Introduction to Neural Networks
U. Minn. Psy 5038
Daniel Kersten
Belief Propagation

Initialize

■ **Read in Statistical Add-in packages:**

```
Off[General::spell1];
<< Statistics`NormalDistribution`
```

```
<< Statistics`ContinuousDistributions`
<< Statistics`MultinormalDistribution`
```

```
Off[General::spell1];
<< Graphics`MultipleListPlot`
```

Last time

Generative modeling: Multivariate gaussian, mixtures

- **Drawing samples**
- **Mixtures of gaussians**
- **Will mixture distributions in the next lecture on EM application to segmentation**

Introduction to Optimal inference and task

Optimal Inference and task dependence: Fruit example

(due to James Coughlan; see Yuille, Coughlan, Kersten & Schrater).

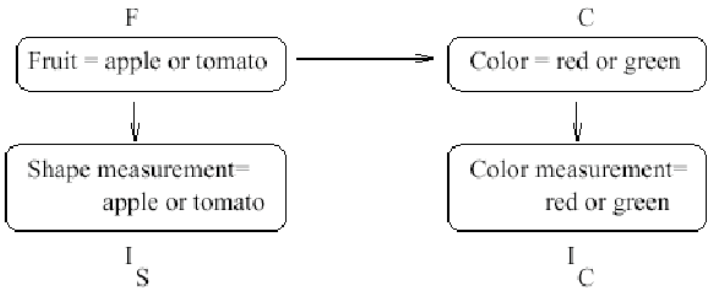


Figure from Yuille, Coughlan, Kersten & Schrater.

The graph specifies how to decompose the joint probability:

$p[F, C, Is, Ic] = p[Ic \mid C]\, p[C \mid F]\, p[Is \mid F]\, p[F]$

The prior model on hypotheses, F & C

More apples (F=1) than tomatoes (F=2), and:

```
ppF[F_] := If[F == 1, 9 / 16, 7 / 16];
TableForm[Table[ppF[F], {F, 1, 2}], TableHeadings -> {{"F=a", "F=t"}}]
```

F=a	$\frac{9}{16}$
F=t	$\frac{7}{16}$

The conditional probability **cpCF[C|F]**:

```
cpCF[F_, C_] := Which[F == 1 && C == 1, 5 / 9, F == 1 && C == 2, 4 / 9,
   F == 2 && C == 1, 6 / 7, F == 2 && C == 2, 1 / 7];
TableForm[Table[cpCF[F, C], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
```

	C=r	C=g
F=a	$\frac{5}{9}$	$\frac{4}{9}$
F=t	$\frac{6}{7}$	$\frac{1}{7}$

So the joint is:

```
jpFC[F_, C_] := cpCF[F, C] ppF[F];
TableForm[Table[jpFC[F, C], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
```

	C=r	C=g
F=a	$\frac{5}{16}$	$\frac{1}{4}$
F=t	$\frac{3}{8}$	$\frac{1}{16}$

We can marginalize to get the prior probability on color alone is:

```
ppC[C_] := Sum[jpFC[F, C], {F, 1, 2}]
```

Question: Is fruit identity independent of material color--i.e. is F independent of C?

■ **Answer**

No.

```
TableForm[Table[jpFC[F, C], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
TableForm[Table[ppF[F] ppC[C], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
```

	C=r	C=g
F=a	$\frac{5}{16}$	$\frac{1}{4}$
F=t	$\frac{3}{8}$	$\frac{1}{16}$

	C=r	C=g
F=a	$\frac{99}{256}$	$\frac{45}{256}$
F=t	$\frac{77}{256}$	$\frac{35}{256}$

The generative model: Imaging probabilities

Suppose that we have gathered some "image statistics" which provides us knowledge of how the image measurements for shape Is, and for color Ic depend on the type of fruit F, and material color, C. For simplicity, our measurements are discrete and binary (a more realistic case, they would have continuous values), say Is = {am, tm}, and Ic = {rm, gm}.

P(I_S=am,tm | F=a) = {11/16, 5/16}

P(I_S=am,tm | F=t) = {5/8, 3/8}

P(I_C=rm,gm | C=r) = {9/16, 7/16}

P(I_C=rm,gm | C=g) = {1/2, 1/2}

We use the notation am, tm, rm, gm because the measurements are already suggestive of the likely cause. So there is a correlation between apple and apple-like shapes, am; and between red material, and "red" measurements. In general, there may not be an obvious correlation like this.

We define a function for the probability of Ic given C, **cpIcC[Ic | C]**:

```
cpIcC[Ic_, C_] := Which[Ic == 1 && C == 1, 9 / 16, Ic == 1 && C == 2, 7 / 16,
  Ic == 2 && C == 1, 1 / 2, Ic == 2 && C == 2, 1 / 2];
TableForm[Table[cpIcC[Ic, C], {Ic, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"Ic=rm", "Ic=gm"}, {"C=r", "C=g"}}]
```

	C=r	C=g
Ic=rm	$\frac{9}{16}$	$\frac{7}{16}$
Ic=gm	$\frac{1}{2}$	$\frac{1}{2}$

The probability of Is conditional on F is **cpIsF[Is | F]**:

```
cpIsF[Is_, F_] := Which[Is == 1 && F == 1, 11 / 16, Is == 1 && F == 2, 5 / 8,
   Is == 2 && F == 1, 5 / 16, Is == 2 && F == 2, 3 / 8];
TableForm[Table[cpIsF[Is, F], {Is, 1, 2}, {F, 1, 2}],
 TableHeadings -> {{"Is=am", "Is=tm"}, {"F=a", "F=t"}}]
```

