Introduction to Neural Networks
U. Minn. Psy 5038

Gaussian generative models, learning, and inference

Initialize standard library files:

```
Off[General::spell1];
```

Last time

Review of probability and statistics

Basic rules of probability

Basic rules of inference: Given a joint distribution, condition on what you know, and integrate out what you don't care about.

...but easier said than done.

Rationale: The tools of probabilistic modeling can provide a deeper understanding of neural networks. They also provide quantitative descriptions closer to functional behaviors.

Today

Examples of computations on continuous probabilities
Examples of computations on discrete probabilities
Low dimensional sampling
Introduction to Bayes learning

Mathematica functions for multivariate distributions & exploring marginals

Multivariate gaussian probability density

An *n*-variate multivariate gaussian (multinormal) distribution with mean vector μ and covariance matrix Σ is denoted $N_n(\mu, \Sigma)$. The density is:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}\,\text{Det}[\Sigma]^{1/2}}\,\text{Exp}\left[-\frac{1}{2}(\mathbf{x}-\mu)^{T}\,\Sigma^{-1}\,(\mathbf{x}-\mu)\right]$$

With *Mathematica*'s built-in function you can, for example, define the bivaraite probability density function with mean vector $\{\mu_1, \mu_2\}$, and covariance matrix $\{\{\sigma_{11}^2, \rho * \sigma_{11} * \sigma_{22}\}, \{\rho * \sigma_{11} * \sigma_{22}, \sigma_{22}^2\}\}$, where ρ parameterizes correlation.

In[2]:=
```
Σ = {{σ11^2, ρ * σ11 * σ22}, {ρ * σ11 * σ22, σ22^2}};
PDF[MultinormalDistribution[{μ1, μ2}, Σ], {x, y}]
```

Out[3]=
$$\frac{e^{\frac{1}{2}\left(-\frac{(\mathbf{x}-\mu_1)\,(\mathbf{y}\,\rho\,\sigma_{11}-\rho\,\mu_2\,\sigma_{11}-\mathbf{x}\,\sigma_{22}+\mu_1\,\sigma_{22})}{(-1+\rho^2)\,\sigma_{11}^2\,\sigma_{22}}-\frac{(\mathbf{y}-\mu_2)\,(-\mathbf{y}\,\sigma_{11}+\mu_2\,\sigma_{11}+\mathbf{x}\,\rho\,\sigma_{22}-\rho\,\mu_1\,\sigma_{22})}{(-1+\rho^2)\,\sigma_{11}\,\sigma_{22}^2}\right)}}{2\pi\sqrt{\sigma_{11}^2\,\sigma_{22}^2-\rho^2\,\sigma_{11}^2\,\sigma_{22}^2}}$$

In[43]:=
```
Σ
```

Out[43]=
$$\{\{\sigma_{11}^2, \rho\,\sigma_{11}\,\sigma_{22}\}, \{\rho\,\sigma_{11}\,\sigma_{22}, \sigma_{22}^2\}\}$$

In[46]:=
```
Covariance[MultinormalDistribution[{μ1, μ2}, Σ]]
```

Out[46]=
$$\{\{\sigma_{11}^2, \rho\,\sigma_{11}\,\sigma_{22}\}, \{\rho\,\sigma_{11}\,\sigma_{22}, \sigma_{22}^2\}\}$$

Examples of PDF, CDF

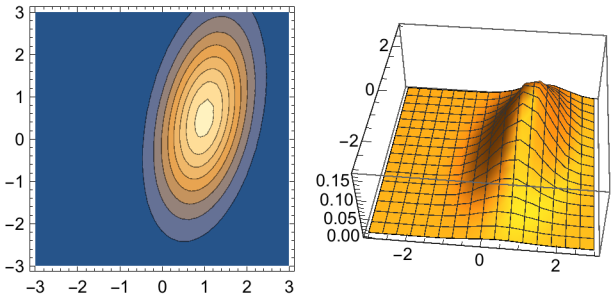
A specific example, MultinormalDistribution[μ, Σ]

In[47]:=
```
m1 = {1,1/2};
r=(1/2)*{{1,2/3},{2/3,4}};
ndist = MultinormalDistribution[m1, r];
pdf = PDF[ndist, {x1, x2}]
```

Out[50]=
$$\frac{3\, e^{\frac{1}{2}\left(-(-1+x1)\left(\frac{9}{4}(-1+x1)-\frac{3}{8}\left(-\frac{1}{2}+x2\right)\right)-\left(-\frac{3}{8}(-1+x1)+\frac{9}{16}\left(-\frac{1}{2}+x2\right)\right)\left(-\frac{1}{2}+x2\right)\right)}}{4\sqrt{2}\,\pi}$$

In[51]:=
```
GraphicsRow[{g1 = ContourPlot[PDF[ndist, {x1, x2}], {x1, -3, 3}, {x2, -3, 3}],
  g13D = Plot3D[PDF[ndist, {x1, x2}], {x1, -3, 3}, {x2, -3, 3}]}]
```

Out[51]=



Calculating probabilities using the CDF

What is the probability of x_1 and x_2 taking on values in the region $x_1 < .5 \cap x_2 < 2$?
Recall that the cumulative distribution function is given by:

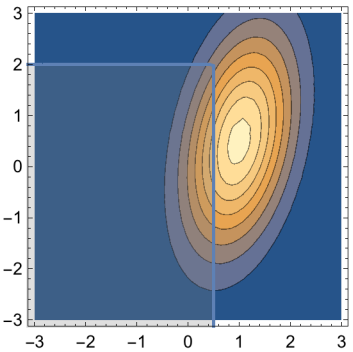
$$\text{CDF}(x_1, x_2) = \int_{-\infty}^{x_2}\int_{-\infty}^{x_1} p(x_1, x_2)\, dx_1\, dx_2$$

So the answer is the volume under the PDF shown below.

In[12]:=
```
grp = RegionPlot[x1 < .5 && x2 < 2, {x1, -4, 4}, {x2, -4, 4},
    PlotStyle → Directive[Opacity[.25], EdgeForm[], FaceForm[Gray]]];

Show[{ g1, grp}, ImageSize → Small]
```

Out[13]=

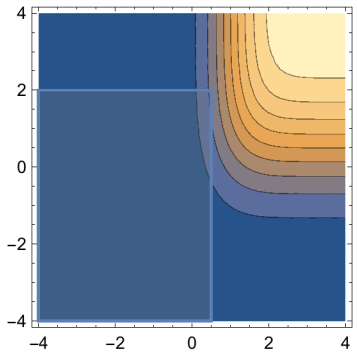


We could numerically integrate to find the volume. Alternatively, the answer can be found from the built-

in cumulative distribution function, or CDF. The value corresponds to the height of the contour at {x_1, x_2} = {.5, 2.0}. Move your mouse cursor over the contour plot below near the {.5, 2.0} point.

