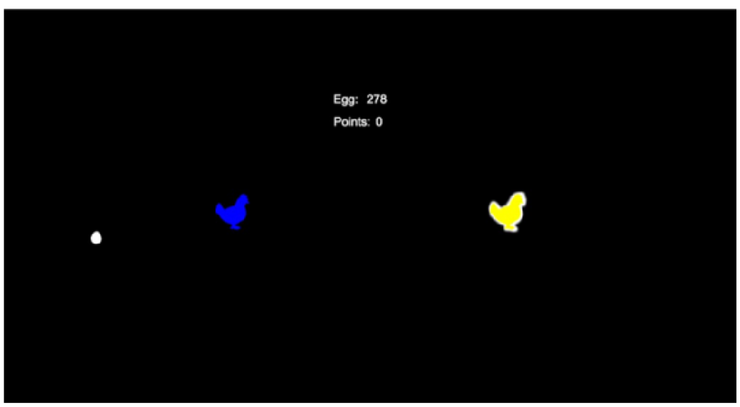
Modeling Updating of Subjective Hazard Rate

Katherine Forthman

12.07.2018

Statement of Original Problem

- In this task, participants are confronted with a simulated environment where there are two choices. The participant is asked to predict the correct choice based on environmental ques and learned expectations from previous trials. In this case, the two choices are two chickens. The egg is generated in some vicinity of the correct chicken, and stands as our environmental que.
- There is a performance trade-off in the learning process between bias (how much decision is based on previous evidence) and variability (how often one's choice changes from one trial to the next).



Normative model

- Asks "how should we do this" rather than "how can we do this".
- The goal is finding the optimal solution to a problem. This is in contrast to a process model, which seeks to describe how a problem is solved.
- Glaze et al. built an optimal change detection algorithm to study how well human subjects detect change.

Problem Analysis

We want to estimate how people respond to randomness in their environment. Do people alter their behavior with respect to randomness accurately? To measure this, we first need to decide what an appropriate response would be. We first want to model ideal prediction of hazard based on information from previous trials and the position of the egg. Then we can use this model to figure out what a subject's predicted hazard rate was based on their choice of the correct chicken.

Assumptions and Simplifications

This model simplifies a function of the human brain to a single equation. The neurons and the networks between them are ignored. There is no brain scan to tell us anything about which synapses are firing. Essentially, this model is not driven by an understanding of the mechanics within the brain. It is based on the observation of human behavior, i.e. the output of the system. This model assumes that humans will use an optimal model when making decisions; a Bayesian model. We will also assume that this model is consistent for everyone, but the parameters of the model (such as hazard rate) are free to vary between individuals.

Procedure Outline

1. Create an ideal observer model for one of the subjects.
2. Estimate H for one of the subjects to better fit the model to their response.
3. Compare subjective H to true H.

The Model

$$L_n = \Psi(L_{n-1}, H) + \mathrm{LLR}_n \qquad (1)$$

- This model uses maximum likelihood estimation to find the optimal solution.
- The adaptive information-accumulation process is based on computing for each trial, n, the logarithm of the posterior odds of the two sources given:
  - all sensory evidence (the egg position, x) collected until that trial
  - knowledge of the current hazard rate, H, which is the expected probability at each time step that the source will switch from one chicken to the other. A lower H means higher stability.
- L_n represents belief of a simulated ideal learner. The sign of L_n indicates which source is currently believed to be correct by the optimal model. L_n is calculated as the logarithm of the posterior odds of the alternative choices.
- L_n is computed using the following equation :

$$L_n = \log\left[\frac{P(\text{right source}|x_{1:n}, H)}{P(\text{left source}|x_{1:n}, H)}\right] = \log\left[\frac{P(\text{right source}|x_{1:n}, H)}{1 - P(\text{right source}|x_{1:n}, H)}\right] = \Psi(L_{n-1}, H) + \mathrm{LLR}_n \qquad (2)$$

- LLR_n represents the sensory evidence. LLR_n is the log likelihood ratio provided by the egg position on that trial.
- Ψ is the time–varying prior expectation (the logarithm of the prior odds) about the source before observing the new sensory evidence. Ψ is computed using the following equation:

$$\Psi(L_{n-1}, H) = L_{n-1} + \log\left[\frac{1-H}{H} + \exp(-L_{n-1})\right] - \log\left[\frac{1-H}{H} + \exp(L_{n-1})\right] \tag{3}$$

