神經與行為模型建構
(Neural & Behavioral Modeling)

課號：Psy5352　　　　教室： 普 101
識別碼：227U2810　　　時間： 一 234


計算認知神經科學來囉
！

Modeling Principles

Response Regulation

Competitive Inhibition

Recurrent Excitation

Representation & Coding (1/3)

不管是現實或模型可分為 localist 與 distributed

小美	1	0	0	0
小明	0	1	0	0
老張	0	0	1	0
老黃	0	0	0	1

小美	1	1	0	0
小明	0	1	1	0
老張	0	0	1	1
老黃	1	0	0	1

Localist coding 非常精確但不穩固

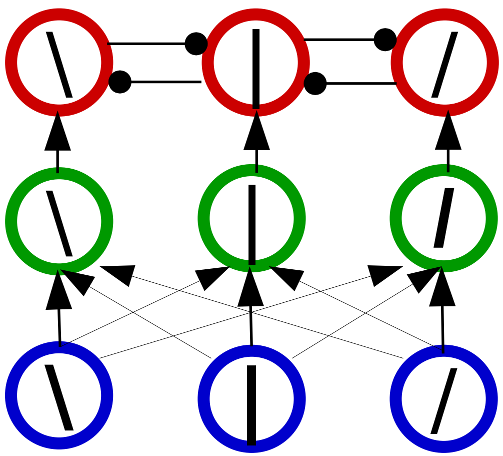
祖母神經元死了就無法辨認祖母了

Distributed coding 較有效率但較不精確

可用 N 個神經元表徵 N 個以上的東西

Representation & Coding (2/3)

模型常簡化未知的表徵為 localist(如 input/output)

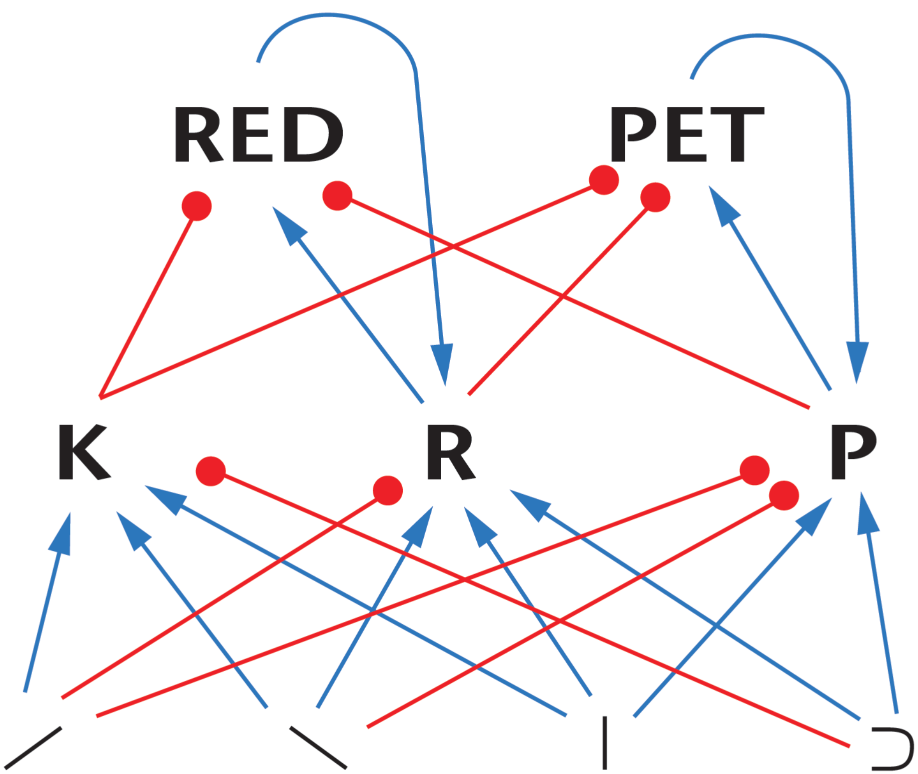



```
# Case 1: x=[0,4]
x,y=arange(0,4,0.1),array([])
for c in range(1,4):
    y=g(x,c) # Gaussian-like responses
    plot(x,y) # tuning/response function of y
legend(['y1','y2','y3'])
# Case 2: x=1
x,y=1,[g(x,1),g(x,2),g(x,3)]
z=around(y==max(y)) # winner takes all
```

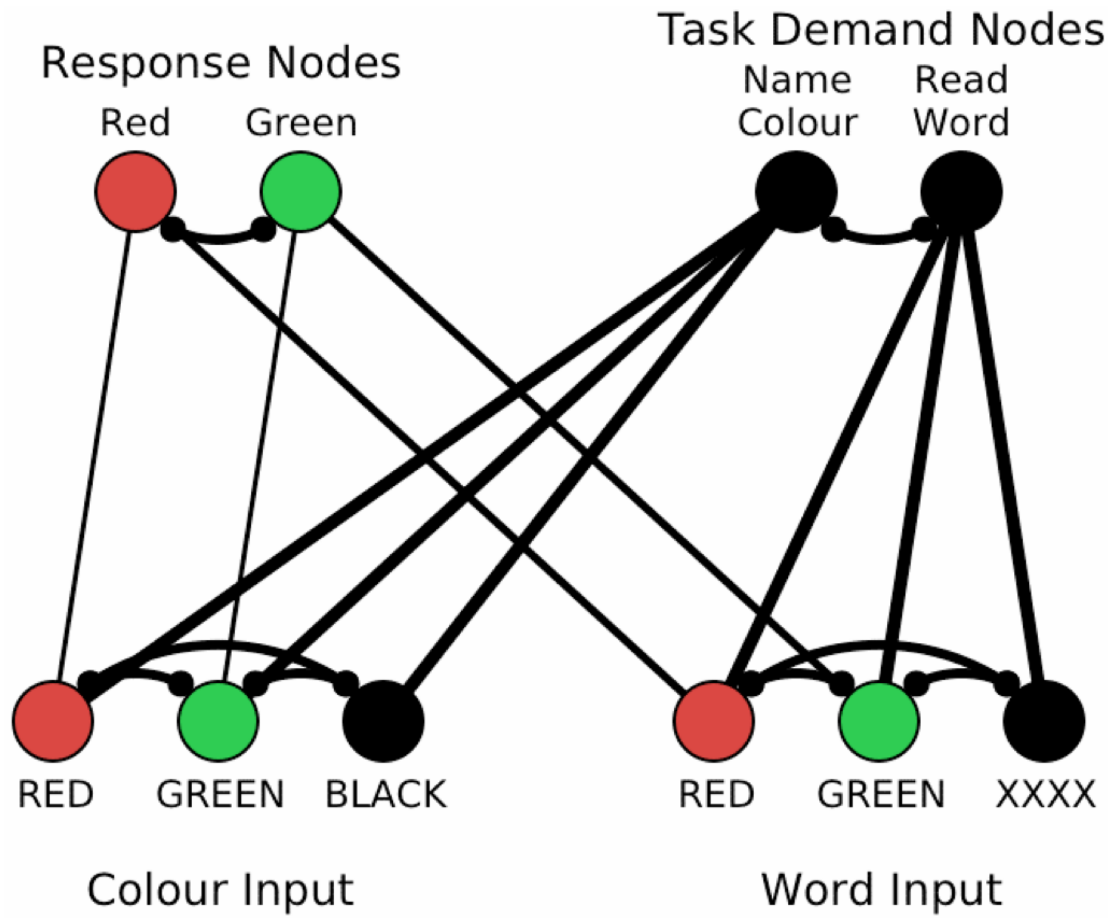
Representation & Coding (3/3)