	F=a	F=t
Is=am	$\frac{11}{16}$	$\frac{5}{8}$
Is=tm	$\frac{5}{16}$	$\frac{3}{8}$

The total joint probability

We now have enough information to put probabilities on the 2x2x2 "universe" of possibilities, i.e. all possible combinations of fruit, color, and image measurements. Looking at the graphical model makes it easy to use the product rule to construct the total joint, which is:

p[F, C, Is, Ic] = p[Ic | C] p[C | F] p[Is | F] p[F]:

```
jpFCIsIc[F_, C_, Is_, Ic_ ] := cpIcC[ Ic, C ] cpCF[F, C ] cpIsF[Is, F ] ppF[F ]
```

Usually, we don't need the probabilities of the image measurements (because once the measurements are made, they are fixed and we want to compare the probabilities of the hypotheses. But in our simple case here, once we have the joint, we can calculate the probabilities of the image measurements through marginalization p(Is,Ic)=$\sum_C \sum_F p(F, C, \text{Is}, \text{Ic})$, too:

```
jpIsIc[Is_, Ic_] := Sum[Sum[jpFCIsIc[F, C, Is, Ic ], {C, 1, 2}], {F, 1, 2}]
```

Three MAP tasks

We are going to show that the best guess (i.e. maximum probability) depends on the task.

■ Define argmax[] function:

```
argmax[x_] := Position[x, Max[x]];
```

■ Pick most probable fruit AND color--Answer "red tomato"

First, suppose the task is to make the best bet as to the fruit AND material color. To make it concrete, suppose that we see an "apple-like shape" with a reddish color, i.e., we measure Is=am=1, and Ic = rm=1. The measurements suggest "red apple", but to find the most probable, we need to take into account the priors too in order to make the best guesses.

Using the total joint, p(F,C | Is, Ic) = $\frac{p(F,C,\text{Is},\text{Ic})}{p(\text{Is},\text{Ic})} \propto$ p(F,C,Is=1, Ic=1)

```
TableForm[jpFCIsIcTable = Table[jpFCIsIc[F, C, 1, 1 ], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
Max[jpFCIsIcTable]
argmax[jpFCIsIcTable]
```

	C=r	C=g
F=a	$\frac{495}{4096}$	$\frac{77}{1024}$
F=t	$\frac{135}{1024}$	$\frac{35}{2048}$

$$\frac{135}{1024}$$

```
{{2, 1}}
```

"Red tomato" is the most probable once we take into account the difference in priors.

Calculating p(F,C | Is, Ic). We didn't actually need p(F,C | Is, Ic), but we can calculate it by conditioning the total joint on the probability of the measurments:

```
jpFCcIsIc[F_, C_, Is_, Ic_] := jpFCIsIc[F, C, Is, Ic ] / jpIsIc[Is, Ic]
```

```
TableForm[
 jpFCcIsIcTable = Table[jpFCcIsIc[F, C, 1, 1 ], {F, 1, 2}, {C, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}, {"C=r", "C=g"}}]
Max[jpFCcIsIcTable]
argmax[jpFCcIsIcTable]
```

	C=r	C=g
F=a	$\frac{55}{157}$	$\frac{308}{1413}$
F=t	$\frac{60}{157}$	$\frac{70}{1413}$

$$\frac{60}{157}$$

```
{{2, 1}}
```

■ Pick most probable color--Answer "red"

Same measurements as before. But now suppose we only care about the true material color, and not the identity of the object. Then we want to integrate out or marginalize with respect to the shape or fruit-type variable, F. In this case, we want to maximize the posterior:

p(C | Is=1, Ic=1)=$\sum_{F=1}^{2}$ $p(F, C \mid \text{Is} = 1, \text{Ic} = 1)$

```
pC[C_, Is_, Ic_] := Sum[jpFCcIsIc[F, C, Is, Ic ], {F, 1, 2}]
```

```
TableForm[pCTable = Table[pC[C, 1, 1 ], {C, 1, 2}],
 TableHeadings -> {{"C=r", "C=g"}}]
Max[pCTable]
argmax[pCTable]
```

C=r	$\frac{115}{157}$
C=g	$\frac{42}{157}$

$$\frac{115}{157}$$

```
{{1}}
```

Answer is that the most probable material color is C = r, "red".

■ Pick most probable fruit--Answer "apple"

Same measurements as before. But now, we don't care about the material color, just the identity of the fruit. Then we want to integrate out or marginalize with respect to the material variable, C. In this case, we want to maximize the posterior:

p(F | Is, Ic)

```
pF[F_, Is_, Ic_] := Sum[jpFCcIsIc[F, C, Is, Ic], {C, 1, 2}]
```

```
TableForm[pFTable = Table[pF[F, 1, 1], {F, 1, 2}],
 TableHeadings -> {{"F=a", "F=t"}}]
Max[pFTable]
argmax[pFTable]
```

```
F=a   803/1413
F=t   610/1413
```

```
803/1413
```

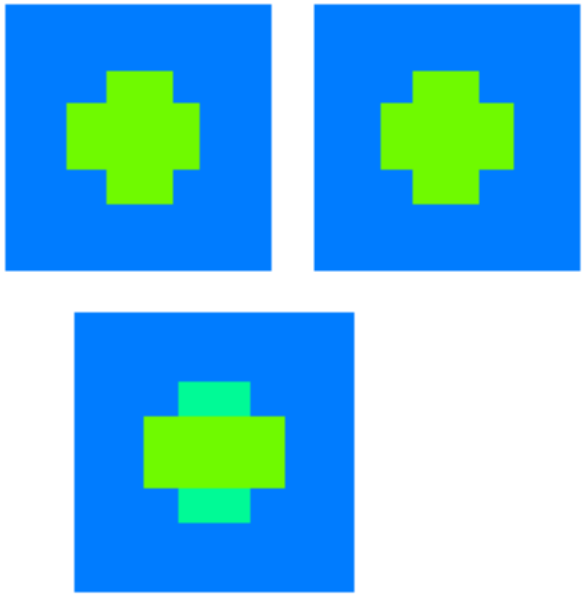
```
{{1}}
```

The answer is "apple". So to sum up, for the same data measurements, the most probable fruit AND color is "red tomato", but the most probable fruit is "apple"!

