# Lecture 4 - Model fit diagnostics

Summary of model fitting steps:

1. collect data into a single data frame

```r
times = c(1,3,6,9,12,18) # retention intervals (in seconds)
numRecall = c(94, 77, 40, 26, 24, 16) # number of words correctly recalled
X = data.frame(times, numRecall)
```

2. build objective function for each model (negative log likelihood)

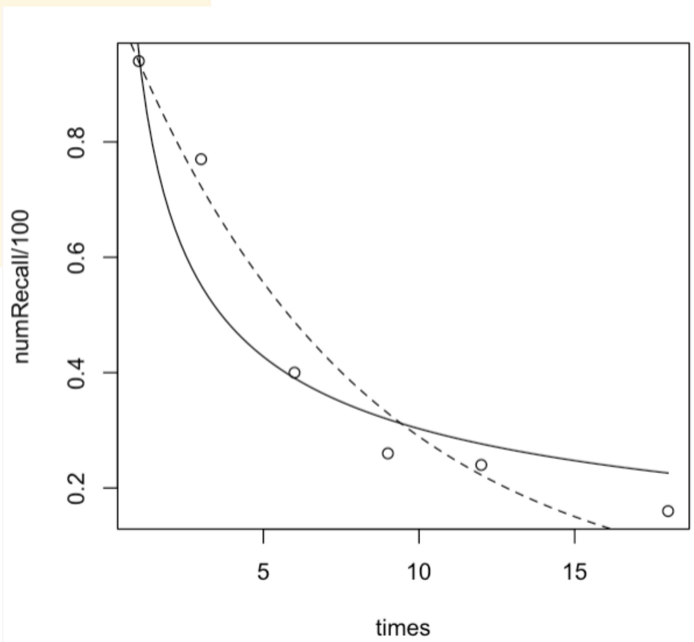
```r
nll.power = function(data, pars){
  a = pars[1]
  b = pars[2]
  t = data$times
  x = data$numRecall
  tmp1 = log(choose(100,x))
  tmp2 = x * log(a*t^b)
  tmp3 = (100-x) * log(1-a*t^b)
  return(-1*sum(tmp1 + tmp2 + tmp3))
}

nll.exp = function(data, pars){
  a = pars[1]
  b = pars[2]
  t = data$times
  x = data$numRecall
  tmp1 = log(choose(100,x))
  tmp2 = x * log(a*b^t)
  tmp3 = (100-x) * log(1-a*b^t)
  return(-1*sum(tmp1 + tmp2 + tmp3))
}
```

## 3. fit the models (using optim function)

```r
a_init = runif(1)
b_init = runif(1)
initPar = c(a_init, b_init) # collect a,b into one parameter vector

model1 = optim(par = initPar,
      fn = nll.power,
      data = X)

model2 = optim(par = initPar,
               fn = nll.exp,
               data = X)
```

## 4. plot models with data to visually assess fit

```r
# let's plot on top of original data
plot(numRecall/100 ~ times)

# first, extract parameters from power model
a = model1$par[1]
b = model1$par[2]
curve(a*x^b,
      from=0, to=18,
      add=T)

# next, extract parameters from exponential model
a = model2$par[1]
b = model2$par[2]
curve(a*b^x,
      from=0, to=18,
      add=T,
      lty=2) # this makes a dashed line
```



Lets compare what we found with Myung:



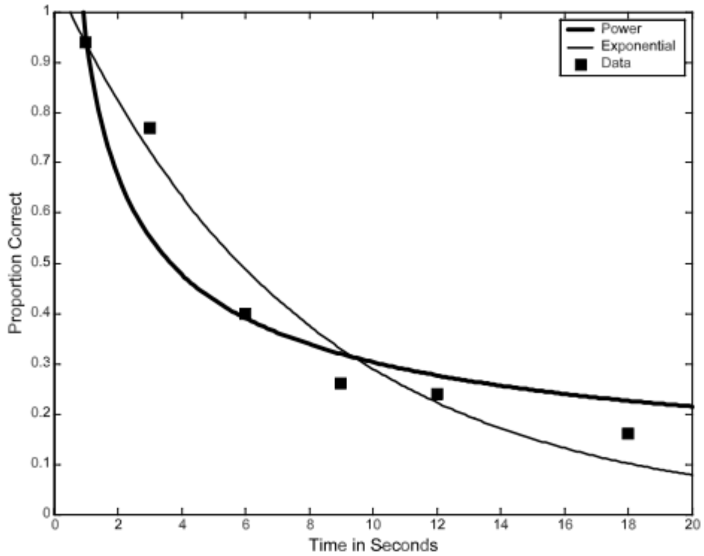



Fig. 4. Modeling forgetting data. Squares represent the data in Murdock (1961). The thick (respectively, thin) curves are best fits by the power (respectively, exponential) models.