偏心智的模型內因不知神經表徵而多為 localist

Recognition model

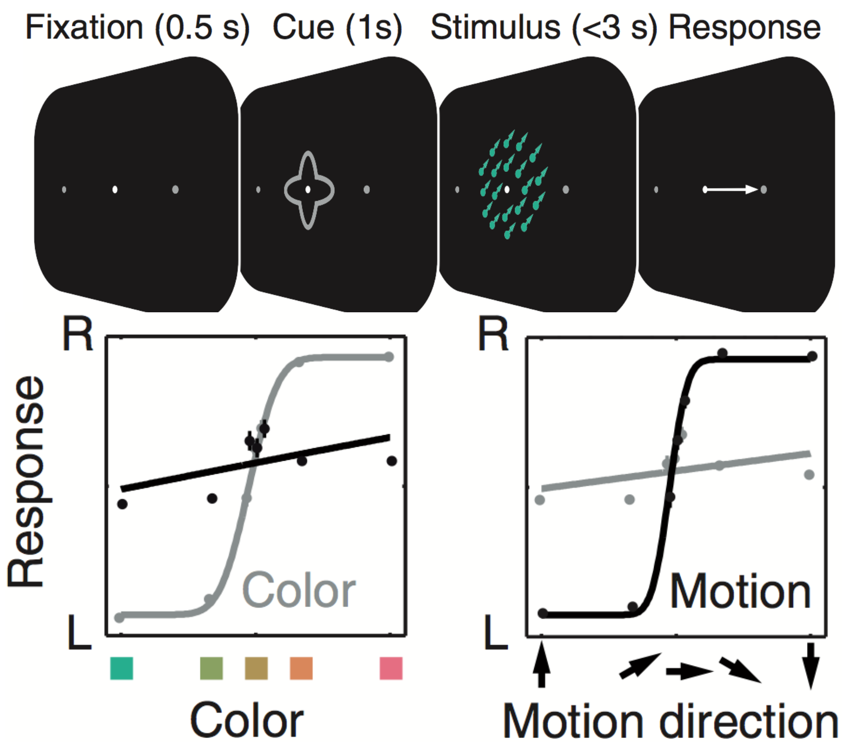


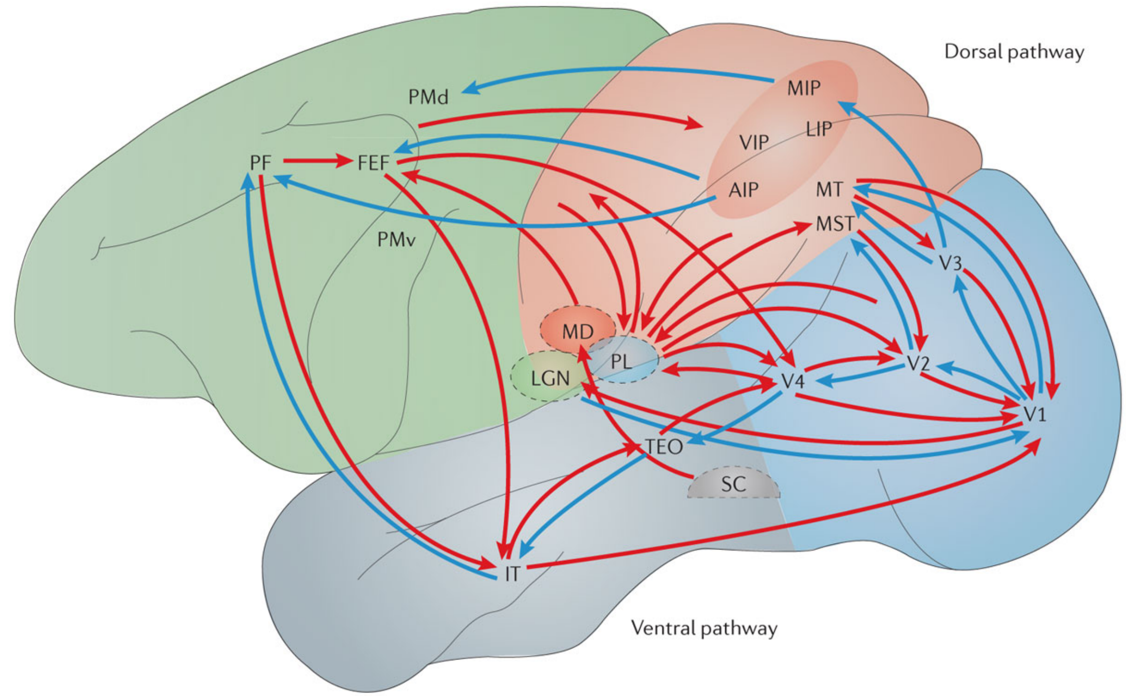
Stroop model

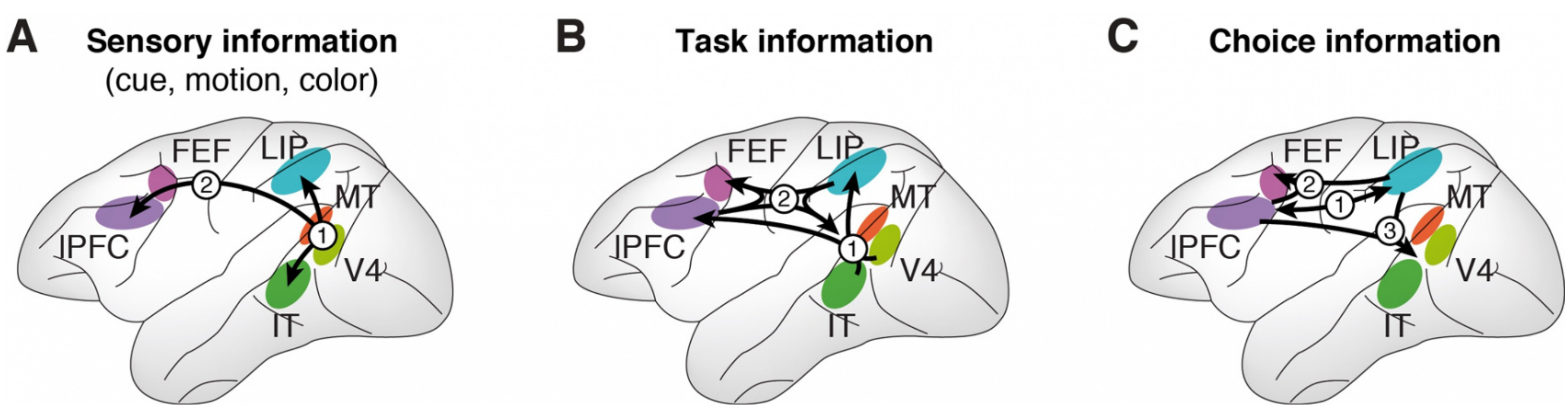


CCN 模型建構的原則 (1/3)