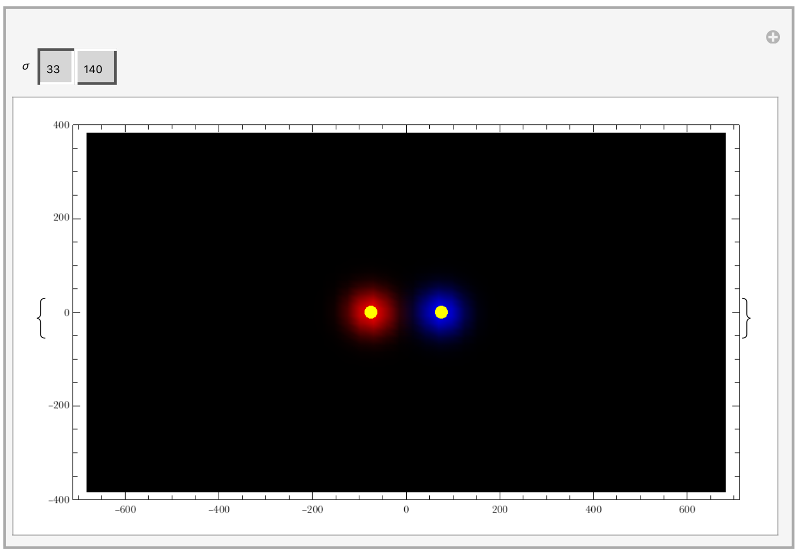
- Ψ is the key feature of this model. The Ψ function controls the dynamics of information accumulation as a function of learned H.
- The prior belief, Ψ, approaches a stabilizing boundary with time whose height directly depends on H.
- To summarize, the belief of the ideal-observer, L_n, is a function of temporal (the prior belief, Ψ) and spatial (the log-likelihood ratio, LLR_n) evidence.

Choosing the Parameters of the Environment

We need to decide the egg distribution around each chicken. It is necessary that the eggs are not generated outside the screen. We also need two comparative egg distributions, one with overlap and one without. Below is an exploration of the two standard deviations we chose.

In[95]:=
```
Manipulate[
 Module[{pL, pR, rFun, bFun, cFun, x, y, w = 1366, h = 768, d = 75},
  {
   pL[x_, y_] := PDF[NormalDistribution[-d, σ], x] * PDF[NormalDistribution[0, σ], y];
   pR[x_, y_] := PDF[NormalDistribution[d, σ], x] * PDF[NormalDistribution[0, σ], y];
   rFun[x_, y_] := Round[pL[x, y] / pL[-d, 0], .001];
   bFun[x_, y_] := Round[pR[x, y] / pR[d, 0], .001];
   cFun = Function[{x, y}, RGBColor[rFun[x, y], 0, bFun[x, y]]];
   Show[
    ParametricPlot[{x, y}, {x, -w / 2, w / 2}, {y, -h / 2, h / 2}, ColorFunctionScaling →
      False, ColorFunction → cFun, Axes → False, PlotPoints → 50, ImageSize → Large],
    ListPlot[{{-d, 0}, {d, 0}}, PlotStyle → Yellow]
   ]
  }
 ],
 {σ, {33, 140}}
]
```

Out[95]=



Qualitative & Numeric Behavior

Exploration of the model.

In[96]:=
```
Remove["Global`*"]
```

First, set the parameters.

In[97]:=
```
μ = 75;
σ = 33;
h = .1;
m = 100;
```

Then, generate some random data.

