

Parameter Identification and Objective Functions

Q550: Models in Cognitive Science



Qualitative vs. Quantitative

- Decay example: qualitative comparison (H_0)
 - this rarely happens, and we rely on a quantitative comparison of models: which best fits data?
- Quant predictions must be evaluated based on *optimal set of parameters*. Otherwise, we could reject a good model simple b/c we selected poor parameters
- Comparison should not be dependent on researcher's arbitrary selection of parameters (compare to regression)

Quantitative Comparisons Require:

1. *Reliable empirical data*
2. *Objective function of (mis)fit*
3. *Optimal parameters*
4. *Quantitative comparison technique*

Types of Parameters:

- **Task Parameters**
 - Fixed; set from values used in experiment
- **Assumptions**
 - Randomized or sampled; representation, environmental structure, etc.
- **Cognitive Parameters**
 - Free (estimated from data); Representative of cognitive process--criterion, attention weight, process rates, etc.
- **Scaling Parameters**
 - Free; Undesirable parameters that aspire to become cognitive process parameters, but we don't know what process or why

Outline:

- *Simple process model and experimental data*
- *Identify, fix, and free parameters*
- *Objective functions*
- *Hand fitting, grid search, optimization algorithms*

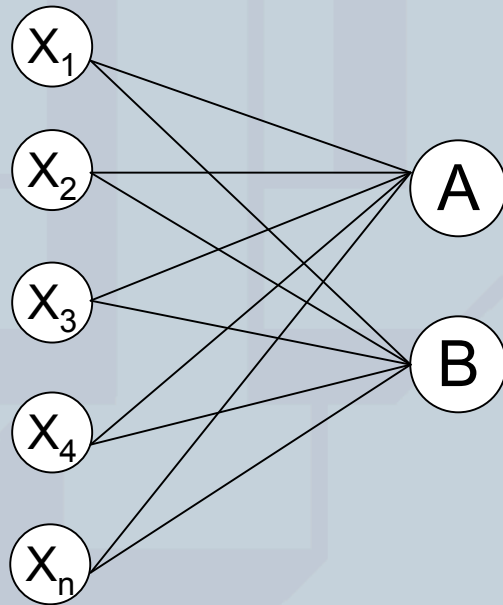
Generate data from model w/ given parameters and then try to reconstruct the generating parms from the data

Classification/Retention Model

- **Ss are trained to classify MDS stimuli (mushrooms, fish, etc.) into two categories**
 - Gaussian representation--not fully linearly separable (depends on σ^2), but we'll use a simple SLP neural net and delta learning rule
- **2 Groups: Sober/Drunk**
 - Learn in state to asymptotic performance
 - Generalization stimuli
 - Test retention of learning over 0-10 week delay
 - We get a group x delay interaction on performance

Our SLP Classification Model:

- Input Gaussian vectors
- Two output nodes; delta update on connection weights
- Luce ratio choice rule determines p-correct given activation-- sensitivity parameter b (cf. d')



$$o_i = \sum_{j=1}^{N_x} x_j w_{i,j}$$

$$\Delta w_{ij} = \alpha(t_i - o_i)x_j$$

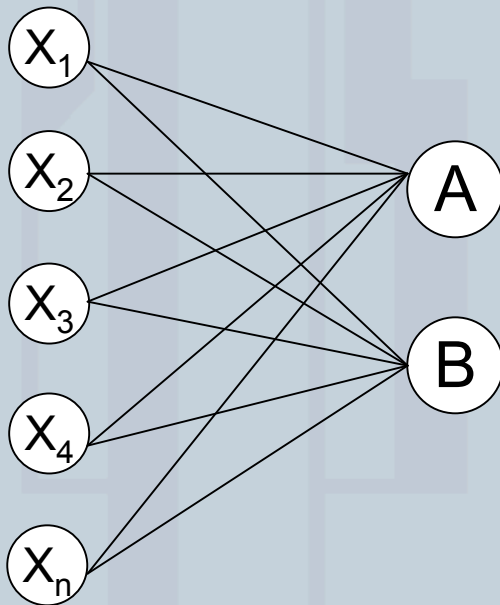
See Classify.f95

Our SLP Classification Model:

- More realistic decision rule: The probability of choosing category A is based on a ratio of strength of the output activations
- The ratio rule (from Luce, 1959):

$$p(A | x) = \frac{e^{bo_A}}{e^{bo_A} + e^{bo_B}}$$

Our SLP Classification Model:



$$o_i = \sum_{j=1}^{N_x} x_j w_{i,j}$$

$$p(A | x) = \frac{e^{bo_A}}{e^{bo_A} + e^{bo_B}}$$

$$\Delta w_{ij} = \alpha(t_i - o_i)x_j$$

See Classify.f95

Pseudocode:

Constants: N_In = 100 ! # input nodes
N_Out = 2 ! # output nodes
Alpha = 0.01 ! Learning rate parameter

Data Structures: Prototype[N_Out, N_In]
Exemplar[N_In]
Weight[N_In, N_Out]
Output[N_Out]

Tools: **Distort**(Vector, D, μ , σ)
Random_Vector(D)
Classify(Exemplar, Weight, Output)

$$o_i = \sum_{j=1}^{N_x} x_j w_{i,j} \quad p(A|x) = \frac{e^{bo_A}}{e^{bo_A} + e^{bo_B}}$$

Update_Weights(Weight, Error, Exemplar, Alpha)

$$w_{ij} = w_{ij} + \alpha(t_i - o_i)x_j$$

Pseudocode:

Prototype[1, :] = **Random_Vector**(N_In)

Prototype[2, :] = **Random_Vector**(N_In)

FOR $i = 1$ to N_Train **do**

! Flip a coin to pick a category (result=x):

Exemplar = **Distort**(Prototype[x, :], 10, 0, 1) *! using z-dist*

Classify(Exemplar, Weight, Output) $\longrightarrow o_i = \sum_{j=1}^{Nx} x_j w_{i,j}$

Error[1] = true_val - Output[1]

Error[2] = true_val - Output[2]

Update_Weights(Weight, Error, Exemplar, Alpha) $w_{ij} = w_{ij} + \alpha(t_i - o_i)x_j$

ENDDO

! Weights are now clamped:

FOR $i = 1$ to N_Test **do**

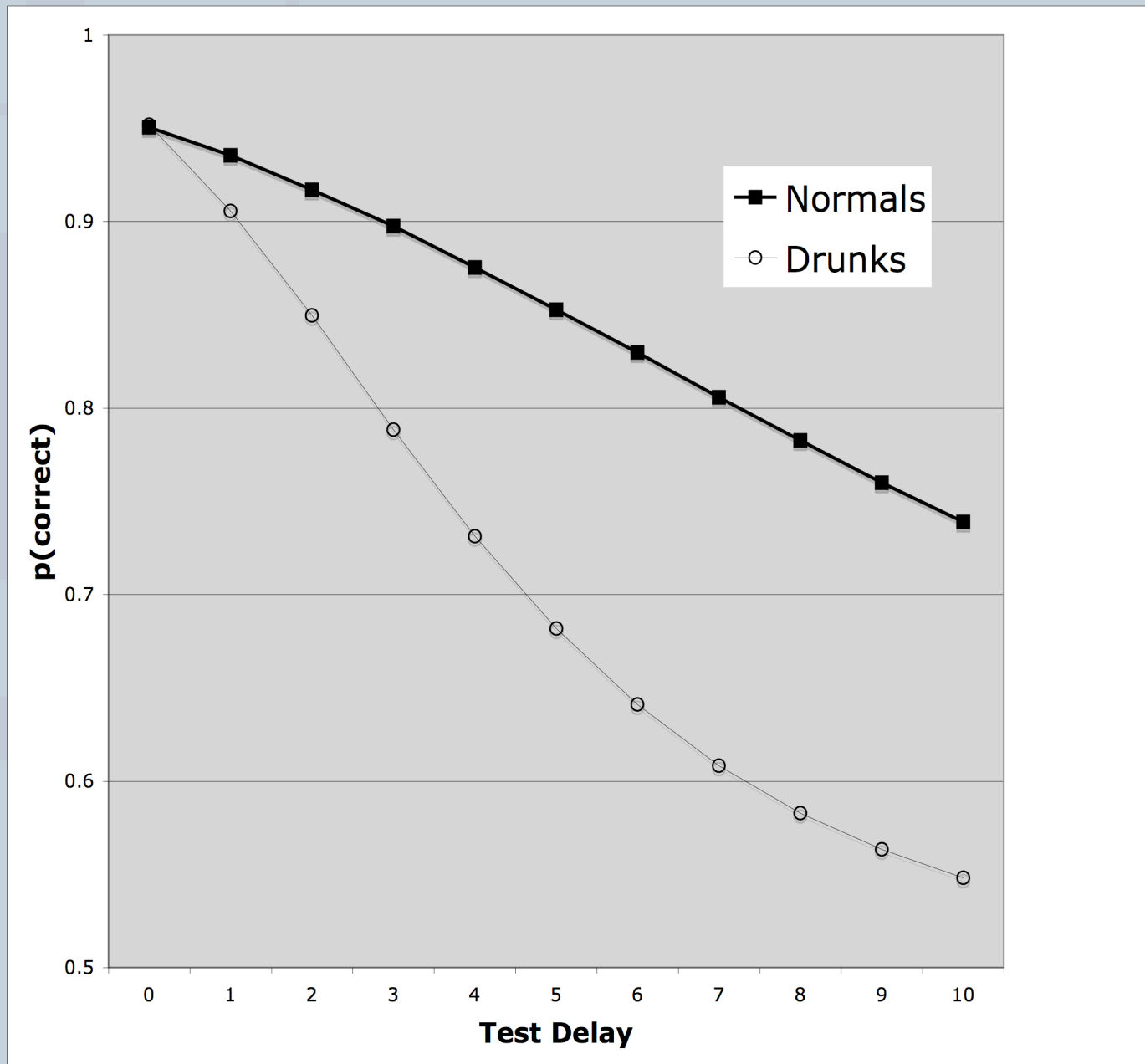
! Flip a coin to pick a category (result=x):

Exemplar = **Distort**(Prototype[x, :], 10, 0, 1) *! using z-dist*

Classify(Exemplar, Weight, Output) $\longrightarrow p(A|x) = \frac{e^{bo_A}}{e^{bo_A} + e^{bo_B}}$

ENDDO

Data:



Classification+Retention Model

- (Classification?)
- Simple hypothesis: Connection weights decay back to zero as a function of time
- Weights immediately after learning: w_{ij}
- After a delay of t weeks, the connection weights are:

$$w_{ij}(t) = \gamma^t w_{ij}$$

- Since $o_i = \sum x_j w_{ij}$, $o_i(t) = \sum x_j w_{ij} \gamma^t = o_i \gamma^t$

- So the ratio rule becomes: $p[A | x(t)] = \frac{e^{bo_A \gamma^t}}{e^{bo_A \gamma^t} + e^{bo_B \gamma^t}}$

Classification Model

$$p[A | x(t)] = \frac{e^{bo_A \gamma^t}}{e^{bo_A \gamma^t} + e^{bo_B \gamma^t}}$$

- We could rely on the equation or, if we're unsure, actually implement the decay in the model:
 1. **Train the classification to asymptotic performance, matching the learning rate of Ss**
 2. **At each time delay,**
 3. **Test on generalization stimuli**

Do this while varying the decay and sensitivity parameters. What are the likely parameter values for each group to produce the data if our assumptions are correct.

