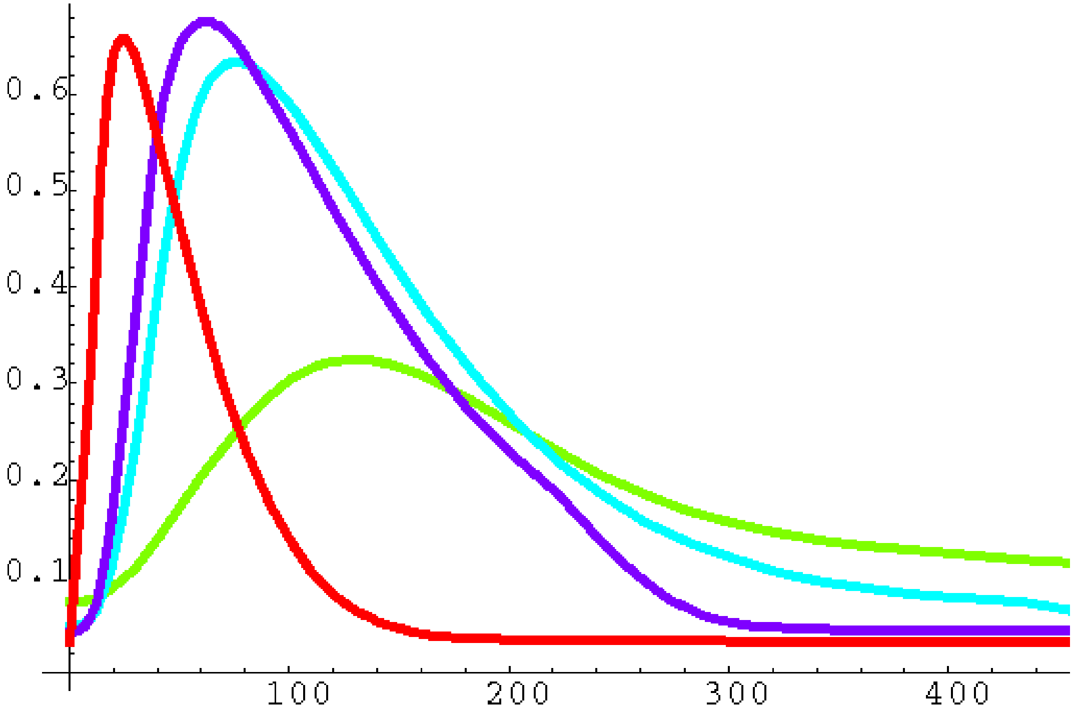


```
DisplayTogether[
  Table[{
    Plot[r[t] /. sol[[s]] // Evaluate, {t, t0, tend}, PlotRange → {{t0, 200}, {0, 1}}, PlotStyle → pstyle[s]]
   }, {s, Length[inputs]}], AxesLabel → {"Time(sec)", "r"}];
```




```
sims = popsim /@ inputs;
rsims = PickResponders /@ sims;
```

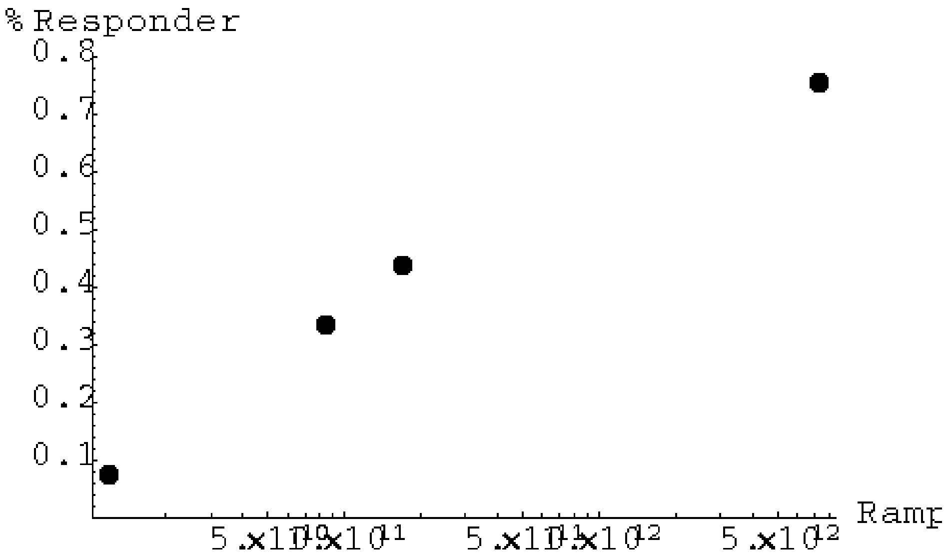
```
g = DisplayTogether[Plot[Mean[rsims[[#]]], {t, 0, tend}, PlotStyle → pstyle[#]] &~Array~Length[inputs]];
```




```
dat = Mean /@ rsims /. t → ti;
Export["fig5.fiveRamps.amps.csv", datᵀ];
Export["fig5.fiveRamps.amps.gif", g];
```

$$\text{dat} = \frac{\text{Length[rsims[[\#]]]}}{\text{Length[sims[[\#]]]}} \;\&\sim\text{Array}\sim\text{Length[sims]};$$

```
LogLinearListPlot[{rampSpeeds, dat}ᵀ, PlotStyle → PointSize[.03],
  PlotRange → {0, 0.8}, AxesLabel → {"Ramp Speed", "% Responder"}];
```

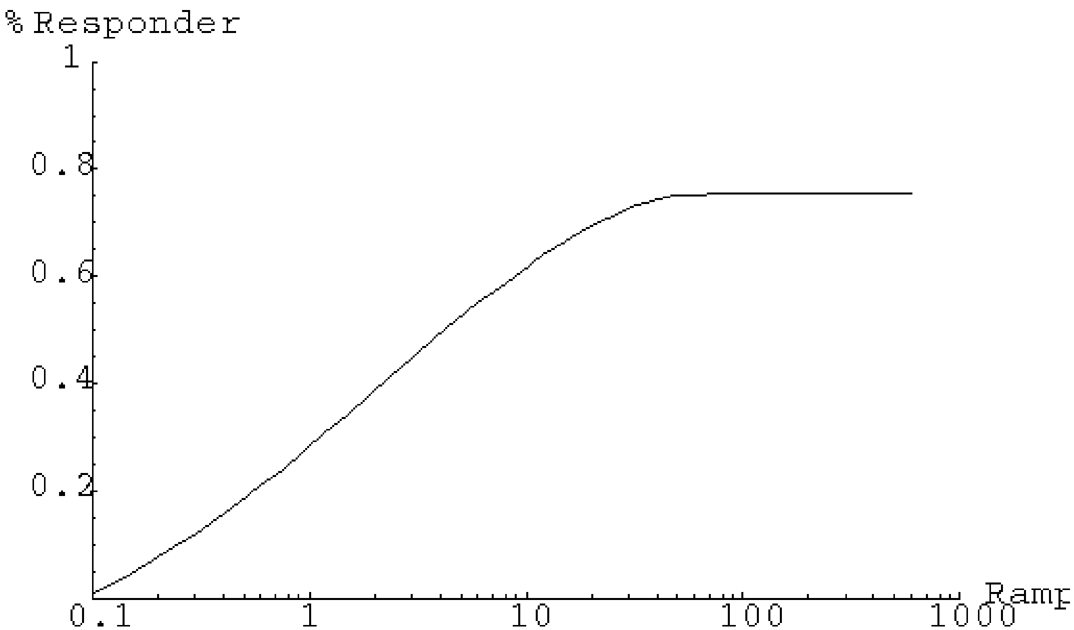


Finer

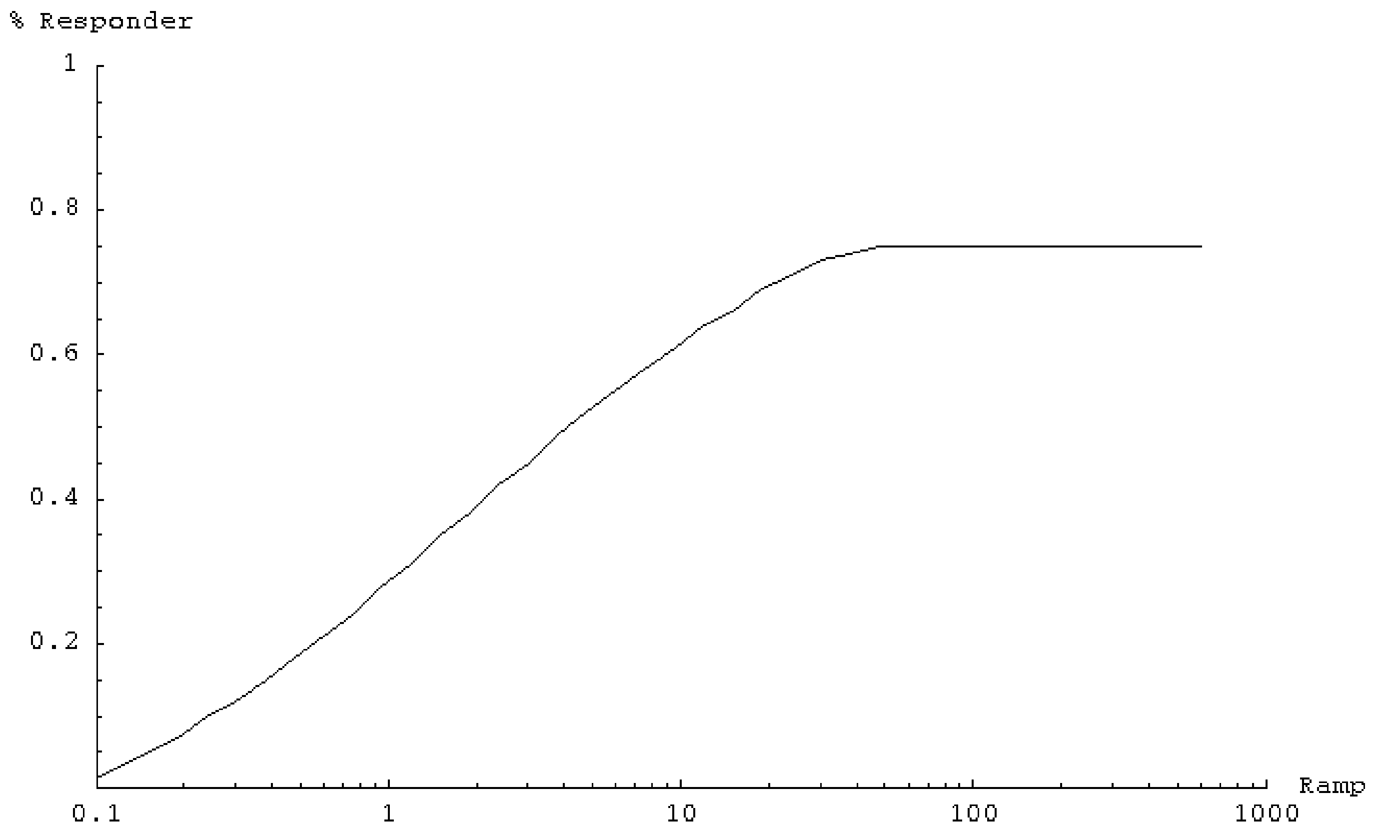
```
rampSpeeds = 10^Range[-3., 1, .1] * 10^-9 // N;
inputs = L -> Clip[# * t, {0., maxC}] & /@ rampSpeeds;
rampSpeeds = rampSpeeds * (60 * 10^9);
```

```
sims = popsim /@ inputs;
rsims = PickResponders /@ sims;
dat = Length[rsims[[#]]] / Length[sims[[#]]] &~Array~Length[sims];
```

```
g = LogLinearListPlot[{rampSpeeds, dat}ᵀ, PlotJoined → True,
   PlotRange → {{10^-1, 10^3}, {0, 1}}, AxesLabel → {"Ramp Speed", "% Responder"}];
```



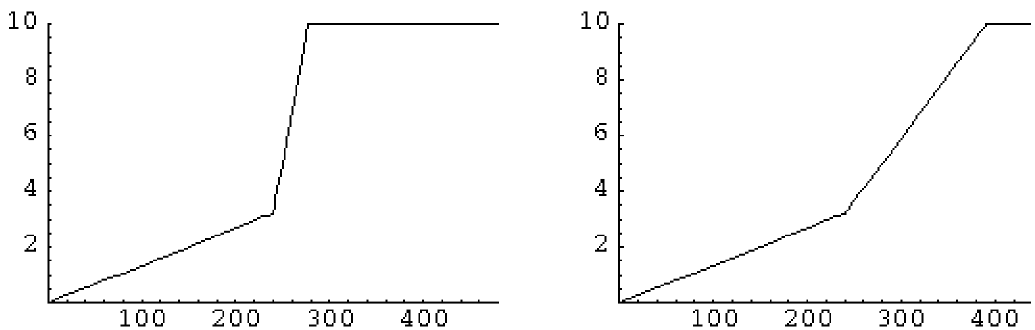
q




```
Export["fig5cRampSpeed.gif", g];
Export["fig5cRampSpeed.csv", {rampSpeeds, dat}ᵀ // N];
```

