noise exercises

Feb 13 2012
Jim Mahoney

I did both of these analytically with *Mathematica* to handle the algrebra and draw plots.

symmetric binary channel

X = 0 or 1 = sent bit
Y = 0 or 1 = received bit
f = probability of bit flip during transmit.

P(Y|X) = probability of y given x

$$\texttt{PyGx[f\_] :=} \begin{pmatrix} 1-f & f \\ f & 1-f \end{pmatrix}$$

P(X) = probability of x

$$\texttt{Px[z\_] :=} \begin{pmatrix} z \\ 1-z \end{pmatrix}$$

P(Y) = probability of y

```
Py = PyGx[f] . Px[z]
```

```
{{f (1 - z) + (1 - f) z}, {(1 - f) (1 - z) + f z}}
```

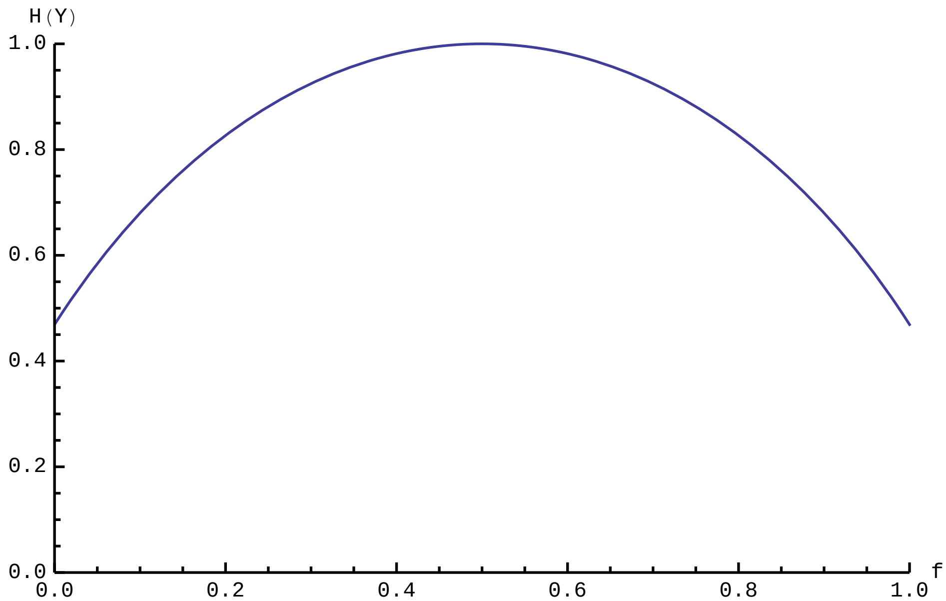
H(Y) = entropy of y

```
Hy = -Total[Py * Log[2, Py]]
```

$$\left\{-\frac{(f(1-z)+(1-f)z)\,\text{Log}[f(1-z)+(1-f)z]}{\text{Log}[2]} - \frac{((1-f)(1-z)+fz)\,\text{Log}[(1-f)(1-z)+fz]}{\text{Log}[2]}\right\}$$

Plot of H(Y) vs z for a given value of f

```
Plot[Hy /. f → 0.1, {z, 0, 1},
 PlotRange → {{0, 1}, {0, 1}}, AxesLabel → {"f", "H(Y)"}]
```

