Statistics with R

Logistic Regression

Zhuanghua Shi (Strongway)

18 June 2018

Logisitic regression

- One of more independent variables X
- A binary output Y

Instead of model Y directly, we model the log odds of the event

$$Z = \ln \frac{P}{1-P}$$

where P is the probability of event. We want to estimate the relationship:

$$Z = b_0 + b_1 X + \epsilon$$

The above equation can be modeled using the `glm()` function by setting `family` to `binomial`.

An example

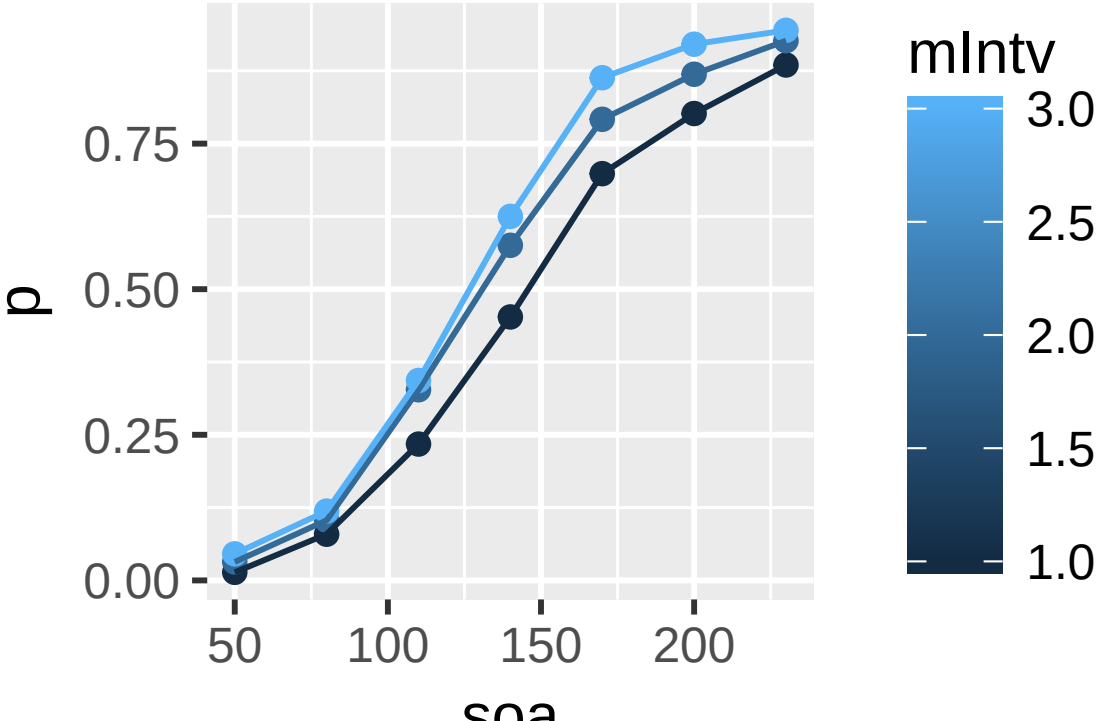
- Ternus apparent motion
  - typical two types of motion percepts - Group vs. Element motion
- The motion percept is mainly determined by the inter-stimulus interval (ISI)
  - https://en.wikipedia.org/wiki/Ternus_illusion
- Chen et al. conducted one experiment on how auditory trains influence on visual apparent motion
  - source available at github: https://github.com/msenselab/temporal_averaging

Visualize the data

First, let's visualize the data

```
dat = read.csv('https://raw.githubusercontent.com/msenselab

dat %>% group_by(mIntv, soa) %>%
  summarise(p = mean(resp)) %>%
  ggplot(aes(soa, p, color = mIntv, group = mIntv)) +
    geom_point() + geom_line()
```