We no longer care about alpha at test (served purpose)

See Retention.f95, and Parameter_Space.f95 for fits to data

Alternate Models

- The ***null model***:
 - One free parameter: the mean
 - Assumes $p(\text{correct})$ is constant across time delays (no effect)
 - Simplest possible model, and provides lower bound for measuring model fit

- The ***saturated model***:
 - Free parameters = data points
 - New free parameter for each data point perfectly reproduces the data, and sets an upper bound for model fit

How far above the null model and below the saturated model is our retention model? ...need fit measure

- The improvement in fit of our cognitive model over the null indicates the amount of the treatment effect predicted by the cognitive model
- The improvement of the saturated model over the cognitive model indicates the amount of the treatment effect left unexplained by the cognitive model

To determine the performance of our model, we need:

- 1. Observed data**
- 2. A cognitive model w/ identifiable parameters**
- 3. Statistical upper and lower bound models (null and saturated)**
- 4. An objective function of (mis)fit**
- 5. A parameter optimization algorithm**

Objective Functions:

Objective functions map parameters onto fit indices: for each combination of parameter values, the predictions are computed, and the fit to the data is measured.

1. Least-squares objective (SSE)
2. Weighted least-squares (WSSE)
3. Log-likelihood (G^2)

$$b = 3.0$$

$$\gamma = 0.7$$

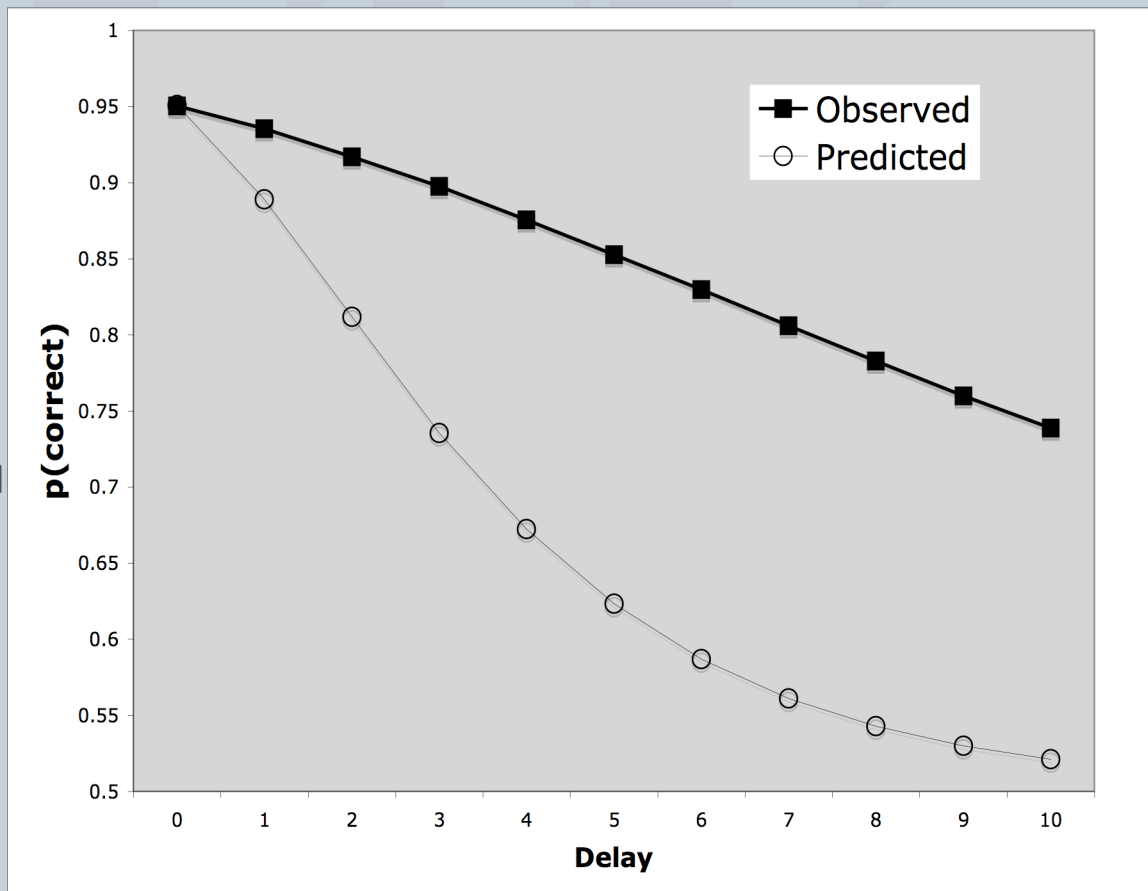
Delay	Observed	Predicted
0	0.9505	0.9511
1	0.9354	0.8889
2	0.917	0.8117
3	0.8976	0.7356
4	0.8754	0.6722
5	0.8526	0.6233
6	0.8298	0.5868
7	0.8058	0.5611
8	0.7826	0.5429
9	0.7599	0.53
10	0.7388	0.5211