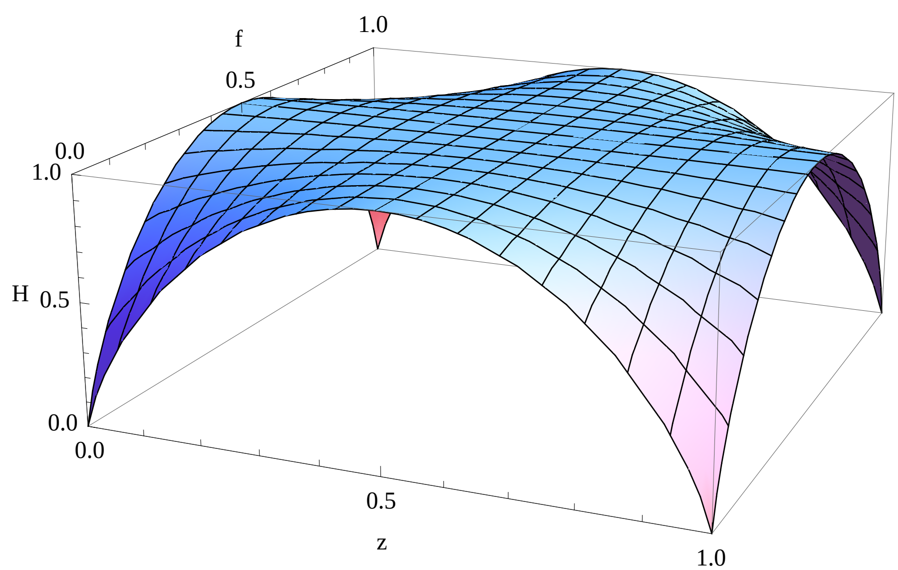


3D plot of H(Y), f, z .

At f=0.5, Y = 0,1 with 50/50 chance no matter what is sent.

Since this is perfectly uncertain, it also has information 1 by our definitions ... even though it has nothing to do with the sent bits.

```
Plot3D[Hy, {z, 0, 1}, {f, 0, 1},
 PlotRange → {{0, 1}, {0, 1}, {0, 1}}, AxesLabel → {"z", "f", "H"}]
```



P(X,Y) = probability of x and y = joint probability distribution

```
PyAx = Table[PyGx[f][[i, j]] * Px[z][[j]], {i, 1, 2}, {j, 1, 2}]
```

```
{{{(1 - f) z}, {f (1 - z)}}, {{f z}, {(1 - f) (1 - z)}}}
```

H(Y|X) = entropy of y given x
Note that since x is given, and the noise is the same for 0's and 1's, it makes sense that this is independent of z.

```
HyGx = Simplify[Total[-Flatten[PyAx] * Flatten[Log[2, PyGx[f]]]]]
```

$$\frac{(-1+f)\,\mathrm{Log}[1-f] - f\,\mathrm{Log}[f]}{\mathrm{Log}[2]}$$

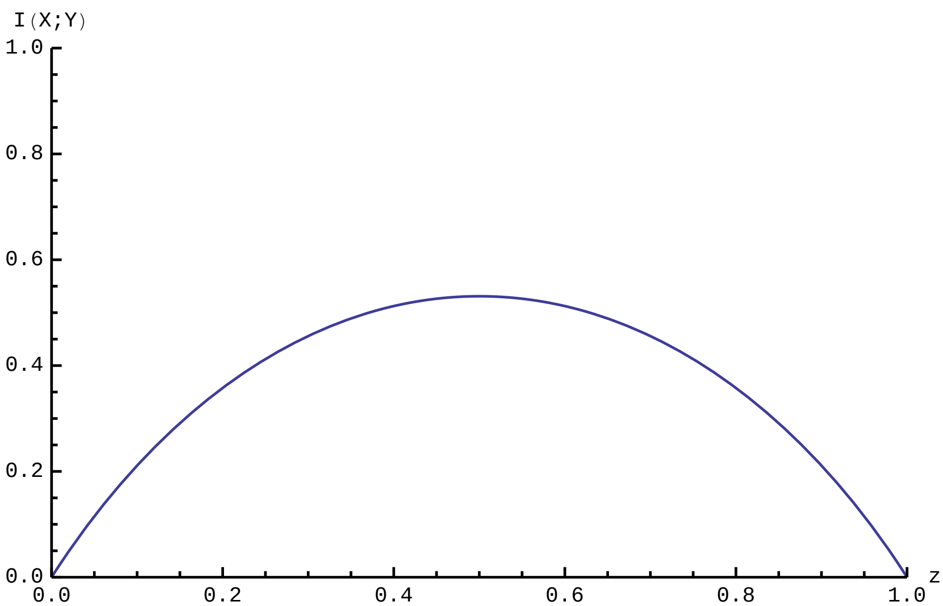
I(X;Y) = mutual entropy = channel capacity for a given P(X).

```
Ixy = Hy - HyGx
```