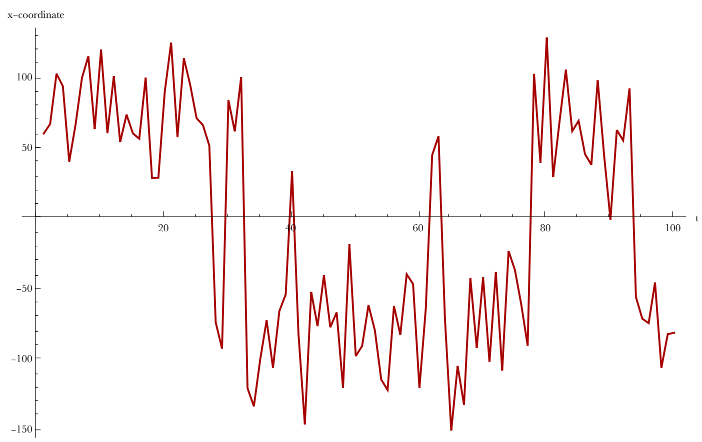
In[101]:=
```
r = RandomVariate[UniformDistribution[{0, 1}], m];
```

In[102]:=
```
switch = Table[If[r[[i]] > h, 0, 1], {i, 1, m}];
dist = {};
val = 1;
For[i = 1, i ≤ m, i++,
 If[switch[[i]] == 1,
  val = If[val == 1, 2, 1]
 ];
 dist = Append[dist, val]
]
x = Table[If[dist[[i]] == 1, RandomVariate[NormalDistribution[μ, σ], 1],
    RandomVariate[NormalDistribution[-μ, σ], 1]], {i, 1, m}];
x = Transpose[x] // Flatten;
```

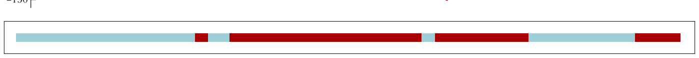
The following plot shows the x-coordinate of the egg in each trial for our random data. The array plot below it shows which chicken produced the egg.

In[108]:=
```
ListPlot[Table[x[[i]], {i, 1, m}], Joined → True,
 AxesLabel → {"t", "x-coordinate"}, ImageSize → Large]
ArrayPlot[{dist}, ImageSize → Large]
```

Out[108]=



Out[109]=



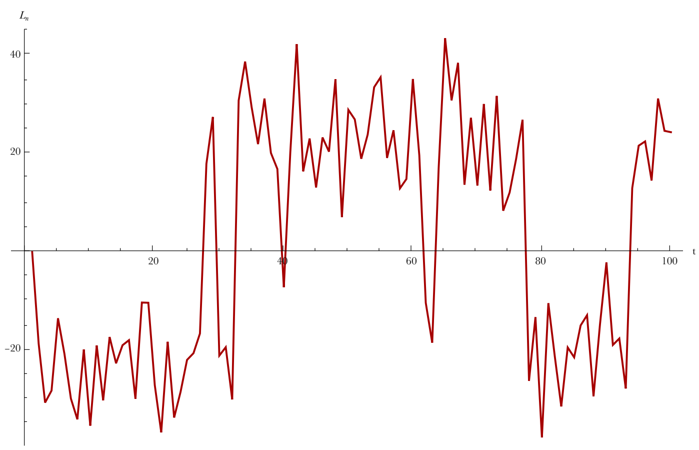
Now, based on the data above, the model will make a prediction of which chicken the egg came from.

In[110]:=
```
l[1] = 0;
For[n = 2, n ≤ m, n++,
 ψ[n] = l[n - 1] + Log[(1 - h)/h + Exp[-l[n - 1]]] - Log[(1 - h)/h + Exp[l[n - 1]]] // N;
 llr[n] = -2 * (LogLikelihood[NormalDistribution[μ, σ], {x[[n]]}] -
      LogLikelihood[NormalDistribution[-μ, σ], {x[[n]]}]) // N;
 l[n] = ψ[n] + llr[n];
]
```

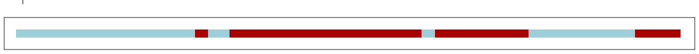
The plot below demonstrates how the belief of the model changes over time in response to changes in the environment.