解剖限定結構；電生理限定計算；目標模擬行為

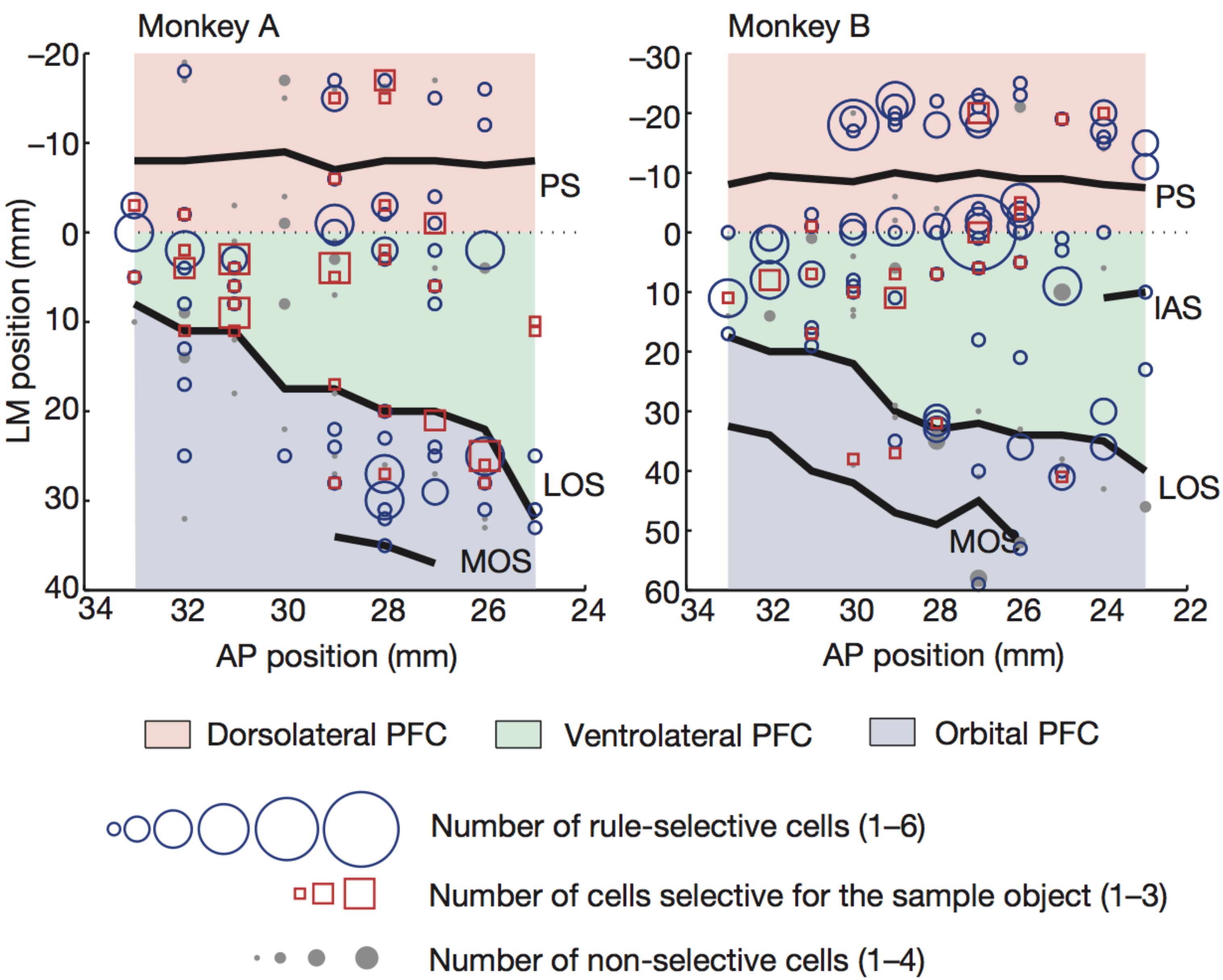






CCN 模型建構的原則 (2/3)

偏神經的模型內模擬一個腦區中的 N 類神經元



CCN 模型建構的原則 (3/3)

通常臨近的腦區或神經元才彼此相連 (locality)

違反局部性的
on-center
off-surround:

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i) I_i - x_i \sum_{k=1}^{n} DI_k$$

遵守局部性的
on-center
off-surround:

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i) \sum_{k=1}^{n} C_{ik} I_k - x_i \sum_{k=1}^{n} D_{ik} I_k$$

$$C_{ik} = Ce^{-\mu(i-k)^2}; D_{ik} = De^{-\nu(i-k)^2}$$

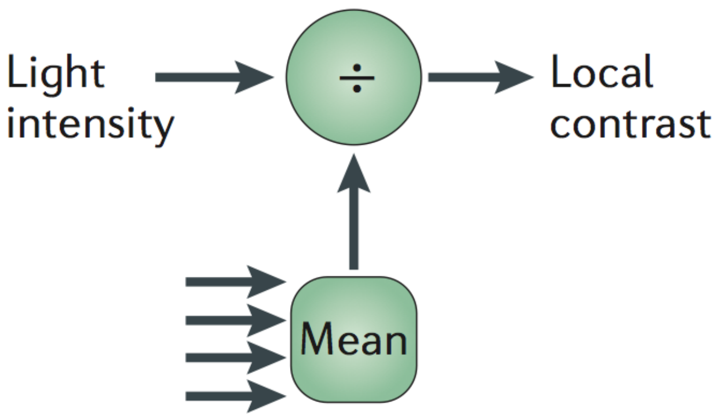
違反局部性的算法有<u>反傳播學習</u>，全域的<u>贏者全拿</u>等

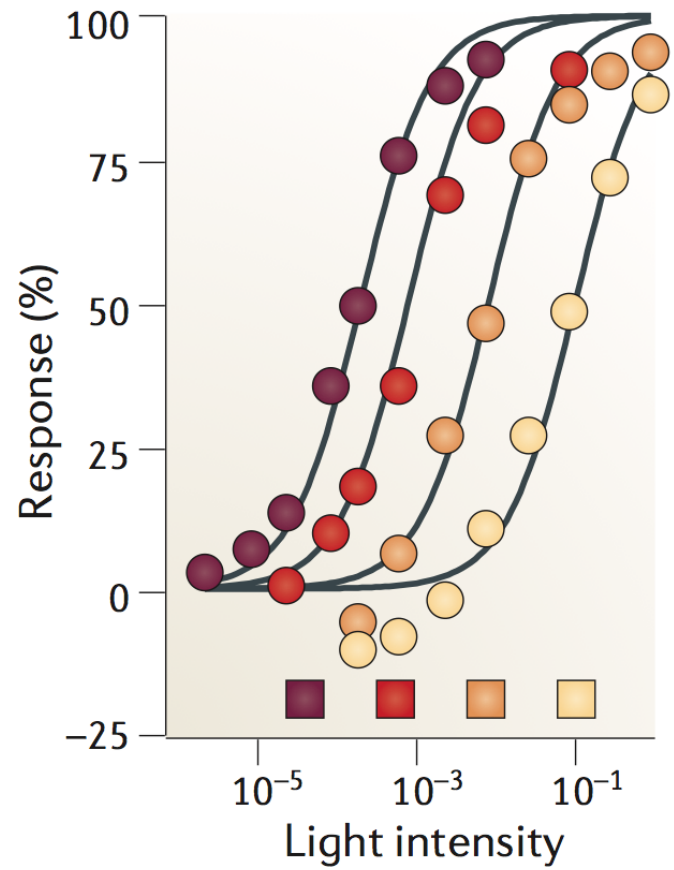
Modeling Principles
Response Regulation
Competitive Inhibition
Recurrent Excitation

