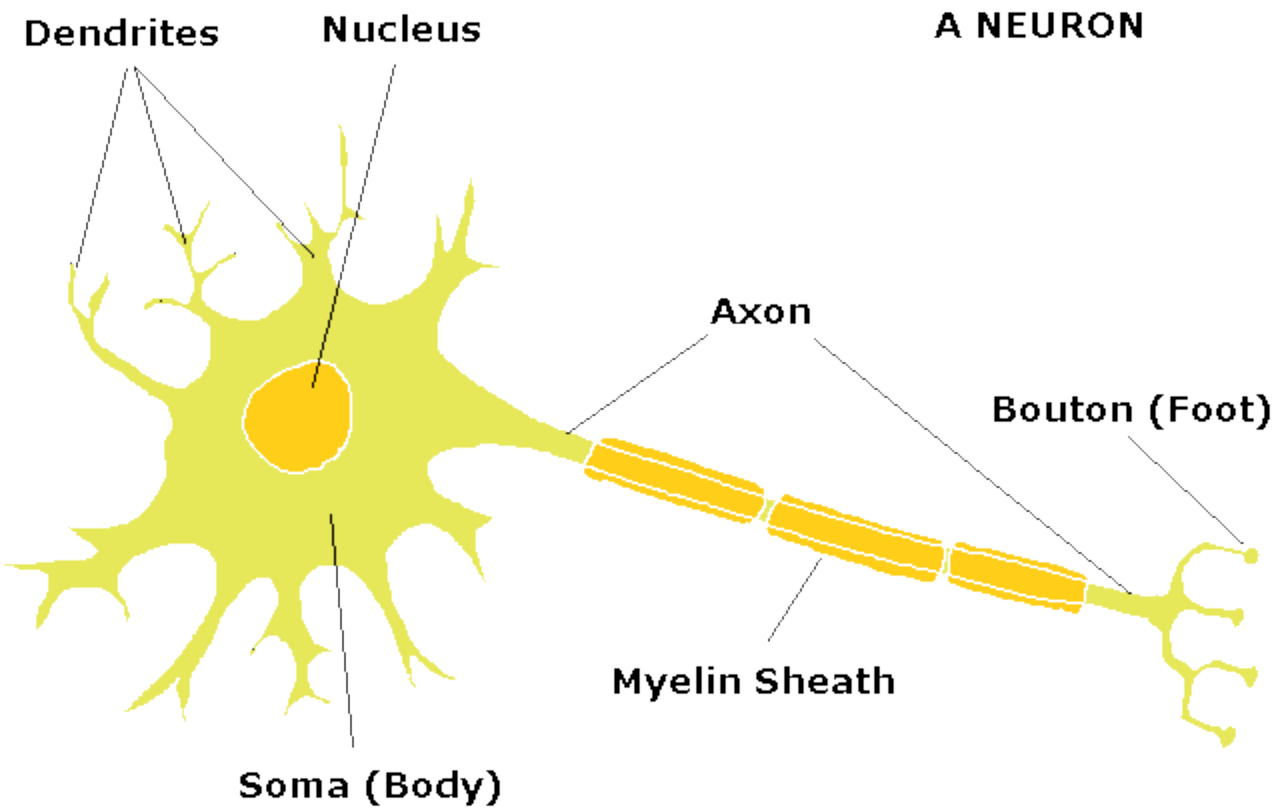
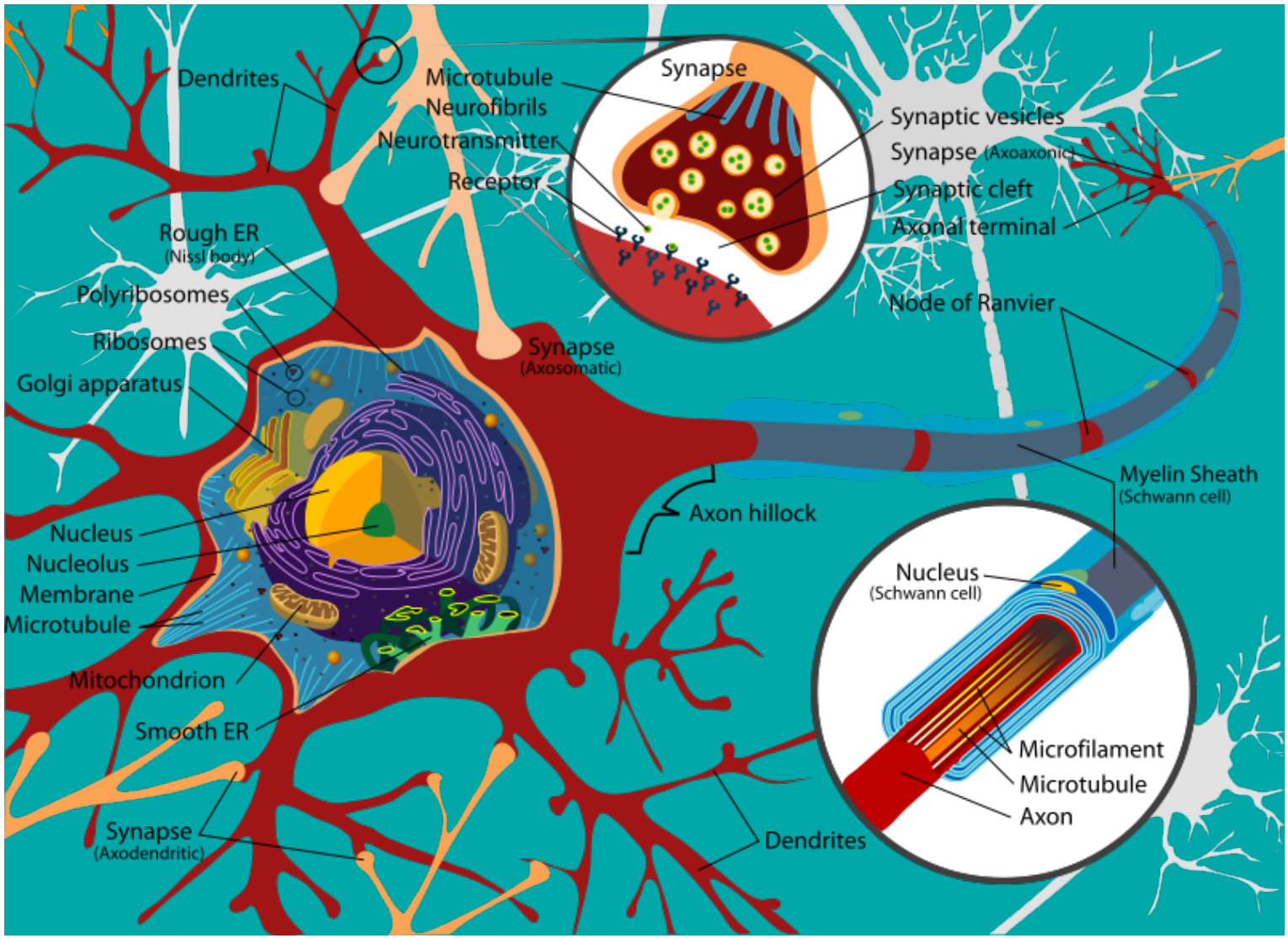
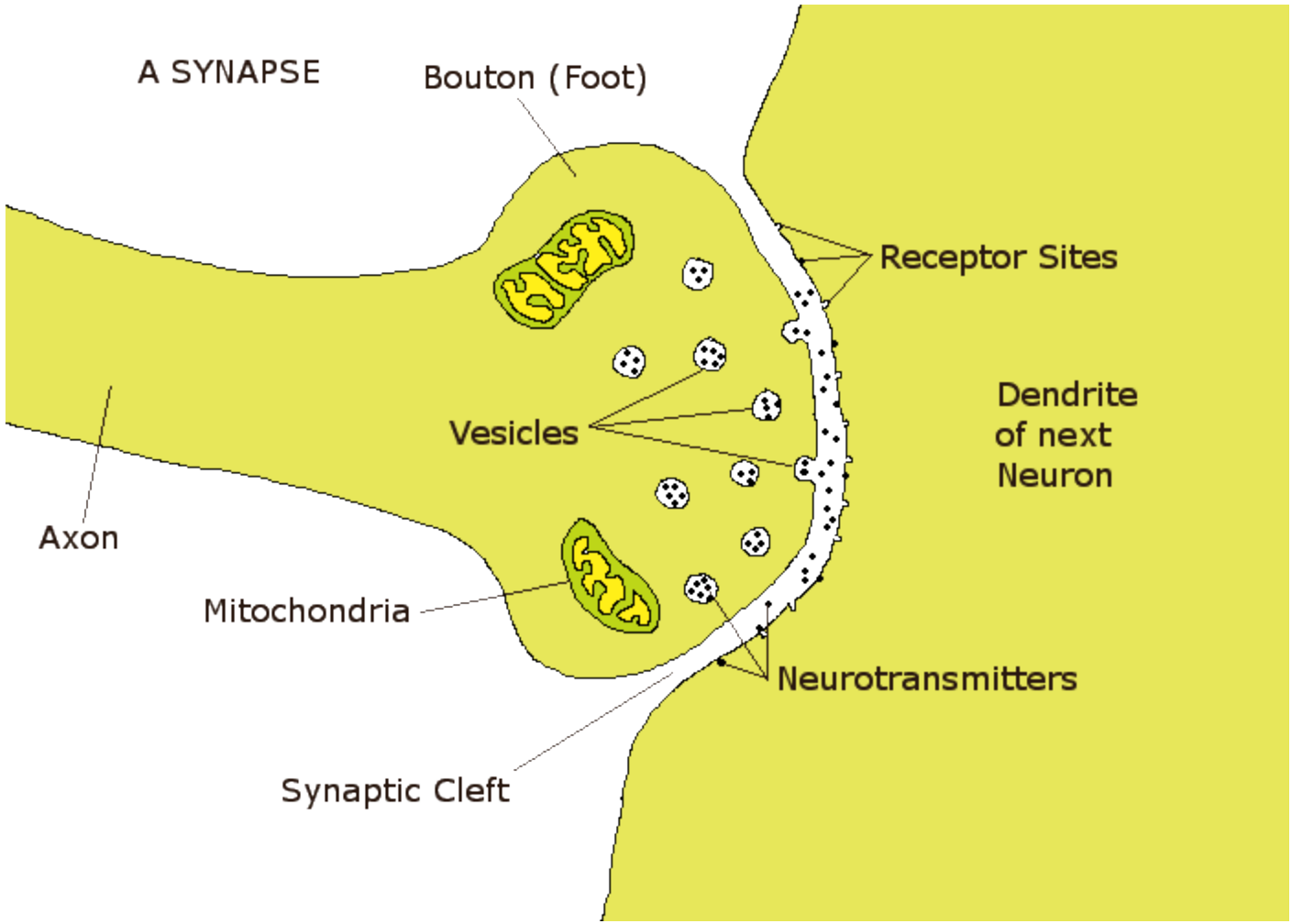
Modeling Delay in Spiking Neurons