Table 1
Summary fits of Murdock (1961) data for the power and exponential models under the maximum likelihood estimation (MLE) method and the least-squares estimation (LSE) method.

| | MLE | | LSE | |
|---|---|---|---|---|
| | Power | Exponential | Power | Exponential |
| Loglik/SSE ($r^2$) | −313.37 (0.886) | −305.31 (0.963) | 0.0540 (0.894) | 0.0169 (0.967) |
| Parameter $w_1$ | 0.953 | 1.070 | 1.003 | 1.092 |
| Parameter $w_2$ | 0.498 | 0.131 | 0.511 | 0.141 |

*Note:* For each model fitted, the first row shows the maximized log-likelihood value for MLE and the minimized sum of squares error value for LSE. Each number in the parenthesis is the proportion of variance accounted for (i.e. $r^2$) in that case. The second and third rows show MLE and LSE parameter estimates for each of $w_1$ and $w_2$. The above results were obtained using Matlab code described in the appendix.

Environment Files Plots Packages Help Viewer

Import Dataset 152 MiB List

R Global Environment

```
Data
model1            List of  5
  $ par        : num [1:2] 0.953 -0.498
  $ value      : num 26.7
  $ counts     : Named int [1:2] 53 NA
  ..- attr(*, "names")= chr [1:2] "function" "gradient"
  $ convergence: int 0
  $ message    : NULL
model2            List of  5
  $ par        : num [1:2] 1.07 0.877
  $ value      : num 18.7
  $ counts     : Named int [1:2] 65 NA
  ..- attr(*, "names")= chr [1:2] "function" "gradient"
  $ convergence: int 0
  $ message    : NULL
```

These are roughly the same, but why are they different?

Question: why did we take the log of the likelihood function?

Answer: because multiplying small numbers $< 1$ makes even smaller numbers!

Explanation: Recall that we have 6 observations

$$x = (x_1, x_2, x_3, x_4, x_5, x_6)$$

If we assume they are all independent, we mutiply the likelihoods:

$$L\left(a,b \,\middle|\, x = (x_1, x_2, x_3, x_4, x_5, x_6)\right) = \prod_{i=1}^{6} L\left(a,b \,\middle|\, x_i\right)$$

Each of the individual likelihoods $L(a,b \mid x_i)$ is a small number $(\ll 1)$, so the product approaches 0 exponentially fast. Computers cannot handle numbers that are too small (or too big).

Trick: convert multiplication to addition with the logarithm:

$$\log L(a,b \mid x_1, x_2, x_3, x_4, x_5, x_6) = \log \left[ \prod_{i=1}^{6} L(a,b \mid x_i) \right]$$

$$= \sum_{i=1}^{6} \log L(a,b \mid x_i)$$

Now, when we add numbers (regardless of their size), the sum always has a bigger magnitude.

Computers can handle this.

Today's topic: how do we assess the fit of our models?

Use information criteria:

* AIC - Akaike Information Criterion, or
* BIC - Bayesian Information Criterion

Basic idea - representing data with a model results in information loss. The best model is the one which loses the least information

Workflow:

1. calculate AIC (or BIC) for each model
2. model with smallest value is best fit.

least "information loss"

## Definitions

Let $k$ = # parameters in model

$\hat{L}$ = value of likelihood function at MLE.

Then:

$$AIC = 2k - 2\log\hat{L}$$

$$BIC = k\log(N) - 2\log\hat{L}$$

$N$ = # observations

## Computing AIC / BIC

Model 1:

$k = 2$ parameters

$N = 6$ observations

$\log\hat{L} = -26.7$

$$AIC_1 = 2k - 2\log(\hat{L})$$
$$= 2(2) - 2(-26.7)$$
$$= 57.4$$

Model 2:

$k = 2$ parameters

$N = 6$ observations

$\log\hat{L} = -18.7$

$$AIC_2 = 2k - 2\log(\hat{L})$$
$$= 2(2) - 2(-18.7)$$
$$= 41.4$$

## BIC is similar:

$$BIC_1 = k \log N - 2 \log(\hat{L})$$
$$= 2 \log(6) - 2(-26.7)$$
$$= 56.98$$

$$BIC_2 = k \log N - 2 \log(\hat{L})$$
$$= 2 \log(6) - 2(-18.7)$$
$$= 40.98$$

## Which one to use?

* matter of choice.
* AIC easy to use, but BIC better accounts for model complexity (it more severely penalizes more complex models)
* BIC gives us a Bayes factor:

$$BF_{21} = \frac{p(\text{data} \mid m_2)}{p(\text{data} \mid m_1)} \approx \exp\left(\frac{BIC_1 - BIC_2}{2}\right)$$

For our models, this gives:

$$BF_{21} \approx \exp\left(\frac{BIC_1 - BIC_2}{2}\right)$$

$$= \exp\left(\frac{56.98 - 40.98}{2}\right)$$

$$= \exp\left(\frac{16}{2}\right)$$

$$= \exp(8)$$

$$\approx 2981.$$

Interpretation: the observed data are approximately 3000 times more likely under Model 2 (the exponential model) than under Model 1 (the power model).