看到這行字了嗎？

Normalization=Adaptation

神經反應會適應刺激動態調整反應大小

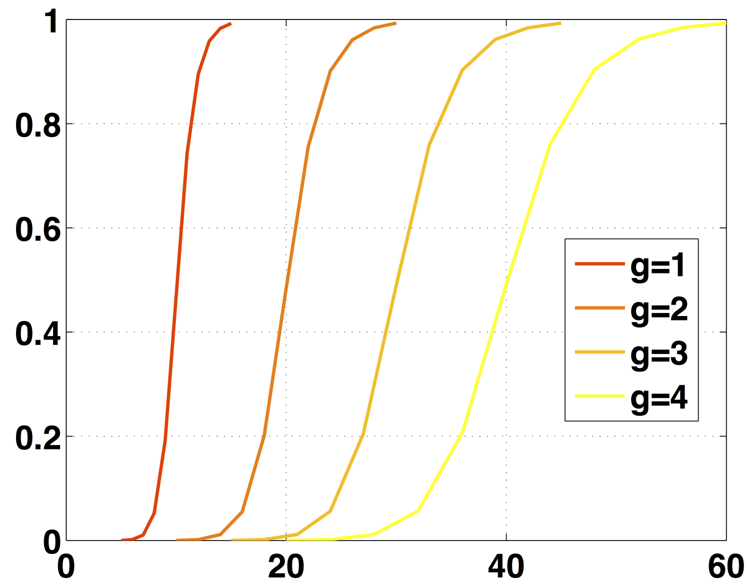




否則在刺激太弱 / 強時無法區分不同刺激相對大小

Phase Shift 的模擬

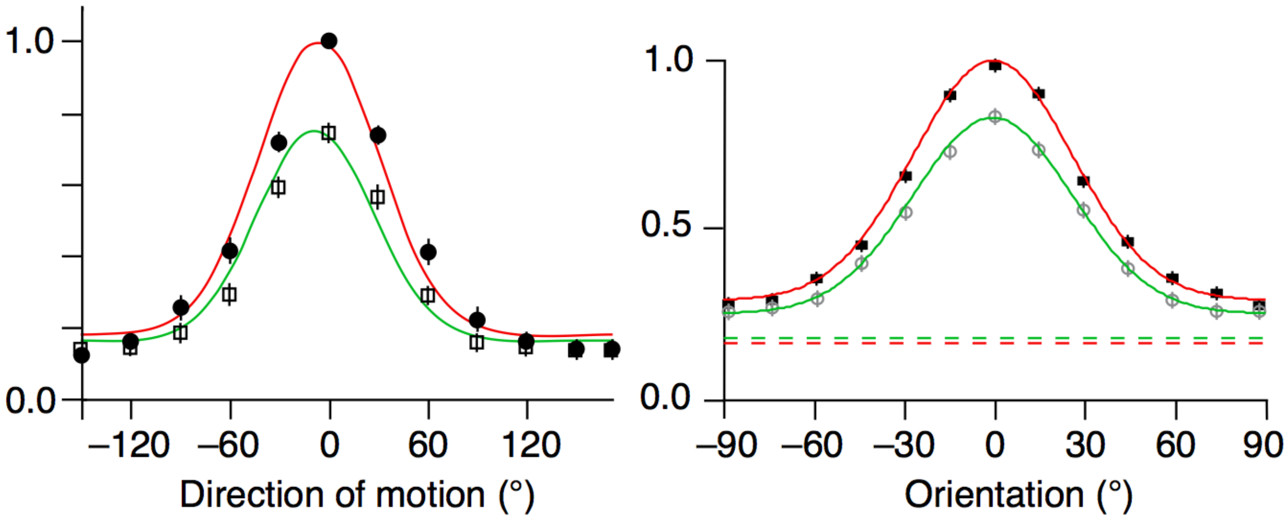
$dy/dt=-Ay+(1-y)*E-y*I=0 \Rightarrow y_\infty=E/(A+E+I)$

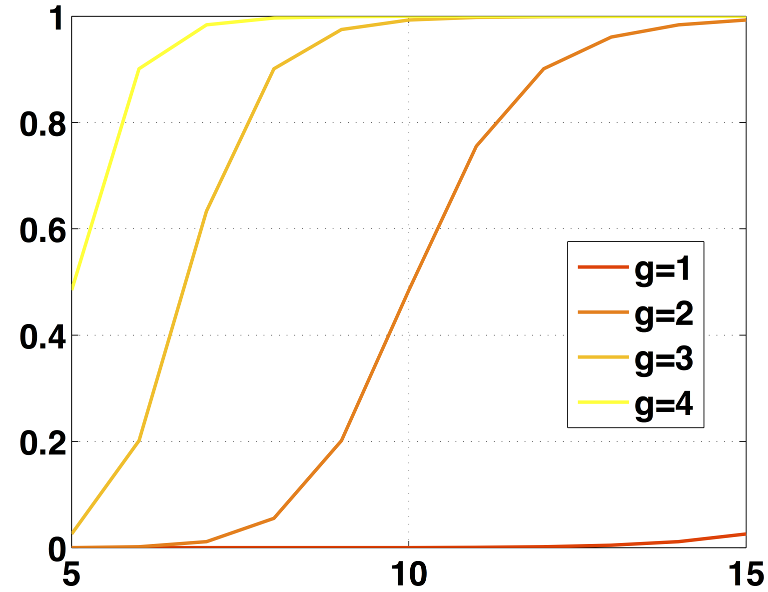



```
A,I0,E0=.1,10,arange(6,16)
for g in range(1,5): # multiplicative gains
    I,E=g*I0,g*E0 # mutual inhibitions also ↑
    y=sigmoid(E/(A+E+I),50,0.5);
    plot(E,y,color=[1,g/4,0]) # specify [r,g,b]
legend(['g=1','g=2','g=3','g=4']);
```

Attention=Gain Control (1/2)

注意力會用乘法增強神經反應的對比

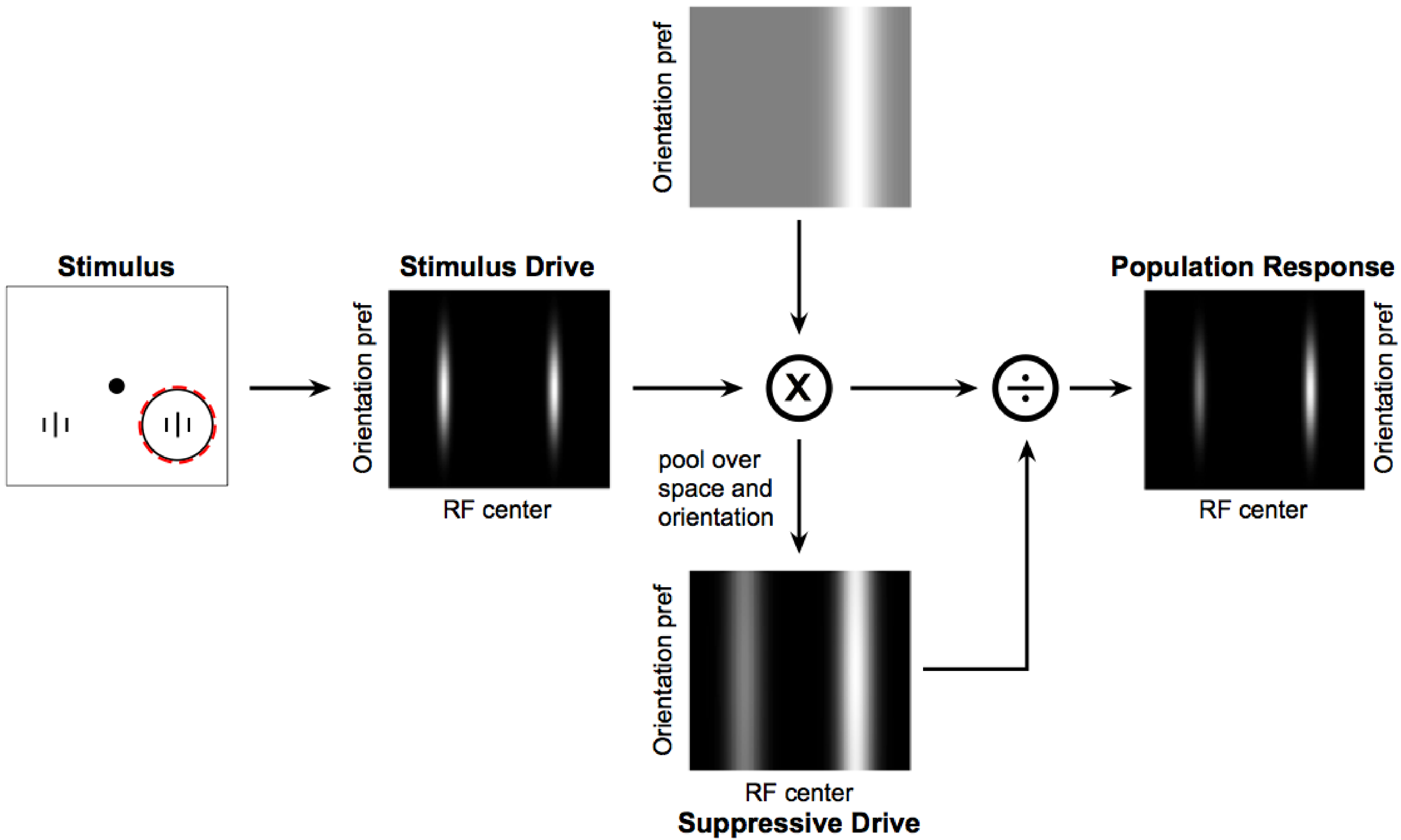





```
A,I0,E0=.1,10,arange(6,16)
for g in range(1,5): # multiplicative gains
    I,E=2*I0,g*E0
    y=sigmoid(E/(A+E+I),50,0.5);
    plot(E0,y,color=[1,g/4,0]) # specify [r,g,b]
legend(['g=1','g=2','g=3','g=4']);
```

Attention=Gain Control (2/2)