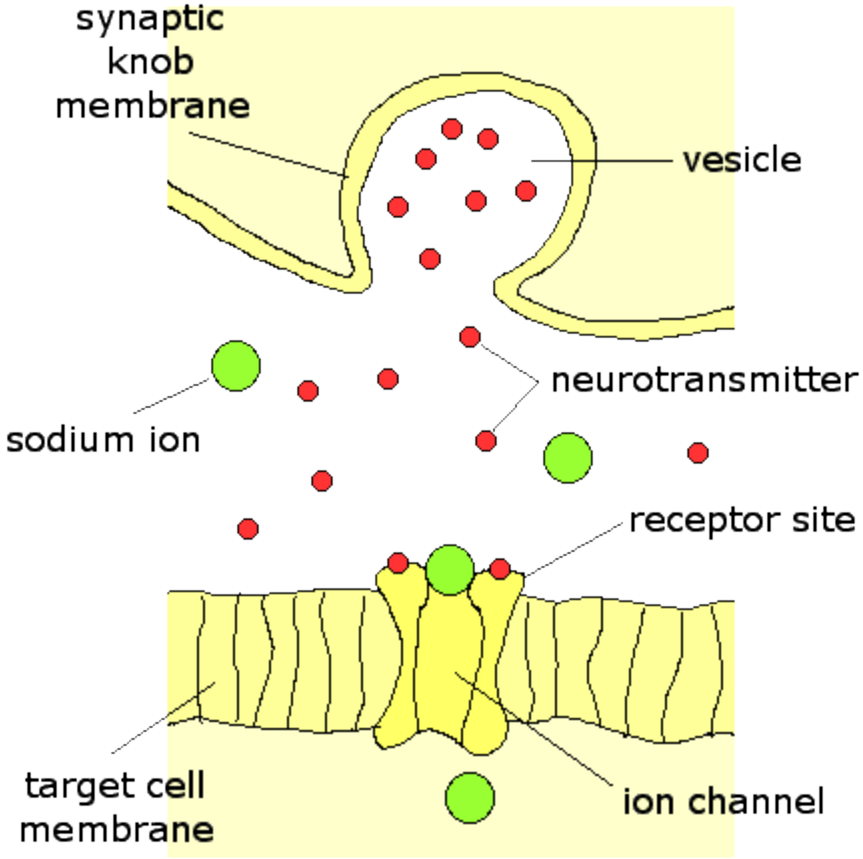
Misha Shvartsman
Dwight Nelson and Sean Ewen
University of St. Thomas
Midwest Numerical Analysis Day
May 3, 2008