In[112]:=
```
ListPlot[Table[l[i], {i, 1, m}], Joined → True, AxesLabel → {"t", "Ln"}, ImageSize → Large]
ArrayPlot[{dist}, ImageSize → Large]
```

Out[112]=



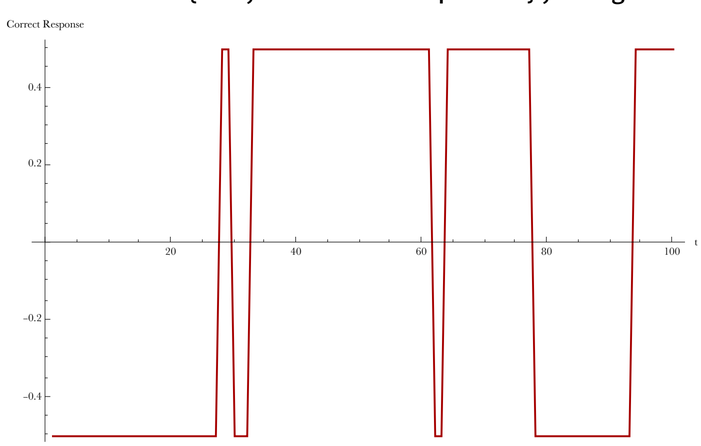
Out[113]=



It is clear that it matches the correct answer very well:

In[114]:=
```
ListPlot[dist - 1.5, Joined → True,
 AxesLabel → {"t", "Correct Response"}, ImageSize → Large]
```

Out[114]=



Percent correct:

In[115]:=
```
pred = Table[If[l[i] > 0, 2, 1], {i, 1, m}];
Sum[If[dist[[i]] == pred[[i]], 1, 0], {i, 1, m}] / m // N
```
Out[116]=
```
0.99
```

The model is clearly a good predictor of the correct chicken under these conditions! But what if we try another sigma or another hazard rate?

The behavior of an imperfect predictor.

We have seen that this model performs well when it has the right parameters. But humans may have imperfect estimations of H. How does this affect the model?
You can play with the values of σ, hTrue, and hSubj to find out.

In[117]:=
```
Remove["Global`*"]
```

First, set the parameters.

In[118]:=
```
μ = 75;
σ = 140;
hTrue = .1;
hSubj = .3;
m = 100;
```

Then, generate some random data.

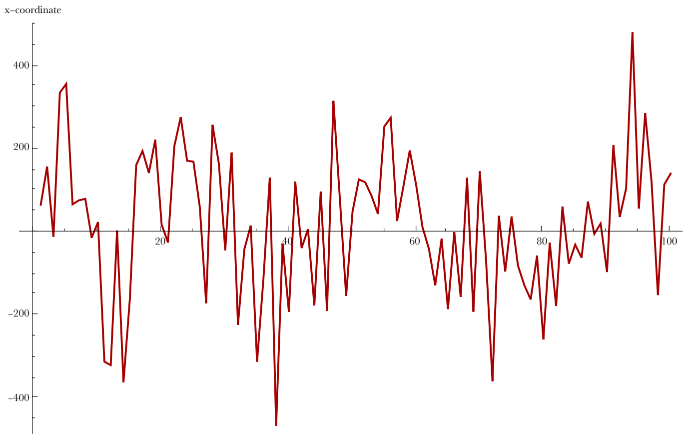
In[123]:=
```
r = RandomVariate[UniformDistribution[{0, 1}], m];
```

In[124]:=
```
switch = Table[If[r[[i]] > hTrue, 0, 1], {i, 1, m}];
dist = {};
val = 1;
For[i = 1, i ≤ m, i++,
 If[switch[[i]] == 1,
  val = If[val == 1, 2, 1]
 ];
 dist = Append[dist, val]
]
x = Table[If[dist[[i]] == 1, RandomVariate[NormalDistribution[μ, σ], 1],
    RandomVariate[NormalDistribution[-μ, σ], 1]], {i, 1, m}];
x = Transpose[x] // Flatten;
```

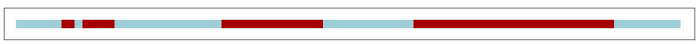
The following plot shows the x-coordinate of the egg in each trial

In[130]:=
```
ListPlot[Table[x[[i]], {i, 1, m}], Joined → True,
  AxesLabel → {"t", "x-coordinate"}, ImageSize → Large]
ArrayPlot[{dist}, ImageSize → Large]
```