注意力會用乘法增強知覺的對比



行為是 bottom-up 與 top-down 神經計算的總效果

神經反應調節的總結

假設 Top-down Excitation 是 G，則比較好的方程是：

$$dy/dt=-Ay+(1-y)*E*G-y*I$$

或：

$$dy/dt=-Ay+(1-y)*E*(1+G)-y*I$$

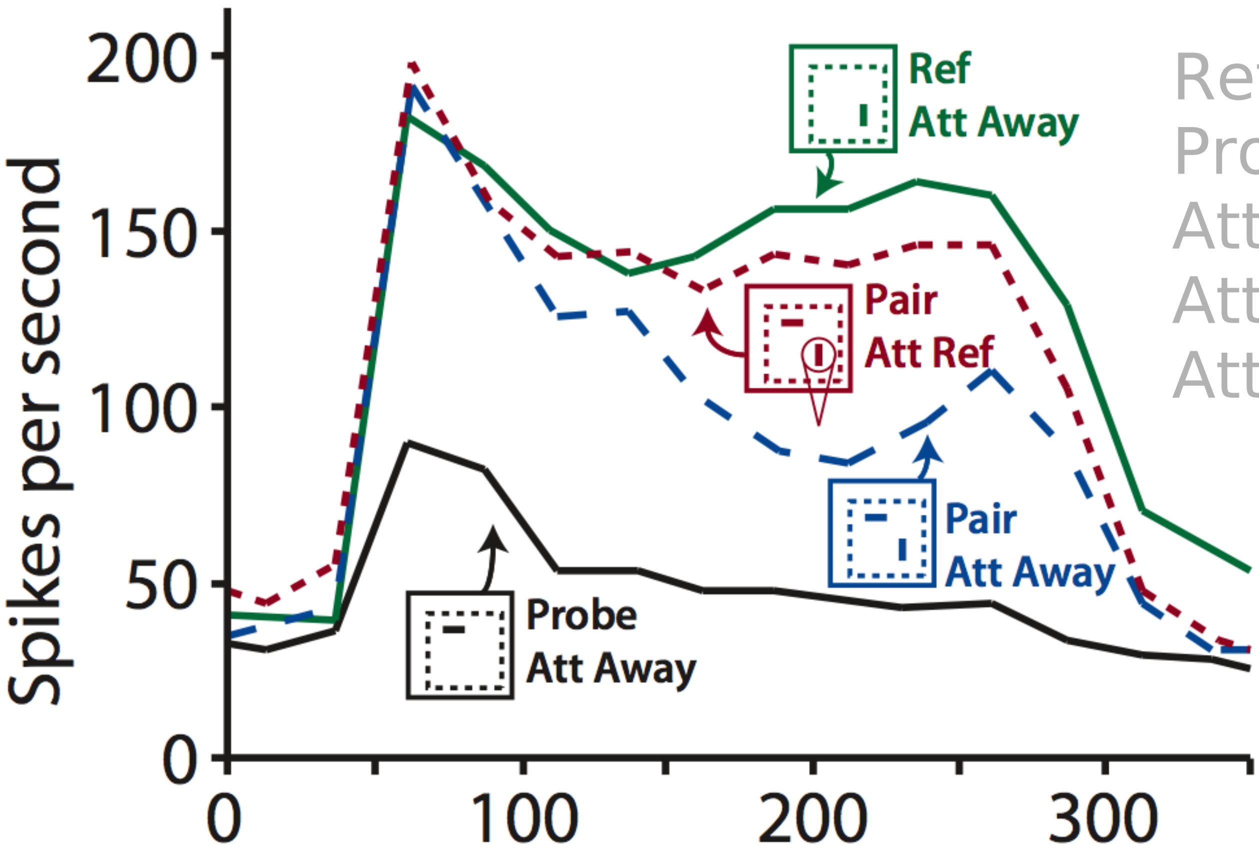
而不是：

$$dy/dt=-Ay+(1-y)*(E+G)-y*I$$

因為當 E=0 時 G 會使 y 產生幻覺

本週作業

先比較綠線與黑線來了解此神經元的 tuning property:



Ref=reference/vertical bar
Probe=probe/horizontal bar
Att=attention
Att Ref=attend reference
Att Away=attend away