A NEURON
Dendrites
Nucleus
Axon
Bouton (Foot)
Myelin Sheath
Soma (Body)


Dendrites
Microtubule
Neurofibrils
Neurotransmitter
Receptor
Synapse
Synaptic vesicles
Synapse (Axoaxonic)
Synaptic cleft
Axonal terminal
Rough ER
(Nissl body)
Polyribosomes
Ribosomes
Golgi apparatus
Synapse
(Axosomatic)
Node of Ranvier
Myelin Sheath
(Schwann cell)
Axon hillock
Nucleus
Nucleolus
Membrane
Microtubule
Mitochondrion
Smooth ER
Nucleus
(Schwann cell)
Microfilament
Microtubule
Axon
Synapse
(Axodendritic)
Dendrites


A SYNAPSE
Bouton (Foot)
Receptor Sites
Vesicles
Dendrite
of next
Neuron
Axon
Mitochondria
Neurotransmitters
Synaptic Cleft

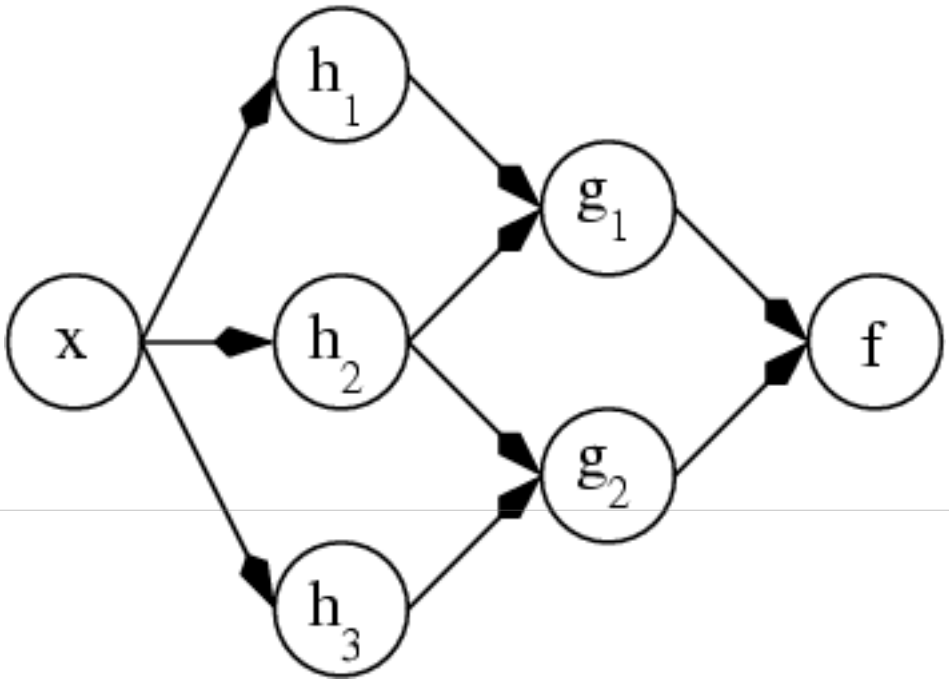
http://www.blackwellpublishing.com/matthews/actionp.html
http://www.blackwellpublishing.com/matthews/channel.html