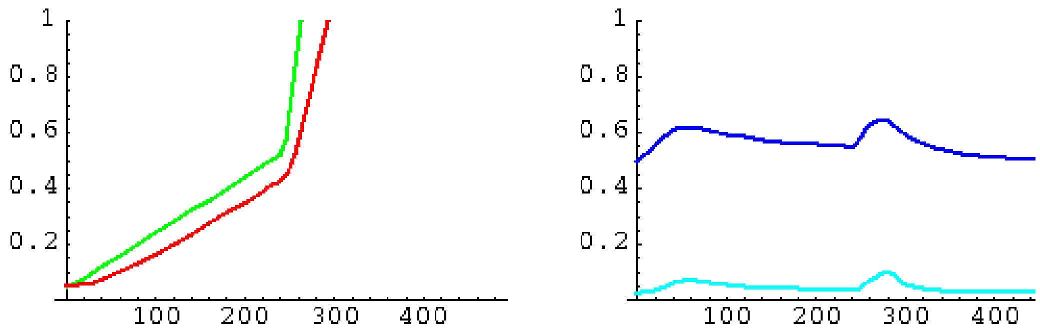
Fig 5c: Double Ramp

```
maxC = 10.;
tend = 60 * 8;
tq = 60 * 4;

nMmin = 1 / 60;
rateInit = 0.8;
rate1 = 11.0;
rate2 = 2.7;
doubleramp = Function[r,
   Function[t, Clip[If[t < tq, rateInit * nMmin * t, r * nMmin * (t - tq) + rateInit * tq * nMmin], {0, maxC}]]
  ];

DisplayTogetherArray[
  Plot[#, {t, t0, tend}, PlotRange → {{t0, tend}, {0, maxC}}] & /@ {doubleramp[rate1][t], doubleramp[rate2][t]}];
```




```
p = Plot[#1 /. sol, {t, t0, tend}, PlotRange → {All, {0, 1}}, PlotStyle → {Thickness[.01], #2}] &;
```

```
sol = runsim[L → doubleramp[rate1][t]];
g = DisplayTogetherArray[
   DisplayTogether[{p[a[t], Green], p[i[t], Red]}],
   DisplayTogether[{p[r[t], Blue], p[amp[t], Cyan]}]];
```

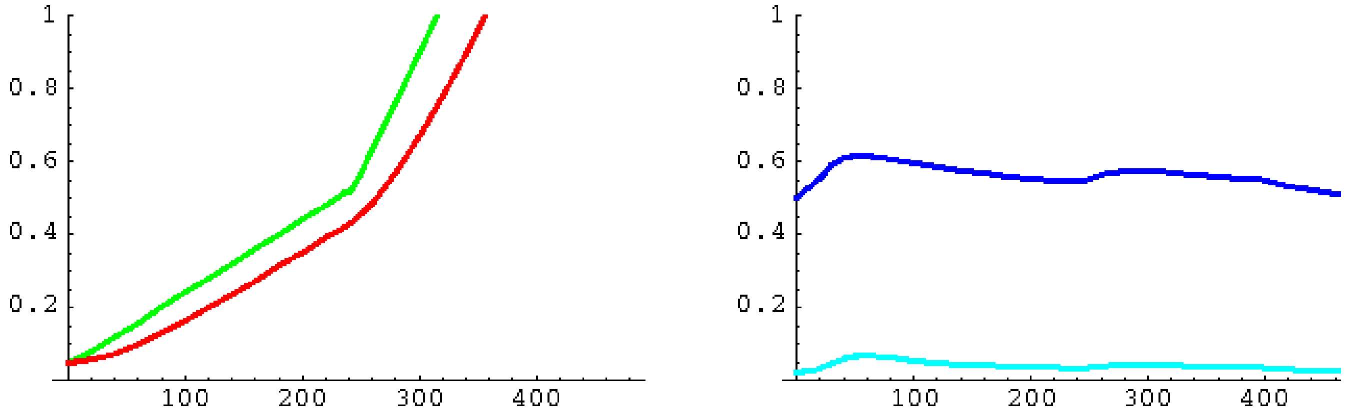



```
ti = Range[t0, tend, 1];
dat = {ti, a[t], i[t], r[t]} /. sol /. t → ti;
Export["fig5c-double.ramp-a.i.r.gif", g]
Export["fig5c-double.ramp-a.i.r.csv", datᵀ]
```

```
fig5c-double.ramp-a.i.r.gif
```

```
fig5c-double.ramp-a.i.r.csv
```

```
sol = runsim[L → doubleramp[rate2][t]];
g = DisplayTogetherArray[DisplayTogether[{p[a[t], Green], p[i[t], Red]}], DisplayTogether[{p[r[t], Blue], p[amp[t], Cyan]}]];
```



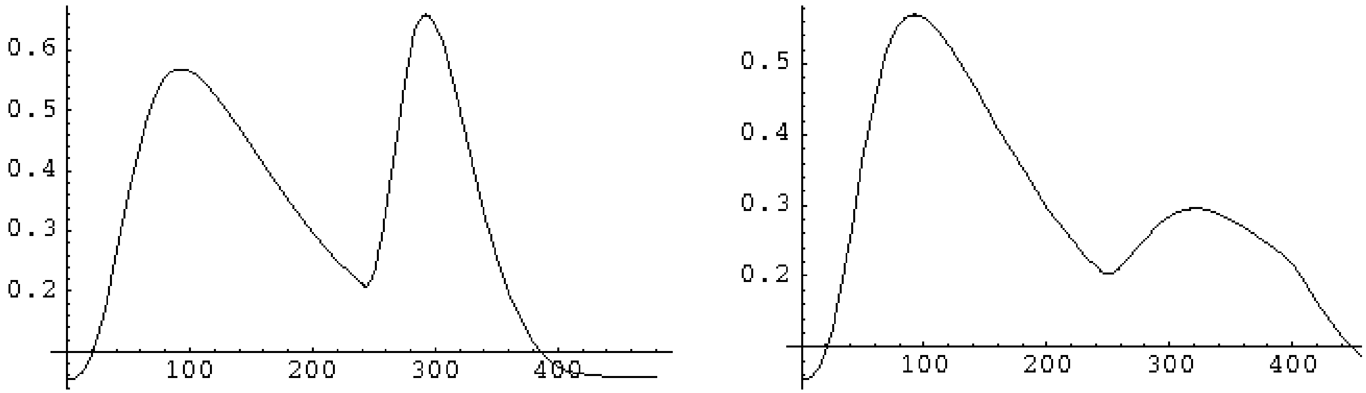

```
dat = {ti, a[t], i[t], r[t]} /. sol /. t → ti;
Export["fig5c-double.ramp2-a.i.r.gif", g]
Export["fig5c-double.ramp2-a.i.r.csv", datᵀ]
```

```
fig5c-double.ramp2-a.i.r.gif
```

```
fig5c-double.ramp2-a.i.r.csv
```

```
sims = popsim[L → #] & /@ {doubleramp[rate1][t], doubleramp[rate2][t]};
rsims = PickResponders /@ sims;
```

```
g = DisplayTogetherArray[Plot[Mean[rsims[[#]]], {t, 0, tend}, PlotRange → All] & /@ Range[2]];
```

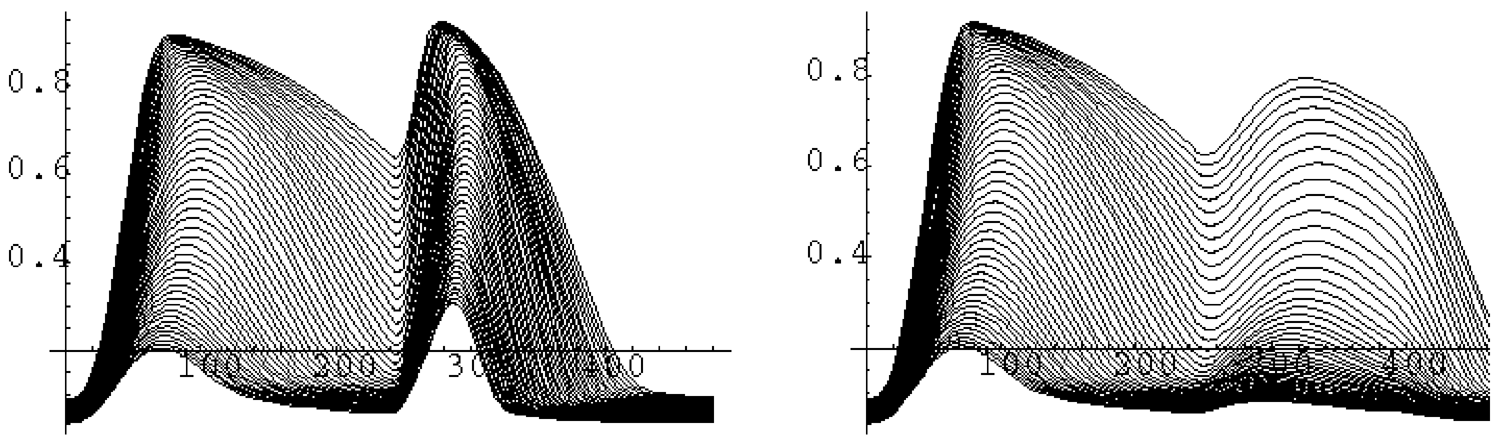



```
dat = {ti, Mean[rsims[[1]]] /. t → ti, Mean[rsims[[2]]] /. t → ti};
Export["fig5c-double.ramp-amp.gif", g]
Export["fig5c-double.ramp-amp.csv", datᵀ]
```

fig5c-double.ramp-amp.gif

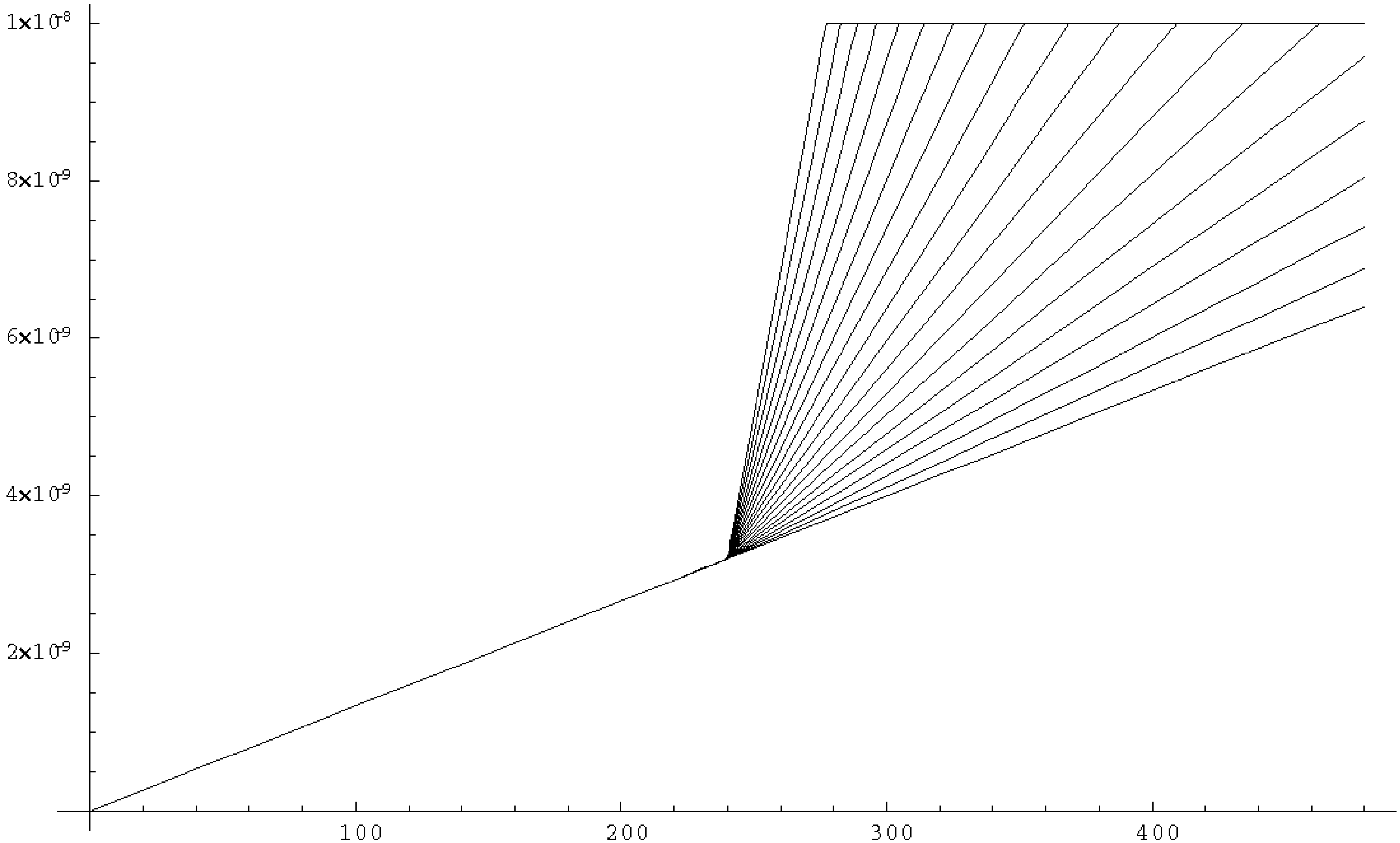
fig5c-double.ramp-amp.csv

```
DisplayTogetherArray[Plot[Evaluate@rsims[[#]], {t, 0, tend}, PlotRange → All] & /@ Range[2]];
```