再參考 Reynolds & Desimone (1999)圖 10 模擬上圖

Modeling Principles

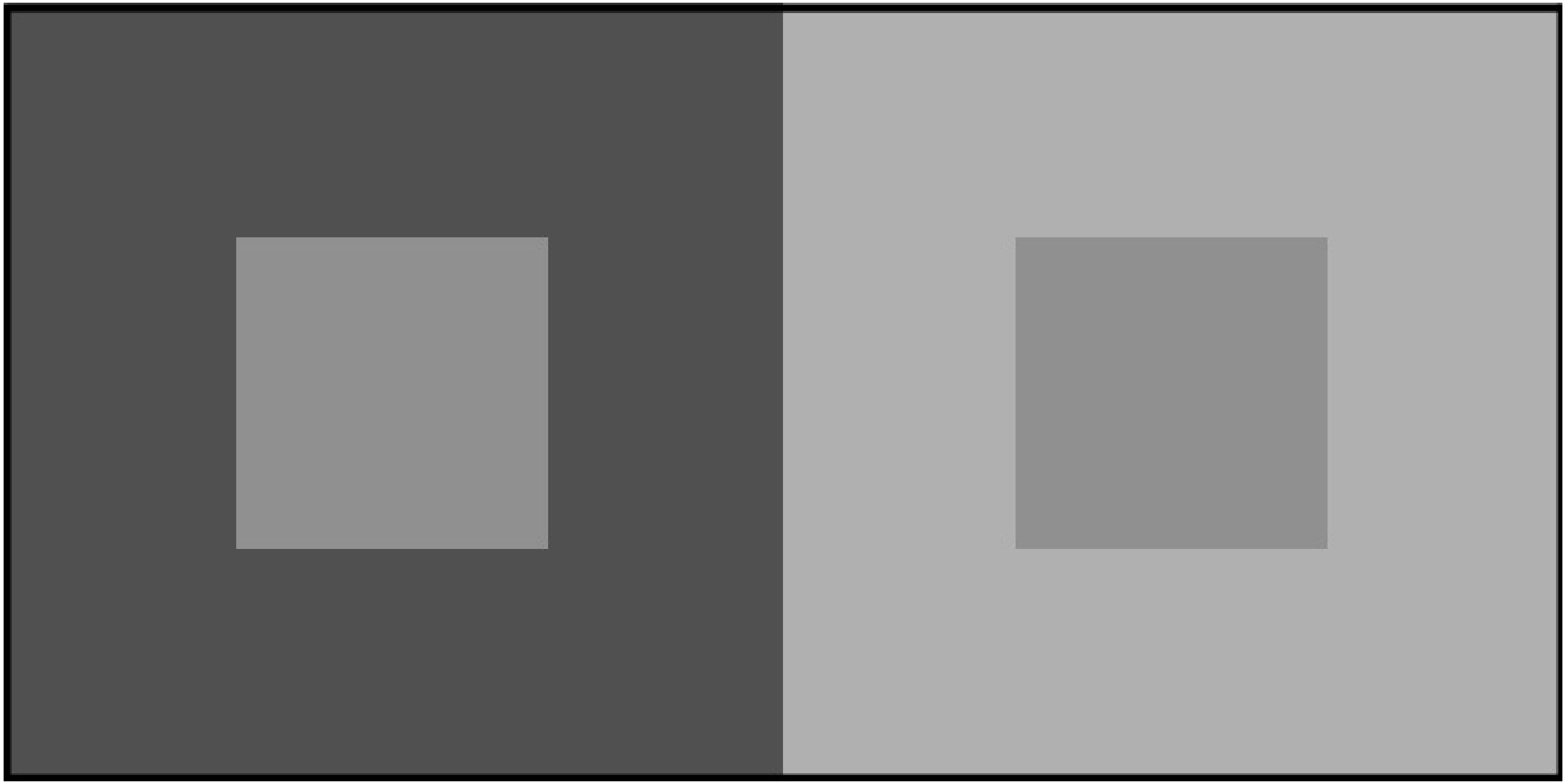
Response Regulation

Competitive Inhibition

Recurrent Excitation

Simultaneous Contrast

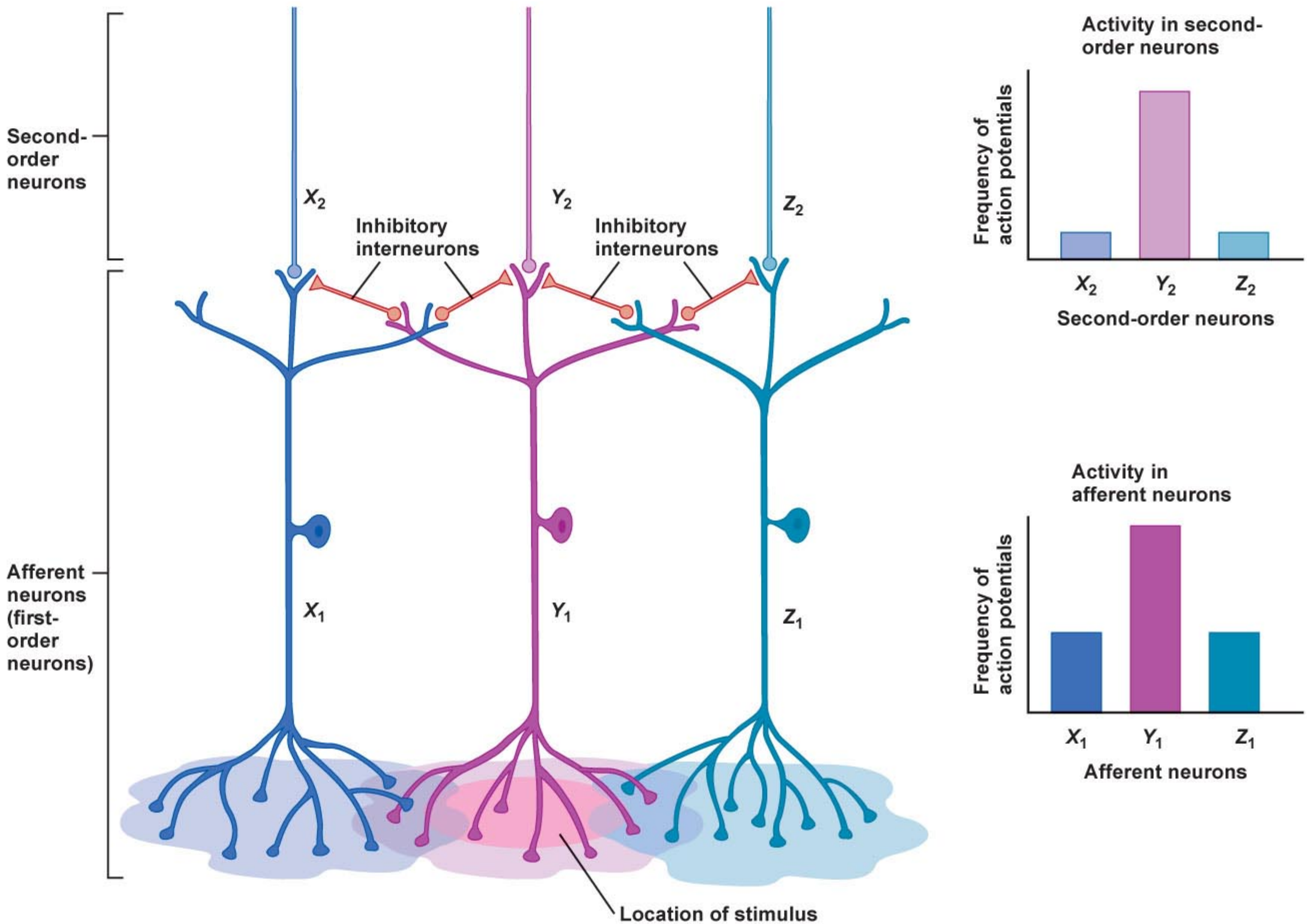
下圖中的兩小塊灰片的 lightness 相同



環境的 lightness≠ 知覺的 brightness

神經間的競爭抑制可加強反應的對比

此機制可用來解釋 simultaneous contrast



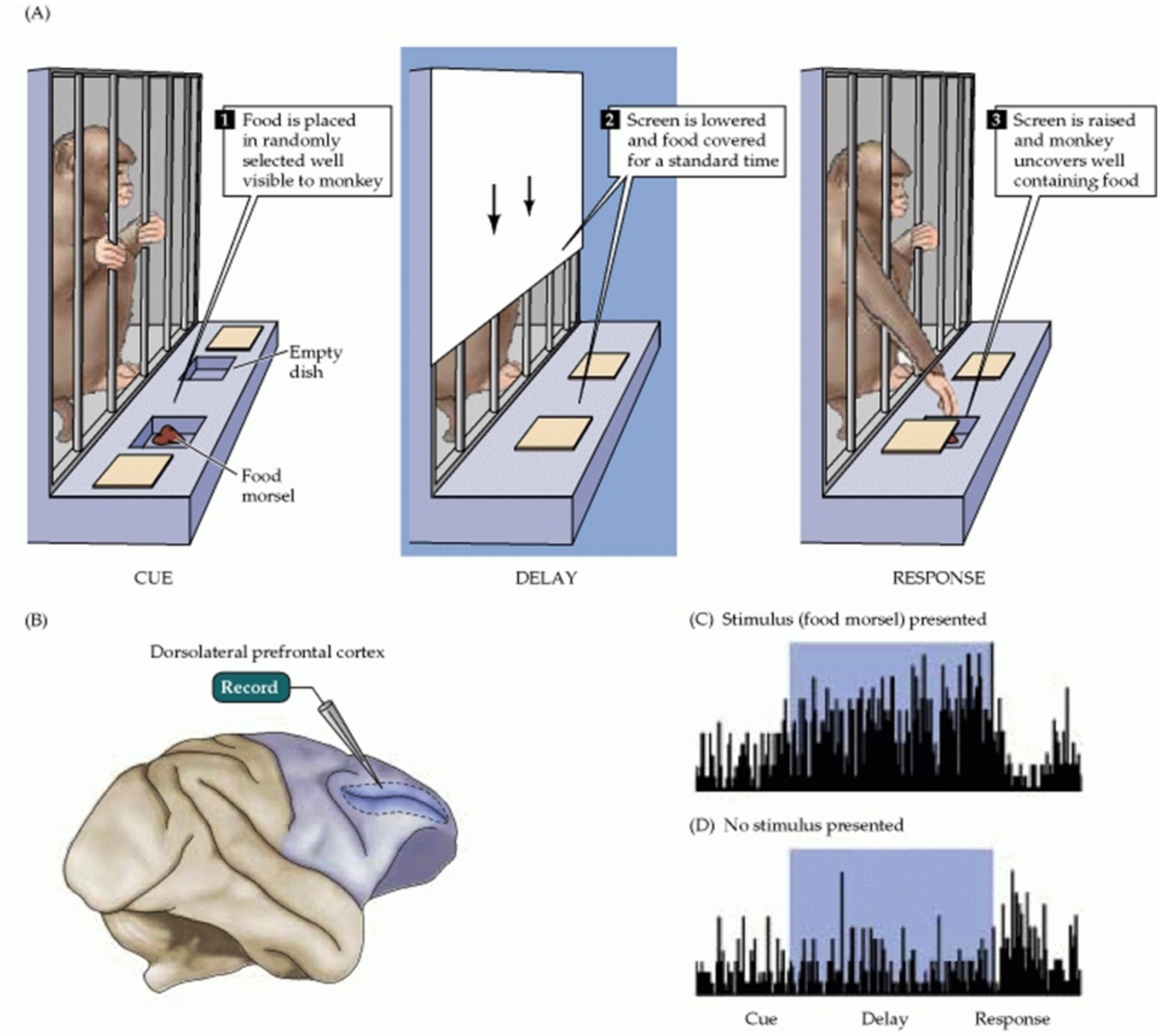
抑制造成對比加強的模擬


```
x=[0,0]; y=[0,0]; s=[1,10]; dt=0.1
for t in arange(0,5,dt):
    x[0]=x[0]+dt*(-x[0]+(1-x[0])*s[0])
    x[1]=x[1]+dt*(-x[1]+(1-x[1])*s[1])
    y[0]=y[0]+dt*(-y[0]+(1-y[0])*S[0]-y[0]*y[1])
    y[1]=y[1]+dt*(-y[1]+(1-y[1])*S[1]-y[1]*y[0])
    clf(); plot([1,2],x,'-o'); plot([1,2],y,'-o')
    ylim([0,1]); legend(['x','y']); title('t='+str(t))
    display(gcf()); clear_output(wait=True)
```

Modeling Principles
Response Regulation
Competitive Inhibition
Recurrent Excitation