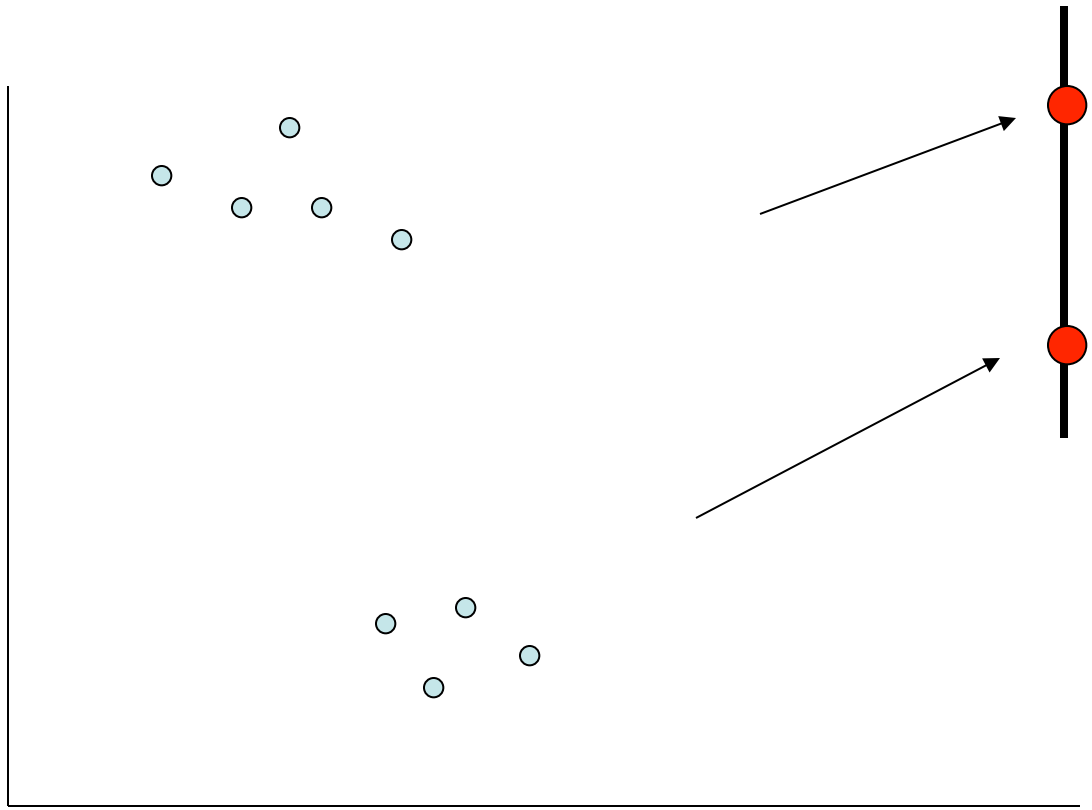
ANN DEPENDENCY GRAPH

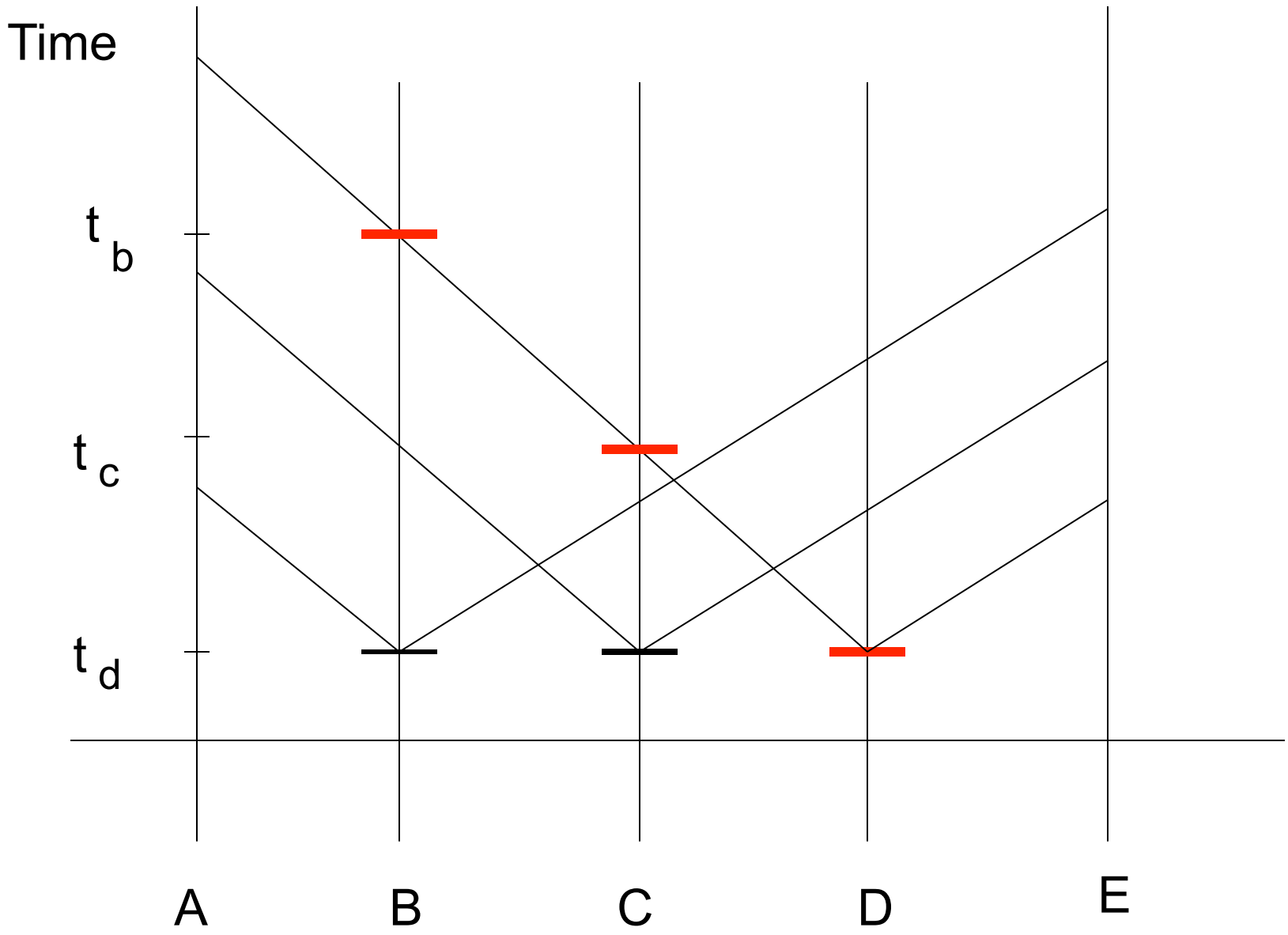
The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