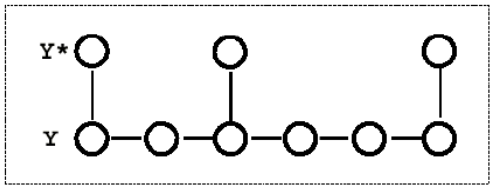
■ Important "take-home message": *Optimal inference depends on the precise definition of the task*

Try expressing the consequences using the frequency interpretation of probability.

Interpolation using smoothness revisited: Gradient descent



For simplicity, we'll assume 1-D as in the lecture on sculpting the energy function. In anticipation of formulating the problem in terms of a graph that represents conditional probability dependence, we represent *observable* depth cues by y^*, and the true ("*hidden*") depth estimates by y.



(Figure from Weiss (1999).

First-order smoothness

Recall that the energy or cost function is given by:

$$J(Y) = \sum_k w_k (y_k - y_k^*)^2 + \lambda \sum_i (y_i - y_{i+1})^2$$

where w_k = xs[[k]] is the indicator function, and y_k^* = d, are the data values.

Gradient descent gives the following local update rule:

$$y_k \leftarrow y_k + \eta_k(\lambda(\frac{y_{k-1} + y_{k+1}}{2} - y_k) + w_k(y_k^* - y_k))$$

As before, λ controls the degree of smoothness, i.e. smoothness at the expense of fidelity to the data.

Gauss-Seidel: η[k_]:=1/(λ+xs[[k]]);

Successive over-relaxation (SOR): η2[k_]:=1.9/(λ+xs[[k]]);

A simulation: Straight line with random missing data points

■ Make the data

We return to the problem of interpolating a set of points with missing data, marked by an indicator function with the following notation:

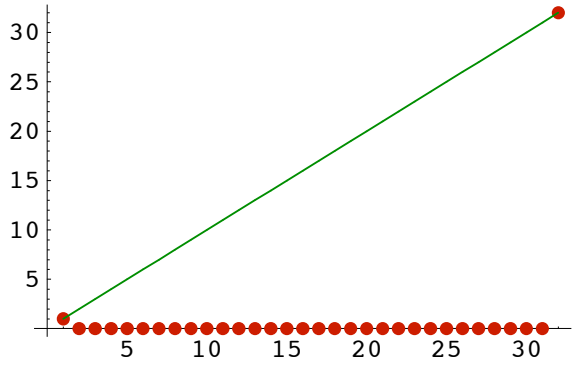
w_k = **xs[[k]], y* = data, y=f.**

We'll assume the true model is that f=y=j, where j=1 to size. data is a function of the sampling process on f=j

```
size = 32;
xs = Table[0, {i,1,size}]; xs[[1]]=1;xs[[size]]=1;(*xs[[size/2]]=1;*)
data = Table[N[j] xs[[j]],
{j, 1, size}];
g3 = ListPlot[Table[N[j],{j,1,size}],PlotJoined→True,
    DisplayFunction→Identity,PlotStyle→{RGBColor[0,.5,0]}];
g2 = ListPlot[data,PlotJoined->False,
PlotStyle->{RGBColor[.75,.0,0]}, Prolog-> AbsolutePointSize[5],
DisplayFunction->Identity];
```

The green line shows the a straight line connecting the data points. The red dots on the abscissa mark the points where data is missing.

```
Show[g2,g3,DisplayFunction->$DisplayFunction];
```



Let's set up two matrices, **Tm** and **Sm** such that the gradient of the energy is equal to:

Tm . f - Sm . f.

Sm will be our filter to exclude non-data points. **Tm** will express the "smoothness" constraint.

```
Sm = DiagonalMatrix[xs];
Tm = Table[0,{i,1,size},{j,1,size}];
For[i=1,i<=size,i++,Tm[[i,i]] = 2];
Tm[[1,1]]=1;Tm[[size,size]]=1; (*Adjust for the boundaries*)
For[i=1,i<size,i++, Tm[[i+1,i]] = -1];
For[i=1,i<size,i++, Tm[[i,i+1]] = -1];
```

Check the update rule code for small size=10:

```
Clear[f, d, λ]
(λ * Tm.Array[f, size] - Sm.((Array[d, size]) - Array[f, size])) //
 MatrixForm
```

■ Run gradient descent

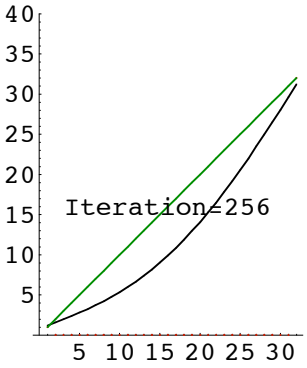
```
Clear[Tf,f1];
dt = 1; λ=2;
Tf[f1_] := f1 - dt*(1/(λ+xs))*(Tm.f1 - λ*Sm.(data-f1));
```

We will initialize the state vector to zero, and then run the network for **iter** iterations:

```
iter=256;
f = Table[0,{i,1,size}];
result = Nest[Tf,f,iter];
```

Now plot the interpolated function.

```
g1 = ListPlot[result,PlotJoined->True,
AspectRatio->Automatic,PlotRange->{{1,size},{1,size}},
DisplayFunction->Identity];
Show[{g1,g2,g3,Graphics[{Text["Iteration="<>ToString[iter],{size/2,size/2
}]}]}, DisplayFunction->$DisplayFunction,PlotRange->{0,40}];
```



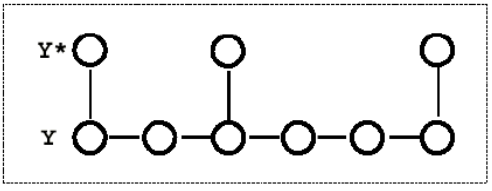
Try starting with f = random values between 0 and 40. Try various numbers of iterations.