Out[130]=



Out[131]=



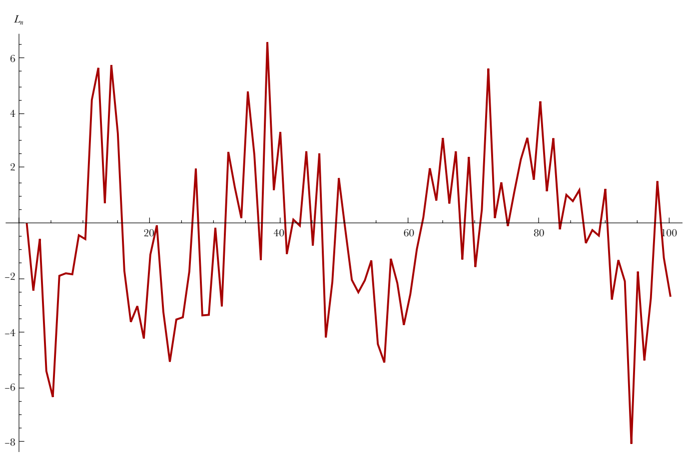
Now, based on the data above, the model will make a prediction of which chicken the egg came from.

In[132]:=
```
l[1] = 0;
For[n = 2, n ≤ m, n++,
 ψ[n] = l[n - 1] + Log[(1 - hSubj)/hSubj + Exp[-l[n - 1]]] - Log[(1 - hSubj)/hSubj + Exp[l[n - 1]]] // N;
 llr[n] = -2 * (LogLikelihood[NormalDistribution[μ, σ], {x[[n]]}] -
      LogLikelihood[NormalDistribution[-μ, σ], {x[[n]]}]) // N;
 l[n] = ψ[n] + llr[n];
]
```

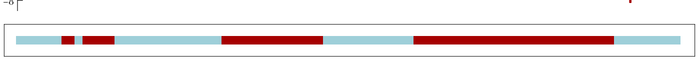
The plot below demonstrates how the belief of the model changes over time in response to changes in the environment.

In[134]:=
```
ListPlot[Table[l[i], {i, 1, m}], Joined → True, AxesLabel → {"t", "Lₙ"}, ImageSize → Large]
ArrayPlot[{dist}, ImageSize → Large]
```

Out[134]=



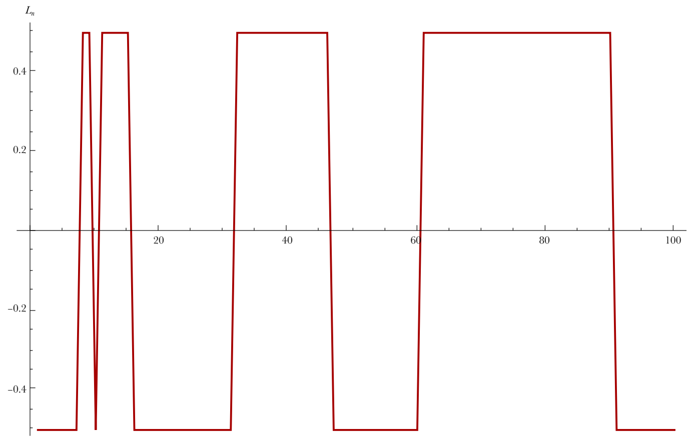
Out[135]=



It is clear to see that it matches the correct answer somewhat.

In[136]:=
```
ListPlot[dist - 1.5, Joined → True, AxesLabel → {"t", "Ln"}, ImageSize → Large]
```

Out[136]=



In[137]:=
```
pred = Table[If[l[i] > 0, 2, 1], {i, 1, m}];
```

Percent correct:

In[138]:=
```
Sum[If[dist[[i]] == pred[[i]], 1, 0], {i, 1, m}] / m // N
```

Out[138]=
```
0.83
```

Note that with a small σ, it doesn't matter how poor the prediction of H; the eggs appear so close to the chickens that H ends up having no influence on the decision. This is because LLR_n will always be larger in magnitude than ψ.

Analysis of Model

Chicken task data was collected on 41 participants. This data was pre-processed in Matlab. Here, we will explore how altering the parameters affect the model's similarity to participant response.

Loading in the Data

The processed data is in the form of a csv. We will now load this data into Matlab. Because this is only an exploration, we will focus on the results of one participant. Each participant had to perform the task under several different parameters. We will load in the results for two different sets of parameters.