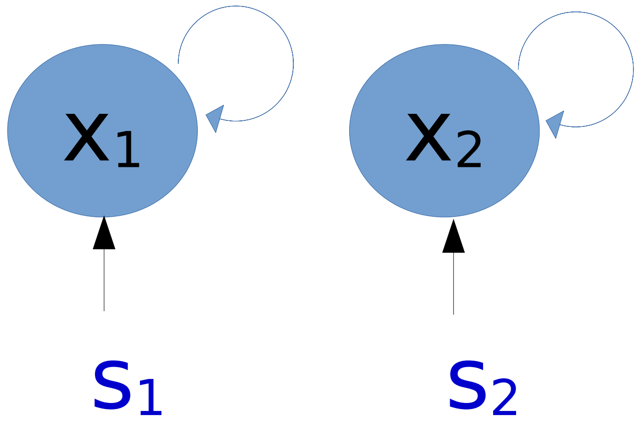
短期記憶與時間前後的脈絡

外界刺激消失後神經反應仍能維持數秒



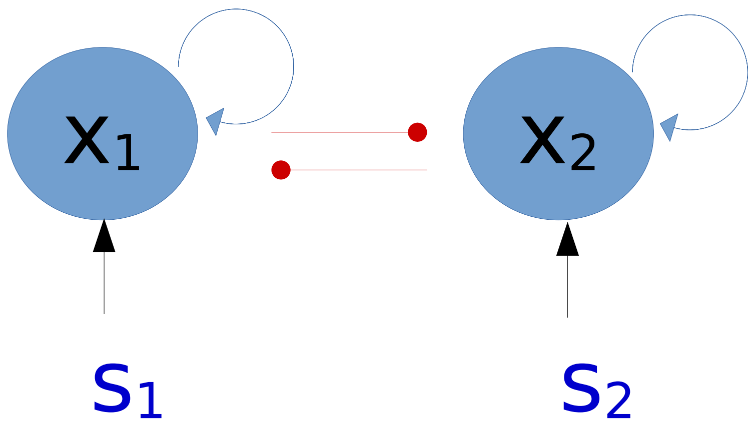
Self-Recurrent Excitations

自我連結的刺激可維持反應卻無法維持刺激對比



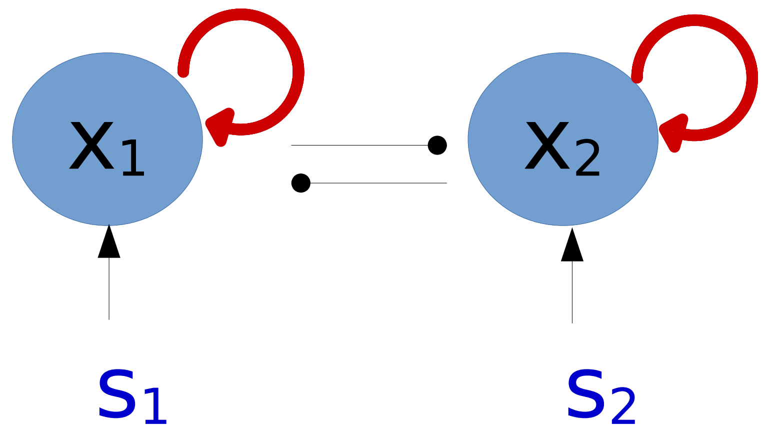

```
x=[0,0]; dt=0.1
for t in arange(0,10,dt):
    s=[1,10] if t<1 else [0,0]
    x[0]=x[0]+dt*(-0.1*x[0]+(1-x[0])*(s[0]+x[0]))
    x[1]=x[1]+dt*(-0.1*x[1]+(1-x[1])*(s[1]+x[1]))
    clf(); plot([1,2],x,'-o')
    ylim([0,1]); title('t='+str(t))
    display(gcf()); clear_output(wait=True)
```

可加入競爭抑制來加強對比




```
x=[0,0]; dt=0.1
for t in arange(0,10,dt):
  s=[1,10] if t<1 else [0,0]
  x[0]+=dt*(-0.1*x[0]+(1-x[0])*(s[0]+x[0])-x[0]*x[1])
  x[1]+=dt*(-0.1*x[1]+(1-x[1])*(s[1]+x[1])-x[1]*x[0])
  clf(); plot([1,2],x,'-o')
  ylim([0,1]); title('t='+str(t))
  display(gcf()); clear_output(wait=True)
```

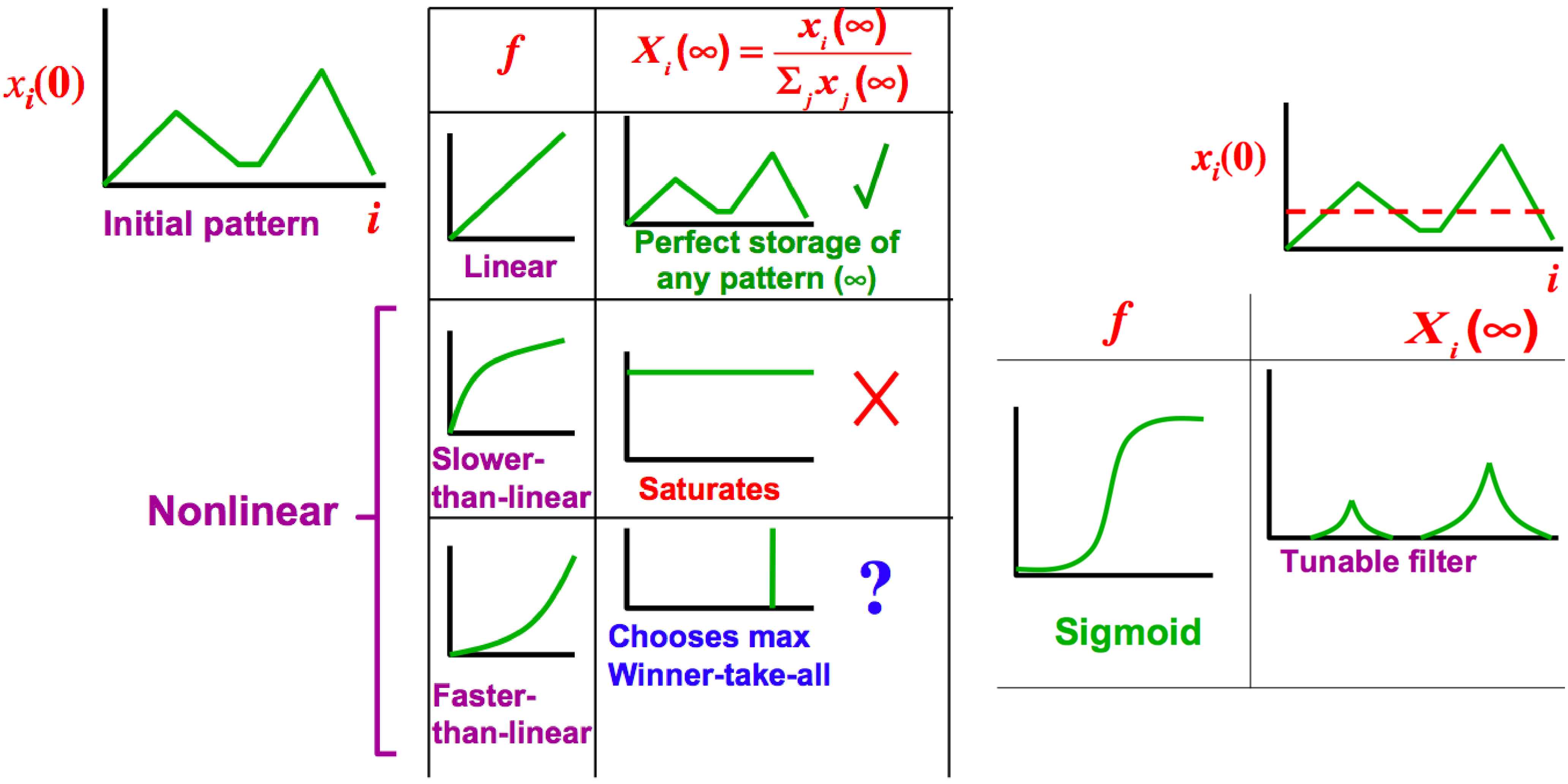
對比加強到極致可產生贏者全拿現象




```
x=[0,0]; dt=0.1
for t in arange(0,10,dt):
  s=[1,10] if t<1 else [0,0]
  x[0]+=dt*(-0.1*x[0]+(1-x[0])*(s[0]+x[0]**2)-x[0]*x[1])
  x[1]+=dt*(-0.1*x[1]+(1-x[1])*(s[1]+x[1]**2)-x[1]*x[0])
  clf(); plot([1,2],x,'-o')
  ylim([0,1]); title('t='+str(t))
  display(gcf()); clear_output(wait=True)
```