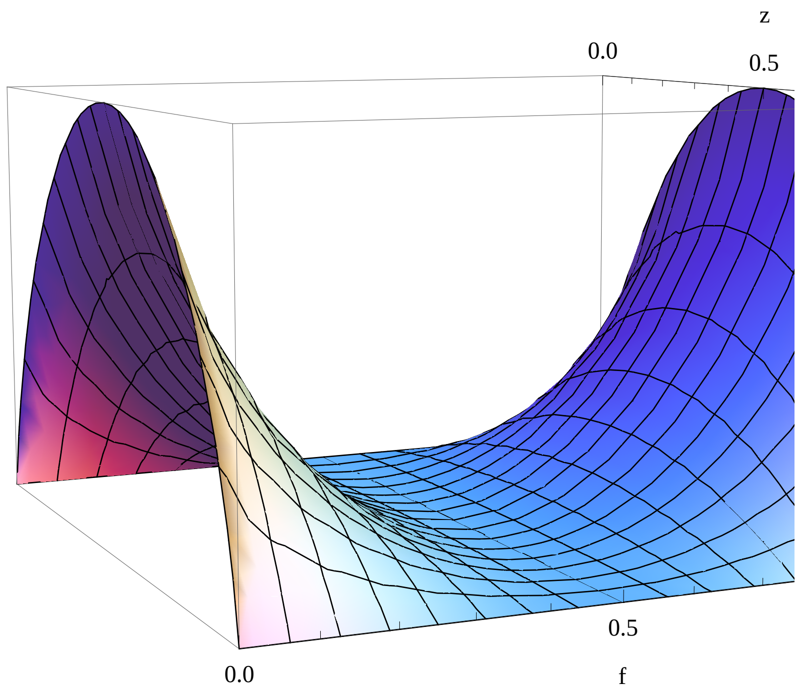
$$\left\{-\frac{(-1+f)\,\mathrm{Log}[1-f] - f\,\mathrm{Log}[f]}{\mathrm{Log}[2]} - \frac{1}{\mathrm{Log}[2]}(f\,(1-z) + (1-f)\,z)\,\mathrm{Log}[f\,(1-z) + (1-f)\,z] - \frac{1}{\mathrm{Log}[2]}((1-f)\,(1-z) + f\,z)\,\mathrm{Log}[(1-f)\,(1-z) + f\,z]\right\}$$

Plot of capacity vs z for a given value for the noise.

```
Plot[Ixy /. f → 0.1, {z, 0, 1},
 PlotRange → {{0, 1}, {0, 1}}, AxesLabel → {"z", "I(X;Y)"}]
```