Try different sampling functions xs[[i]].

Belief Propagation

Same interpolation problem, but now using belief propagation

Example is taken from Yair Weiss.(Weiss, 1999)



Probabilistic generative model

$$\texttt{data[[i]]} = \texttt{y}^{*}\texttt{[i]} = \texttt{xs[[i]]}\ \texttt{y[[i]]} + \texttt{dnoise, dnoise} \sim \texttt{N[0,}\ \sigma_{\texttt{D}}\texttt{]} \quad (1)$$

$$\texttt{y[[i + 1]]} = \texttt{y[[i]]} + \texttt{znoise, znoise} \sim \texttt{N[0,}\ \sigma_{\texttt{R}}\texttt{]} \quad (2)$$

The first term is the "data formation" model, i.e. how the data is directly influenced by the interaction of the underlying causes, y with the sampling and noise. The second term reflects our prior assumptions about the smoothness of y, i.e. nearby y's are correlated, and in fact identical except for some added noise. So with no noise the prior reflects the assumption that lines are horizontal--all y's are the same.

Some theory

We'd like to know the distribution of the random variables at each node i, conditioned on all the data: I.e. we want the posterior

p(Y_i=u |all the data)

If we could find this, we'd be able to: 1) say what the most probable value of the y value is, and 2) give a measure of confidence

Let p(Y_i=u |all the data) be normally distributed: NormalDistribution[μi,σi].

Consider the ith unit. The posterior p(Y_i=u|all the data) =

$$\text{p}(Y_i=\text{u}|\text{all the data}) \propto \text{p}(Y_i=\text{u}|\text{data before i})\ \text{p}(\text{data at i}|\ Y_i=\text{u})\ \text{p}(Y_i=\text{u}|\text{data after i}) \quad (3)$$

Suppose that p(Y_i=u | data before i) is also gaussian:

$$\text{p}(Y_i=\text{u}|\text{data before i}) = \alpha[\text{u}] \sim \text{NormalDistribution}[\mu\alpha,\sigma\alpha]$$

and so is probability conditioned on the data after i:

$$\text{p}(Y_i=\text{u}|\text{data after i}) = \beta[\text{u}] \sim \text{NormalDistribution}[\mu\beta,\sigma\beta]$$

And the noise model for the data:

$$\text{p}(\text{data at i}|\ Y_i=\text{u}) = \text{L[u]} \sim \texttt{NormalDistribution[yp,}\ \sigma_{\texttt{D}}\texttt{]}$$

yp=data[[i]]

So in terms of these functions, the posterior probability of the ith unit taking on the value u can be expressed as proportional to a product of the three factors:

$$\text{p}(Y_i=\text{u}|\text{all the data}) \propto \alpha[\text{u}]*\text{L[u]}*\beta[\text{u}] \quad (4)$$

```
αudist = NormalDistribution[μα, σα];
α[u] = PDF[αudist, u];

Ddist = NormalDistribution[yp, σD];
L[u] = PDF[Ddist, u];

βudist = NormalDistribution[μβ, σβ];
β[u] = PDF[βudist, u];

α[u] * L[u] * β[u]
```

$$\frac{e^{-\frac{(u-\mu\alpha)^2}{2\sigma\alpha^2}-\frac{(u-\mu\beta)^2}{2\sigma\beta^2}-\frac{(u-y_p)^2}{2\sigma_D^2}}}{2\sqrt{2}\,\pi^{3/2}\,\sigma\alpha\,\sigma\beta\,\sigma_D}$$

This just another gaussian distribution on Y_i=u. What is its mean and variance? Finding the root enables us to complete the square to see what the numerator looks like. In particular, what the mode (=mean for gaussian) is.

```
Solve[-D[-(u - μα)^2/(2 σα^2) - (u - μβ)^2/(2 σβ^2) - (u - yp)^2/(2 σD^2), u] == 0, u]
```

$$\left\{\left\{u \to \frac{\frac{\mu\alpha}{\sigma\alpha^2}+\frac{\mu\beta}{\sigma\beta^2}+\frac{y_p}{\sigma_D^2}}{\frac{1}{\sigma\beta^2}+\frac{1}{\sigma_D^2}+\frac{1}{\sigma\alpha^2}}\right\}\right\}$$

The update rule for the variance is:

$$\sigma^2 \text{ -> } \frac{1}{\sigma\alpha^2}+\frac{1}{\sigma\beta^2}+\frac{1}{\sigma_D^2}$$

How do we get $\mu\alpha$, $\mu\beta$, $\sigma\alpha$, $\sigma\beta$?

We express the probability of the ith unit taking on the value **u** in terms of the values of the neighbor before, conditioning on what is known (the observed measurements), and marginalizing over what isn't (the previous "hidden" node value, **v,** at the i-1th location).