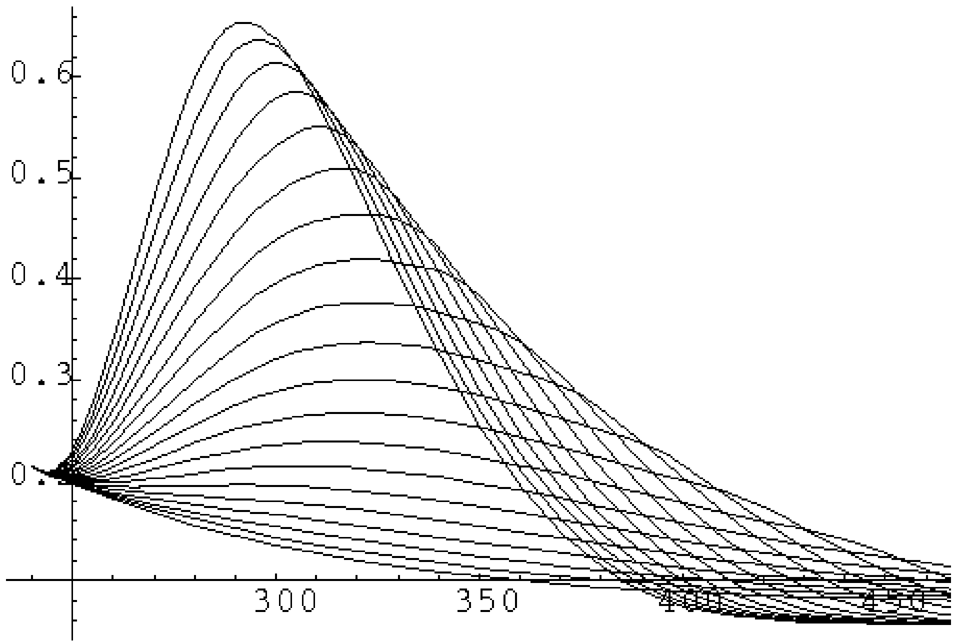
■ Finer

```
rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20];
Plot[doubleramp[#][t] & /@ rampspeed // Evaluate, {t, 0, tend}, PlotRange → All];
```



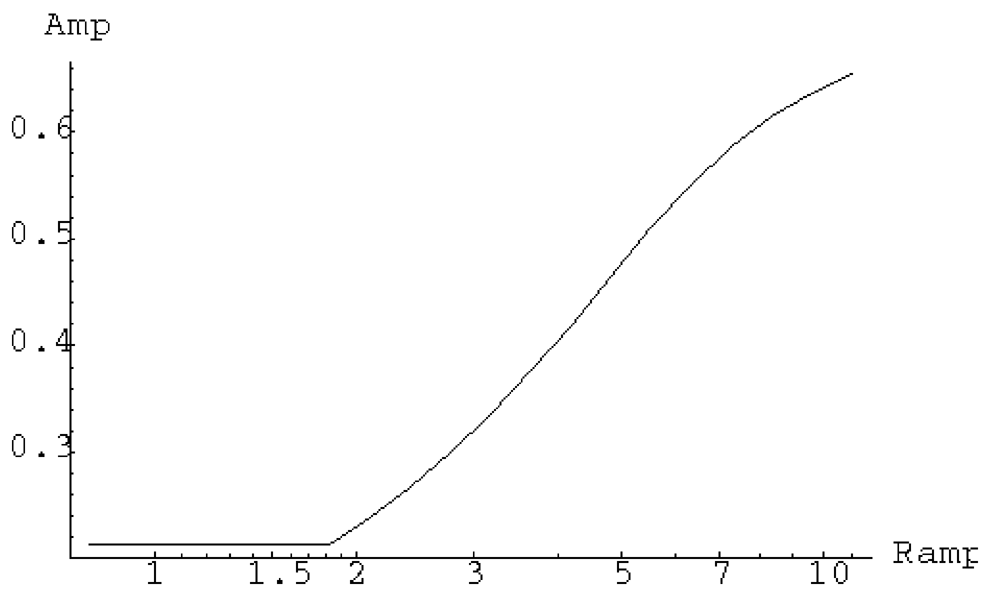
```
sims = popsim[L -> doubleramp[#][t]] & /@ rampspeed;
rsims = PickResponders /@ sims;
mrsims = Mean /@ rsims;
```

```
Plot[mrsims // Evaluate, {t, tq, tend}, PlotRange → All];
```




```
dat = Max[# /. t → Range[tq, tend, 1]] & /@ mrsims;
```

```
g = LogLinearListPlot[{rampspeed, dat}ᵀ, PlotJoined → True, AxesLabel → {"Ramp Speed", "Amp"}];
```




```
Export["fig5c-double.ramp-variation.gif", g]
Export["fig5c-double.ramp-variation.csv", {rampspeed, dat}ᵀ]
```

```
fig5c-double.ramp-variation.gif
```

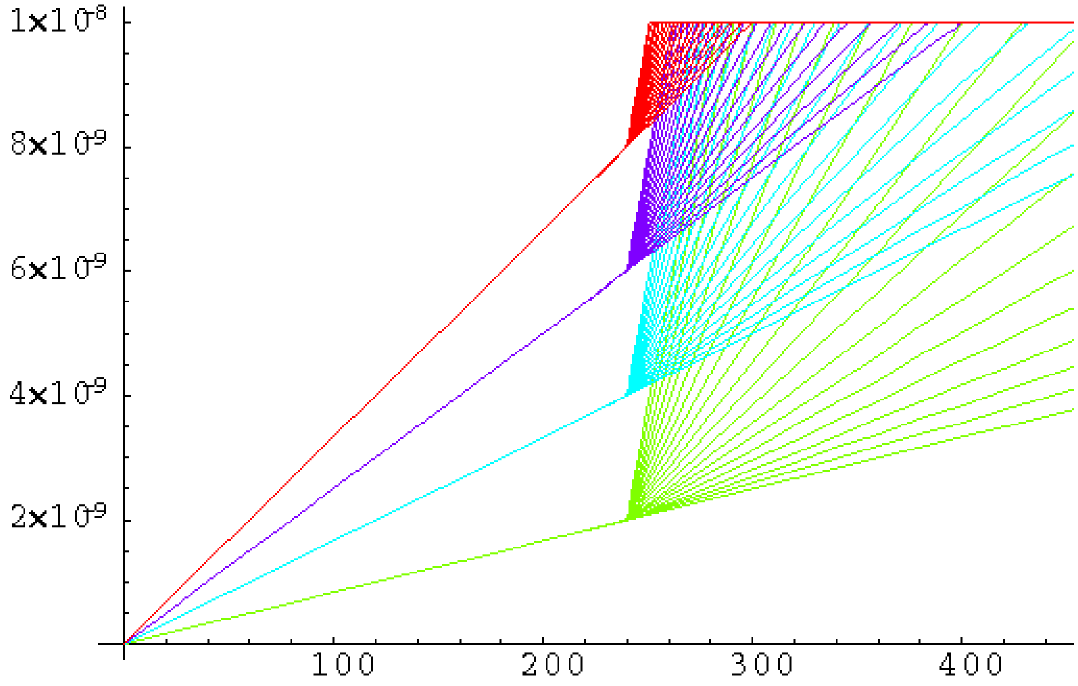
```
fig5c-double.ramp-variation.csv
```

■ Varying initial ramp speed

```
initRampSpeeds = Range[0.5, 2, .5]
len = Length[initRampSpeeds];
```

```
{0.5, 1., 1.5, 2.}
```

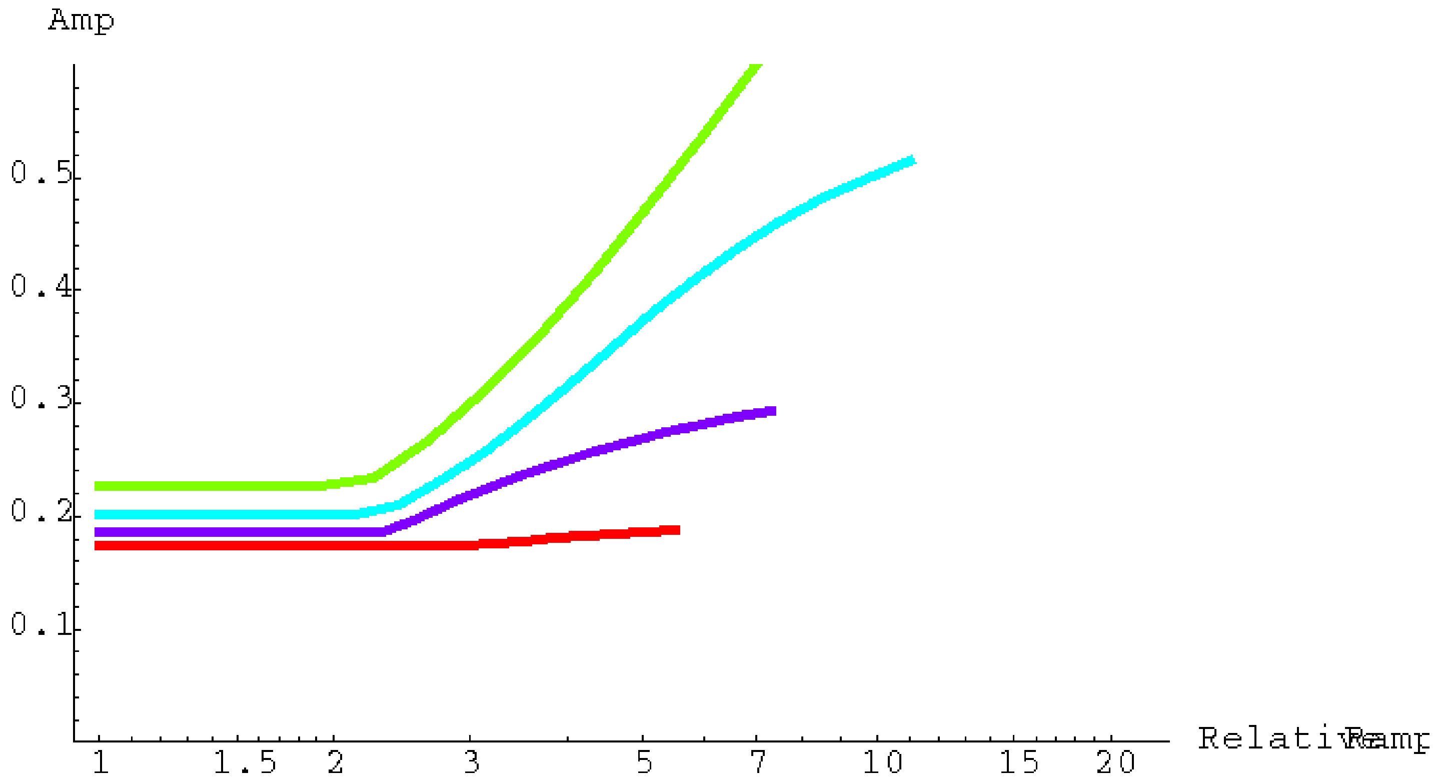
```
g = DisplayTogether[
   Function[r,
     rateInit = initRampSpeeds[[r]];
     rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20];
     Plot[doubleramp[#][t] & /@ rampspeed // Evaluate, {t, 0, tend}, PlotRange → All, PlotStyle → Hue[r / len]]
    ] ~ Array ~ len];
```




```
ti = Range[t0, tend, 1];
dat = Function[r,
    rateInit = initRampSpeeds[[r]];
    rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20];
    Function[s, doubleramp[s][#] & /@ ti] /@ rampspeed
   ] ~ Array ~ len;
dat = Flatten[dat, 1];
```

```
Export["variable-initial-ramp.gif", g];
Export["variable-initial-ramp.csv", datᵀ];

dat = Function[r,
    rateInit = r;
    rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20]; sims = popsim[L -> doubleramp[#][t]] & /@ rampspeed;
    rsims = PickResponders /@ sims;
    mrsims = Mean /@ rsims;
    dat = Max[# /. t → Range[tq, tend, 1]] & /@ mrsims;
    {rampspeed/rateInit, dat}ᵀ
   ] /@ initRampSpeeds;

Dimensions[dat]

{4, 20, 2}
```

```
g = DisplayTogether[Function[r,
      LogLinearListPlot[dat[[r]], PlotRange → {0, 0.6}, PlotJoined → True,
       AxesLabel → {"Relative Ramp Speed", "Amp"}, PlotStyle → {Thickness[.01], Hue[r / len]}]
     ] ~
     Array ~
     len];
```