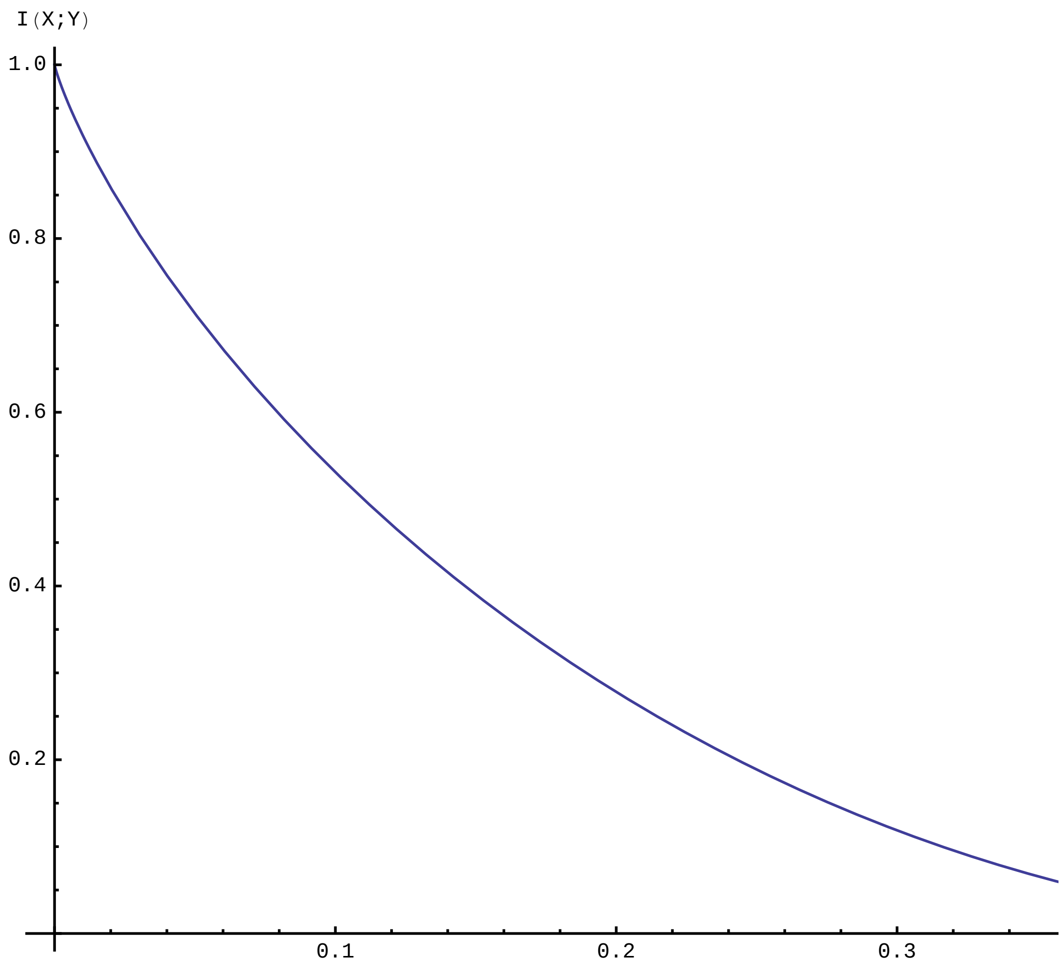


```
Plot3D[Ixy, {z, 0, 1}, {f, 0, 1},
 PlotRange → {{0, 1}, {0, 1}, {0, 1}},
 BoxRatios → {1, 1, 0.6},
 AxesLabel → {"z", "f", "I"}]
```



Finally, a plot of the max capacity (i.e. z=0.5) as a function of the noise probability. This is the big result : how many bits of information per bit of data can we send through this channel as a function of the probability f of a bit flip error.

```
Plot[Ixy /. z → 0.5, {f, 0, 0.5}, AxesLabel → {"f", "I(X;Y)"}]
```



Jim’s 3 symbol example

X = (1,2,3) = sent symbol
Y = (a,b,c) = received bit

P(Y|X) = probability of y given x

$$\texttt{PyGx[f\_] :=} \begin{pmatrix} 1-f & f/2 & f/3 \\ f & 1-f & 2f/3 \\ 0 & f/2 & 1-f \end{pmatrix}$$

P(X) = probability of x

$$\texttt{Px[z\_, o\_] :=} \begin{pmatrix} z \\ o \\ 1-z-o \end{pmatrix}$$

P(Y) = probability of y

```
Py = PyGx[f] . Px[z, o]
```

$$\left\{\left\{\frac{f\,o}{2}+\frac{1}{3}f\,(1-o-z)+(1-f)\,z\right\},\ \left\{(1-f)\,o+\frac{2}{3}f\,(1-o-z)+f\,z\right\},\ \left\{\frac{f\,o}{2}+(1-f)\,(1-o-z)\right\}\right\}$$