Build logistic model

- Logistic regression uses function `glm()`
  - `family` objects provide a convenenient way to specify the details of the models: binomial, gaussian, Gamma, inverse.gaussian, poisson, quasibinomial et.
  - Within the `family`, you can link a specific function, e.g.,

binomial(link = 'logit')binomial(link = 'probit')

```
dat %>% filter(sub == 1, mIntv ==1) %>%
  group_by(soa) %>% summarise(p = mean(resp)) ->sub11
gsub1= glm(p ~ soa, family = binomial(link = 'logit'), data
kable(tidy(gsub1))
```

term	estimate	std.error	statistic	p.value
(Intercept)	-6.612804	4.9781960	-1.328354	0.1840613
soa	0.035382	0.0274964	1.286784	0.1981694
# fit from raw	data			

Interpreting the parameters

$$\ln \frac{P}{1-P} = b_0 + b_1 \cdot X$$

- When $P = 0.5$, we have $X_{0.5} = -b_0/b_1$ as the threshold (PSE).
- When $P = 0.75$, $X_{0.75} = (ln3 - b_0)/b_1$. So we can obtain the JND: $JND = ln3/b_1$

Build a function for tidyverse

- Using `glm()` build a function
  - doing logistic regression
  - estimate parameters
  - calculate PSE and JND

```
fitPsy <- function(df){
  glm(resp ~ soa, family = binomial(link = 'logit'), data =
    tidy(.) %>% select(term, estimate) %>%
    spread(term, estimate) %>%
    rename(b1 = soa, b0 = `(Intercept)`) %>%
    mutate(pse = -b0/b1, jnd = log(3)/b1)
}
```

Build a pipe for analysis

- nest each subject, each condition
- estimate logistic function
- return the paramters

```
dat %>% group_by(sub, mIntv) %>%
  nest() %>%
  mutate(est = map(data, fitPsy)) %>%
  unnest(est, .drop = TRUE) -> thresholds
```

Repeated-measures ANOVA

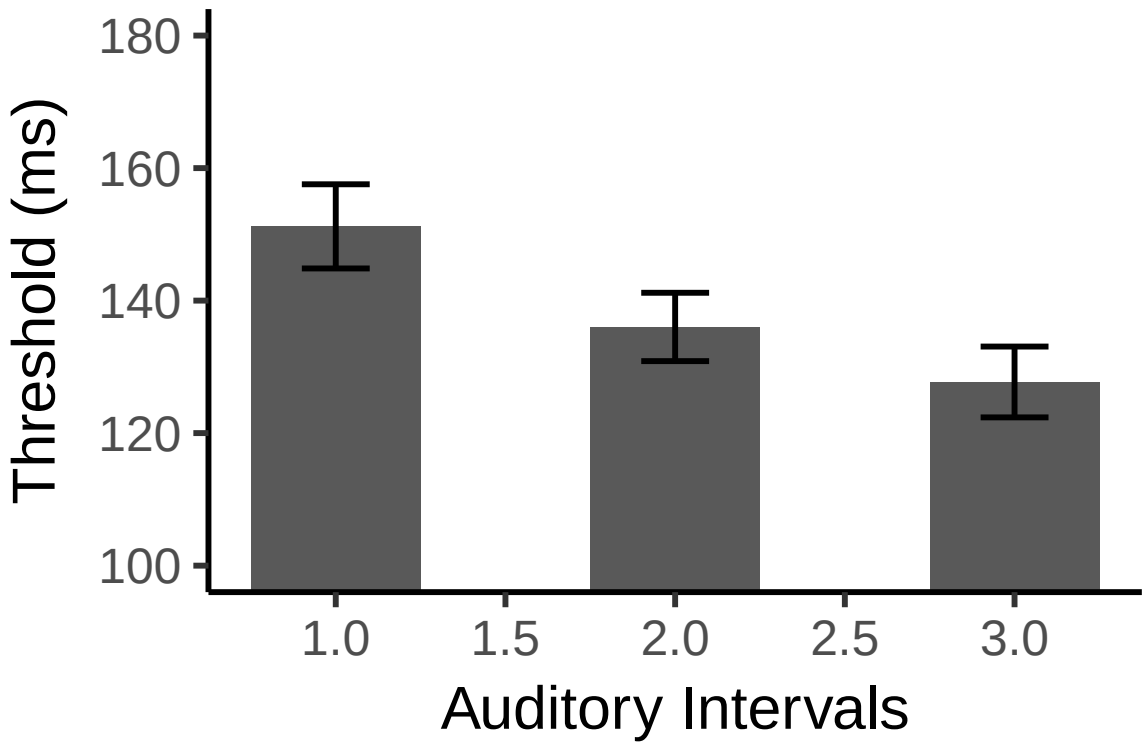
- Test if the auditory train has any impact on visual motion percept.

```
av1 = ezANOVA(data = thresholds,
         dv = pse,
         wid = sub,
         within = mIntv)
kable(av1$ANOVA)
```