$$f(x) = f(g_1(x), g_2(x), \ldots, g_n(x)) = K\left(\sum w_i g_i(x)\right)$$

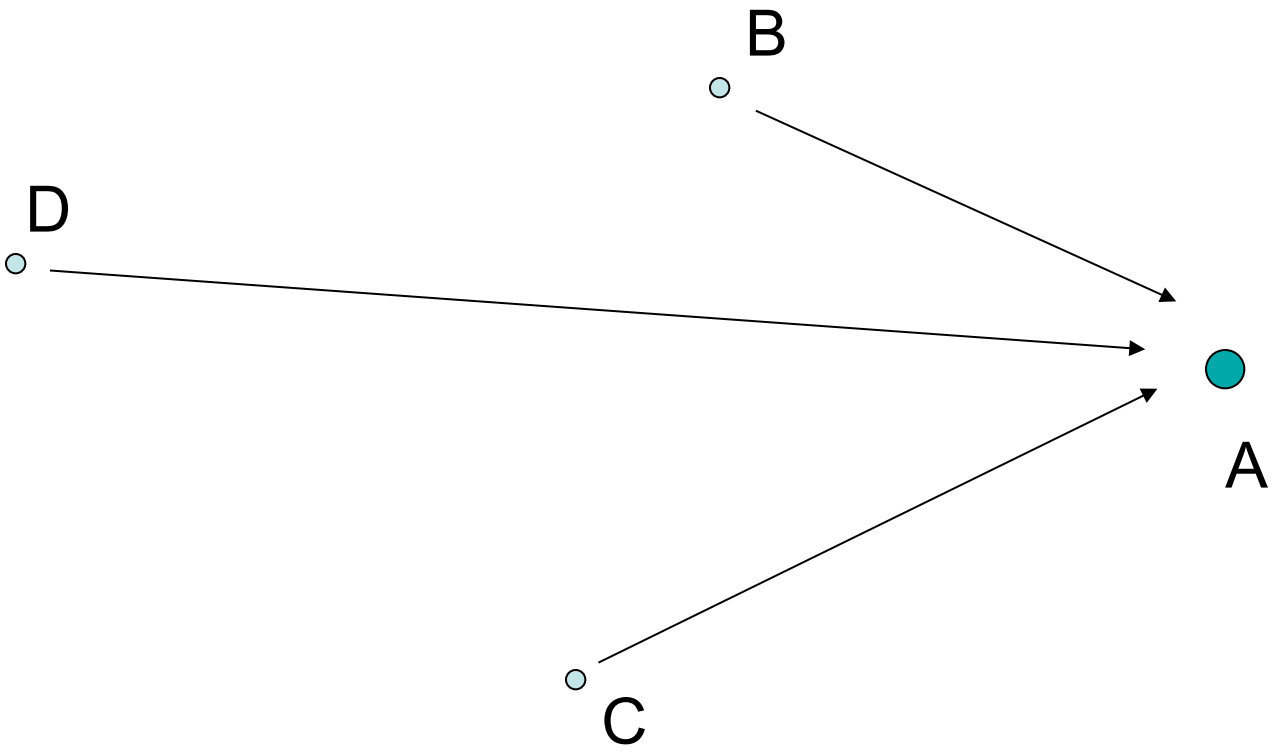


TRIAL AND ERROR
SUPEVISED LEARNING




Time
t b
t c
t d
A
B
C
D
E

CONDUCTION DELAY MATCHING



Quantum Brain

$$\Psi = \sum_{i=1}^{\infty} c_i \Psi_i$$

Synaptic Weight

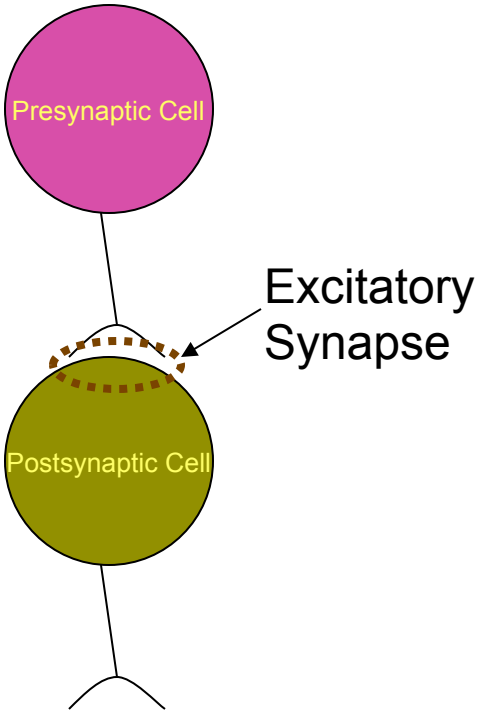
n = number of release sites

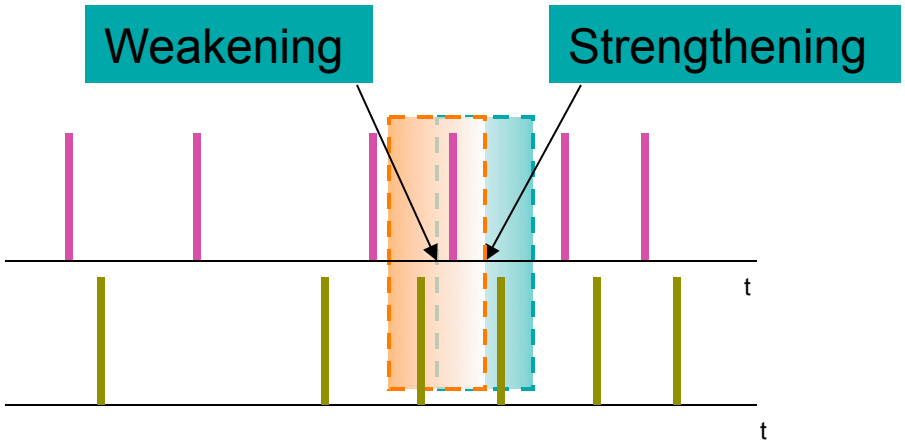
p = probability of synaptic release per site