In[14]:=
```
gcdf = ContourPlot[CDF[ndist, {x1, x2}], {x1, -4, 4}, {x2, -4, 4},ImageSize → Small];
Show[{ gcdf, grp}, ImageSize → Small]
```

Out[15]=



A more precise answer is:

In[52]:=
```
CDF[ndist, {.5, 2.0}]
```

Out[52]=
```
0.225562
```

Finding the mode

The mode is the value of the random variable with the highest probability or probability density. For discrete distributions, think of it as the most frequent value.

(Sometimes the word “mode” is used to refer to a *local* maximum in a density function. Then the distribution is called multimodal. If it has two modes, it is called bimodal.) There may not be a unique mode--the uniform distribution is an extreme case of this.

For the Gaussian case, the mode vector corresponds to the mean vector. But we can pretend we don't know that, and use the **FindMaximum[]** function to find the maximum and the coordinates where the max occurs:

In[17]:=
```
FindMaximum[PDF[ndist, {x1, x2}], {{x1, 0}, {x2, 0}}]
```

Out[17]=
```
{0.168809, {x1 → 1., x2 → 0.5}}
```

Marginals

Suppose we want the distribution for just x_1, PDF[x_1] = marginal[x_1]. This is called calculating the marginal distribution of x_1. How do we find it? It is obtained by integrating out the other variable, x_2. I.e.Integrate PDF[x_1, x_2] with respect to x_2. Similarly, we can calculate PDF[x_2].

In[69]:=
```
Clear[x1, x2];
marginal[x1_] := ∫_-∞^∞ PDF[ndist, {x1, x2}] dx2;
marginal2[x2_] := ∫_-∞^∞ PDF[ndist, {x1, x2}] dx1;
```

What is the mode of PDF[x_2]?

In[72]:=
```
FindMaximum[marginal[x1], {{x1, 0}}]
FindMaximum[marginal2[x2], {{x2, 0}}]
```

Out[72]= {0.56419, {x1 → 1.}}

Out[73]= {0.282095, {x2 → 0.5}}

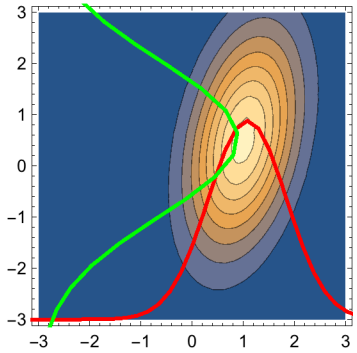
Now let's plot up the marginals on top of the contour plot of the joint distribution.

In[61]:=
```
mt = Table[{x1, marginal[x1]}, {x1, -3, 3, .2}];
g2 = ListPlot[mt, Joined → True, PlotStyle → {Red, Thick}, Axes → False];
```

In[63]:=
```
mt2 = Table[{x2, marginal2[x2]}, {x2, -3, 3, .4}];
g3 = ListPlot[mt2, Joined → True, PlotStyle → {Green, Thick}, Axes → False];
```

In[65]:=
```
theta = Pi / 2;
Show[g1, Epilog → {Inset[g2, {0, -3}, {0, 0}], Inset[g3, {-3, 0}, {0, 0},
    Automatic, {{Cos[theta], Sin[theta]}, {Sin[theta], -Cos[theta]}}]}]
```

Out[66]=



Drawing samples

As we've used in earlier lectures, drawing samples is done by:

In[76]:=
```
RandomVariate[ndist]
```

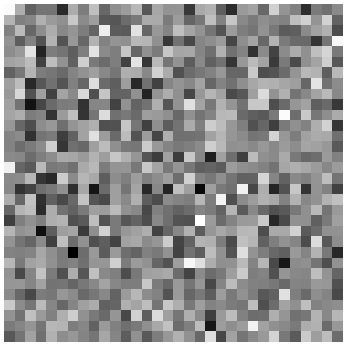
Out[76]= {2.11531, 0.392123}

(RandomReal[ndist] **also works, but RandomVariate[]** is preferred.)

Here's an example of a "white gaussian noise" image. It is called "white" because there are no correlations between the pixel intensities. Each one is drawn independently of any of the others.

In[30]:=
```
width = 32;
data = RandomVariate[NormalDistribution[0, 1], width * width];
Image[Partition[data, width]] // ImageAdjust
```

Out[32]=



Mixtures of gaussians with MultinormalDistribution[]

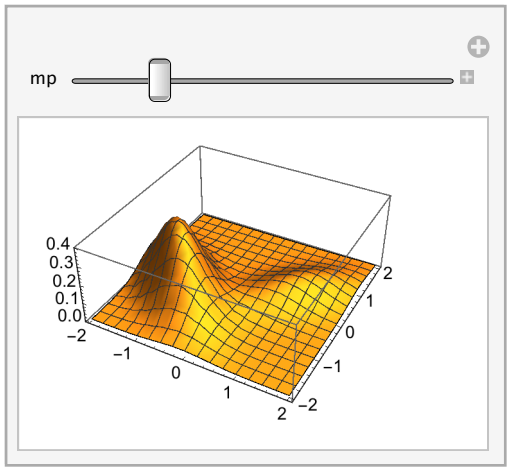
Multivariate gaussian distributions are often inadequate to model real-life problems, that for example might involve more than one mode or might have non-gaussian properties. One solution is to approximate more general distributions by a sum or mixture of gaussians.

