

In[170]:=

```
BasicEntropy[p_] := -p * Log[2, p]
```

In[205]:=

```
TransmittedInformation[{{pys_, pyw_}, {pns_, pnw_}}] :=  
  BasicEntropy[pys + pyw] + BasicEntropy[pns + pnw] + BasicEntropy[pys + pns] +  
  BasicEntropy[pyw + pnw] -  
  BasicEntropy[pys] - BasicEntropy[pyw] - BasicEntropy[pns] -  
  BasicEntropy[pnw]  
(*This is the transinformation aka  
  mutual information in a 2x2 joint probability*)
```

In[206]:=

```
RestrictedInformation[a_, b_, p_] :=  
  TransmittedInformation[{{a * p, b * (1 - p)}, {(1 - a) * p, (1 - b) * (1 - p)}}]  
(*a 2x2 joint probability is four nonnegative  
  numbers that satisfy the constraint that they  
  sum to 1; this leaves three free variables. We  
  can parameterize this as the probability p  
  that there is a signal,  
  and conditional probabilities b that given a signal the receiver says  
  yes and the probability a that given no signal the receiver says yes*)
```

In[207]:=

```
RestrictedInformation[.4, .6, .5]
```

Out[207]=

```
0.0290494
```

In[173]:=

```
ImplicitContourDerivative[a_, b_, p_] =  
  -Simplify[D[RestrictedInformation[a, b, p], a]] /  
  Simplify[D[RestrictedInformation[a, b, s], b]];  
(*basic multivariable calculus tells  
  us that the slope of a level contour of a function  
  f(a,b) is given by the differential equation  $\frac{db}{da} = -((\partial f / \partial a) / (\partial f / \partial b))$ ;  
  in our case the amounts to the nonlinear differential equation
```

$$\frac{db}{da} = - \frac{s \operatorname{Log}\left[\left(\frac{a}{1-a}\right) \left(\frac{(1-b) + (b-a)s}{b - (b-a)s}\right)\right]}{(1-s) \left(\operatorname{Log}\left[\left(\frac{b}{1-b}\right) \left(\frac{(1-b) + (b-a)s}{b - (b-a)s}\right)\right]\right)} *)$$

In[174]:=

```
bprime[a_, s_] := ImplicitContourDerivative[a, b[a], s]  
bNprime[a_, b_, n_, s_] := bNprime[a, b, n, s] =  
  (D[bNprime[a, b, n - 1, s] /. b -> b[a], a] /. b'[a] -> bprime[a, s]) /. b[a] -> b  
bNprime[a, b, 1, s] := bprime[a, s] /. b[a] -> b  
(*this is the mathematica expression  
  of the nonlinear differential equation above*)
```

```

TaylorCoefficient[x0_, y0_, n_, s0_] :=
  bNprime[a, b, n, s] / n! /. {a → x0, b → y0, s → s0}
TaylorSeries[x_, x0_, y0_, n_, s_] :=
  y0 + Sum[TaylorCoefficient[x0, y0, k, s] * (x - x0) ^ k, {k, 1, n}]
(*This function gives a Taylor series approximation of the
  isoinformation contour that passes through the point x0,y0*)

```

Above is "mathematica" code for a function which gives the nth order Taylor series approximation to that isoinformation contour of the transmitted information function which passes through the point (x0,y0) when the probability of a "signal" is "s", while the probability of merely noise is "1-s". This code can be memory intensive, you are advised allocate sufficient memory to the task, and/or to start with "n" small and work up to larger values. Below are: 1) Numerically computed contours, 2) a fifth order Taylor approximation to the isoinformation contour and 3) A plot of this approximation, all in the case that "s"=.5. BE AWARE that these approximations will always give inaccurate results near where the isoinformation contour intersects the lines x=0, x=1, y=0, y=1.