In[139]:=
```
Remove["Global`*"]
```

In[140]:=
```
myData = Import["/Users/Katie/Dropbox (LIBR)/02_Projects/Chicken
    Task/Chicken_code/Data/Raw/A1KFIQBETMB81I_T1.csv"];
myData[[1 ;; 11]] // TableForm
```

Out[141]//TableForm=

egg_x	egg_y	correct_choice	H_true	sigma	prediction	reaction_time
-42	-36	1	0.05	33	1	3.153
-125	9	1	0.05	33	1	2.634
-21	-19	1	0.05	33	1	1.572
-89	-80	1	0.05	33	1	1.205
-104	42	1	0.05	33	1	1.023
-78	-22	1	0.05	33	1	1.222
54	49	2	0.05	33	2	1.514
61	-15	2	0.05	33	2	0.758
147	73	2	0.05	33	2	1.259
124	24	2	0.05	33	2	0.637

In[142]:=
```
myData2 = Import["/Users/Katie/Dropbox (LIBR)/02_Projects/Chicken
    Task/Chicken_code/Data/Raw/A1KFIQBETMB81I_T2.csv"];
myData2[[1 ;; 11]] // TableForm
```

Out[143]//TableForm=

egg_x	egg_y	correct_choice	H_true	sigma	prediction	reaction_time
-149	215	1	0.05	140	1	0.802
-258	-69	1	0.05	140	1	0.334
62	-1	1	0.05	140	2	0.142
-168	-245	1	0.05	140	1	0.56
44	5	1	0.05	140	2	1.026
-29	54	1	0.05	140	1	0.413
-8	-31	1	0.05	140	1	0.503
-314	142	1	0.05	140	1	0.37
-89	102	1	0.05	140	1	0.502
-417	-74	1	0.05	140	1	0.341

Validation of Model

Does this model accurately predict participant response?

Lets start with the first set of data, which has a small sigma value.

First, set the parameters.

In[144]:=
```
μ = 75;
σ = myData[[2, 5]];
h = myData[[2, 4]];
m = 199;
```

Then, define the x-values of the egg.

In[148]:=
```
x = myData[[2 ;; 200, 1]];
```

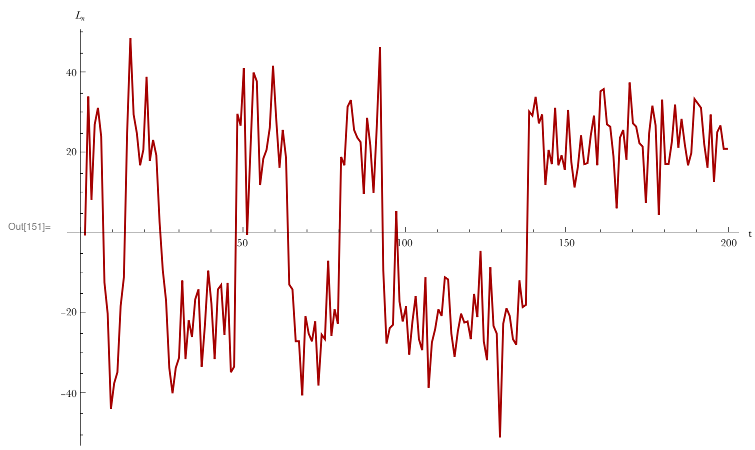
Now, based on the data above, the model will make a prediction of which chicken the egg came from.

In[149]:=
```
l[1] = 0;
For[n = 2, n ≤ m, n++,
 ψ[n] = l[n - 1] + Log[(1 - h)/h + Exp[-l[n - 1]]] - Log[(1 - h)/h + Exp[l[n - 1]]] // N;
 llr[n] = -2 * (LogLikelihood[NormalDistribution[μ, σ], {x[[n]]}] -
      LogLikelihood[NormalDistribution[-μ, σ], {x[[n]]}]) // N;
 l[n] = ψ[n] + llr[n];
]
```

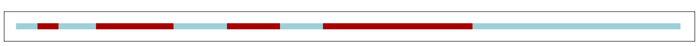
The plot below demonstrates how the belief of the model changes over time in response to changes in the environment.