q = measure of postsynaptic effect

$$R = npq$$

Spike Time Dependent Plasticity





Integrate and Fire Models

Non Leaky Integrate and Fire

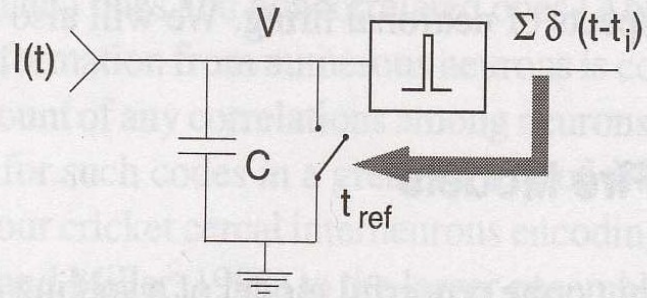
$$C(dV/dt) = I(t)$$

Forgetful or Leaky Integrate and Fire

$$C(dV/dt)+V(t)/R = I(t)$$

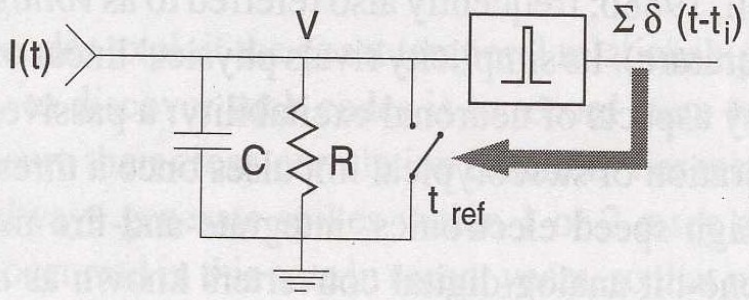
A)

Perfect Integrate-and-Fire Unit



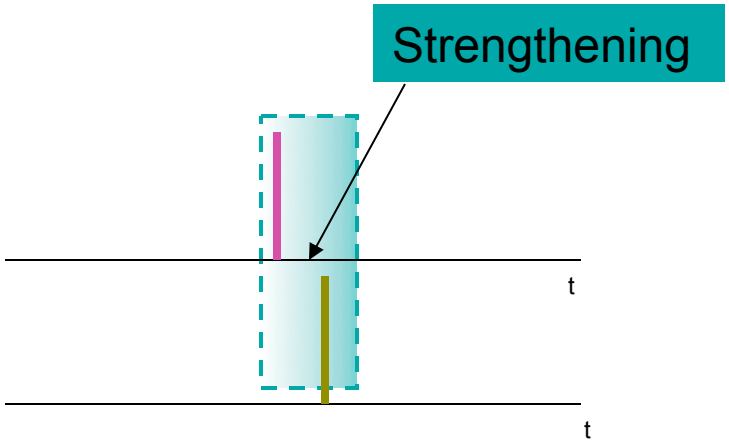
B)