Least Squares:

Sum of squared residuals (SSE)

$$SSE = \text{sum}[(\text{Obs}-\text{Pred})^{**2}]$$

$$SSE = \sum [(p_t - P(t))^2]$$



Delay	Observed	Predicted
0	0.9505	0.9511
1	0.9354	0.8889
2	0.917	0.8117
3	0.8976	0.7356
4	0.8754	0.6722
5	0.8526	0.6233
6	0.8298	0.5868
7	0.8058	0.5611
8	0.7826	0.5429
9	0.7599	0.53
10	0.7388	0.5211

Least Squares:

Sum of squared residuals (SSE)

$$SSE = \text{sum}[(\text{Obs}-\text{Pred})^{**2}]$$

$$SSE = \sum [(p_t - P(t))^2]$$

$$b = 3.0$$

$$\gamma = 0.7$$

Delay	Observed	Predicted	sqr(O-P)
0	0.9505	0.9511	3.6E-07
1	0.9354	0.8889	0.00216225
2	0.917	0.8117	0.01108809
3	0.8976	0.7356	0.026244
4	0.8754	0.6722	0.04129024
5	0.8526	0.6233	0.05257849
6	0.8298	0.5868	0.059049
7	0.8058	0.5611	0.05987809
8	0.7826	0.5429	0.05745609
9	0.7599	0.53	0.05285401
10	0.7388	0.5211	0.04739329
		SUM:	0.40999391

Least Squares

One problem with the SSE is that it penalizes all errors the same. Some errors may be more serious depending on the precision of estimation of p_t (n_{ct}/n)

In our retention model (if true), the variance in p_t will increase as a function of t

Errors at longer delays should be given less weight b/c they are more likely to be the result of estimation error. Failing to take variance into account makes SSE an inefficient estimator of prediction error.

Weighted Least Squares Objective

Weighted least squares (WSSE) is also computed between squared deviations between obs and pred, but these deviations are weighted by the inverse of the variance of the prediction.

$$WSSE = \sum \left(\frac{p_t - P(t)}{\sigma^2} \right)^2 = \sum \frac{1}{\sigma_t^2} [p_t - P(t)]^2$$

In this case, we know the variance at time t is $\sigma_t^2 = \frac{p_t q_t}{n}$

$$WSSE = \sum \left(\frac{p_t - P(t)}{\sigma^2} \right)^2 = \sum \frac{1}{\sigma_t^2} [p_t - P(t)]^2$$

$b = 3.0$

$\gamma = 0.7$

Delay	Observed	Predicted	Var	W(Obs-Pred)
0	0.9505	0.9511	0.00465088	0.01664303
1	0.9354	0.8889	0.00987568	22.1703207
2	0.917	0.8117	0.01528431	47.4640738
3	0.8976	0.7356	0.01944926	69.3783052
4	0.8754	0.6722	0.02203472	85.0418095
5	0.8526	0.6233	0.02347971	95.3723788
6	0.8298	0.5868	0.02424658	100.441158
7	0.8058	0.5611	0.02462668	98.7316147
8	0.7826	0.5429	0.02481596	93.2983455
9	0.7599	0.53	0.02491	85.178598
10	0.7388	0.5211	0.02495548	76.1000667
WSSE:				773.193314