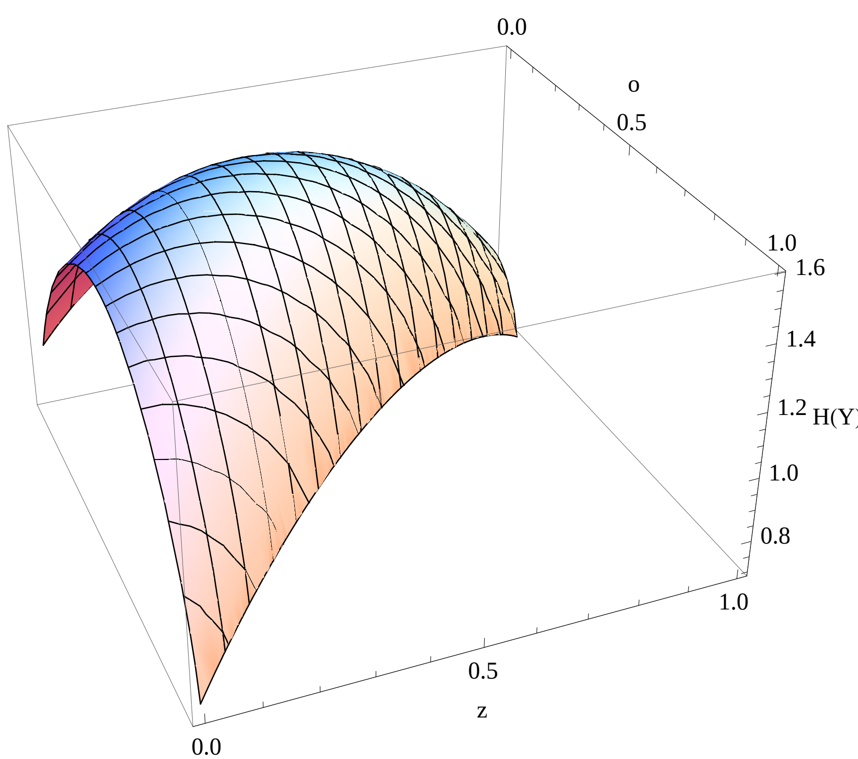
H(Y) = entropy of y

```
Hy = -Total[Py * Log[2, Py]]
```

$$\left\{-\frac{1}{\text{Log}[2]}\left(\frac{f\,o}{2}+(1-f)\,(1-o-z)\right)\text{Log}\left[\frac{f\,o}{2}+(1-f)\,(1-o-z)\right]-\frac{1}{\text{Log}[2]}\left(\frac{f\,o}{2}+\frac{1}{3}f\,(1-o-z)+(1-f)\,z\right)\text{Log}\left[\frac{f\,o}{2}+\frac{1}{3}f\,(1-o-z)+(1-f)\,z\right]-\frac{1}{\text{Log}[2]}\left((1-f)\,o+\frac{2}{3}f\,(1-o-z)+f\,z\right)\text{Log}\left[(1-f)\,o+\frac{2}{3}f\,(1-o-z)+f\,z\right]\right\}$$

Plot of H(Y) vs (z,o) for a given value of f

```
Plot3D[Hy /. f → 0.2, {z, 0, 1}, {o, 0, 1},
 BoxRatios → {1, 1, 0.6},
 AxesLabel → {"o", "z", "H(Y)"}]
```



P(X,Y) = probability of x and y = joint probability distribution

```
PyAx = Table[PyGx[f][[i, j]] * Px[z, o][[j]], {i, 1, 3}, {j, 1, 3}]
```

$$\left\{\left\{\{(1-f)\,z\},\ \left\{\frac{f\,o}{2}\right\},\ \left\{\frac{1}{3}\,f\,(1-o-z)\right\}\right\},\ \left\{\{f\,z\},\ \{(1-f)\,o\},\ \left\{\frac{2}{3}\,f\,(1-o-z)\right\}\right\},\ \left\{\{0\},\ \left\{\frac{f\,o}{2}\right\},\ \{(1-f)\,(1-o-z)\}\right\}\right\}$$