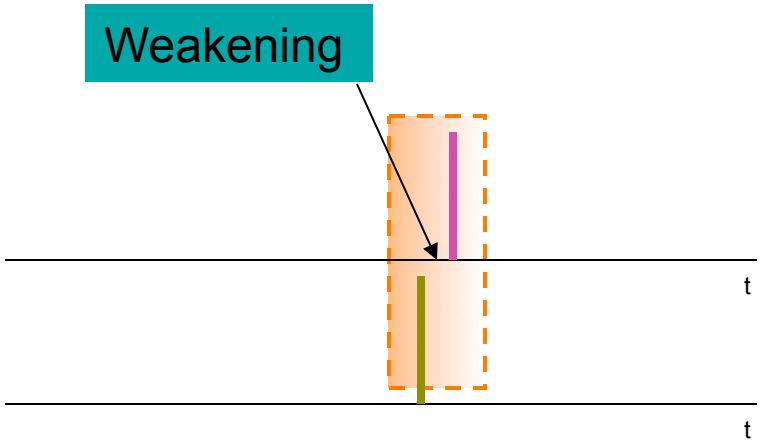
Leaky Integrate-and-Fire Unit

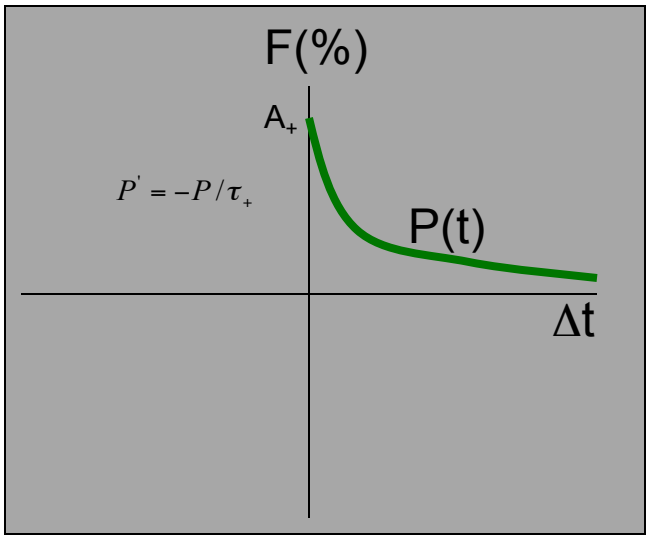


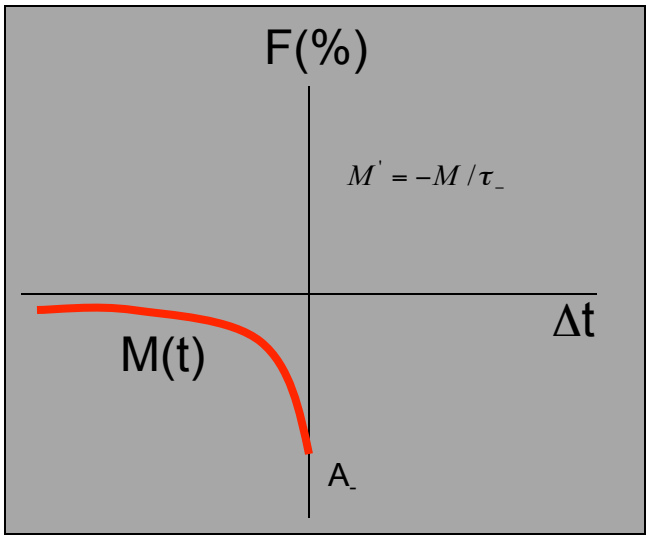
Song, Miller, Abbott Model for

$$F(\Delta t) = \begin{cases} A_+ \exp(-\Delta t / \tau_+) & if\ \Delta t > 0 \\ -A_- \exp(\Delta t / \tau_-) & if\ \Delta t \le 0 \end{cases}$$









FitzHugh-Nagumo Model

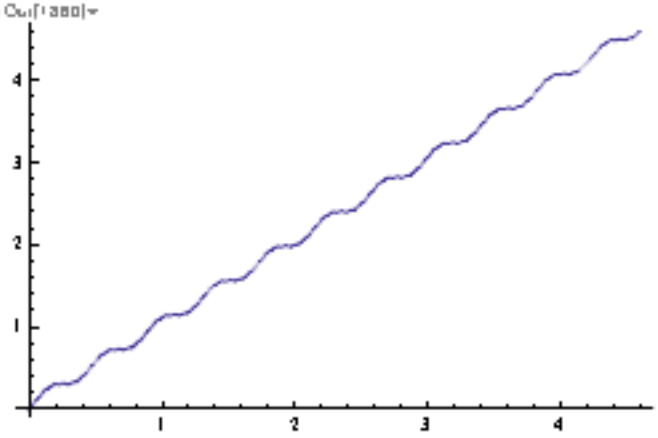
$$\dot{V} = f(V) - W + I$$

$$\dot{W} = a(bV - cW)$$

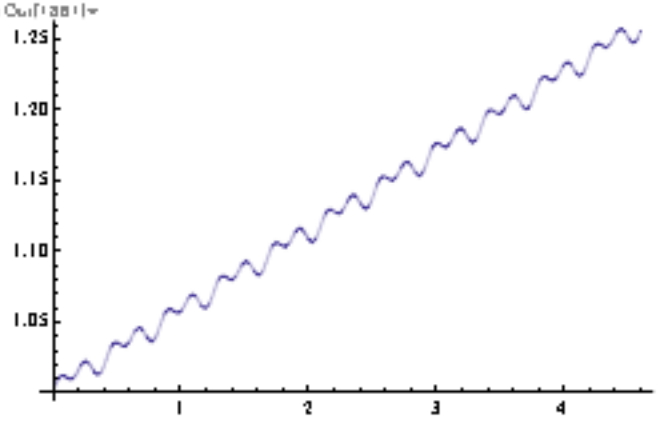
(Debug) In[352]:=

```
Directory[];
SetDirectory["/Users/user/Desktop/Delay"];
Clear[t0, T, tau, P1, P2, P3, P4, P5, P6, P7, P8, P9, z, G, H];
G[z_] = -2 * Sin[30 * z]; H[z_] = Cos[15 * z]; T = 4.6; t0 = 0; tau = 30; K = 1;
(*System with delay - NDelayDSolve Routine*)
Clear[x, t, y, s];
s = NDelayDSolve[{v'[t] == v[t - tau] - (v[t - tau])^3 / 3 - w[t - tau] + K,
    w'[t] == 0.08 * (v[t - tau] + 0.7 - 0.8 w[t - tau])}, {v → G, w → H}, {t, t0, T}];
P7 = Plot[Evaluate[v[t] /. s], {t, t0, T}];
P8 = Plot[Evaluate[w[t] /. s], {t, t0, T}];
P9 = Plot[Evaluate[{v[t], w[t]} /. s], {t, t0, T}, PlotStyle → {Red, Red}];
Show[P7]
Show[P8]
Show[P9]
```