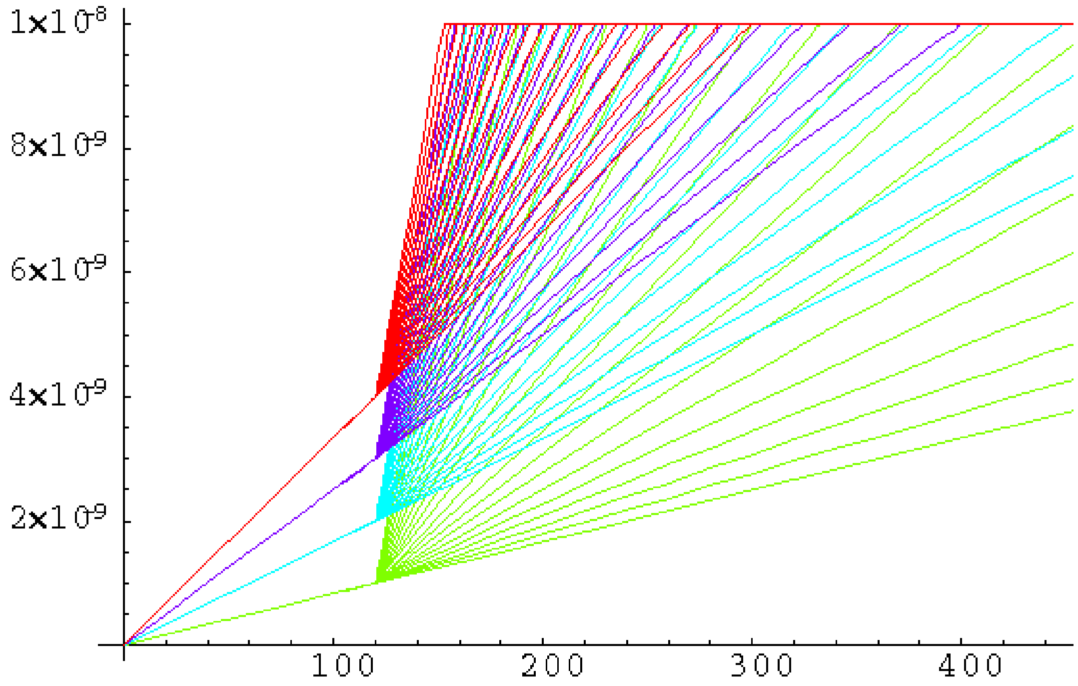

```
Export["variable-initial-ramp-2nd-peak.gif", g];
Export["variable-initial-ramp-2nd-peak.csv", Flatten[Transpose[dat, {1, 3, 2}], 1]ᵀ];
```

Varying initial ramp speed, half as short initial ramp

```
tq = tq / 2;
initRampSpeeds = Range[0.5, 2, .5]
len = Length[initRampSpeeds];
```

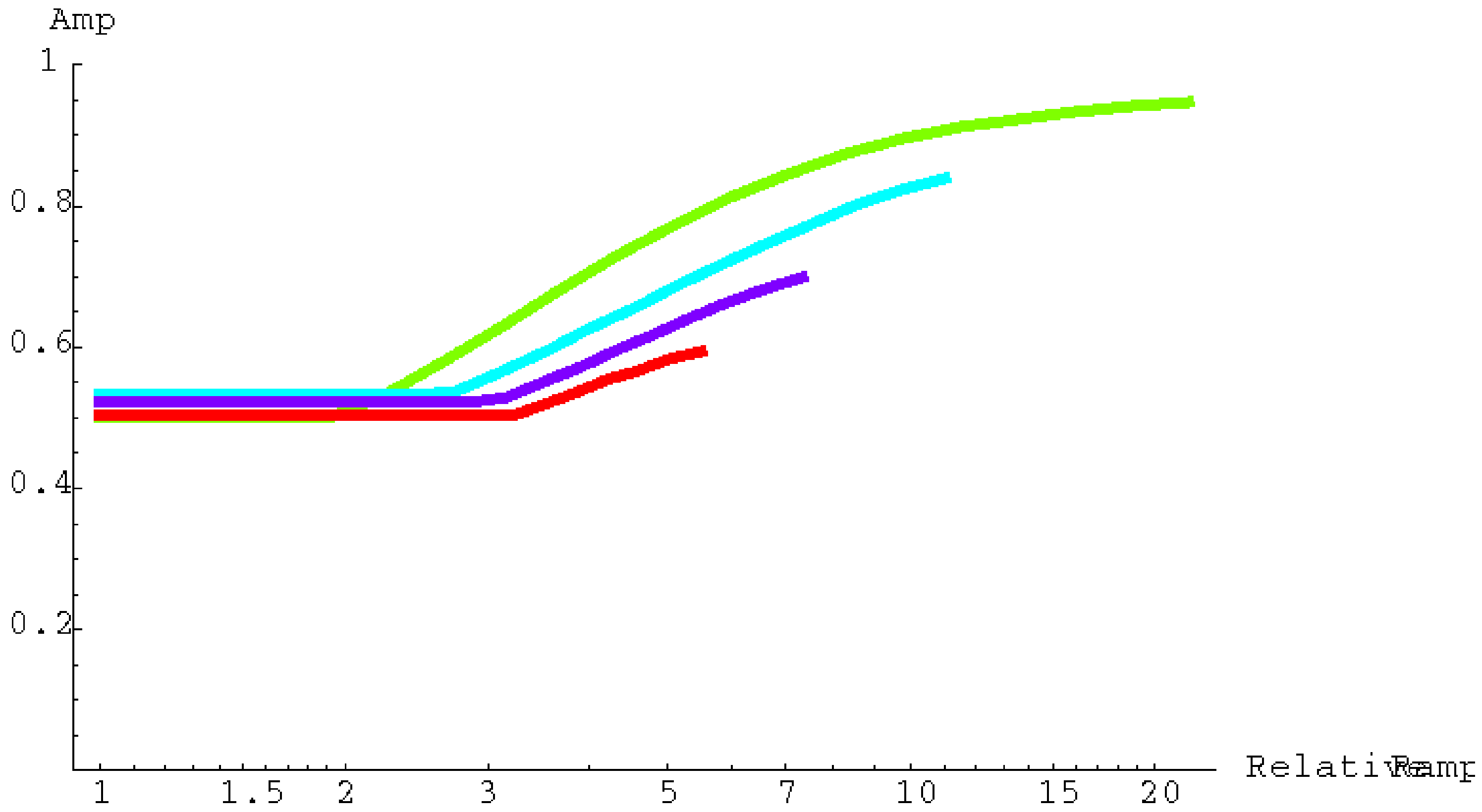
```
{0.5, 1., 1.5, 2.}
```

```
g = DisplayTogether[
   Function[r,
     rateInit = initRampSpeeds[[r]];
     rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20];
     Plot[doubleramp[#][t] & /@ rampspeed // Evaluate, {t, 0, tend}, PlotRange → All, PlotStyle → Hue[r / len]]
   ]~Array~len];
```



```
dat = Function[r,
    rateInit = initRampSpeeds[[r]];
    rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20];
    Function[s, doubleramp[s][#] & /@ ti] /@ rampspeed
   ]~Array~len;
dat = Flatten[dat, 1];

Export["variable-initial-ramp2.gif", g];
Export["variable-initial-ramp2.csv", datᵀ];

dat = Function[r,
    rateInit = r;
    rampspeed = 10^RangeI[Log[10, rateInit], Log[10, rate1], 20]; sims = popsim[L -> doubleramp[#][t]] & /@ rampspeed;
    rsims = PickResponders /@ sims;
    mrsims = Mean /@ rsims;
    dat = Max[# /. t → Range[tq, tend, 1]] & /@ mrsims;
    {rampspeed/rateInit, dat}ᵀ
   ] /@ initRampSpeeds;

Dimensions[dat]

{4, 20, 2}
```

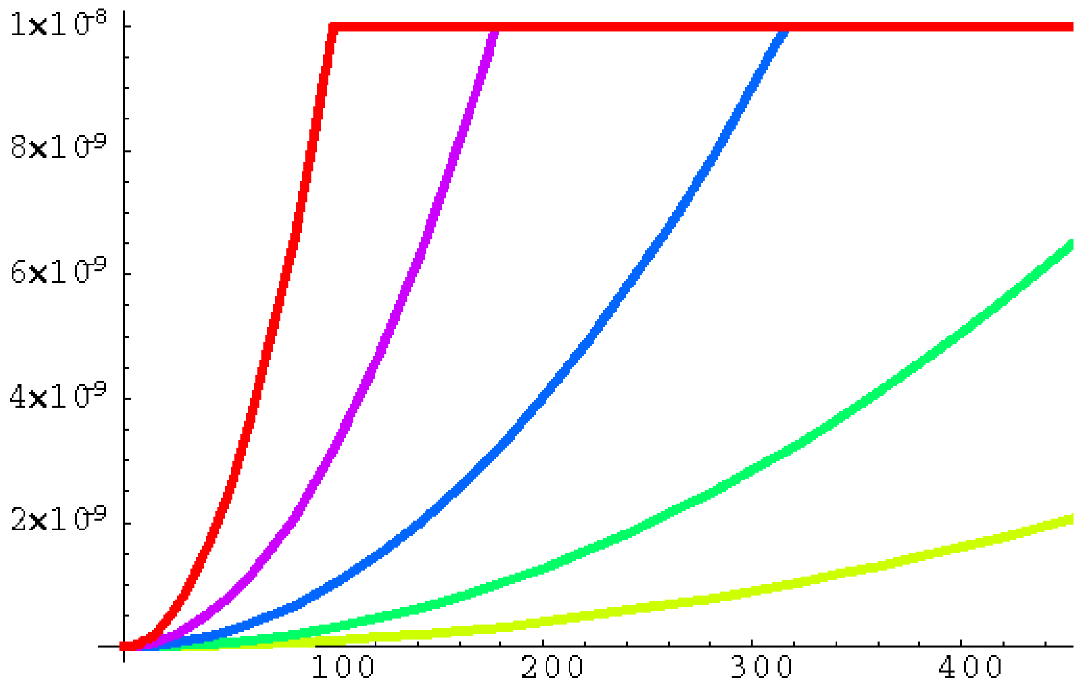
```
g = DisplayTogether[Function[r,
     LogLinearListPlot[dat[[r]], PlotRange → {0, 1}, PlotJoined → True,
      AxesLabel → {"Relative Ramp Speed", "Amp"}, PlotStyle → {Thickness[.01], Hue[r / len]}]
    ] ~
    Array ~
    len];
```



```
Export["variable-initial-ramp-2nd-peak2.gif", g];
Export["variable-initial-ramp-2nd-peak2.csv", Flatten[Transpose[dat, {1, 3, 2}], 1]ᵀ];
```

Parabolic Input

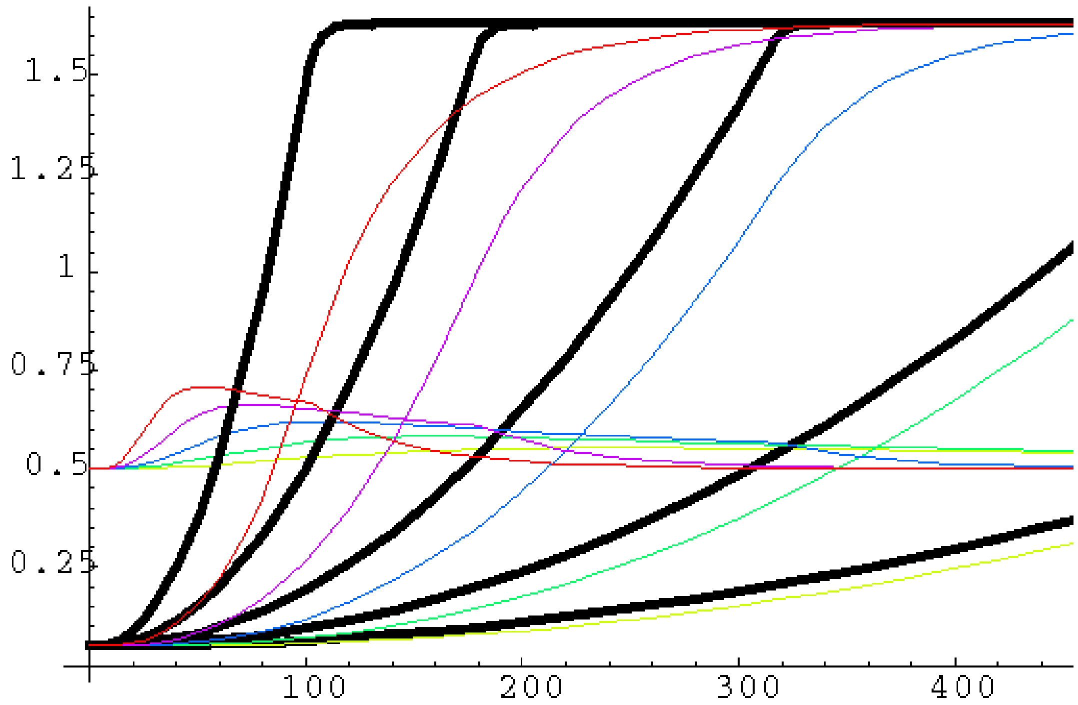
```
tend = 8 * 60;
maxC = 10 * 10^-9.;
inputs = Table[Clip[10.^α (t - 0)^2 UnitStep[t - 0], {0., maxC}], {α, -14, -12, .5}];
len = Length[inputs];
g = DisplayTogether[Table[
    Plot[inputs[[r]] // Evaluate, {t, t0, tend}, PlotStyle → {Thickness[.01], Hue[r / len]}], {r, len}]];
```




```
ti = Range[t0, tend, 1];
```

```
dat = Prepend[inputs, ti] /. t → ti;
Export["parabolic-inputs.gif", g];
Export["parabolic-inputs.csv", datᵀ];
```