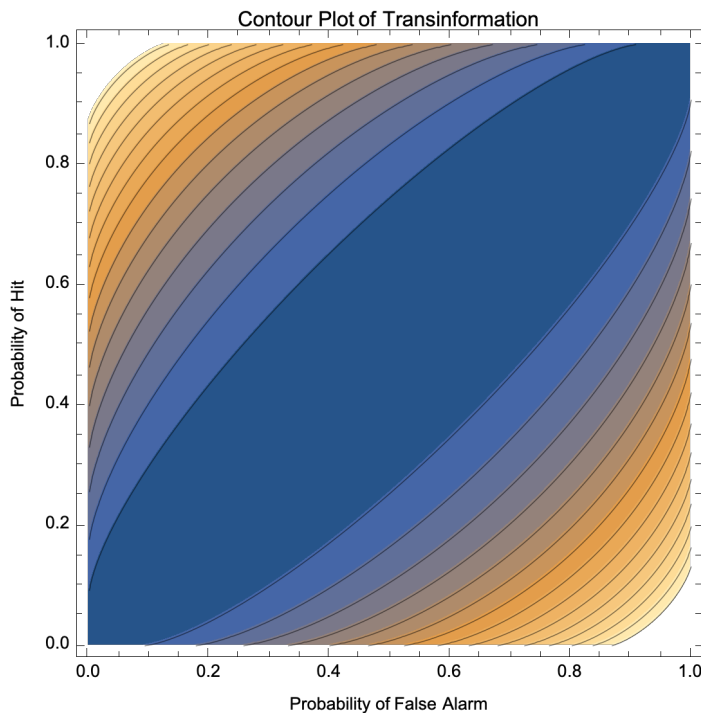
In[218]:=

```

ContourPlot[RestrictedInformation[a, b, .5],
  {a, 0.0001, .9999}, {b, 0.0001, .9999},
  PlotPoints → 50, Contours → 20, PlotLabel → "Contour Plot of Transinformation",
  FrameLabel → {"Probability of False Alarm", "Probability of Hit"}]
(*Below is a contour plot of the transinformation,
  the contours being the isoinformation curves*)

```

Out[218]=



In[212]:=

```

TaylorSeries[x, .25, .75, 5, .5]
RestrictedInformation[.25, .75, .5]
(*this uses the fifth order Taylor polynomial to compute an
approximation of the contour that passes through the point a=.25,
b=.75 when p=.5. The transinformation for this case is 0.188722*)

```

Out[212]=

```

0.75 + 1. (-0.25 + x) - 1.21365 (-0.25 + x)^2 +
1.47295 (-0.25 + x)^3 - 3.80799 (-0.25 + x)^4 + 9.52554 (-0.25 + x)^5

```

Out[213]=

```

0.188722

```

```

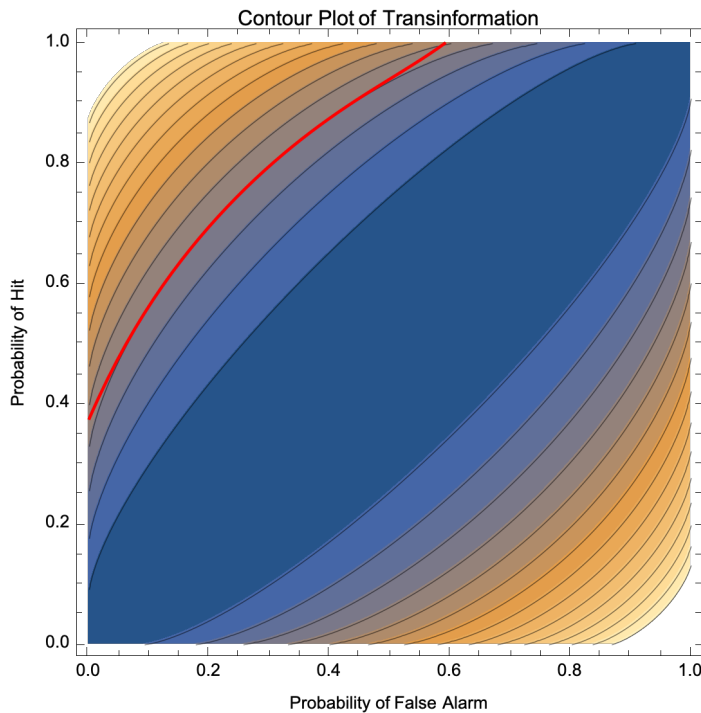
f[x_] := 0.75` + 1.` (-0.25` + x) -
1.2136523021691166` (-0.25` + x)^2 + 1.4729519105603963` (-0.25` + x)^3 -
3.8079867959311535` (-0.25` + x)^4 + 9.525541162887286` (-0.25` + x)^5
(*we encapsulate the Taylor polynomial above as a function f(x)*)

```

In[221]:=

```
Show[ContourPlot[RestrictedInformation[a, b, .5],
  {a, 0.0001, .9999}, {b, 0.0001, .9999},
  PlotPoints → 50, Contours → 20, PlotLabel → "Contour Plot of Transinformation",
  FrameLabel → {"Probability of False Alarm", "Probability of Hit"}, Plot[f[x],
  {x, 0, .75}, PlotRange → {0, 1}, AspectRatio → 1, Frame → True, PlotStyle → Red]]
(*Here we lay the Taylor polynomial approximation to the contour in
  red on top of the contours portrayed above. We see that, as promised,
  the Taylor polynomial is quite close to the contour near the point (.25,.75),
  but diverges significantly as the contour nears x=0 and near y=1*)
```