Effect	DFn	DFd	F	p	p<.05	ges
mIntv	1	20	21.81965	0.0001469	*	0.5217559

Visualize the thresholds

```
thresholds %>% group_by(mIntv) %>%
  summarise(mpse = mean(pse), n = n(),
            se = sd(pse)/sqrt(n-1)) %>%
  ggplot(aes(mIntv, mpse)) + geom_bar(stat = 'identity', wi
  geom_errorbar(aes(ymin = mpse - se, ymax = mpse+se), widt
  coord_cartesian(ylim = c(100, 180)) + theme_classic() +
  xlab('Auditory Intervals') + ylab('Threshold (ms)')
```



Second approach with `quickpsy`

- `quickpsy` by Daniel Linares
  - http://dlinares.org/quickpsy.html

A package for quickly fitting and plotting psychometric functions.

- Fits and plots multiple conditions
- Calculates parametric and non-parametric bootstrap confidence intervals
- Guess and lapses can be fixed or free as parameters
- Performs goodness-of-fit
- compute AIC

QuickPsy

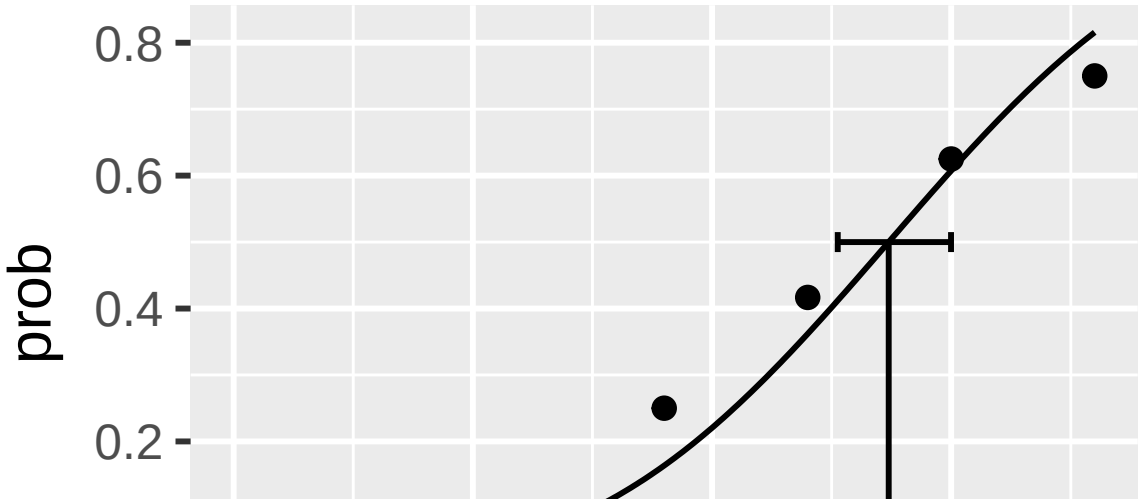
- QuickPsy uses a general psychometric function

$$\Psi(x) = \gamma + (1 - \gamma - \lambda) f(x)$$

where γ is the guess rate, λ is the lapse rate, and $f(.)$ is a sigmoidal-shape function.

```
library(quickpsy)

pest = quickpsy(sub1, x = soa, k = resp)
plotcurves(pest)
```