In[151]:=
```
ListPlot[Table[l[i], {i, 1, m}], Joined → True, AxesLabel → {"t", "Ln"}, ImageSize → Large]
ArrayPlot[{myData[[2 ;; 200, 3]]}, ImageSize → Large]
```

Out[151]=



Out[152]=



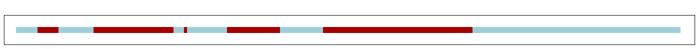
In[153]:=
```
pred = Table[If[l[i] > 0, 1, 2], {i, 1, m}];
```

Below are array plots comparing the model response, participant response, and correct response. It is clear that all are very similar.

In[154]:=
```
Print["Participant Response:"]
ArrayPlot[{myData[[2 ;; 200, 6]]}, ImageSize → Large]
Print["Model Response:"]
ArrayPlot[{pred}, ImageSize → Large]
Print["Correct Response:"]
ArrayPlot[{myData[[2 ;; 200, 3]]}, ImageSize → Large]
```

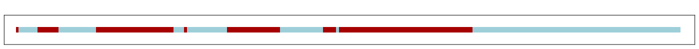
Participant Response:

Out[155]=



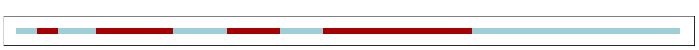
Model Response:

Out[157]=



Correct Response:

Out[159]=



How often the model matched the participant:

In[160]:=
```
Sum[If[myData[[i + 1, 6]] == pred[[i]], 1, 0], {i, 1, m}] / m // N
```
Out[160]=
```
0.984925
```

Now, lets compare the model to the same participant, but a larger sigma value,

In[161]:=
```
μ = 75;
σ = myData2[[2, 5]];
h = myData2[[2, 4]];
m = 199;
```

Then, define the x-values of the egg.

In[165]:=
```
x = myData2[[2 ;; 200, 1]];
```

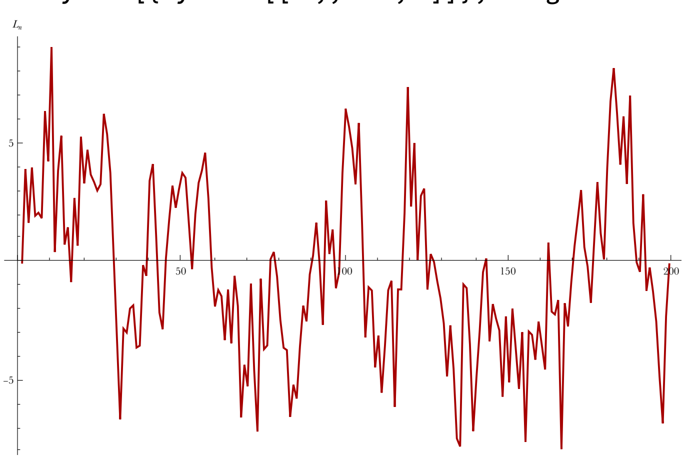
Now, based on the data above, the model will make a prediction of which chicken the egg came from.

In[166]:=
```
l[1] = 0;
For[n = 2, n ≤ m, n++,
 ψ[n] = l[n - 1] + Log[(1 - h)/h + Exp[-l[n - 1]]] - Log[(1 - h)/h + Exp[l[n - 1]]] // N;
 llr[n] = -2 * (LogLikelihood[NormalDistribution[μ, σ], {x[[n]]}] -
       LogLikelihood[NormalDistribution[-μ, σ], {x[[n]]}]) // N;
 l[n] = ψ[n] + llr[n];
]
```

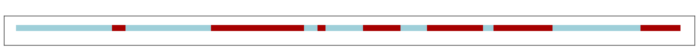
The plot below demonstrates how the belief of the model changes over time in response to changes in the environment.

In[168]:=
```
ListPlot[Table[l[i], {i, 1, m}], Joined → True, AxesLabel → {"t", "Ln"}, ImageSize → Large]
ArrayPlot[{myData2[[2 ;; 200, 3]]}, ImageSize → Large]
```

Out[168]=



Out[169]=



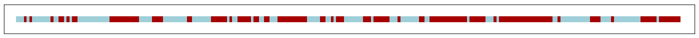
In[170]:=
```
pred = Table[If[l[i] > 0, 1, 2], {i, 1, m}];
```

Below are array plots comparing the model response, participant response, and correct response.