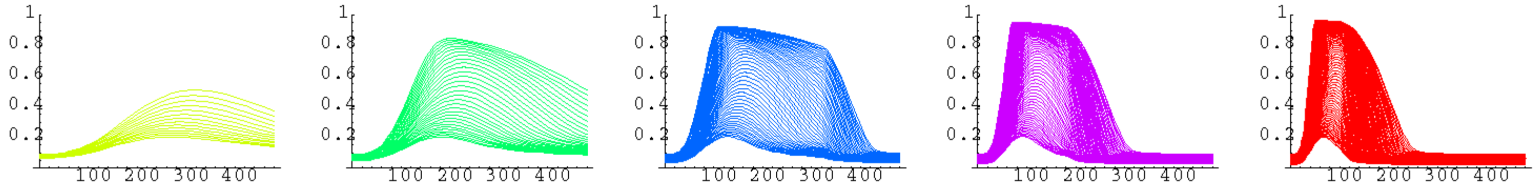
```
sims = runsim[L → #] & /@ inputs;
DisplayTogether[Table[Plot[{a[t], i[t], r[t]} /. sims[[k]] // Evaluate,
   {t, t0, tend}, PlotStyle → {Thickness[.01], Hue[k / len], Hue[k / len]}], {k, len}]]
```



- Graphics -

```
sims = popsim[L → #] & /@ inputs;
rsims = PickResponders[#, tend] & /@ sims;
mrsims = Mean /@ rsims;
```

```
g = DisplayTogetherArray[Table[
    Plot[rsims[[r]] // Evaluate, {t, t0, tend}, PlotRange → {0, 1}, PlotStyle → {Hue[r / len]}]
    , {r, len}]];
```



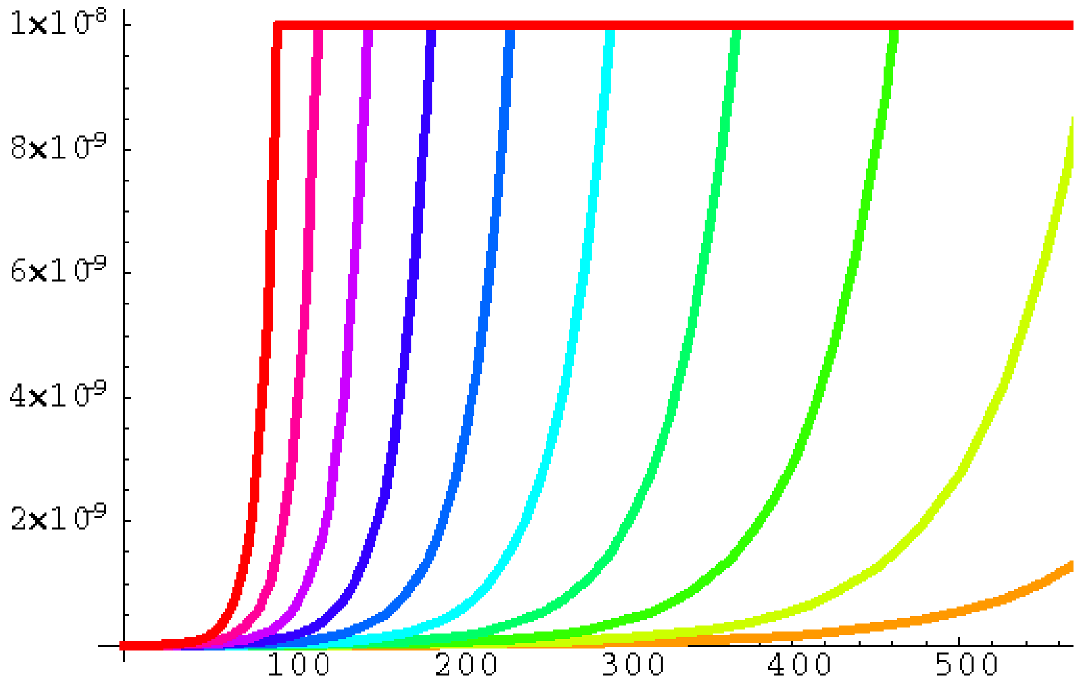

```
dat = Prepend[mrsims, ti] /. t → ti;
Export["parabolic-inputs-amp.gif", g];
Export["parabolic-inputs-amp.csv", datᵀ];
```

```
Dimensions[dat]
```

{6, 481}

Exponential Input

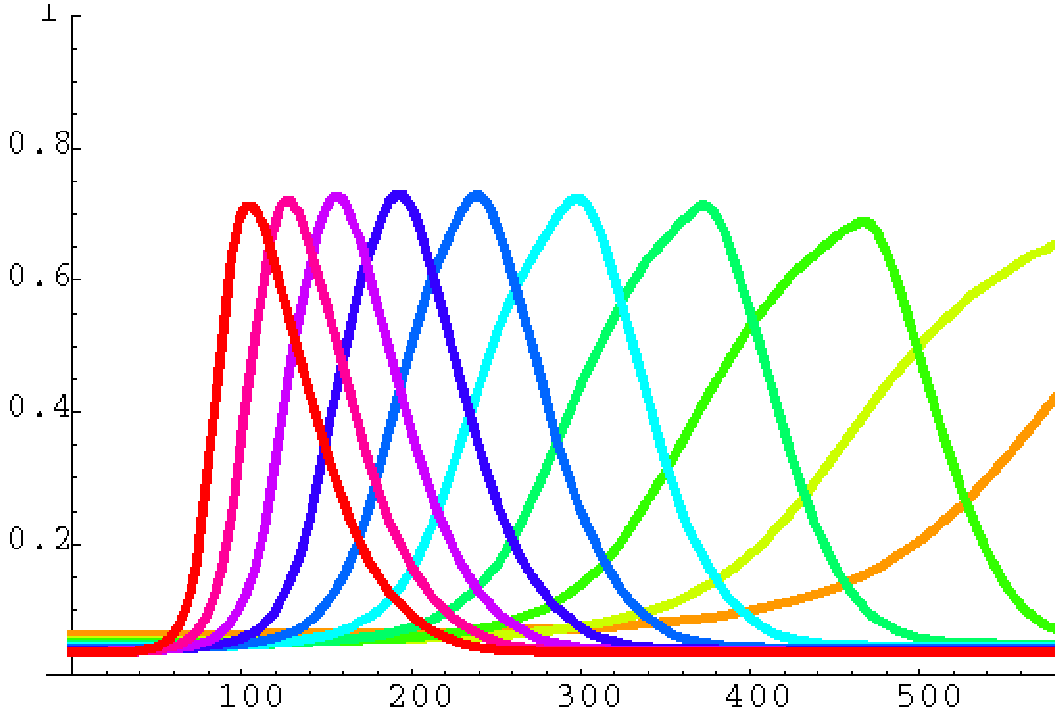
```
inputs = Table[Clip[10.^-12 Exp[10^α t], {0., maxC}], {α, -1.9, -1, .1}];
len = Length[inputs];
g = DisplayTogether[Table[
    Plot[inputs[[r]] // Evaluate, {t, t0, tend}, PlotStyle → {Thickness[.01], Hue[r / len]}], {r, len}]];
```




```
dat = Prepend[inputs, ti] /. t → ti;
Export["exponential-inputs.gif", g];
Export["exponential-inputs.csv", dat^T];
```

```
sims = popsim[L → #] & /@ inputs;
rsims = PickResponders[#, tend] & /@ sims;
mrsims = Mean /@ rsims;
```

```
g = DisplayTogether[Table[
    Plot[mrsims[[r]] // Evaluate, {t, t0, tend}, PlotRange → {0, 1}, PlotStyle → {Thickness[.01], Hue[r / len]}]
    , {r, len}]];
```




```
dat = Prepend[mrsims, ti] /. t → ti;
Export["exponential-inputs-amp.gif", g];
Export["exponential-inputs-amp.csv", datᵀ];
```

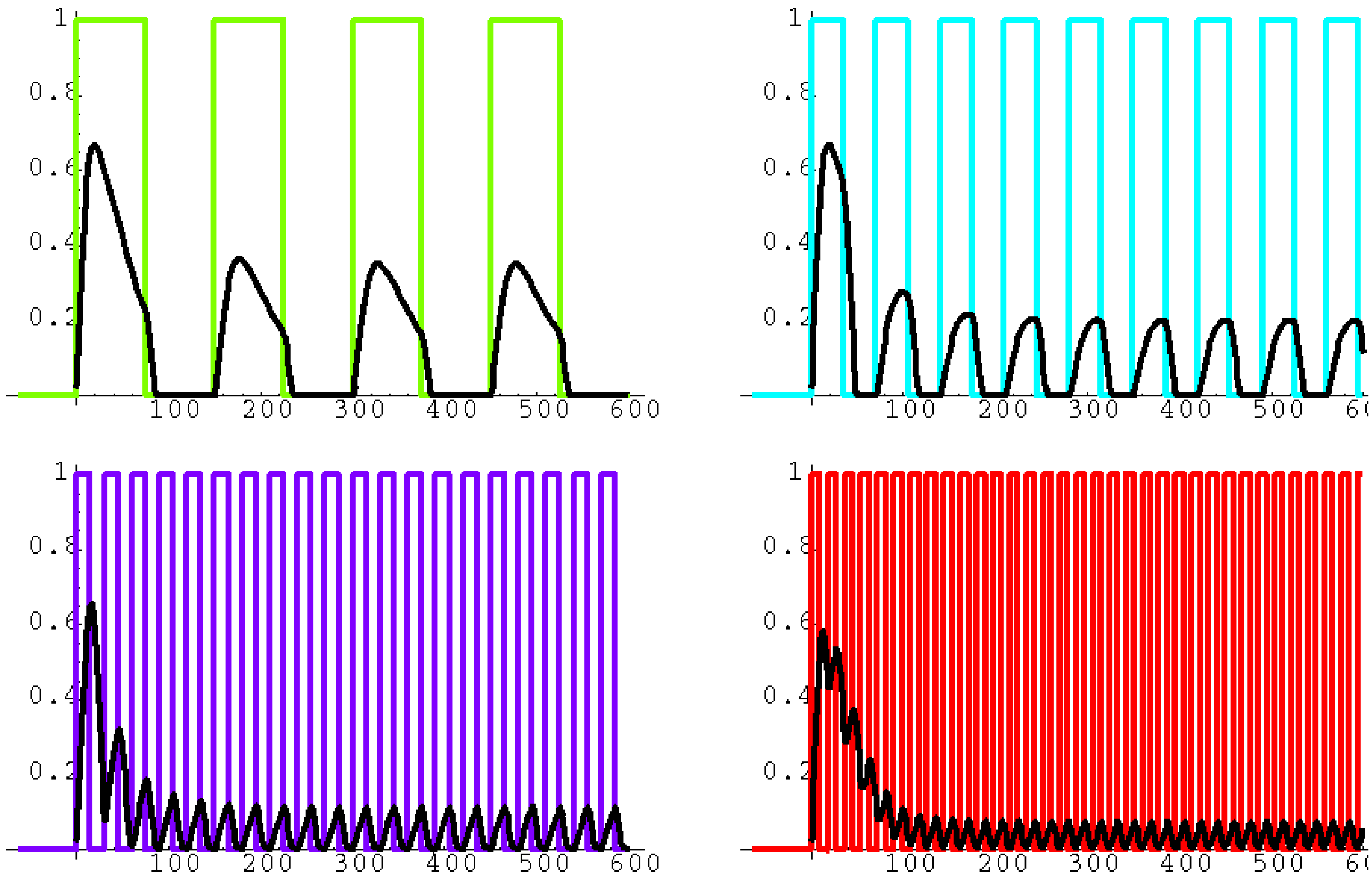
Fig 5d: Pulse

```
tend = 60 * 10;
pstyle = {Thickness[.01], Hue[# / Length[inputs]]} &;
```

$$\text{inputs} = L \to 100. * 10^{-9} * \frac{\text{Sign@Sin}\left[\frac{2\pi t}{\#}\right] + 1}{2} \text{UnitStep}[t] \ \&\ /@\ (2 * \{75, 35, 15, 9\});$$

```
sims = popsim[#] & /@ inputs;
rsims = PickResponders /@ sims;

ti = Range[t0, tend, .5];

Function[s,
   dat = {t, L /. inputs[[s]], Mean[rsims[[s]]]} /. t → ti // Transpose;
   Export["fig5.pulse.mamp." <> ToString[s] <> ".csv", dat];
  ] ~Array~ Length[rsims];
```

```
DisplayTogetherArray[DisplayTogether[
    Plot[10^7 inputs[[#, 2]], {t, -60, tend}, PlotStyle → pstyle[#], PlotPoints → 100, PlotRange → All],
    Plot[rsims[[#]] // Mean // Evaluate, {t, t0, tend}, PlotPoints → 100, PlotRange → {0, 1}, PlotStyle → Thickness[0.01]]] &~
  Array~Length@inputs // Partition[#, 2] &];
```



```
sol = runsim[#] & /@ inputs;
DisplayTogetherArray[DisplayTogether[
    Plot[10^7 inputs[[#, 2]], {t, -60, tend}, PlotStyle → pstyle[#], PlotPoints → 100, PlotRange → {{t0, tend}, {0, 1}}],
    Plot[r[t] /. sol[[#]] // Evaluate, {t, t0, tend}, PlotPoints → 100, PlotRange → {0, 1}, PlotStyle → Thickness[0.01]]] &~
  Array~Length@inputs // Partition[#, 2] &];
```