(Debug) Out[380]:=

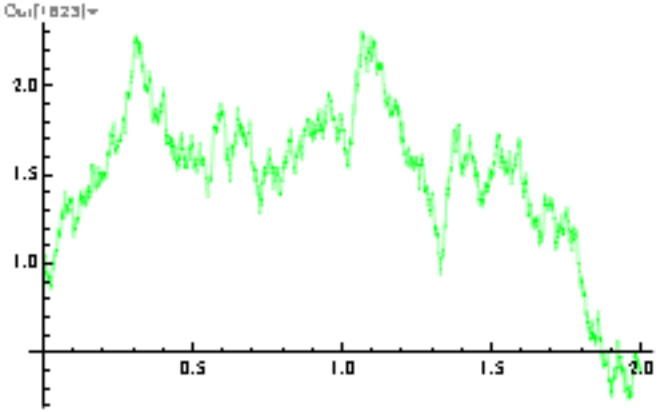


(Debug) Out[381]:=

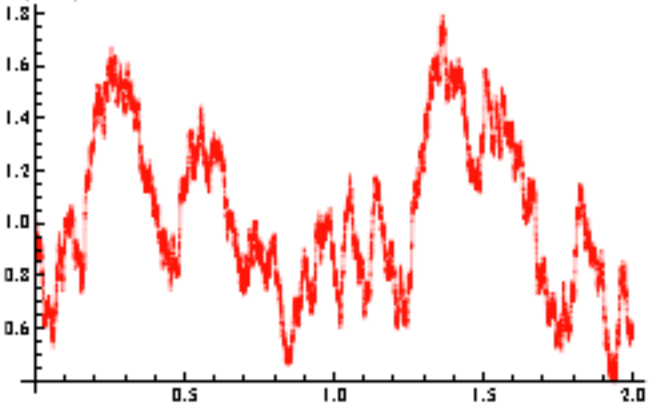


```
EM[{t0_, x0_}, h_, n_] :=
  (EMStep[{t_, x_}] := N[{t + h, x + Sqrt[h] * Random[NormalDistribution[0, 1]]}];
   NestList[EMStep, {t0, x0}, n]);
Clear[h, n, x, t, T, K, t0, x0]; t0 = 0; x0 = 1; T = 2; K = 1; h = 1 / K; n = T / h;
em1 = EM[{t0, x0}, h, n]; K = 10; h = 1 / K; n = T / h; em2 = EM[{t0, x0}, h, n];
K = 100; h = 1 / K; n = T / h; em3 = EM[{t0, x0}, h, n]; K = 1000; h = 1 / K; n = T / h;
em4 = EM[{t0, x0}, h, n]; K = 10000; h = 1 / K; n = T / h; em5 = EM[{t0, x0}, h, n];
P1 = ListPlot[em1, Joined → True, PlotStyle → RGBColor[0, 0, 0]];
P2 = ListPlot[em2, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
P3 = ListPlot[em3, Joined → True, PlotStyle → RGBColor[0, 0, 1]];
P4 = ListPlot[em4, Joined → True, PlotStyle → RGBColor[0, 1, 0]]
P5 = ListPlot[em5, Joined → True, PlotStyle → RGBColor[1, 0, 0]]
```

(Debug) Out[1823]=



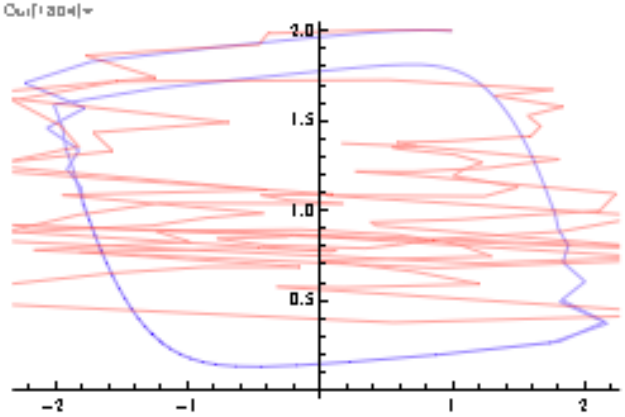
(Debug) Out[1824]=



(Debug) In[1335]:=

```
Clear[em1, em2, EM, EMS, f, g, b, t, u, v, t0, v0,
  w0, K, T, h, n, t1, t2, t3, t4, t5, t6, P1, P2, P3, P4, P5, P6];
EM[f_, g_, {t0_, v0_, w0_}, h_, n_] := (EMStep[{t_, v_, w_}] :=
    N[{t + h, v + h * f[t, v, w], w + h * g[t, v, w]}]; NestList[EMStep, {t0, v0, w0}, n]);
EMS[f_, g_, b_, {t0_, v0_, w0_}, h_, n_] := (EMStep[{t_, v_, w_}] :=
    N[{t + h, v + h * f[t, v, w] + Sqrt[h] * b[t, v, w] * Random[NormalDistribution[0, 1]],
      w + h * g[t, v, w]}]; NestList[EMStep, {t0, v0, w0}, n]);
f[t_, v_, w_] := v - (v^3 / 3) - w + K; g[t_, v_, w_] := 0.08 * (v + 0.7 - 0.8 w); b[t_, v_, w_] = 1;
v0 = 1; w0 = 2; t0 = 0; T = 60; n = 100; K = 1; h = T / n;
em1 = EM[f, g, {t0, v0, w0}, h, n];
t1 = Table[{em1[[i, 1]], em1[[i, 2]]}, {i, 1, n + 1}];
P1 = ListPlot[t1, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t2 = Table[{em1[[i, 1]], em1[[i, 3]]}, {i, 1, n + 1}];
P2 = ListPlot[t2, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t3 = Table[{em1[[i, 2]], em1[[i, 3]]}, {i, 1, n + 1}];
P3 = ListPlot[t3, Joined → True, PlotStyle → RGBColor[0, 0, 1]];
em2 = EMS[f, g, b, {t0, v0, w0}, h, n];
t4 = Table[{em2[[i, 1]], em2[[i, 2]]}, {i, 1, n + 1}];
P4 = ListPlot[t4, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t5 = Table[{em2[[i, 1]], em2[[i, 3]]}, {i, 1, n + 1}];
P5 = ListPlot[t5, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t6 = Table[{em2[[i, 2]], em2[[i, 3]]}, {i, 1, n + 1}];
P6 = ListPlot[t6, Joined → True, PlotStyle → RGBColor[1, 0, 0]];
Show[P3, P6]
```

(Debug) Out[1304]:=



```
t4 = Table[{em2[[i, 1]], em2[[i, 2]]}, {i, 1, n + 1}];
P4 = ListPlot[t4, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t5 = Table[{em2[[i, 1]], em2[[i, 3]]}, {i, 1, n + 1}];
P5 = ListPlot[t5, Joined → True, PlotStyle → RGBColor[0, 1, 0]];
t6 = Table[{em2[[i, 2]], em2[[i, 3]]}, {i, 1, n + 1}];
P6 = ListPlot[t6, Joined → True, PlotStyle → RGBColor[1, 0, 0]];
Show[P3]
Show[P6]
```