We have three terms to worry about that depend on nodes in the neighborhood preceding i:

$$\alpha[u] = \int_{-\infty}^{\infty} \alpha_p[v] * S[u] * L[v]\, dv \propto \int_{-\infty}^{\infty} e^{-\frac{(v-y_p)^2}{2\sigma_D^2}-\frac{(u-v)^2}{2\sigma_R^2}-\frac{(v-\mu\alpha_p)^2}{2\sigma\alpha_p^2}}\, dv \qquad (5)$$

$\alpha_p = \alpha_{i-1}$. S[u] is our smoothing term, or transition probability : S[u] = p(u | v). L[] is the likelihood of the previous data node, given its hidden node value, v.

```
Rdist = NormalDistribution[v, σR];
S[u] = PDF[Rdist, u];

αvdist = NormalDistribution[μαp, σαp];
αp[v] = PDF[αvdist, v];

Lp[v] = PDF[Ddist, v];
```

```
Integrate[αp[v] * S[u] * Lp[v], {v, -Infinity, Infinity}]
```

$$\frac{1}{2\sqrt{2}\,\pi^{3/2}\,\sigma_D\,\sigma_R\,\sigma\alpha_p}\,\text{If}\left[\text{Re}\left[\frac{1}{\sigma_D^2}+\frac{1}{\sigma_R^2}+\frac{1}{\sigma\alpha_p^2}\right]>0,\right.$$

$$\frac{e^{-\frac{(u-\mu\alpha_p)^2\,\sigma_D^2+\mu\alpha_p^2\,\sigma_R^2+u^2\,\sigma\alpha_p^2+y_p^2\,(\sigma_R^2+\sigma\alpha_p^2)-2\,y_p\,(\mu\alpha_p\,\sigma_R^2+u\,\sigma\alpha_p^2)}{2\,(\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2))}}\sqrt{2\,\pi}}{\sqrt{\frac{1}{\sigma_D^2}+\frac{1}{\sigma_R^2}+\frac{1}{\sigma\alpha_p^2}}},$$

$$\left.\int_{-\infty}^{\infty} e^{-\frac{(v-y_p)^2}{2\,\sigma_D^2}-\frac{(u-v)^2}{2\,\sigma_R^2}-\frac{(v-\mu\alpha_p)^2}{2\,\sigma\alpha_p^2}}\,dv\right]$$

■ Some uninspired *Mathematica* manipulations

To find an expression for the mode of the above calculated expression for α[u]

```
D[-((u - μαp)^2 σD^2 + μαp^2 σR^2 + u^2 σαp^2 + yp^2 (σR^2 + σαp^2) - 2 yp (μαp σR^2 + u σαp^2))/
    (2 (σR^2 σαp^2 + σD^2 (σR^2 + σαp^2))), u]
```

$$-\frac{2\,(u-\mu\alpha_p)\,\sigma_D^2+2\,u\,\sigma\alpha_p^2-2\,y_p\,\sigma\alpha_p^2}{2\,(\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2))}$$

```
Solve[-% == 0, u]
```

$$\left\{\left\{u\to\frac{\frac{\mu\alpha_p\,\sigma_D^2}{\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2)}+\frac{y_p\,\sigma\alpha_p^2}{\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2)}}{\frac{\sigma_D^2}{\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2)}+\frac{\sigma\alpha_p^2}{\sigma_R^2\,\sigma\alpha_p^2+\sigma_D^2\,(\sigma_R^2+\sigma\alpha_p^2)}}\right\}\right\}$$

```
Simplify[(μα_p σ_D^2 / (σ_R^2 σα_p^2 + σ_D^2 (σ_R^2 + σα_p^2)) + y_p σα_p^2 / (σ_R^2 σα_p^2 + σ_D^2 (σ_R^2 + σα_p^2))) / (σ_D^2 * σα_p^2)]
```

$$\frac{\mu\alpha_p\,\sigma_D^2 + y_p\,\sigma\alpha_p^2}{\sigma_D^2\,\sigma_R^2\,\sigma\alpha_p^4 + \sigma_D^4\,\sigma\alpha_p^2\,(\sigma_R^2 + \sigma\alpha_p^2)}$$

```
Simplify[(σ_D^2 / (σ_R^2 σα_p^2 + σ_D^2 (σ_R^2 + σα_p^2)) + σα_p^2 / (σ_R^2 σα_p^2 + σ_D^2 (σ_R^2 + σα_p^2))) / (σ_D^2 * σα_p^2)]
```

$$\frac{\sigma_D^2 + \sigma\alpha_p^2}{\sigma_D^2\,\sigma_R^2\,\sigma\alpha_p^4 + \sigma_D^4\,\sigma\alpha_p^2\,(\sigma_R^2 + \sigma\alpha_p^2)}$$

```
(μα_p σ_D^2 + y_p σα_p^2) / (σ_D^2 σ_R^2 σα_p^4 + σ_D^4 σα_p^2 (σ_R^2 + σα_p^2)) / ((σ_D^2 + σα_p^2) / (σ_D^2 σ_R^2 σα_p^4 + σ_D^4 σα_p^2 (σ_R^2 + σα_p^2)))
```

$$\frac{\mu\alpha_p\,\sigma_D^2 + y_p\,\sigma\alpha_p^2}{\sigma_D^2 + \sigma\alpha_p^2}$$

So we now have rule that tells us how to update the α(u)=p(yi=uldata before i), in terms of the mean and variance parameters of the previous node:

$$\mu\alpha \leftarrow \frac{\mu\alpha_p\,\sigma_D^2 + y_p\,\sigma\alpha_p^2}{\sigma_D^2 + \sigma\alpha_p^2} = \frac{\frac{\mu\alpha_p\,\sigma_D^2}{\sigma\alpha_p^2\,\sigma_D^2} + \frac{y_p\,\sigma\alpha_p^2}{\sigma\alpha_p^2\,\sigma_D^2}}{\frac{\sigma_D^2}{\sigma\alpha_p^2\,\sigma_D^2} + \frac{\sigma\alpha_p^2}{\sigma\alpha_p^2\,\sigma_D^2}} = \frac{\frac{\mu\alpha_p}{\sigma\alpha_p^2} + \frac{y_p}{\sigma_D^2}}{\frac{1}{\sigma\alpha_p^2} + \frac{1}{\sigma_D^2}}$$