Likelihood Objective:

- Compute the likelihood that a model would have generated the observed data, given the parameter set
- We don't need to estimate the μ or σ , lets just list each trial with the observed and predicted values

Trial	Delay	Response	Prediction
1	0	1	0.9511
2	0	0	0.0489
3	0	1	0.9511
4	0	1	0.0489
5	0	1	0.9511
6	0	0	0.0489
.			
.			
310	2	1	0.8177
311	2	0	0.1823
312	2	0	0.1823
.			
.			
.			
N	10	0	0.4789

$$L_R = \prod_{i=1}^N \text{Pred}_i$$

- Obviously, $L_R = \prod_{i=1}^N \text{Pred}_i$ often turns out to be a very small number, so it is more convenient to work with the natural logarithm of the product...hence the name:

Log Likelihood

- The log of a product = sum of logs, so we can just log each predicted value and then sum the $\ln(\text{pred})$:

Trial	Delay	Response	Prediction	Ln(Pred)
1	0	1	0.9511	-0.0501361
2	0	0	0.0489	-3.0179779
3	0	1	0.9511	-0.0501361
4	0	1	0.0489	-3.0179779
5	0	1	0.9511	-0.0501361
6	0	0	0.0489	-3.0179779
.				
.				
310	2	1	0.8177	-0.2012598
311	2	0	0.1823	-1.7021016
312	2	0	0.1823	-1.7021016
.				
.				
.				
N	10	0	0.4789	-0.7362635

Sum[Ln(Pred)]: -13.546068

- Simplify by counting n-correct and n-incorrect for each condition:

$$\ln(L_R) = \sum n_{1,t} \cdot \ln[P(t)] + n_{0,t} \cdot \ln[Q(t)] = -13.54$$

- So is -13.54 good? That all depends on how good a model could do: enter the saturated model (Observed relative frequencies are used as the prediction for each delay condition:

$$\ln(L_S) = \sum n_{1,t} \cdot \ln[p_d] + n_{0,t} \cdot \ln[q_d]$$

Trial	Delay	Response	Prediction	Obs-Freq	LnSat
1	0	1	0.9511	0.9505	-0.0507671
2	0	0	0.0489	0.0495	-3.0057826
3	0	1	0.9511	0.9505	-0.0507671
4	0	1	0.0489	0.9505	-0.0507671
5	0	1	0.9511	0.9505	-0.0507671
6	0	0	0.0489	0.0495	-3.0057826
.					
.					
310	2	1	0.8177	0.917	-0.0866478
311	2	0	0.1823	0.083	-2.4889147
312	2	0	0.1823	0.083	-2.4889147
.					
.					
.					
N	10	0	0.4789	0.2612	-1.3424689
				Sum:	-12.62158

$$\ln(L_S) = \sum n_{1,t} \cdot \ln[p_d] + n_{0,t} \cdot \ln[q_d] = -12.62$$

$$\ln(L_R) = \sum n_{1,t} \cdot \ln[P(t)] + n_{0,t} \cdot \ln[Q(t)] = -13.54$$

- With maximum likelihood, we use the G^2 statistic:

$$G^2 = 2[\ln(L_S) - \ln(L_R)]$$

In this case: $G^2 = 2[-12.62 - (-13.54)] = 0.92$

This measures the lack of fit between the cognitive model and the saturated model. We will try to select parameters in our cognitive model that minimize G^2

Maximum likelihood actually means minimal G^2

Comparing Objective Functions

Least Squares

- Easy calculation, but inefficient as an estimator (need large N to be confident)
- Watch any case where your condition values do not have homogeneity of variance (probability, RT, etc.)

Weighted Least Squares

- Use for probabilities or RTs
- SSE and WSSE are equivalent for large sample sizes

Log-Likelihood

- Bernoulli process assumes independence of trials (but are they? 1/f noise and fatigue, etc.)
- Both WSSE and G^2 are distributed on χ^2 and for large sample sizes, SSE=WSSE