In[33]:=
```
Clear[mix];
r1=0.4*{{1,.6},{.6,1}};
r2=0.4*{{1,-.6},{-.6,1}};
m1 = {1,.5}; m2 = {-1,-.5};
ndist1 = MultinormalDistribution[m1, r1];
ndist2 = MultinormalDistribution[m2, r2];
```

In[39]:=
```
mix[x_,mp_] := mp*PDF[ndist1, x] + (1-mp)*PDF[ndist2, x];
```

In[40]:=
```
Manipulate[gg1 = Plot3D[mix[{x1, x2}, mp], {x1, -2, 2},
    {x2, -2, 2}, PlotRange → Full, ImageSize → Small], {{mp, .2}, 0, 1}]
```

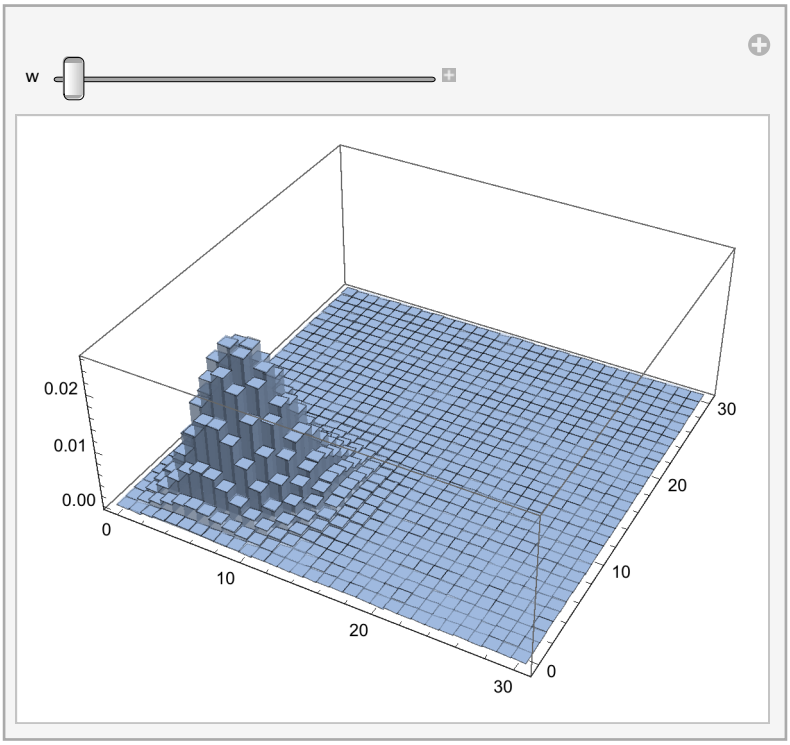
Out[40]=



Mathematica has a built-in function for defining mixture distributions:

In[41]:=
```
𝒟[w_] = MixtureDistribution[{w, 1 - w}, {MultivariatePoissonDistribution[7, {9, 10}],
    MultivariatePoissonDistribution[2, {5, 4}]}];
Manipulate[DiscretePlot3D[PDF[𝒟[w], {x, y}], {x, 0, 30},
  {y, 0, 30}, ExtentSize → Full], {w, 0, 1}]
```

Out[42]=



See Zoran and Weiss (2011) for an interesting application of mixture models to discovering visual features in natural images.

Sampling in low dimensions

Why sample? Rationale for Monte Carlo.

Inferring means, marginalization require integrations or sums.
But Integrations, especially in high dimensional spaces can be computationally hard.
Data synthesis can provide a “reality” check.

Later we’ll look at the problem of sampling in high dimensions.

Let’s look at some methods of sampling. We first go over the simple one dimensional (univariate) case for densities.

Density mapping theorem

Suppose we have a change of variables that maps a discrete set of x's uniquely to y's: X->Y.

Discrete random variables

No change to probability function. The mapping just corresponds to a change of labels, so the probabilities p(X)=p(Y).

Continuous random variables

In this case, the form of probability density function changes because we require the probability "mass" to be unchanged:
p(x)dx = p(y)dy

Suppose, y=f(x)

$p_Y(y)\,\delta y = p_X(x)\,\delta x$

Transformation of variables is useful in making random number generators for probability densities other than the uniform distribution, such as a Gaussian.

Below we'll need to use the cumulative distribution function: CDF(x) = prob $(X < x) = \int_{-\infty}^{x} p(X)\,dX$

Univariate sampling

Making a univariate (scalar) gaussian random number generator:

We assume we have a random number generator that provides uniformly distributed numbers between 0 and 1. How can we get numbers that are Gaussian distributed?

Well, the easiest way is to use a built-in function:

```
RandomVariate[NormalDistribution[0, 1]]
```

```
0.726157
```

but we'd like to better understand some principles behind generating random numbers for a specified distribution.

Method 1: Just for Gaussian. Use Central Limit Theorem

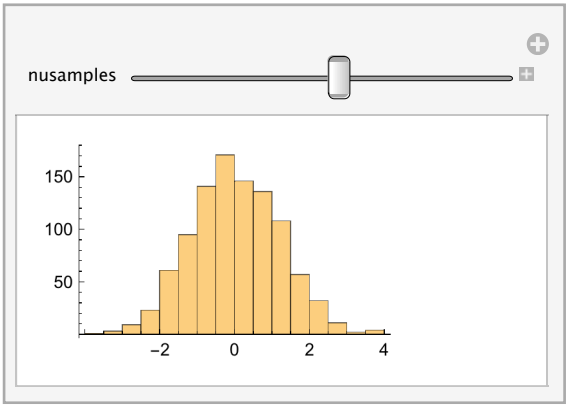
If all we want to do is make a Gaussian random number generator from a uniformly distributed generator, we can use the Central Limit Theorem. The Central Limit Theorem says that the sum of a sufficiently large number of independent random variables drawn from the same underlying distribution (with finite mean and variance), will be approximately normally distributed. Note: These draws are said to be *"i.i.d.", meaning "independent, identically distributed".*

The approximation gets better as the number of samples increases.

Try the cell below with nusamples = 1, 2, .., 10,...

In[83]:=
```
Manipulate[zl = Table[Sum[RandomReal[] - nusamples/2, {i, 1, nusamples}], {1000}];
 Histogram[zl, ImageSize → Small], {nusamples, 1, 30, 1}]
```

Out[83]=



Method 2: Use Density Mapping theorem. More general.

We'll use the density mapping theorem to turn uniformly distributed random numbers **RandomReal[]** into gaussian distributed random numbers with mean =0 and standard deviation =1.

$$p_Y(y)\,\delta y = p_X(x)\,\delta x$$

$$p_Y(y)\,\frac{\delta y}{\delta x} = p_X(x)$$

Suppose $p_Y(y)$=1 (over the unit interval, but zero elsewhere). Then

$$y(x) = \int_{-\infty}^{x} p_x(x')\,dx' = P(x)$$

Thus if we sample from the uniform distribution to get y, x should be distributed according to $p_X(x)$. To do this, we need a mapping from y → x. This is given by the inverse cumulative distribution, i.e. $P^{-1}(y)$.

Let's implement this. The quick way is to use *Mathematica*'s built-in function, InverseErf[ ], to get the inverse cumulative.