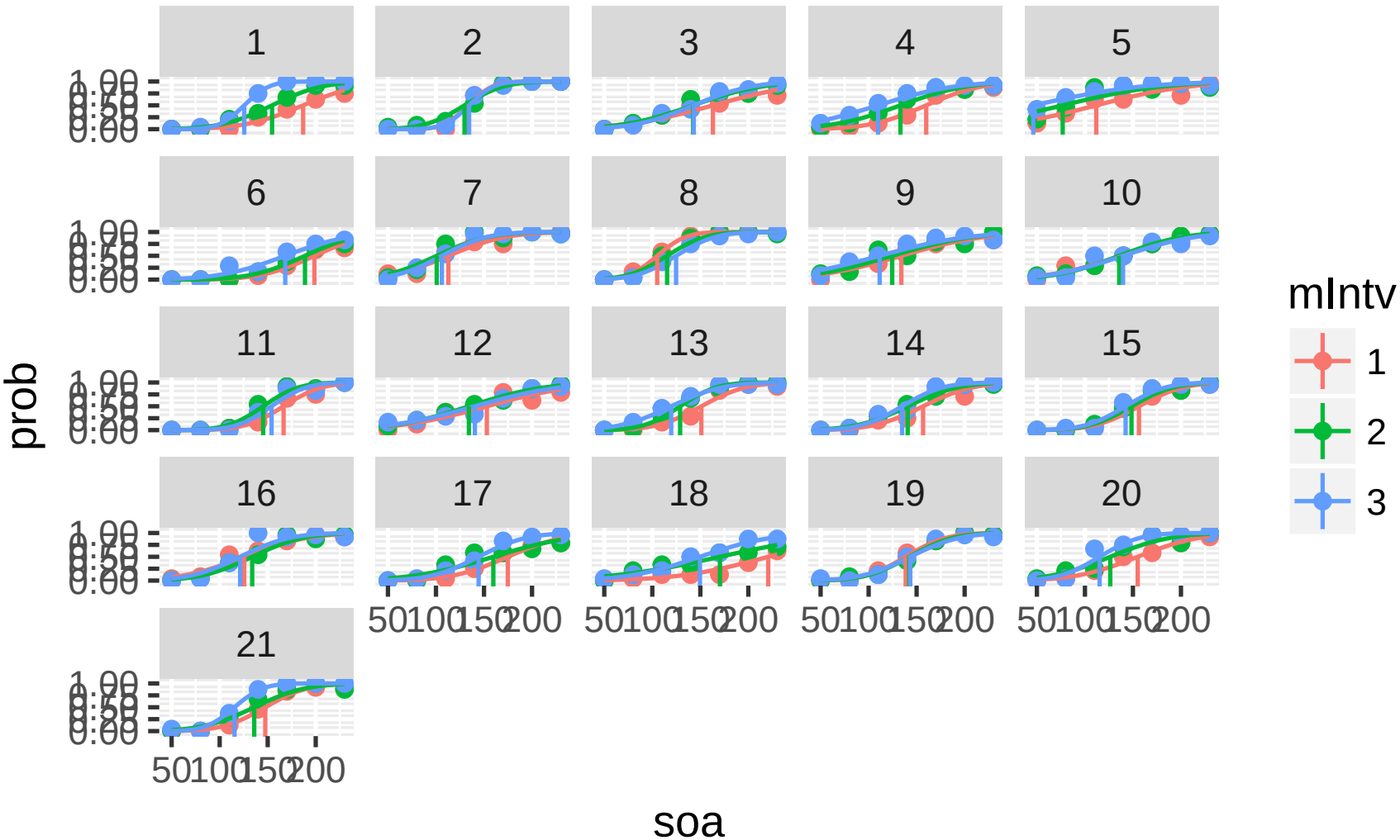
Paramters of quickpsy

- grouping: name of the grouping variables, accept multiple conditions
- within: name of the within-factor variable
- between: name of the between variable
- fun: name of the shape of the curve. (cum_normal_fun, logistic_fun, weibull_fun)

```
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                grouping = .(Direction, WaveForm, TempFreq)
```