Signal Function 的影響

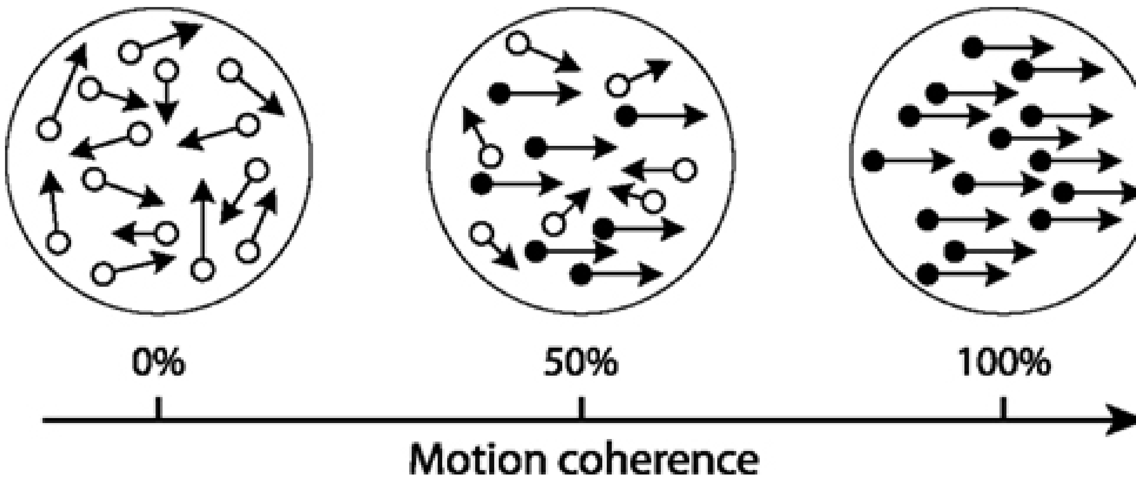
Grossberg (1973)闡明了 sigmoid function 的妙用

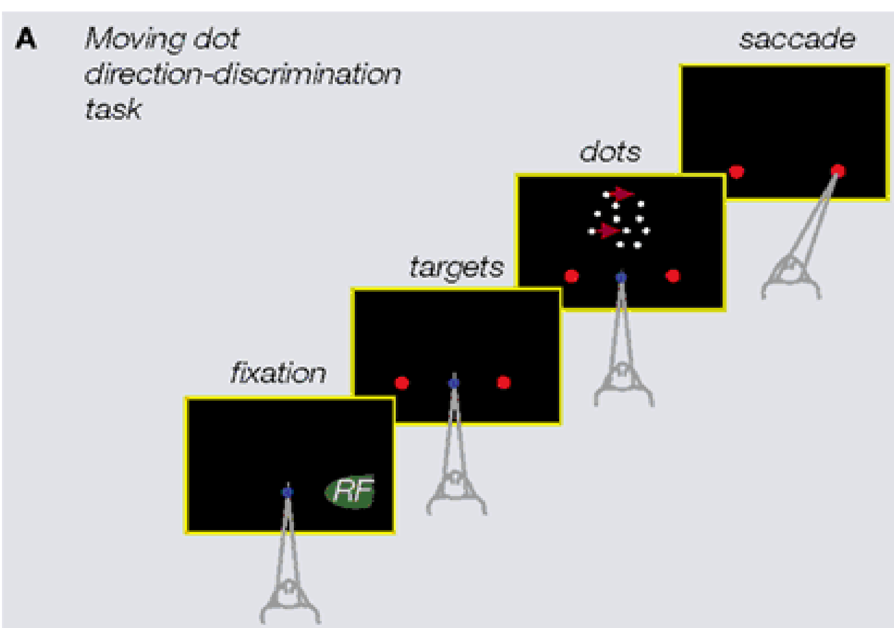


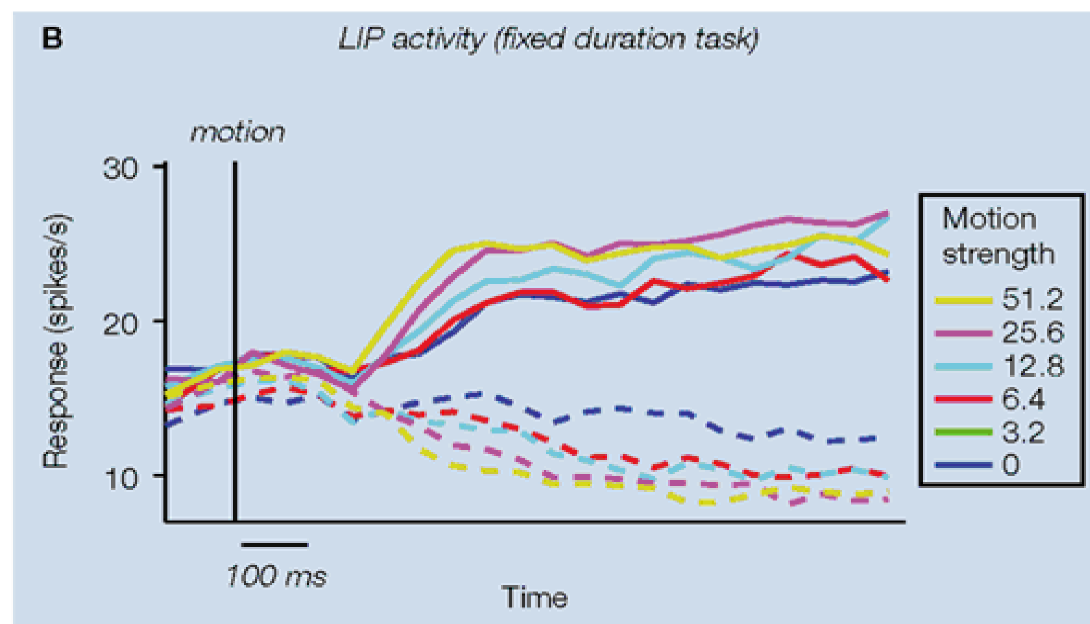
sigmoid 前段壓抑雜訊；中段保留訊號；後段限定上界

Perceptual Decision Making

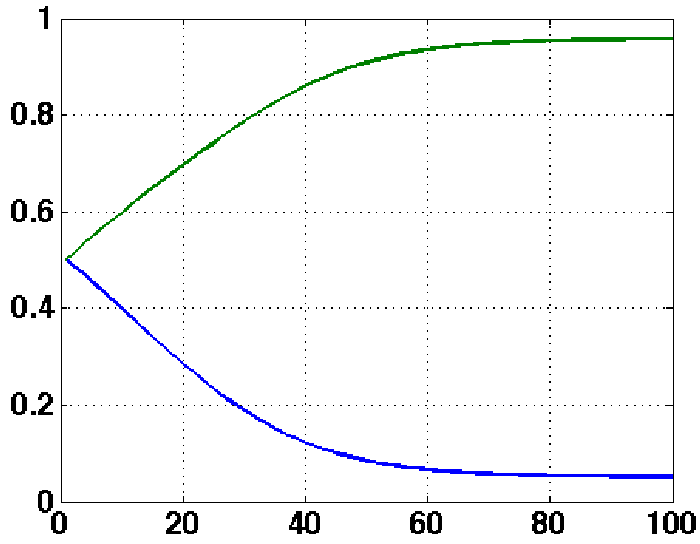
外界刺激太微弱需要長時間的訊息整合才能確定







知覺決策的模擬：贏者全拿




```
s=5e-2*array([10,1]) # weak stimuli
x=[0.5,0.5]
dt=0.1
x0_history=[x[0]]; x1_history=[x[1]]
for t in arange(0,10,dt):
 x[0]+=x[0]+dt*(-.01*x[0]+(1-x[0])*(s[0]+x[0]**2)-x[0]*x[1])
 x0_history.append(x[0])
 x[1]+=x[1]+dt*(-.01*x[1]+(1-x[1])*(s[1]+x[1]**2)-x[1]*x[0])
 x1_history.append(x[1])
plot(x0_history,'g'); plot(x1_history,'b')
```

Game Over