Method 2a: Applied to Gaussian

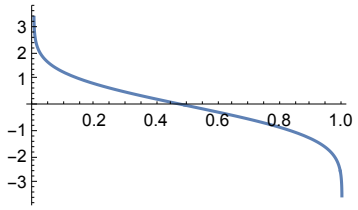
InverseErf[ ] is the inverse of the classic “error function”:

$$erf(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\,dt$$

We can use this to define a function for the inverse cumulative of a gaussian:

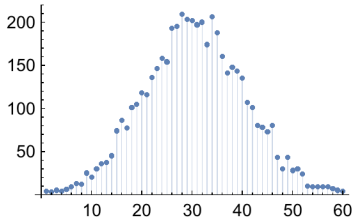
In[87]:=
```
Clear[z];
z[p_] := √2 InverseErf[1 - 2 p];
Plot[z[y], {y, 0, 1}]
```

Out[89]=



In[90]:=
```
binsize = 0.1;
zl = Table[z[RandomReal[]], {5000}];
freq = BinCounts[zl, {-3, 3, binsize}];
ListPlot[freq, Filling → Axis]
```

Out[93]=



Method 2b: Cumulative from scratch: Works for almost any low dimensional distribution.

Suppose we have a discrete approximation of any cumulative distribution. How can we generate samples? For illustration purposes, we'll illustrate the method with a discretization of the Gaussian.

Our first goal is to produce a discrete approximation to the cumulative gaussian. To review where things come from, we'll start with the definition of a Gaussian, and make sure it is normalized.

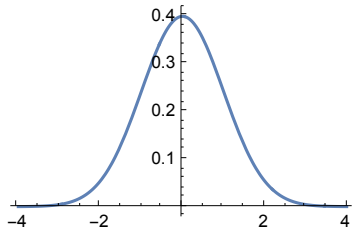
In[94]:=
```
Integrate[Exp[-(x - x0)^2 / (2 * σ^2)], {x, -Infinity, Infinity}]
```

Out[94]=
$$\text{ConditionalExpression}\left[\frac{\sqrt{2\,\pi}}{\sqrt{\frac{1}{\sigma^2}}}, \text{Re}\left[\frac{1}{\sigma^2}\right] \geq 0\right]$$

Let x0=0 and σ=1:

In[95]:=
```
Plot[e^(-x1^2/2) / √(2 π), {x1, -4, 4}]
```

Out[95]=



Note that **Plot[PDF[NormalDistribution[0,1],x1],{x1,-4,4}]**; gives the same thing using the built-in

normal distribution function.

Cumulative gaussian

In[96]:=
```
Clear[cumulgauss, x, x1];
cumulgauss[x_] := NIntegrate[Exp[-(x1^2) / 2] / (Sqrt[2 * Pi]), {x1, -Infinity, x}]
```

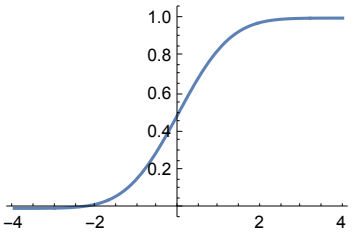
In[98]:=
```
cumulgauss[Infinity]
```

Out[98]=
```
1.
```

We can plot up cumulgauss:

In[99]:=
```
Plot[cumulgauss[x], {x, -4, 4}]
```

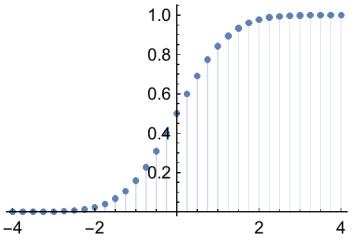
Out[99]=



Now make a discrete version of the cumulative distribution:

In[100]:=
```
lcumulgauss = Table[{x, cumulgauss[x]}, {x, -4.`, 4.`, 0.25`}];
ListPlot[lcumulgauss, Filling → Axis]
```

Out[101]=



Remember the main reason we did the above is to now show how we can go from an arbitrary histogram to drawing samples. I.e. it doesn't have to be a gaussian at this stage, the cumulative could have come from some any one dimenstional histogram from data gathered elsewhere.

Make inverse cumulative gaussian table

Here is a useful trick whenever you want an inverse function, given a discrete representation.

In[102]:=
```
invlcumulgauss = RotateLeft[lcumulgauss, {0, 1}];
```

To see what this does, evaluate:

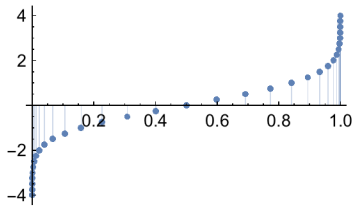
In[103]:=
```
{{x1, y1}, {x2, y2}, {x3, y3}}
RotateLeft[{{x1, y1}, {x2, y2}, {x3, y3}}, {0, 1}]
```

Out[103]=
```
{{x1, y1}, {x2, y2}, {x3, y3}}
```

Out[104]=
```
{{y1, x1}, {y2, x2}, {y3, x3}}
```

In[106]:=
```
ListPlot[invlcumulgauss, Filling → Axis]
```

Out[106]=



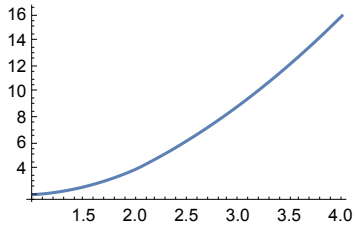
Make a smooth interpolated function from the discrete inverse cumulative

Another useful trick.

Interpolation works by fitting polynomial curves to the data. Try the test below with various interpolation orders (the default is 3)

In[107]:=
```
test = Interpolation[{{1, 2.}, {2, 4}, {3, 9}, {4, 16}}, InterpolationOrder → 2];
Plot[test[x], {x, 1, 4}]
```

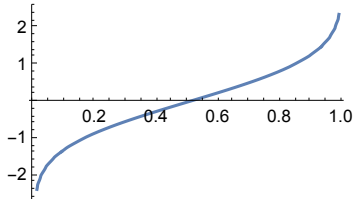
Out[108]=



In[109]:=
```
interinvlcumulgauss = Interpolation[invlcumulgauss];
```

In[110]:=
```
Plot[interinvlcumulgauss[x], {x, 0.01, 0.99}]
```

Out[110]=



Draw samples with a standard deviation of Sqrt[10]

In[113]:=
```
Round[10 interinvlcumulgauss[RandomReal[]]]
```

Out[113]= 6

Draw a bunch of samples, and plot up histogram

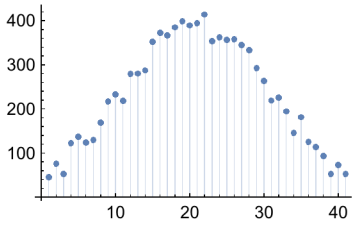
In[114]:=
```
z = Table[Round[10 interinvlcumulgauss[RandomReal[]]], {10 000}];
domain = Range[-20, 20];
Freq = (Count[z, #1] &) /@ domain;
```