Using quickpsy with tidyverse

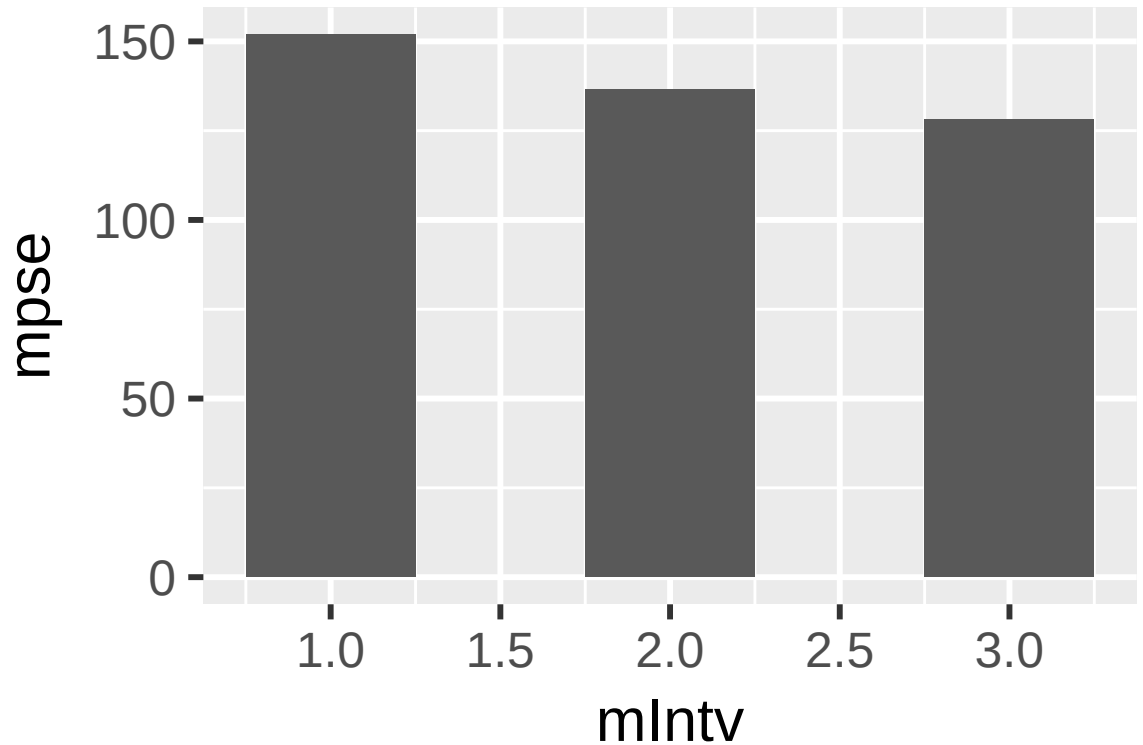
```
quickpsy(dat, soa, resp, grouping = .(sub),
         within = .(mIntv), bootstrap = 'none') -> pses
plotcurves(pses)
```



Visualize the thresholds

- output of the quickpsy has multiple tables
  - .$thresholds contain the estimated thresholds

```
pses$thresholds %>% group_by(mIntv) %>%
  summarise(mpse = mean(thre)) %>%
  ggplot(aes(mIntv, mpse)) + geom_bar(stat = 'identity', wi
```



Let's practice

- Next two weeks - Jan Nasemann will provide related tutorials