The update rule for the variance is:

$$\sigma\alpha^2 \leftarrow \sigma_R^2 + \frac{1}{\frac{1}{\sigma_D^2} + \frac{1}{\sigma\alpha_p^2}}$$

A similar derivation gives us the rules for $\mu\beta$, $\sigma\beta^2$

$$\mu\beta \leftarrow \frac{\frac{\mu\beta_a}{\sigma\beta_a^2} + \frac{y_a}{\sigma_D^2}}{\frac{1}{\sigma\beta_a^2} + \frac{1}{\sigma\alpha_a^2}}$$

$$\sigma\beta^2 \leftarrow \sigma_R^2 + \frac{1}{\frac{1}{\sigma_D^2} + \frac{1}{\sigma\beta_a^2}}$$

Where the subscript index p (for "previous", i.e. unit i-1) is replaced by a (for "after", i.e. unit i+1).

Recall that sometimes we have data and sometimes we don't. So replace:

$$y_p \rightarrow \texttt{xs[i - 1] data[i - 1]} = w_{i-1}\,y^*_{i-1} \qquad (6)$$

And similarly for y_a.

■ Summary of update rules

The ratio, $\left(\frac{\sigma_D}{\sigma_R}\right)^2$ plays the role of λ above. If $\sigma_D^2 >> \sigma_R^2$, there is greater smoothing. If $\sigma_D^2 << \sigma_R^2$, there is more fidelity to the data. (Recall $y^* \rightarrow$ data.$w_k \rightarrow$ xs[[k]])

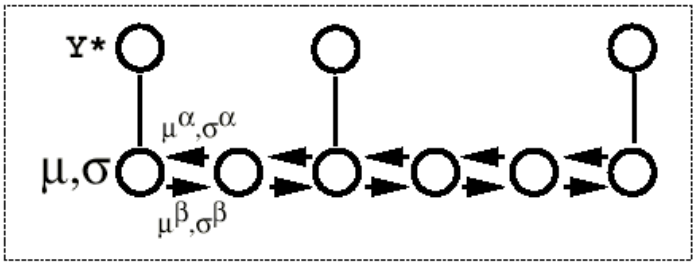
We'll follow Weiss, and also make a (hopefully not too confusing) notation change to avoid the square superscripts for $\sigma_D^2 \text{->} \sigma_D$, $\sigma_R^2 \text{->} \sigma_R$.

$$\mu_i \leftarrow \frac{\frac{w_i}{\sigma_D} Y_i^* + \frac{1}{\sigma_i^\alpha}\mu_i^\alpha + \frac{1}{\sigma_i^\beta}\mu_i^\beta}{\frac{w_i}{\sigma_D} + \frac{1}{\sigma_i^\alpha} + \frac{1}{\sigma_i^\beta}}$$

$$\sigma_i \leftarrow \frac{1}{\frac{w_i}{\sigma_D} + \frac{1}{\sigma_i^\alpha} + \frac{1}{\sigma_i^\beta}}$$

$$\mu_i^\alpha \leftarrow \frac{\frac{1}{\sigma_{i-1}^\alpha}\mu_{i-1}^\alpha + \frac{w_{i-1}}{\sigma_D} Y_{i-1}^*}{\frac{1}{\sigma_{i-1}^\alpha} + \frac{w_{i-1}}{\sigma_D}}$$

$$\sigma_i^\alpha \leftarrow \sigma_R + \left(\frac{1}{\sigma_{i-1}^\alpha} + \frac{w_{i-1}}{\sigma_D}\right)^{-1}$$



A simulation: Belief propagation for interpolation with missing data

■ Initialization

```
μ0 = 1;
μα = 1; σα = 100000; (*large uncertainty *)
μβ = 1; σβ = 100000; (*large*)
σR = 4.0; σD = 1.0;
μ = Table[μ0, {i, 1, size}];
σ = Table[σα, {i, 1, size}];
μα = Table[μ0, {i, 1, size}];
σα = Table[σα, {i, 1, size}];
μβ = Table[μ0, {i, 1, size}];
σβ = Table[σβ, {i, 1, size}];
iter = 0;
i = 1;
j = size;
```

General::spell :
Possible spelling error: new symbol name "σD" is similar to existing symbols {σ, σR}. More…

The code below implements the above iterative equations, taking care near the boundaries. The plot shows the estimates of $y_i =$ μ, and the error bars show $\pm\sigma_i$.