H(Y|X) = entropy of y given x

(First remove the P(Y|X) and P(Y,X) zero terms which give 0*log(0) which *Mathematica* doesn't like.)

```
flatNoZero[x_] := DeleteCases[Flatten[x], 0];

HyGx = Total[-flatNoZero[PyAx] * Log[2, flatNoZero[PyGx[f]]]]
```

$$-\frac{(1-f)\,o\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}-\frac{(1-f)\,(1-o-z)\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}-\frac{(1-f)\,z\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}-\frac{f\,(1-o-z)\,\mathrm{Log}\left[\frac{f}{3}\right]}{3\,\mathrm{Log}[2]}-\frac{f\,o\,\mathrm{Log}\left[\frac{f}{2}\right]}{\mathrm{Log}[2]}-\frac{2\,f\,(1-o-z)\,\mathrm{Log}\left[\frac{2f}{3}\right]}{3\,\mathrm{Log}[2]}-\frac{f\,z\,\mathrm{Log}[f]}{\mathrm{Log}[2]}$$

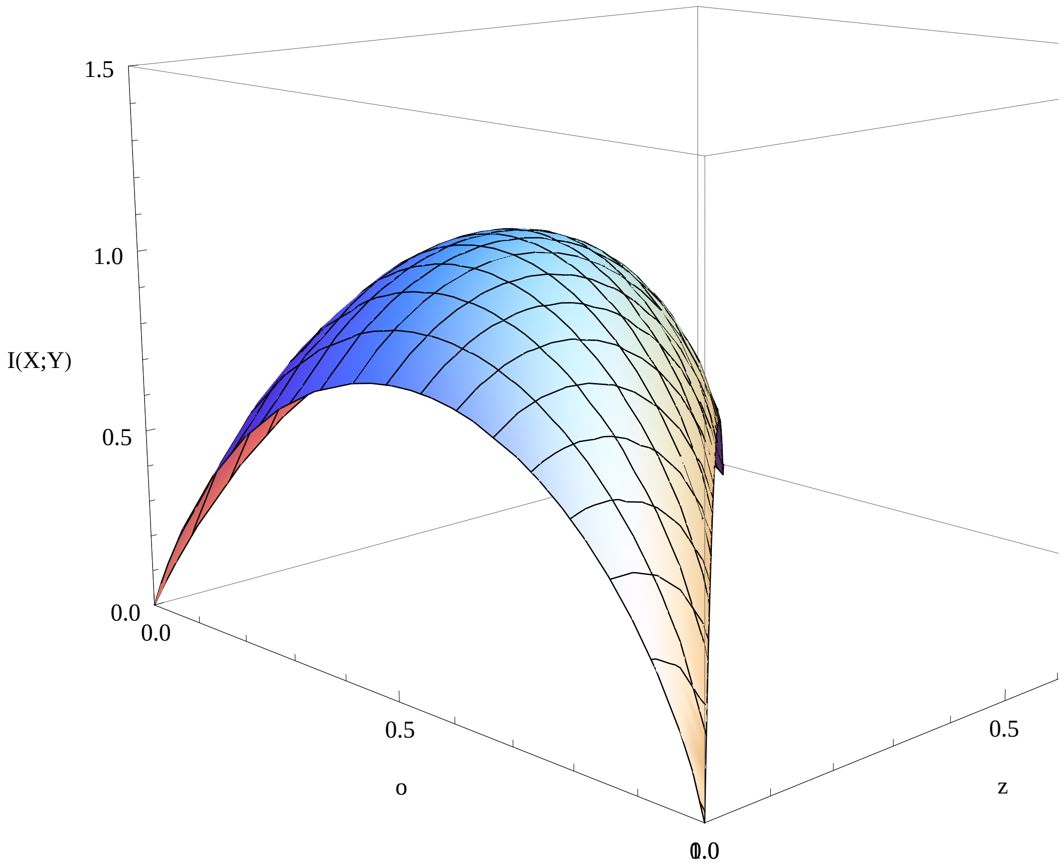
I(X;Y) = mutual entropy = channel capacity for a given P(X).

```
Ixy = Hy - HyGx
```