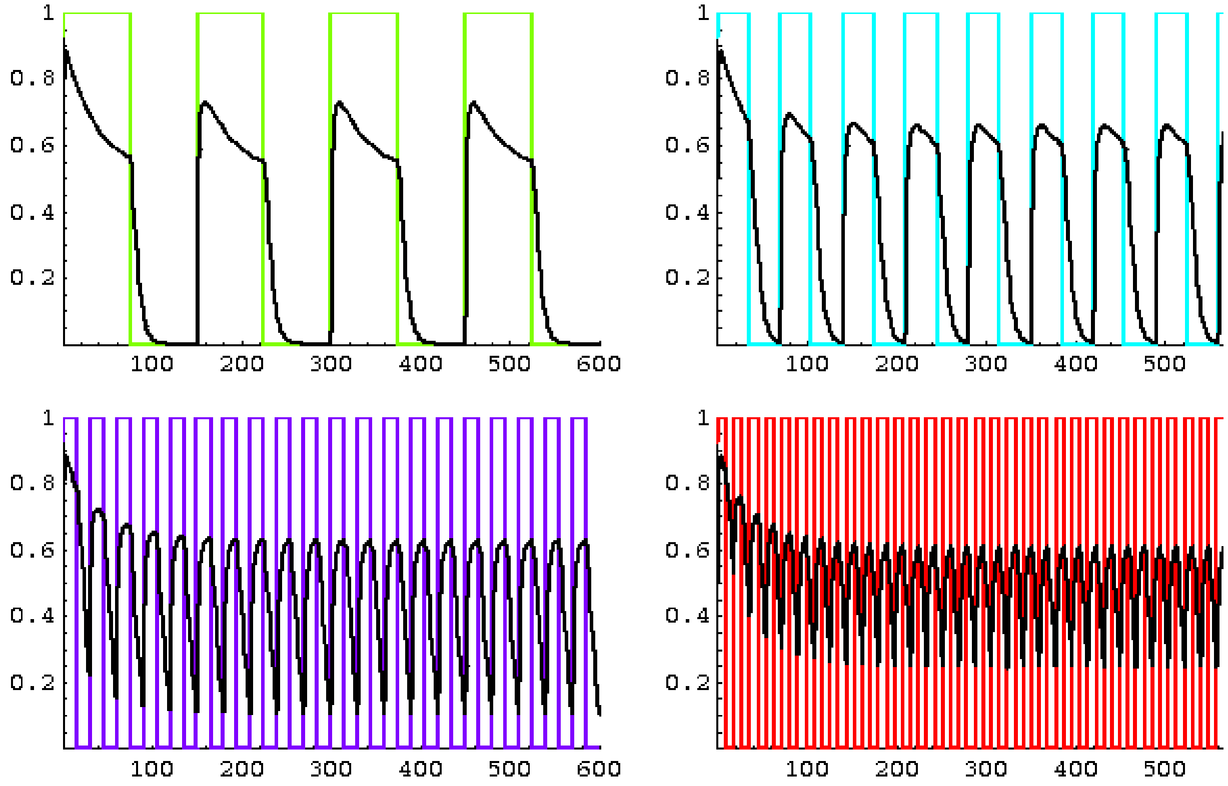
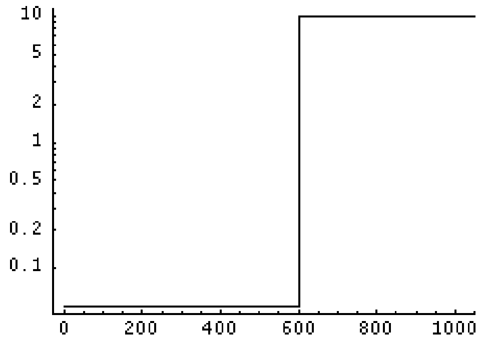


Fig 5e: Double dose

```
tq = 10 * 60;
tend = 2 tq;
input = .05 UnitStep[t, tq - t] + 10 UnitStep[t - tq];
LogPlot[input, {t, 0, tend}]
```

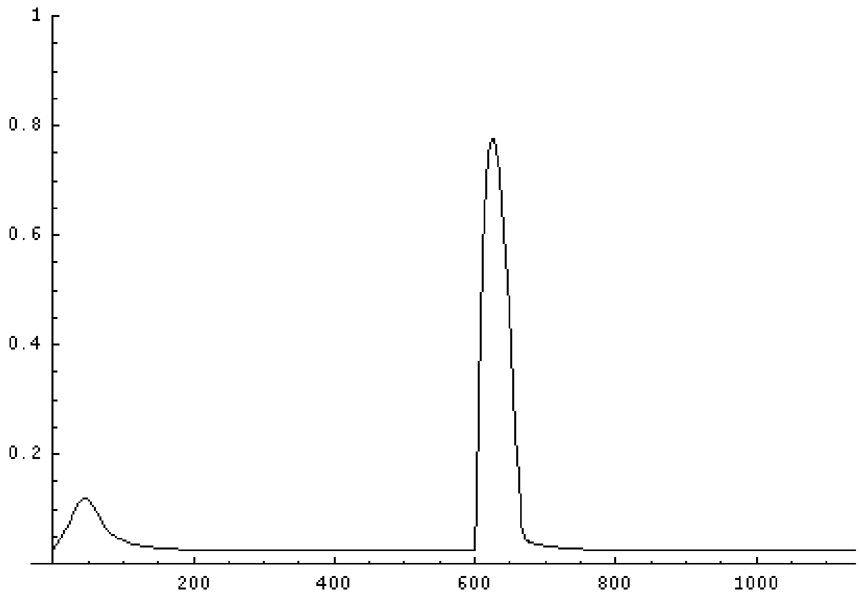



```
- Graphics -
```

```
sim = popsim[L → input];
```

```
rsim = PickResponders[sim];
```

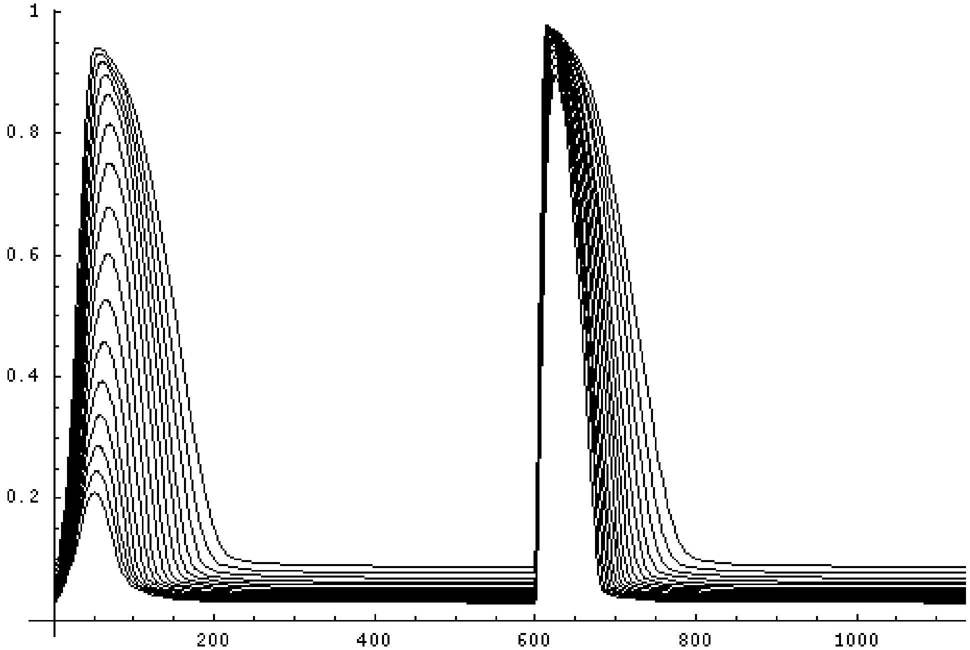
```
Plot[rsim[[20]], {t, 0, tend}, PlotRange → {0, 1}]
```



- Graphics -

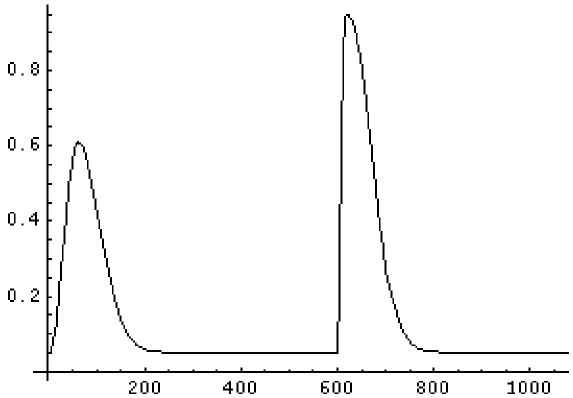
```
rsim = PickResponders[sim];
```

```
Plot[Evaluate@rsim, {t, 0, tend}]
```



- Graphics -

```
Plot[Evaluate@Mean@rsim, {t, 0, tend}]
```



- Graphics -

Spatial gradient

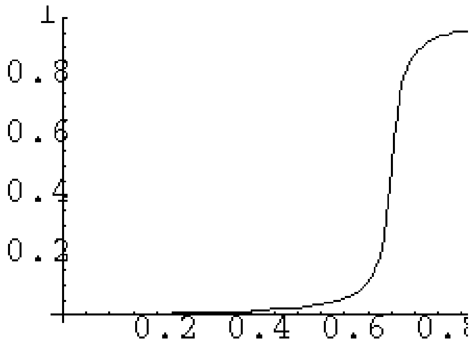
```
SetOptions[ListDensityPlot, Mesh → False, Frame → False];

<< Graphics`PlotField`

ssAmp = amp /. Solve[{0 == sys[[-1, 2]] /. rdt}, amp]
```

$$\left\{\frac{1}{2\,(I2-r)}\left(I2+I2\,Km-r+Km\,r-\sqrt{-4\,Km\,(I2-r)\,r+(-I2-I2\,Km+r-Km\,r)^2}\right),\ \frac{1}{2\,(I2-r)}\left(I2+I2\,Km-r+Km\,r+\sqrt{-4\,Km\,(I2-r)\,r+(-I2-I2\,Km+r-Km\,r)^2}\right)\right\}$$

```
AmpF = Function[{r, i2}, Evaluate@If[r == i2, 0.5, Evaluate[ssAmp[[1]] /. I2 → i2 /. params]]]
```

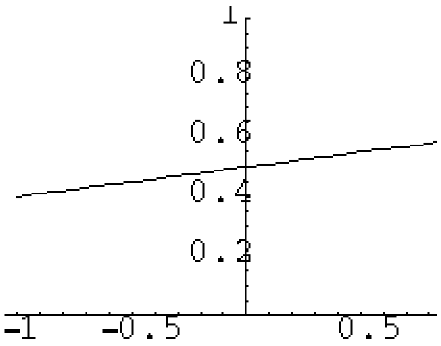
$$\text{Function}\left[\{r,\ i2\},\ \text{If}\left[r == i2,\ 0.5,\ \frac{1.01\,i2-0.99\,r-\sqrt{(-1.01\,i2+0.99\,r)^2-0.04\,(i2-r)\,r}}{2\,(i2-r)}\right]\right]$$

```
Plot[AmpF[r, 0.7], {r, 0, 1}];
```




```
grad[{x_, y_}] := g0 * x + c0;
c0 = 0.5;
g0 = 0.1;
```

```
Plot[grad[{x, 0}], {x, -1, 1}, PlotRange → {0, 1}];
```