Out[221]=



(*The right hand side of the differential equation representing contours is just below*)

$$\left(\frac{p}{1-p} \right) \frac{\log\left[\left(\frac{a}{1-a}\right)\right] - \log\left[\frac{b(1-p)+ap}{1-b(1-p)-ap}\right]}{\log\left[\left(\frac{b}{1-b}\right)\right] - \log\left[\frac{b(1-p)+ap}{1-b(1-p)-ap}\right]}$$

(*when p=1/2 this amounts to the following*)


$$-\frac{\left(\log\left[\frac{a}{1-a}\right] - \log\left[\frac{0.5 \cdot a + 0.5 \cdot b[a]}{1 - 0.5 \cdot a - 0.5 \cdot b[a]}\right]\right)}{\log\left[\frac{b[a]}{1-b[a]}\right] - \log\left[\frac{0.5 \cdot a + 0.5 \cdot b[a]}{1 - 0.5 \cdot a - 0.5 \cdot b[a]}\right]}$$


(*here we obtain a numerical approximation of the differential equation;
there are some computational difficulties near where the contour
encounters conditional probabilities of 0 or 1 but the numerics
are robust enough that useful results are obtained anyway*)

q =

$$\text{NDSolve}\left[\left\{b'[a] == -\frac{\left(\text{Log}\left[\frac{a}{1-a}\right] - \text{Log}\left[\frac{0.5\,a+0.5\,b[a]}{1-0.5\,a-0.5\,b[a]}\right]\right)}{\text{Log}\left[\frac{b[a]}{1-b[a]}\right] - \text{Log}\left[\frac{0.5\,a+0.5\,b[a]}{1-0.5\,a-0.5\,b[a]}\right]}, b[0.25] == .75\right\}, b[a], \{a, 0, .75\}\right]$$

Out[214]=

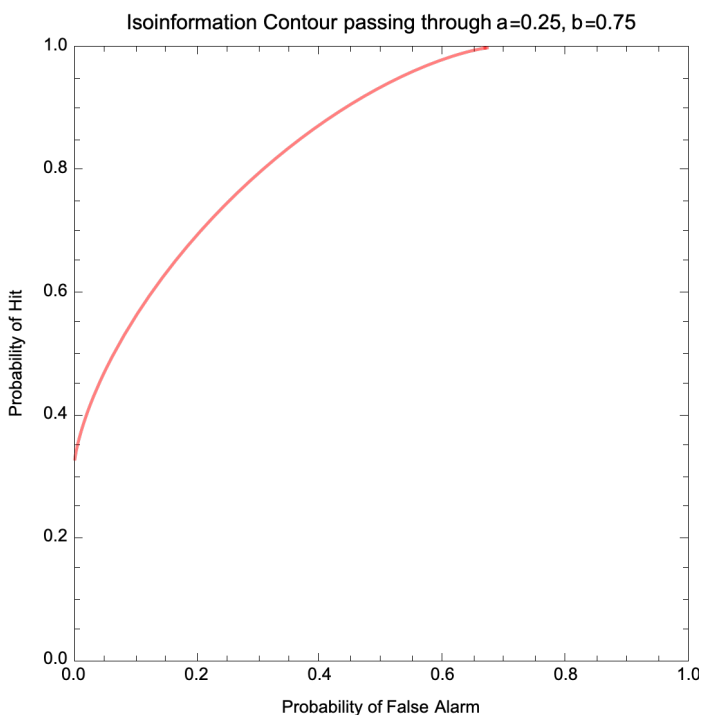
{ {b[a] → InterpolatingFunction[ Domain: {{1.37×10⁻⁶, 0.671}} Output: scalar] [a] } }

Data not in notebook. Store now 

In[222]:=

```
plt = Plot[Evaluate[b[a] /. q], {a, 0, 1}, PlotRange → {{0, 1}, {0, 1}},
  AspectRatio → 1, PlotStyle → {Opacity[.5], Red}, Frame → True,
  PlotLabel → "Isoinformation Contour passing through a=0.25, b=0.75",
  FrameLabel → {"Probability of False Alarm", "Probability of Hit"}]
```

Out[222]=



In[225]:=

```
Show[ContourPlot[RestrictedInformation[a, b, .5],
  {a, 0.0001, .9999}, {b, 0.0001, .9999},
  PlotPoints → 50, Contours → 20,
  PlotLabel → "Approximate Isoinformation Contour passing through a=0.25, b=0.75",
  FrameLabel → {"Probability of False Alarm", "Probability of Hit"}], plt]
(*Here we lay the numerical approximation to the contour in red on top of
the contours portrayed above. We see that the numerical approximation
is more faithful to the correct contour than the Taylor polynomial*)
```

Out[225]=