■ **Belief Propagation Routine: Execute this cell "manually" for each iteration**

$$\mu[[i]] = \frac{\frac{xs[[i]]}{\sigma D} * data[[i]] + \frac{1}{\sigma\alpha[[i]]} * \mu\alpha[[i]] + \frac{1.0}{\sigma\beta[[i]]} * \mu\beta[[i]]}{\frac{xs[[i]]}{\sigma D} + \frac{1}{\sigma\alpha[[i]]} + \frac{1}{\sigma\beta[[i]]}};$$

$$\sigma[[i]] = \frac{1.0}{\frac{xs[[i]]}{\sigma D} + \frac{1}{\sigma\alpha[[i]]} + \frac{1}{\sigma\beta[[i]]}};$$

$$\mu[[j]] = \frac{\frac{xs[[j]]}{\sigma D} * data[[j]] + \frac{1}{\sigma\alpha[[j]]} * \mu\alpha[[j]] + \frac{1.0}{\sigma\beta[[j]]} * \mu\beta[[j]]}{\frac{xs[[j]]}{\sigma D} + \frac{1}{\sigma\alpha[[j]]} + \frac{1}{\sigma\beta[[j]]}};$$

$$\sigma[[j]] = \frac{1.0}{\frac{xs[[j]]}{\sigma D} + \frac{1}{\sigma\alpha[[j]]} + \frac{1}{\sigma\beta[[j]]}};$$

```
nextj = j - 1;
```

$$\mu\alpha[[nextj]] = \frac{\frac{xs[[j]]}{\sigma D} * data[[j]] + \frac{1.0}{\sigma\alpha[[j]]} * \mu\alpha[[j]]}{\frac{xs[[j]]}{\sigma D} + \frac{1}{\sigma\alpha[[j]]}};$$

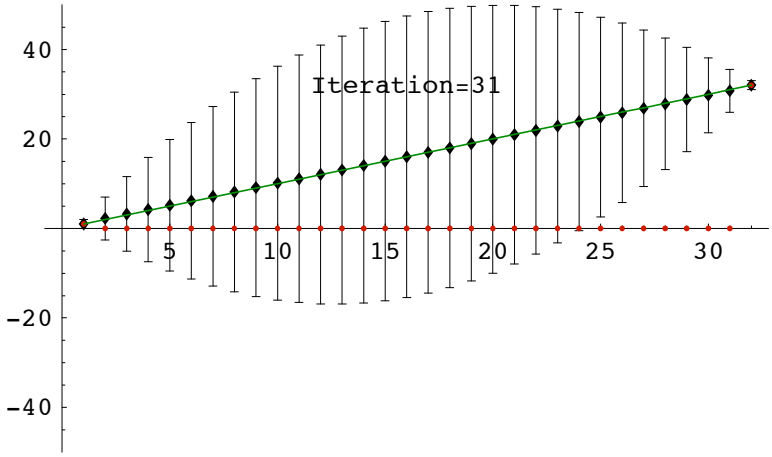
$$\sigma\alpha[[nextj]] = \sigma R + \frac{1.0}{\frac{xs[[j]]}{\sigma D} + \frac{1}{\sigma\alpha[[j]]}};$$

```
nexti = i + 1;
```

$$\mu\beta[[nexti]] = \frac{\frac{xs[[i]]}{\sigma D} * data[[i]] + \frac{1.0}{\sigma\beta[[i]]} * \mu\beta[[i]]}{\frac{xs[[i]]}{\sigma D} + \frac{1}{\sigma\beta[[i]]}};$$

$$\sigma\beta[[nexti]] = \sigma R + \frac{1.0}{\frac{xs[[i]]}{\sigma D} + \frac{1}{\sigma\beta[[i]]}};$$

```
j--;
i++;

iter++;

yfit = Table[{μ[[i1]], ErrorBar[σ[[i1]]]}, {i1, 1, size}];
g1b = MultipleListPlot[yfit, DisplayFunction → Identity];
Show[
  {g1b, g2, g3,
   Graphics[{Text["Iteration=" <> ToString[iter], {size / 2, size}]}]},
  DisplayFunction → $DisplayFunction, PlotRange → {-50, 50}];
```


Iteration=31
40
20
-20
-40
5
10
15
20
25
30

Exercises

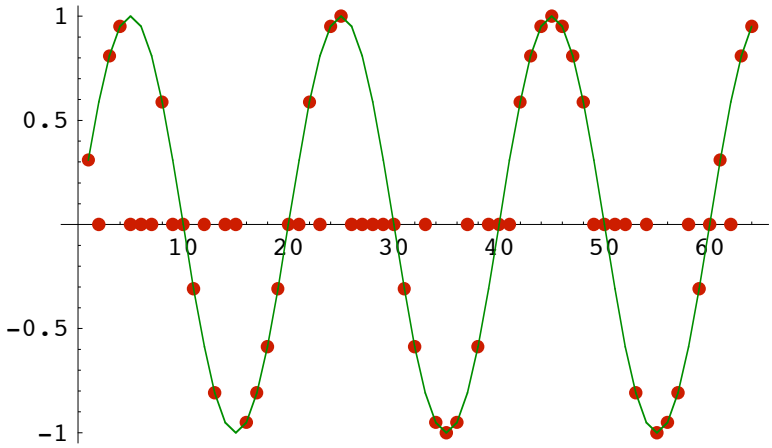
Run the descent algorithm using successive over-relaxation (SOR): η2[k_]:=1.9/(λ+xs[[k]]).
How does convergence compare with Gauss-Seidel?

Run Belief Propation using: σR=1.0; σD=4.0; How does fidelity to the data compare with the original case
(σR=4.0; σD=1.0).