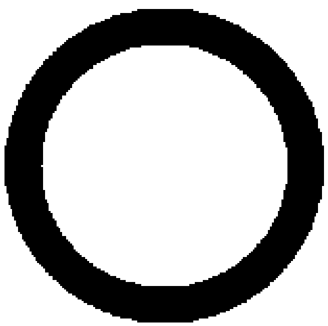


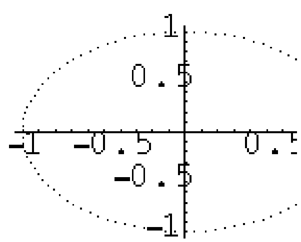

```
dθ = .05
```

0.05

```
Block[{R = 1.4, r1 = 1.3, r2 = 1, dx = 0.01, dθ = .01},
  circ = Table[{Cos[2 π θ], Sin[2 π θ]}, {θ, 0., 1 - dθ / 2., dθ}];
  vec = #/Norm[#] & /@ circ;
  mycoor = Table[{x, y}, {x, -R, R, dx}, {y, -R, R, dx}];
  mycell = Map[Function[c,
      {x, y} = c;
      2 UnitStep[(r1^2 - x^2 - y^2) (-r2^2 + x^2 + y^2)] - 1
    ], mycoor, {2}];
  d = Dimensions[mycell][[1]] / 2;
  pos = Position[mycell, 1];
  θ = ArcTan[Sequence @@ #] & /@ Extract[mycoor, pos];
 ];
Dimensions[mycell]
DisplayTogetherArray[ListDensityPlot[1 - mycell], ListPlot[circ]];
```

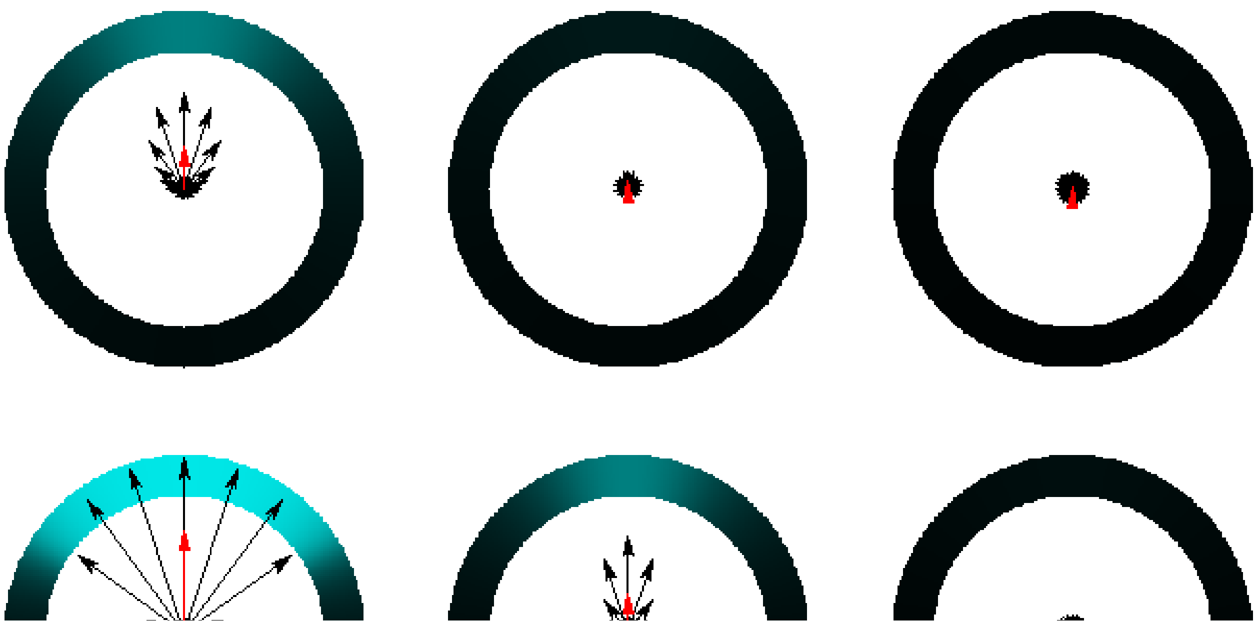
{281, 281}

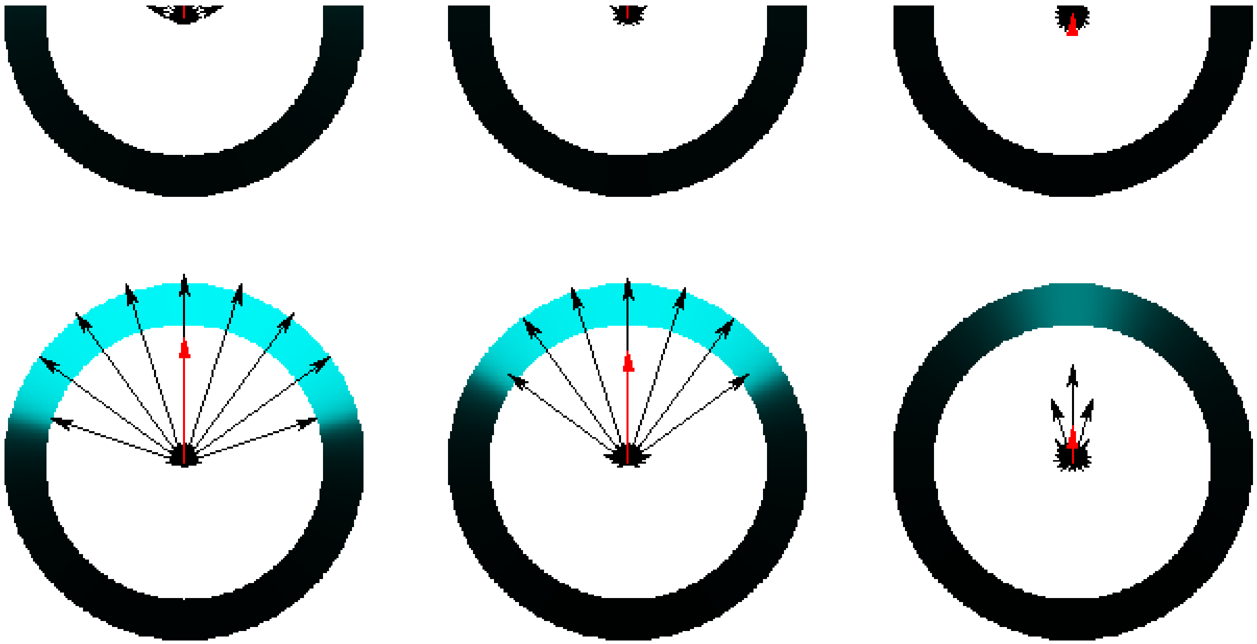





```
setGradient = Function[g,
   g0 = g;
   gcirc = grad /@ circ;
   gradVals = grad /@ Transpose@{Cos[θ], Sin[θ]};
   gradCell = ReplacePart[mycell, gradVals, pos, {Range[Length[gradVals]]}ᵀ];
  ];
setGradient[0.1]
```

```
f = Function[i2,
   ampCell = MapAt[AmpF[#, i2] &, gradCell, pos];
   ampVals = Map[AmpF[#, i2] &, gcirc];
   ampVecs = ampVals * vec;
   Mean[ampVecs][[1]]
  ];
Block[{$DisplayFunction = Identity},
  g = Function[g,
     setGradient[g];
     Function[i2,
       f[i2];
       ListDensityPlot[ampCell, ColorFunction :> (If[# < -0.1, RGBColor[1, 1, 1], RGBColor[0, #, #]] &), ColorFunctionScaling →
         False, PlotRange → {All, All, All}, Epilog → {Arrow[{d, d}, d + d * Reverse@#, HeadCenter → 0.75] & /@ ampVecs,
          Red, Arrow[{d, d}, d + d * 2 Reverse@Mean@ampVecs]}]] /@ {.55, .6, .7}
    ] /@ {.05, .1, .2}
 ];
DisplayTogetherArray[g]
```

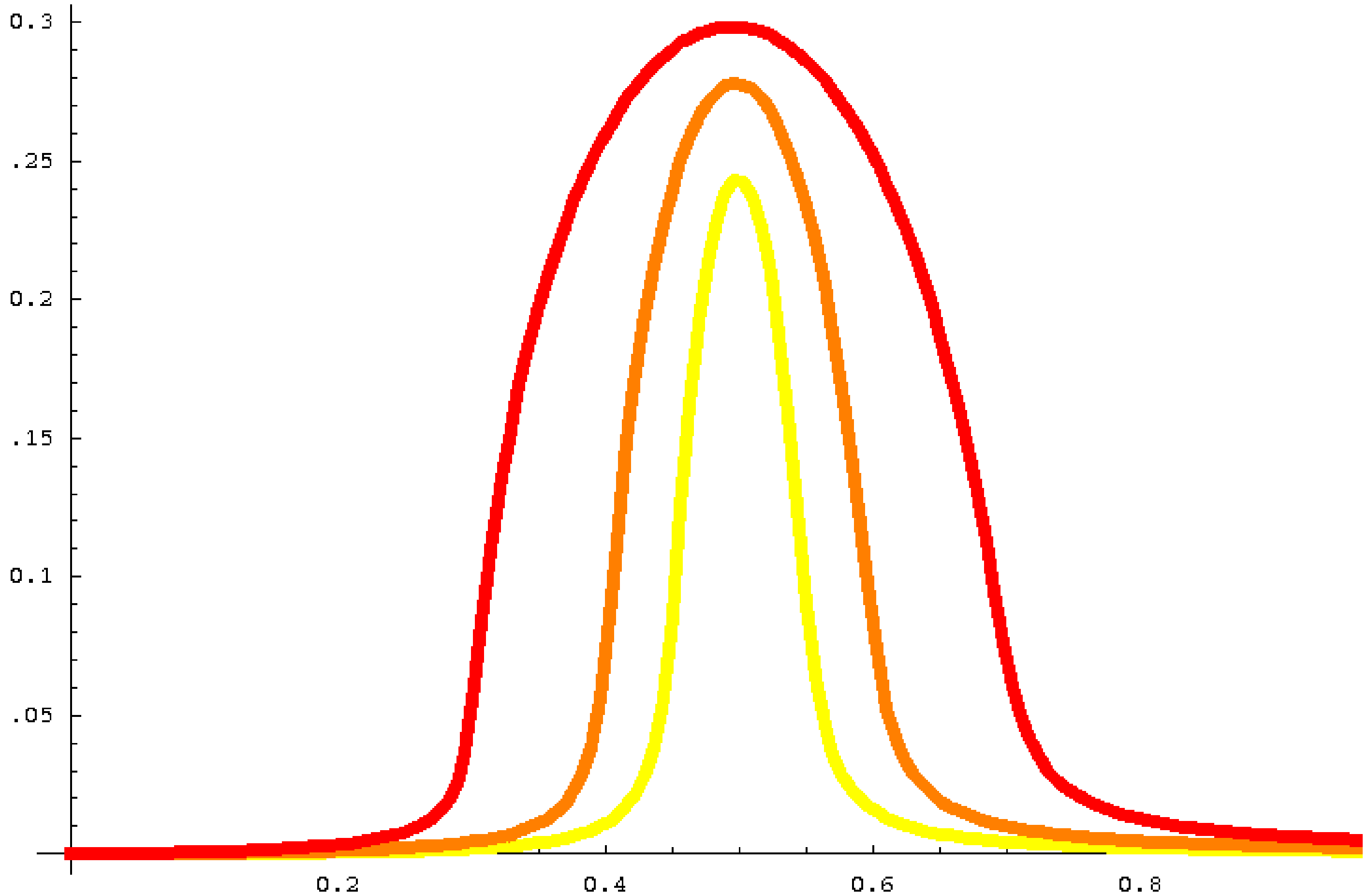




- GraphicsArray -

```
fullI2range = RangeI[0., 1., 150];
setGradient[.05];
dat1 = f /@ fullI2range;
setGradient[.1];
dat2 = f /@ fullI2range;
setGradient[.2];
dat3 = f /@ fullI2range;
```

```
g = DisplayTogether[
    ListPlot[{fullI2range, dat1}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Yellow}, PlotRange → All],
    ListPlot[{fullI2range, dat2}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Orange}],
    ListPlot[{fullI2range, dat3}ᵀ, PlotJoined → True, PlotStyle → {Thickness[.01], Red}]
  ];
```



```
Export["sensing-metric.gif", g]
Export["sensing-metric.csv", {fullI2range, dat1, dat2, dat3}ᵀ]
```

```
sensing-metric.gif
```

```
sensing-metric.csv
```