InterpolatingFunction::dmval :
Input value {0.999992} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

InterpolatingFunction::dmval :
Input value $\{4.80633 \times 10^{-6}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

In[118]:=
```
ListPlot[Freq, Filling → Axis]
```

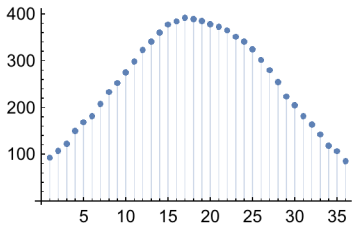
Out[118]=



digression...a quick & dirty way to smooth is to do a moving average

In[119]:=
```
ListPlot[MovingAverage[Freq, 6], Filling → Axis]
```

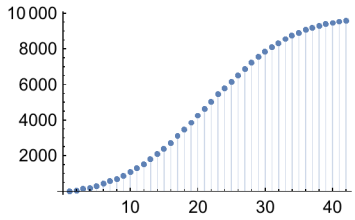
Out[119]=



Plot up cumulative histogram

In[120]:=
```
CumFreq = FoldList[Plus, 0, Freq];
ListPlot[CumFreq, Filling → Axis]
```

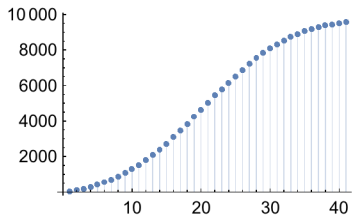
Out[121]=



Same thing, with **Accumulate[]** :

In[122]:=
```
CumFreq = Accumulate[Freq];
ListPlot[CumFreq, Filling → Axis]
```

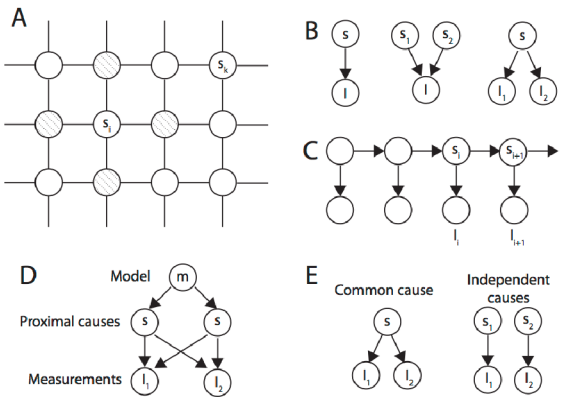
Out[123]=



Graphical Models

Causal structure and conditional independence

In general, we'd like to be able to specify natural patterns such as images by a high-dimensional joint probability. This is usually too difficult in practice, and requires simplification and approximations. If we have some idea of the conditional relationships between various causes and data, this knowledge can be represented in a *directed graph*. (Markov Random Fields are represented by undirected graphs.) The idea is to represent the probabilistic structure of the joint distribution P(S,L,I) by a graphical model that expresses how variables influence each other. Random variables are represented by nodes, and the links or arrows between the nodes represent conditional relationships.



The above figure illustrates several types of graphical structures. A. undirected graph as in a Markov Random Field. B-E illustrate directed graphs: B. Simple influence relationships. C. Markov process. D. Simple hierarchical model. E. Common vs. independent causes.

Conceptual illustration from vision

In[142]:= `Image[ ... ]`

Out[142]=

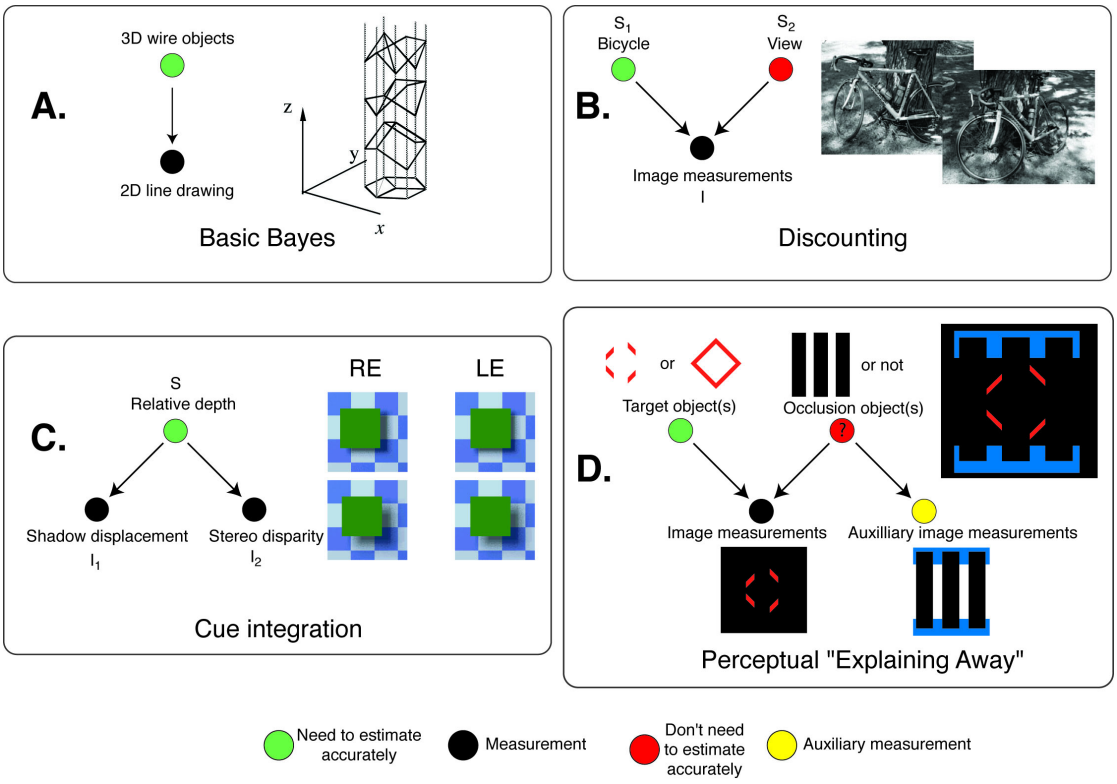
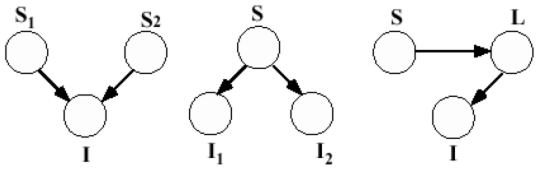


Figure from: Kersten, D., & Yuille, A. (2003). 150–158.