BP with missing sine wave data

■ **Generate sine wave with missing data**

```
size = 64;
xs = Table[Random[Integer,1], {i,1,size}];
data = Table[N[Sin[2 Pi (1/20) j] xs[[j]]],
{j, 1, size}];
g3b = ListPlot[Table[N[Sin[2 Pi (1/20) j]], {j, 1, size}],
        PlotJoined->True,DisplayFunction->Identity,
    PlotStyle->{RGBColor[0,.5,0]}];
g2b = ListPlot[data,PlotJoined->False,
PlotStyle->{RGBColor[.75,.0,0]}, Prolog-> AbsolutePointSize[5],
DisplayFunction->Identity];
Show[g2b,g3b,DisplayFunction->$DisplayFunction];
```

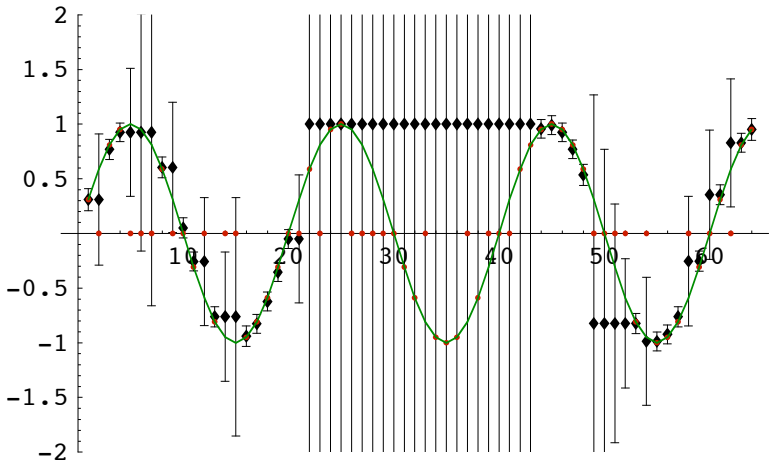


■ Initialize

```
μ0 = 1;
μα = 1; σα = 100000; (*large uncertainty *)
μβ = 1; σβ = 100000;(*large*)
σR = .5; σD = .1;
μ = Table[μ0, {i, 1, size}];
σ = Table[σα, {i, 1, size}];
μα = Table[μ0, {i, 1, size}];
σα = Table[σα, {i, 1, size}];
μβ = Table[μ0, {i, 1, size}];
σβ = Table[σβ, {i, 1, size}];
iter = 0;
i = 1;
j = size;
```

■ **SINE WAVE DEMO: Belief Propagation Routine: Execute this cell "manually" for each iteration**

```
μ[[i]] = (xs[[i]]/σD * data[[i]] + 1/σα[[i]] * μα[[i]] + 1.0/σβ[[i]] * μβ[[i]]) /
    (xs[[i]]/σD + 1/σα[[i]] + 1/σβ[[i]]);
σ[[i]] = 1.0/(xs[[i]]/σD + 1/σα[[i]] + 1/σβ[[i]]);
μ[[j]] = (xs[[j]]/σD * data[[j]] + 1/σα[[j]] * μα[[j]] + 1.0/σβ[[j]] * μβ[[j]]) /
    (xs[[j]]/σD + 1/σα[[j]] + 1/σβ[[j]]);
σ[[j]] = 1.0/(xs[[j]]/σD + 1/σα[[j]] + 1/σβ[[j]]);


nextj = j - 1;
μα[[nextj]] = (xs[[j]]/σD * data[[j]] + 1.0/σα[[j]] * μα[[j]])/(xs[[j]]/σD + 1/σα[[j]]);
σα[[nextj]] = σR + 1.0/(xs[[j]]/σD + 1/σα[[j]]);

nexti = i + 1;
μβ[[nexti]] = (xs[[i]]/σD * data[[i]] + 1.0/σβ[[i]] * μβ[[i]])/(xs[[i]]/σD + 1/σβ[[i]]);
σβ[[nexti]] = σR + 1.0/(xs[[i]]/σD + 1/σβ[[i]]);


j--;
i++;

iter++;

yfit = Table[{μ[[i1]], ErrorBar[σ[[i1]]]}, {i1, 1, size}];
g1bb = MultipleListPlot[yfit, DisplayFunction → Identity];
Show[{g1bb, g2b, g3b}, DisplayFunction → $DisplayFunction,
  PlotRange → {-2, 2}];
```


2
1.5
1
0.5
-0.5
-1
-1.5
-2
10
20
30
40
50
60

Run EM with Generative Model 2. Increase the additive noise. How does attribution accuracy change? ("attribution" means assigning a point to its correct line)

References

Applebaum, D. (1996). Probability and Information . Cambridge, UK: Cambridge University Press.

Frey, B. J. (1998). *Graphical Models for Machine Learning and Digital Communication*. Cambridge, Massachusetts: MIT Press.

Jepson, A., & Black, M. J. (1993). *Mixture models for optical flow computation*. Paper presented at the Proc. IEEE Conf. Comput. Vsion Pattern Recog., New York.

Kersten, D. and P.W. Schrater (2000), *Pattern Inference Theory: A Probabilistic Approach to Vision*, in *Perception and the Physical World*, R. Mausfeld and D. Heyer, Editors. , John Wiley & Sons, Ltd.: Chichester. (pdf)

Kersten, D., & Madarasmi, S. (1995). The Visual Perception of Surfaces, their Properties, and Relationships. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 19*, 373-389.

Madarasmi, S., Kersten, D., & Pong, T.-C. (1993). The computation of stereo disparity for transparent and for opaque surfaces. In C. L. Giles & S. J. Hanson & J. D. Cowan (Eds.), *Advances in Neural Information Processing Systems 5*. San Mateo, CA: Morgan Kaufmann Publishers.

Pearl, Judea. (1997) Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference. (amazon.com link)

Ripley, B. D. (1996). *Pattern Reco gnition and Neural Networks*. Cambridge, UK: Cambridge University Press.

Weiss Y. (1999) Bayesian Belief Propagation for Image Understanding submitted to *SCTV* 1999. (gzipped postscript 297K)

Weiss, Y. (1997). *Smoothness in Layers: Motion segmentation using nonparametric mixture estimation*. Paper presented at the Proceedings of IEEE conference on Computer Vision and Pattern Recognition.

Yuille, A., Coughlan J., Kersten D.(1998) (pdf)

For notes on Graphical Models, see:http://www.cs.berkeley.edu/~murphyk/Bayes/bayes.html