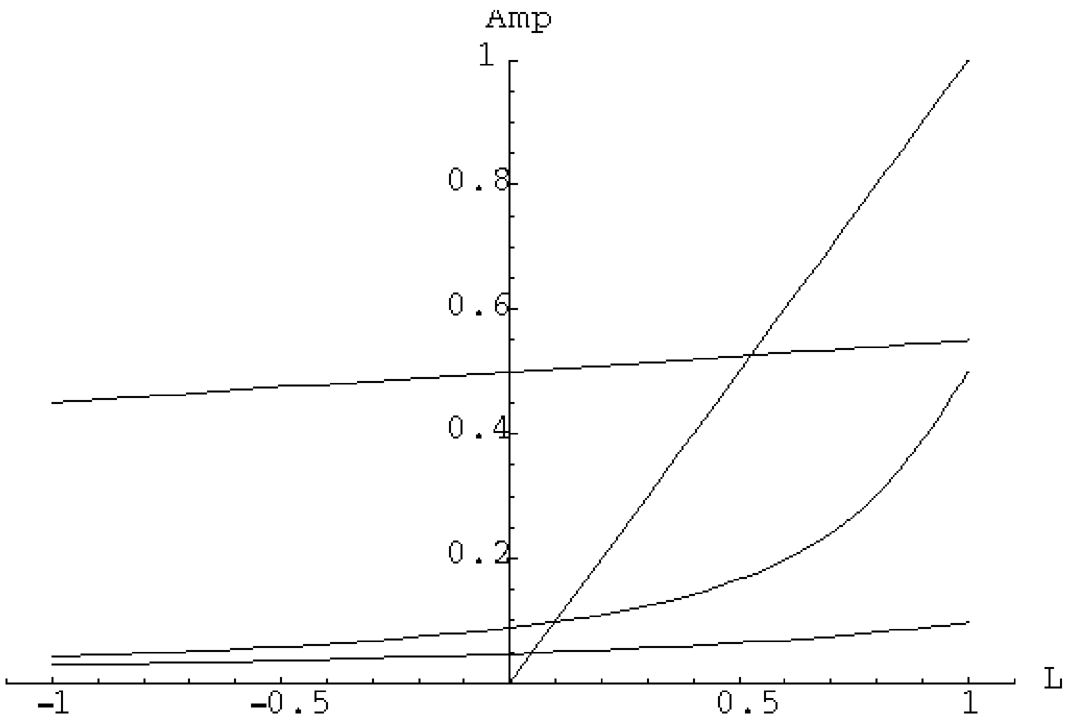
```
circ = Table[{Cos[2 π θ], Sin[2 π θ]}, {θ, 0, 1, .01}];
```

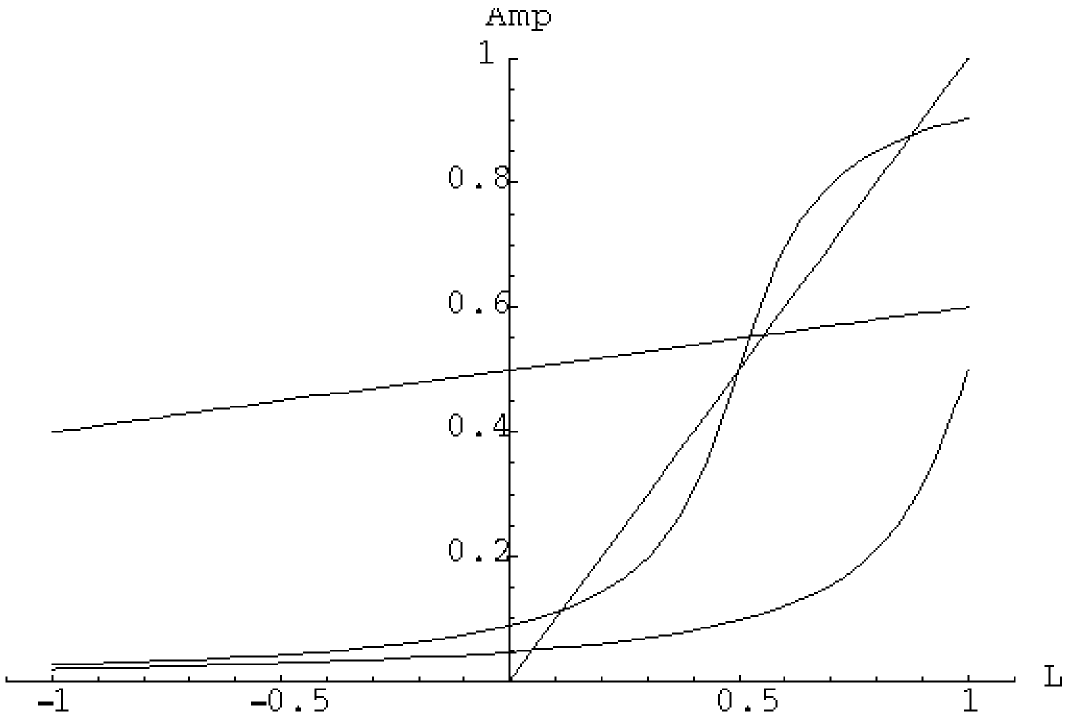
```
dat = Function[g,
    g0 = g;
    gcirc = grad /@ circ;

    dat = Function[i2,
       Map[AmpF[#, i2] &, gcirc]
      ] /@ {.55, .6, .7};
    Join[{circ[[All, 1]], gcirc}, dat]
   ] /@ {.05, .1, .2};
```

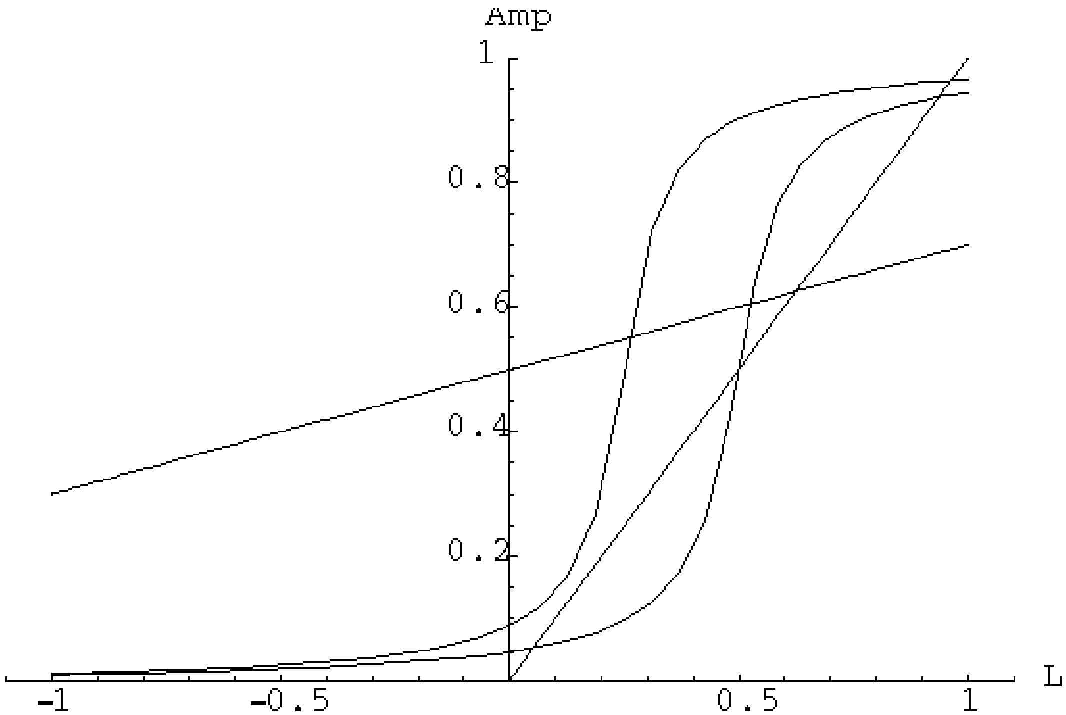
```
Dimensions[dat]
```

```
{3, 5, 101}
```

```
Function[d,
   DisplayTogether[ListPlot[{d[[1]], #}ᵀ, PlotJoined → True,
       PlotRange → {{-1.1, 1.1}, {0, 1}}, AxesLabel → {"LEGI-R", "Amp"}] & /@ Most@d]] /@ dat;
```


Amp
1
0.8
0.6
0.4
0.2
-1
-0.5
0.5
1
L


Amp
1
0.8
0.6
0.4
0.2
-1
-0.5
0.5
1
L



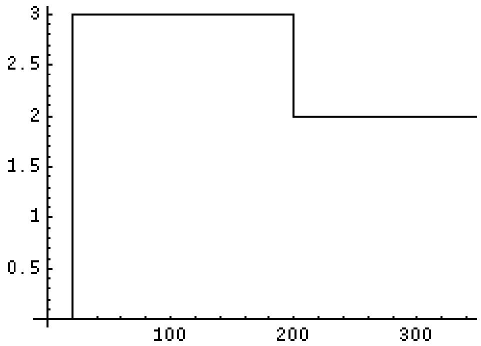

```
Array[
 Export["fig6-spatial.amp.g" <> ToString[#] <> ".csv", dat[[#]]ᵀ] &, 3]
```

{fig6-spatial.amp.g1.csv, fig6-spatial.amp.g2.csv, fig6-spatial.amp.g3.csv}

```
{.05, .1, .2}
```

Non-zero Pulse

```
tend = 400;
tq = 200;
input = L -> 3 UnitStep[t - 20, tq - t] + 2 UnitStep[t - tq];
Plot[L /. input, {t, t0, tend}, PlotRange → All];
```

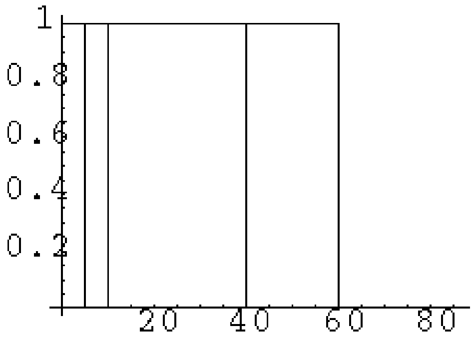



```
sim = popsim[input];

Plot[sim // Evaluate, {t, t0, tend}, PlotRange → All]
```

Transient Pulse

```
tend = 100

100

input = L → UnitStep[t, # - t] & /@ {5, 10, 40, 60}

{L → UnitStep[5 - t, t], L → UnitStep[10 - t, t], L → UnitStep[40 - t, t], L → UnitStep[60 - t, t]}
```

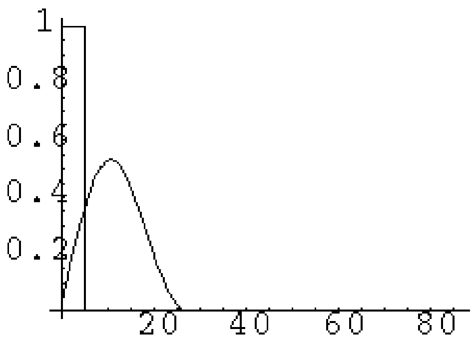
```
Plot[L /. Transpose@{input} // Evaluate, {t, t0, tend}]
```

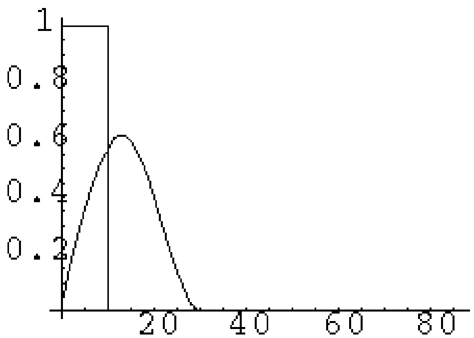


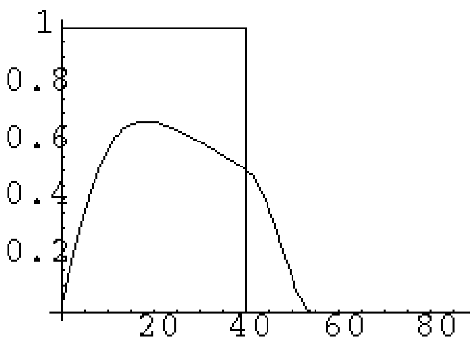
```
- Graphics -
```

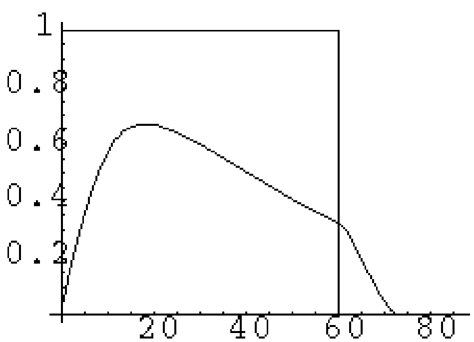
```
sims = popsim /@ input;
rsims = PickResponders /@ sims;
msims = Mean /@ rsims;
```

```
Function[i,
  DisplayTogether[
   Plot[L /. input[[i]] // Evaluate, {t, t0, tend}, PlotRange → All],
   Plot[msims[[i]] // Evaluate, {t, t0, tend}]]] /@ Range[4]
```










```
{- Graphics -, - Graphics -, - Graphics -, - Graphics -}
```

```
tend = 1000;
sim = runsim[input[[1]]];
Plot[{a[t], i[t], r[t]} /. sim // Evaluate, {t, t0, tend}, PlotRange → All]
```

HTML Export

```
tmp = "/tmp/LEGI-Amp/";
NotebookSave[];
If[FileType[tmp] == None, CreateDirectory[tmp]]
HTMLSave["/tmp/LEGI-Amp/index.html"]
Run["scp -rq /tmp/LEGI-Amp lunchbox:public_data/"]
Run["scp -qp /home/abergman/dicty/figs/* lunchbox:public_data/figs/"]
```

```
/tmp/LEGI-Amp/
```

```
/tmp/LEGI-Amp/index.html
```

```
0
```

```
0
```

```
Run["scp -qp /home/abergman/dicty/figs/* lunchbox:public_data/figs/"]
```

```
0
```

```
Run["scp -rq /tmp/LEGI-Amp lunchbox:public_data/"]
Run["scp -qp /home/abergman/dicty/figs/* lunchbox:public_data/figs/"]
```

```
0
```

```
256
```

```
Exit[]
```