Consider three basic building blocks: converging, diverging, and intermediate nodes in the figure below. Multiple (e.g. scene) variables causing a given image measurement (left), a single variable producing multiple image measurements (middle), or a cause indirectly influencing an image measurement through an intermediate variable (right). These are directed graphs because the arrow shows the direction of influence.



For example, these nodes could correspond to:multiple (scene) causes {shape S1, illumination S2 giving rise to the same image measurement, I ; one cause, S influencing more than one image measurement, {color, I1, brightness, I2}; a scene (or other) cause S, {object identity, S} influencing an image measurement (image contour) through an intermediate variable L (3D shape) .

The arrows tell us how to factor the joint probability into conditionals. So for the three examples above, we have:

p(S1, S2, I) = p(I | S1, S2) p(S1) p(S2)
p(S, I1, I2) = p(I1 | S) p(I2 | S) p(S)
p(S, L, dI) = p(I | L) p(L|S) p(S)

Primary, secondary variables.

Influences between variables are represented by conditioning, and a graphical model expresses the conditional independencies between variables. Two random variables may only become independent, however, once the value of some third variable is known. This is called conditional independence. Recall that two random variables are independent if and only if their joint probability is equal to the product of their individual probabilities. Thus, if p(A,B) = p(A)p(B), then A and B are independent. If p(A,B|C) = p(A|C)p(B|C), then A and B are conditionally independent.

Here's an example to help build intuition. When corn prices drop in the summer, hay fever (allergies) incidence goes up. However, if the joint on corn price and hay fever is conditioned on ``ideal weather for corn and ragweed'', the correlation between corn prices and hay fever drops. This is because corn price and hay fever symptoms are conditionally independent.

You might have heard the saying "correlation is not causation". There is a correlation between eating ice cream and drowning. Why? What event could you condition on to make the dependence go away?

What is noise? Primary and secondary variables.

Noise is whatever variables you don't care to estimate, but contributes to the data. These variables are also called nuisance variables or confounding variables. Yuille and Kersten describe variables as primary and secondary in the context of visual inference.

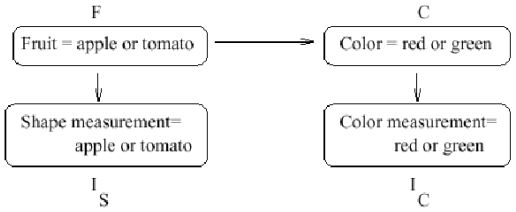
The task determines what is important and what is not. If the task doesn't require you to estimate a random variable (secondary), then it gets integrated out of the joint distribution to produce a marginal distribution over the variable that you are interested in (primary).

One counter-intuitive consequence of this is that the most probable (MAP) estimates are not necessarily consistent across tasks.

This is illustrated in the next section.

Optimal Inference and task dependence: Fruit example

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



The graph specifies how to decompose the joint probability:

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

Generative model: 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], {C, 1, 2}, {F, 1, 2}],
 TableHeadings -> {{"C=r", "C=g"}, {"F=a", "F=t"}}]
```

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

The above conditional is a probability distribution on **C**. So would you expect the sum over a row to be 1? Over a column?

So by the product rule 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	$\frac{3}{8}$	$\frac{1}{16}$

Note that now all the entries sum to 1.

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

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

Which color is a priori more probable?

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

Check whether the joint probability on Fruit and Color can be factored into the product of the prior probabilities on Fruit and Color.

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	$\frac{77}{256}$	$\frac{35}{256}$

Generative model: The likelihood model--probabilities of measurements, i.e. some features given hypotheses

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 , say Is = {am, tm}, and Ic = {rm, gm}. (In a more realistic case, they would have continuous values).

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, rm.
(This may sound too artificial, but a red apple can result in a greenish measurement if illuminated with a greenish light.)
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], {C, 1, 2}, {Ic, 1, 2}],
 TableHeadings -> {{"Ic=rm", "Ic=gm"}, {"C=r", "C=g"}}]
```

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

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[jpFCIsIc[F, C, Is, Ic ], {C, 1, 2}, {F, 1, 2}]
```

Three MAP tasks

We are going to show that the best guess depends on the task.
In other words, given measurements Is, Ic, the most probable choices of fruit and/or color depend on what which combination of hypotheses we care about.

Define argmax[] function:

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

This returns the position of the biggest value in a list.

TASK 1: Pick most probable fruit AND color--Answer "red tomato"

We are given some data--i.e. values of Is and Ic and want to draw some conclusions about what kind of fruit we are looking at, and its material color. First, suppose the task is to make the best bet as to the fruit AND material color from the measurements given. To make it concrete, suppose that we see an "apple-ish 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 guess.
p(F,C | Is, Ic) is given by:

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

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

	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}}
```

So looking at the table we can see that our best bet is “red tomato”.

TASK 2: 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[FCcIsIc[F, C, Is, Ic ], {F, 1, 2}]
```

```
pCTable = Table[pC[C, 1, 1], {C, 1, 2}];
TableForm[pCTable, 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".

TASK 3: Pick most probable fruit--Answer "apple"

Same measurements as before, Is=am=1, and Ic = rm=1. 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[FCcIsIc[F, C, Is, Ic ], {C, 1, 2}]
```

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

F=a	$\frac{803}{1413}$
F=t	$\frac{610}{1413}$

$$\frac{803}{1413}$$

```
{{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*

Further, the above example shows that that most probable answers from the marginals does not have to be consistent with the most probable answers from the joint distribution.

Try expressing the task-dependent consequences using the frequency interpretation of probability.

How to sample from this model?

Appendix

Exercises

Question: Is p(F, C, Is=1, Ic=1) a joint probability on F and C? (Answer: No.)

Redefine **jpFCcIsIcTable** and **jpFCIsIcTable**, without the TableForm[] wrapper and calculate:

```
Total[jpFCcIsIcTable, {2}]
Total[jpFCIsIcTable, {2}]
```

$$\left\{\frac{803}{4096}, \frac{305}{2048}\right\}$$

$$\left\{\frac{803}{4096}, \frac{305}{2048}\right\}$$

Note that we don't always have to calculate the conditional. For example in Task 1 above.