In[171]:=
```
Print["Participant Response:"]
ArrayPlot[{myData2[[2 ;; 200, 6]]}, ImageSize → Large]
Print["Model Response:"]
ArrayPlot[{pred}, ImageSize → Large]
Print["Correct Response:"]
ArrayPlot[{myData2[[2 ;; 200, 3]]}, ImageSize → Large]
```

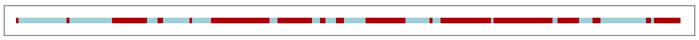
Participant Response:

Out[172]=



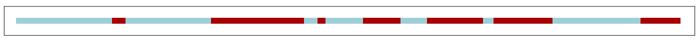
Model Response:

Out[174]=



Correct Response:

Out[176]=



How often the model matched the participant:

In[177]:=
```
Sum[If[myData2[[i + 1, 6]] == pred[[i]], 1, 0], {i, 1, m}] / m // N
```

Out[177]= 0.788945

Note that the model doesn't do too well this time. This is likely because the model has perfect knowledge of H, whereas the participant may be interpreting things as being more/-less random than they truly are. Let's try the model again with a higher H.

In[178]:=
```
h = .3;
l[1] = 0;
For[n = 2, n ≤ m, n++,
 ψ[n] = l[n - 1] + Log[(1 - h)/h + Exp[-l[n - 1]]] - Log[(1 - h)/h + Exp[l[n - 1]]] // N;
 llr[n] = -2 * (LogLikelihood[NormalDistribution[μ, σ], {x[[n]]}] -
      LogLikelihood[NormalDistribution[-μ, σ], {x[[n]]}]) // N;
 l[n] = ψ[n] + llr[n];
]
```

In[181]:=
```
pred = Table[If[l[i] > 0, 1, 2], {i, 1, m}];
```

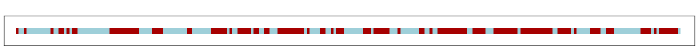
In[182]:=
```
Print["Participant Response:"]
ArrayPlot[{myData2[[2 ;; 200, 6]]}, ImageSize → Large]
Print["Model Response:"]
ArrayPlot[{pred}, ImageSize → Large]
Print["Correct Response:"]
ArrayPlot[{myData2[[2 ;; 200, 3]]}, ImageSize → Large]
```

Participant Response:

Out[183]=

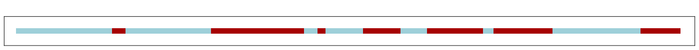
Model Response:

Out[185]=



Correct Response:

Out[187]=



How often the model matched the participant:

In[188]:=
```
Sum[If[myData2[[i + 1, 6]] == pred[[i]], 1, 0], {i, 1, m}] / m // N
```

Out[188]=
```
0.924623
```

This model works much better! This tells us that the participant is interpreting their environment to be more random than it truely is.

Discussion

The proposed model has been demonstrated here as a good model of human decision making. This supports the current theory of the 'Bayesian Brain', which proposes that humans make decisions in a Bayesian manner, weighing past experience to current knowledge.

Future exploration of this model would include many more subjects, and a larger-scale comparison of how well the model fits for a variety of parameters. Furthermore, error in predicted hazard rate could eventually be compared to mental health. Some have proposed that prediction errors could be associated with anxiety.

Nomenclature

Sensory evidence, x - the egg position

Hazard rate, H - the expected probability at each time step that the source will switch from one alternative to the other.

L_n - belief of a simulated ideal learner. The sign of L_n indicates which source is currently believed to be correct by the optimal model.

LLR_n - the sensory evidence. LLR_n is the log likelihood ratio provided by the egg position on that trial.

Ψ - the time–varying prior expectation (the logarithm of the prior odds) about the source before observing the new sensory evidence.

References

1. Glaze CM, Kable JW, Gold JI. Normative evidence accumulation in unpredictable environments. Elife. 2015;4:e08825.
2. Glaze CM, Filipowicz ALS, Kable JW, Balasubramanian V, Gold JI. A bias-variance trade-off governs individual differences in on-line learning in an unpredictable environment. Nature Human Behaviour. 2018;2(3):213-24.