$$\left\{\frac{(1-f)\,o\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}+\frac{(1-f)\,(1-o-z)\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}+\frac{(1-f)\,z\,\mathrm{Log}[1-f]}{\mathrm{Log}[2]}+\frac{f\,(1-o-z)\,\mathrm{Log}\left[\frac{f}{3}\right]}{3\,\mathrm{Log}[2]}+\frac{f\,o\,\mathrm{Log}\left[\frac{f}{2}\right]}{\mathrm{Log}[2]}+\frac{2\,f\,(1-o-z)\,\mathrm{Log}\left[\frac{2f}{3}\right]}{3\,\mathrm{Log}[2]}+\frac{f\,z\,\mathrm{Log}[f]}{\mathrm{Log}[2]}-\frac{1}{\mathrm{Log}[2]}\left(\frac{f\,o}{2}+(1-f)\,(1-o-z)\right)\mathrm{Log}\left[\frac{f\,o}{2}+(1-f)\,(1-o-z)\right]-\frac{1}{\mathrm{Log}[2]}\left(\frac{f\,o}{2}+\frac{1}{3}\,f\,(1-o-z)+(1-f)\,z\right)\mathrm{Log}\left[\frac{f\,o}{2}+\frac{1}{3}\,f\,(1-o-z)+(1-f)\,z\right]-\frac{1}{\mathrm{Log}[2]}\left((1-f)\,o+\frac{2}{3}\,f\,(1-o-z)+f\,z\right)\mathrm{Log}\left[(1-f)\,o+\frac{2}{3}\,f\,(1-o-z)+f\,z\right]\right\}$$

Plot of capacity vs z for a given value for the noise.

```
Plot3D[Ixy /. f → 0.1, {z, 0, 1}, {o, 0, 1},
 BoxRatios → {1, 1, 0.7}, PlotRange → {{0, 1}, {0, 1}, {0, 1.5}},
 AxesLabel → {"o", "z", "I(X;Y)"}]
```



Finally, a plot of the max capacity (i.e. z=0.5) as a function of the noise probability.
This is the big result : how many bits of information per bit of data can we send through this channel as a function of the probability f of a bit flip error.

In this case, *Mathematica* isn't doing a great job of finding the channel max I, which happens at the (z,o) value that maximizes I.
Seems that even when doing it numerically, it keeps heading off into ranges where things aren't defined well.

Here I'm using their FindMaximum to look for a max at a specific value for f, and then building up a table of values.
This expression says to look at Ixy where f is 0.1, starting at (z,o)=(0.1,0.1), and then vary z,o to find the highest value.
What gets returned as a solution has the maximum in its first component.

```
FindMaximum[Ixy /. f → 0.1, {z, 0.1}, {o, 0.1}]
```

```
{1.05387, {z → 0.360735, o → 0.293443}}
```

So here I do that for many values of f, saving the results in a table.

```
result := Table[
  {ff,
   FindMaximum[Ixy /. f → ff, {z, 0.1}, {o, 0.1}][[1]]},
  {ff, 0.02, 0.4, 0.02}]
```

```
result
```

```
{{0.02, 1.43079}, {0.04, 1.31734}, {0.06, 1.21974}, {0.08, 1.13274},
 {0.1, 1.05387}, {0.12, 0.981696}, {0.14, 0.915308}, {0.16, 0.854081},
 {0.18, 0.797586}, {0.2, 0.745526}, {0.22, 0.697716}, {0.24, 0.654061},
 {0.26, 0.61456}, {0.28, 0.579313}, {0.3, 0.548541}, {0.32, 0.522629},
 {0.34, 0.502187}, {0.36, 0.488176}, {0.38, 0.482105}, {0.4, 0.486419}}
```

And then we plot the table.

```
ListPlot[result, AxesLabel → {"f", "max I(X;Y)"}]
```