We can use the fact that 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). I.e. the conditional is propor-

tional to the function specified by the joint evaluated at Is=1, and Ic=1.

```
jpFCcIsIcTable = Table[jpFCIsIc[F, C, 1, 1], {F, 1, 2}, {C, 1, 2}];
TableForm[jpFCIsIcTable, 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}}

Using *Mathematica* lists to manipulate discrete priors, likelihoods, and posteriors

A note on list arithmetic

We haven't done standard matrix/vector operations above to do conditioning. We've take advantage of how *Mathematica* divides a 2x3 array by a 2-element vector:

```
M=Array[m,{2,3}]
X = Array[x,{2}]
```

$$\begin{pmatrix} m(1,1) & m(1,2) & m(1,3) \\ m(2,1) & m(2,2) & m(2,3) \end{pmatrix}$$

$\{x(1), x(2)\}$

```
M/X
```

$$\begin{pmatrix} \frac{m(1,1)}{x(1)} & \frac{m(1,2)}{x(1)} & \frac{m(1,3)}{x(1)} \\ \frac{m(2,1)}{x(2)} & \frac{m(2,2)}{x(2)} & \frac{m(2,3)}{x(2)} \end{pmatrix}$$

Putting the probabilities back together again to get the joint

```
Transpose[Transpose[pHx] px]
```

$$\left(\text{px pHx}^{T}\right)^{T}$$

```
pxH pH
```

pH pxH

Getting the posterior from the priors and likelihoods:

One reason Bayes' theorem is so useful is that it is often easier to formulate the likelihoods (e.g. from a causal or generativemodel of how the data could have occurred), and the priors (often from heuristics, or in computational vision empirically testable models of the external visual world). So let's use *Mathematica* to derive **p(H|x)** from **p(x|H**) and **p(H)** , (i.e. pHx from pxH and pH).

```
px2 = Plus @@ (pxH pH)
```

$$pH + pxH$$

```
Transpose[Transpose[(pxH pH)] / Plus @@ (pxH pH)]
```

$$\frac{(pH\ pxH)^{T\,T}}{pH + pxH}$$

Show that this joint probability has a uniform prior (i.e. both priors equal).

```
p = {{1 / 8, 1 / 8, 1 / 4}, {1 / 4, 1 / 8, 1 / 8}}
```

$$\begin{pmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{8} & \frac{1}{8} \end{pmatrix}$$

Marginalization and conditioning: A small dimensional example using list manipulation in *Mathematica*

A discrete joint probability

All of our knowledge regarding the signal discrimination problem can be described in terms of the joint probability of the hypotheses, **H** and the possible data measurements, **x.** The probability function assigns a number to all possible combinations:

p[H, x]

That is, we are assuming that both the hypotheses and the data are discrete random variables.

$$H = \begin{cases} S1 \\ S2 \end{cases}$$

$$x \in \{1, 2, \ldots\}$$

Let's assume that x can only take on one of three values, 1, 2, or 3. And suppose the joint probability is:

```
p = {{1/12, 1/12, 1/6}, {1/3, 1/6, 1/6}}
```

$$\begin{pmatrix} \frac{1}{12} & \frac{1}{12} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{pmatrix}$$

```
TableForm[p, TableHeadings -> {{"H=S1", "H=S2"}, {"x=1", "x=2", "x=3"}}]
```

	x=1	x=2	x=3
H=S1	$\frac{1}{12}$	$\frac{1}{12}$	$\frac{1}{6}$
H=S2	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$

The total probability should sum up to one. Let's test to make sure. We first turn the list of lists into a singel list of scalars using **Flatten[]**. And then we can sum either with **Apply[Plus,Flatten[p]].**

```
Plus @@ Flatten[p]
```

1

We can pull out the first row of p like this:

```
p[[1]]
```

$\left\{\frac{1}{12}, \frac{1}{12}, \frac{1}{6}\right\}$

$\left\{\mathbf{\frac{1}{12}}, \mathbf{\frac{1}{12}}, \mathbf{\frac{1}{6}}\right\}$

$\left\{\frac{1}{12}, \frac{1}{12}, \frac{1}{6}\right\}$

Is this the probability of x? No. For a start, the numbers don't sum to one. But we can get it through the two processes of marginalization and conditioning.

Marginalizing

What are the probabilities of the data, p(x)? To find out, we use the *sum rule* to sum over the columns:

```
px = Apply[Plus, p]
```

$\left\{\frac{5}{12}, \frac{1}{4}, \frac{1}{3}\right\}$

"Summing over "is also called **marginalization** or **"integrating out".** Note that marginalization turns a probability function with higher degrees of freedom into one of lower degrees of freedom.

What are the prior probabilities? p(H)? To find out, we sum over the rows:

```
pH = Apply[Plus, Transpose[p]]
```

$\left\{\frac{1}{3}, \frac{2}{3}\right\}$

Conditioning

Now that we have the marginals, we can get use the *product rule* to obtain the conditional probability through conditioning of the joint:

$$p[x \mid H] = \frac{p[H,\ x]}{p[H]}$$

In the Exercises, you can see how to use *Mathematica* to do the division for conditioning. The syntax is simple:

```
pxH = p / pH
```

$\begin{pmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$

Note that the probability of x conditional on H sums up to 1 over x, i.e. each row adds up to 1. But, the columns do not. **p[x|H]** is a **probability** function of x, but a **likelihood** function of H. The posterior

probability is obtained by conditioning on x:

$$p[H \mid x] = \frac{p[H,\ x]}{p[x]}$$

Syntax here is a bit more complicated, because the number of columns of px don't match the number of rows of p. We use Transpose[] to exchange the columns and rows of p before dividing, and then use Transpose again to get back the 2x3 form:

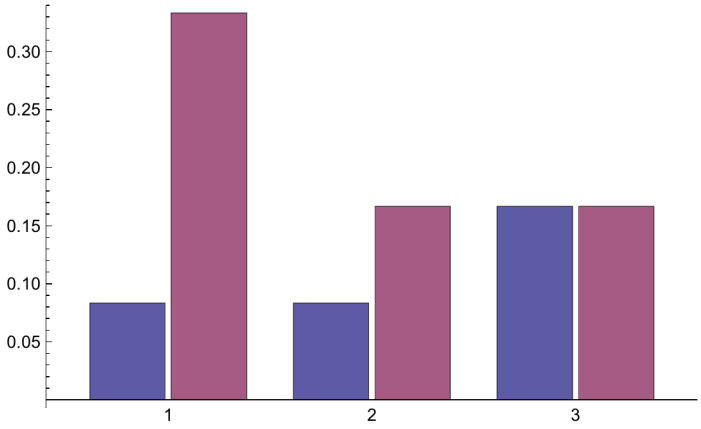
```
pHx = Transpose[Transpose[p] / px]
```

$$\begin{pmatrix} \frac{1}{5} & \frac{1}{3} & \frac{1}{2} \\ \frac{4}{5} & \frac{2}{3} & \frac{1}{2} \end{pmatrix}$$

Plotting the joint

The following BarChart[] graphics function requires in add-in package (**<< Graphics`Graphics`**), which is specified at the top of the notebook. You could also use **ListDensityPlot[]**.

```
BarChart[p[[1]], p[[2]]]
```



Marginalization and conditioning: An example using *Mathematica* functions

A discrete joint probability

All of our knowledge regarding the signal discrimination problem can be described in terms of the joint probability of the hypotheses, **H** and the possible data measurements, **x.** The probability function assigns a number to all possible combinations:

p[H, x]

That is, we are assuming that both the hypotheses and the data are discrete random variables.

$$H = \begin{cases} S1 \\ S2 \end{cases}$$

$$x \in \{1,\ 2,\ \ldots\}$$

Let's assume that x can only take on one of three values, 1, 2, or 3. And suppose the joint probability is:

```
p[H_, x_] := Which[H == 1 && x == 1, 1 / 12, H == 1 && x == 2, 1 / 12, H == 1 && x == 3,
    1 / 6, H == 2 && x == 1, 1 / 3, H == 2 && x == 2, 1 / 6, H == 2 && x == 3, 1 / 6];
```

```
TableForm[Table[p[H, x], {H, 1, 2}, {x, 1, 3}],
 TableHeadings -> {{"H=s1", "H=s2"}, {"X=1", "X=2", "X=3"}}]
```

	X=1	X=2	X=3
H=s1	$\frac{1}{12}$	$\frac{1}{12}$	$\frac{1}{6}$
H=s2	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$

The total probability should sum up to one. Let's test to make sure. We first turn the list of lists into a singel list of scalars using **Flatten[]**. And then we can sum either with **Apply[Plus,Flatten[p]].**

```
Sum[p[H, x], {H, 1, 2}, {x, 1, 3}]
```

```
1
```

We can pull out the first row of p like this:

```
Table[p[1, x], {x, 1, 3}]
```

$$\left\{\frac{1}{12}, \frac{1}{12}, \frac{1}{6}\right\}$$

Is this the probability of x? No. For a start, the numbers don't sum to one. But we can get it through the two processes of marginalization and conditioning.

Marginalizing

What are the probabilities of the data, p(x)? To find out, we use the *sum rule* to sum over the columns:

```
px[x_] := Sum[p[H, x], {H, 1, 2}];
```

```
Table[px[x], {x, 1, 3}]
```

$$\left\{\frac{5}{12}, \frac{1}{4}, \frac{1}{3}\right\}$$

"Summing over "is also called **marginalization** or **"integrating out".** Note that marginalization turns a probability function with higher degrees of freedom into one of lower degrees of freedom.

What are the prior probabilities? p(H)? To find out, we sum over the rows:

```
pH[H_] := Sum[p[H, x], {x, 1, 3}];
```

```
Table[pH[H], {H, 1, 2}]
```

$$\left\{\frac{1}{3}, \frac{2}{3}\right\}$$

Conditioning

Now that we have the marginals, we can get use the *product rule* to obtain the conditional probability through conditioning of the joint:

$$p[x \mid H] = \frac{p[H,\ x]}{p[H]}$$

We use function definition in *Mathematica* to do the division for conditioning. The syntax is simple:

```
pxH[H_, x_] := p[H, x] / pH[H];

Table[pxH[H, x], {H, 1, 2}, {x, 1, 3}]
```

$$\left\{\left\{\frac{1}{4}, \frac{1}{4}, \frac{1}{2}\right\}, \left\{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\right\}\right\}$$

Note that the probability of x conditional on H sums up to 1 over x, i.e. each row adds up to 1. But, the columns do not. **p[x|H]** is a **probability** function of x, but a **likelihood** function of H. The posterior probability is obtained by conditioning on x:

$$p[H \mid x] = \frac{p[H,\ x]}{p[x]}$$

```
pHx[H_, x_] := p[H, x] / px[x];

Table[pHx [H, x], {H, 1, 2}, {x, 1, 3}]
```

$$\left\{\left\{\frac{1}{5}, \frac{1}{3}, \frac{1}{2}\right\}, \left\{\frac{4}{5}, \frac{2}{3}, \frac{1}{2}\right\}\right\}$$

Plotting the joint

We use **ArrayPlot[]**.

```
ArrayPlot[Table[p[H, x], {H, 1, 2}, {x, 1, 3}]]
```

References

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

Cover, T. M., & Joy, A. T. (1991). Elements of Information Theory. New York: John Wiley & Sons, Inc.

Duda, R. O., & Hart, P. E. (1973). Pattern classification and scene analysis . New York.: John Wiley & Sons.

Golden, R. (1988). A unified framework for connectionist systems. Biological Cybernetics, 59, 109-120.

Intrator, N. Combining Exploratory Projection Pursuit and Projection Pursuit Regression. Neural Computation (5):443-455, 1993. http://www.physics.brown.edu/users/faculty/intrator/papers/epp-ppr.ps.gz

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., Mamassian P & Yuille A (2004) Object perception as Bayesian inference. Annual Review of Psychology. (pdf, http://arjournals.annualreviews.org/doi/pdf/10.1146/annurev.psych.55.090902.142005)
Kersten, D., & Yuille, A. (2003). Bayesian models of object perception. Current Opinion in Neurobiology, 13(2) http://gandalf.psych.umn.edu/~kersten/kersten-lab/papers/KerstenYuilleApr2003.pdf
Ma, W. J., Beck, J. M., Latham, P. E., & Pouget, A. (2006). Bayesian inference with probabilistic population codes. Nat Neurosci, 9(11), 1432-1438.
Murphy, K. P. (2012). Machine Learning: a Probabilistic Perspective. MIT Press.
Ripley, B. D. (1996). Pattern Recognition and Neural Networks. Cambridge, UK: Cambridge University Press.
Yuille, A., Coughlan J., Kersten D.(1998) (pdf)
Zhu, S. C., Wu, Y., & Mumford, D. (1998). Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. International Journal of Computer Vision, 27(2), 107–126.
Zoran, D., & Weiss, Y. (2011). From learning models of natural image patches to whole image restoration, 479–486.
http://www.cs.huji.ac.il/~daniez/EPLLICCVCameraReady.pdf