quickpsy: An R Package to Fit Psychometric Functions for Multiple Groups

by Daniel Linares and Joan López-Moliner

Abstract **quickpsy** is a package to parametrically fit psychometric functions. In comparison with previous R packages, **quickpsy** was build to easily fit and plot data for multiple groups. Here, we describe the standard parametric model used to fit psychometric functions and the standard estimation of its parameters using maximum likelihood. We also provide examples of usage of **quickpsy**, including how allowing the lapse rate to vary can sometimes eliminate the bias in parameter estimation, but not in general. Finally, we describe some implementation details, such as how to avoid the problems associated to round-off errors in the maximisation of the likelihood or the use of closures and non-standard evaluation functions.

Introduction

Statistical model

The response of humans, other animals and neurons in a classification task with a binary response variable and a stimulus level as explanatory variable is often binomially modelled as (Watson, 1979; O'Regan and Humbert, 1989; Klein, 2001; Wichmann and Hill, 2001a; Macmillan and Creelman, 2004; Gold and Shadlen, 2007; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012; Lu and Dosher, 2013; Gold and Ding, 2013)

$$f(\mathbf{k};\boldsymbol{\theta}) = \prod_{i=1}^{M} \binom{n_i}{k_i} \psi(x_i;\boldsymbol{\theta})^{k_i} \left(1-\psi\left(x_i;\boldsymbol{\theta}\right)\right)^{n_i-k_i}, \tag{1}$$

where

- f is the probability mass function of the model or the likelihood when considered as a function of the parameters;
- M is the number of stimulus levels used in the classification task;
- x_i is the i*th* stimulus level;
- n_i is the number of times that x_i is presented;
- $\mathbf{k} = (k_1, k_2, \ldots, k_M)$ is the vector of responses with k_i being the number of *Yes-type* (or correct) responses when x_i is presented;
- $\psi(x_i;\boldsymbol{\theta})$ is the probability of responding *Yes* when x_i is presented; it is called the *psychometric function* and has the form

$$\psi(x;\boldsymbol{\theta}) = \psi(x;\alpha,\beta,\gamma,\lambda) = \gamma + (1-\gamma-\lambda)F(x;\alpha,\beta), \tag{2}$$

  where

  - $\boldsymbol{\theta} = (\alpha,\beta,\gamma,\lambda)$ is the vector of parameters that define the parametric family of probability mass functions of the model. α and β are the position and scale parameters. γ and λ are the parameters corresponding to the leftward and rightward asymptote of ψ.
  - F is a function with leftward asymptote 0 and rightward asymptote 1—typically a cumulative probability function with a sigmoidal shape such as the cumulative normal, logistic or Weibull functions.

The model assumes that a given classification response does not depend on previous classifications. This is a idealisation, given the known order effects such as adaptation, fatigue, learning or serial dependence (Kingdom and Prins, 2009; Fründ et al., 2011; Van der Burg et al., 2013; Fischer and Whitney, 2014; Summerfield and Tsetsos, 2015).

Examples

Light detection. To measure the ability of an observer to detect light, a dim flash of light selected at random from 5 different light intensities (x_i; $i = 1 \ldots M$ with $M = 5$) is presented and the observer is

asked to report *Yes* if she has seen it and *No* otherwise. After her response, another intensity, selected at random from the 4 intensities that have not been presented yet, is presented and the observer classifies it again as seen or not seen. Then, the observer performs 3 more classifications until the 5 intensities have been presented. After that, the whole procedure is repeated 20 times using a new random sequence of 5 intensities each time. Using this procedure, which is called the method of constant stimuli (Green and Swets, 1966; Gescheider, 1997; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012), the observer will perform a total of 100 classifications with $n_i = 20$ for all i. k_i will correspond to the number of times that the observer responds *Yes* for each intensity. Because it is expected that, for very low intensities, the observer will never respond *Yes* and for very high intensities the observer will always respond *Yes*, γ and λ are often fixed to 0.

Criterion-independent light detection. In the previous procedure, $\mathbf{k}$ depends on how confident the observer needs to feel to give a *Yes* response—conservative observers will repond *Yes* less often (Green and Swets, 1966; Macmillan and Creelman, 2004; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012; Lu and Dosher, 2013). To avoid criterion-dependent responses, 2-intervals forced choice procedures are often used (Green and Swets, 1966; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012; Lu and Dosher, 2013). These procedures are similar to the criterion-dependent procedure described above, but the stimulus is presented at random in one interval from two intervals presented consecutively (marked with a sound, for example) and the observer needs to decide whether the stimulus was presented in the first or the second interval. Because it is expected that for very low intensities the observer will respond at chance, γ is fixed at $1/2$ (λ is usually fixed to 0). More generally, γ is fixed to $1/m$ when the observer needs to decide in which over m intervals the stimulus was presented.

Light detection with lapses. Sometimes, the observer will miss the flash (because of a blink, for example) or will make an error reporting the response (pressing the wrong response button, for example). To account for these response *lapses*, λ, which corresponds to $(1-\gamma)$ times the lapse rate (Kingdom and Prins, 2009), is not fixed but estimated as a parameter (Wichmann and Hill, 2001a,b).

Point estimation and confidence intervals

The *point estimation* of the parameters $\boldsymbol{\theta}$ of the model in (1), $\hat{\boldsymbol{\theta}}$, is sometimes obtained using Bayesian methods (Kuss et al., 2005; Kingdom and Prins, 2009), but more often using maximum likelihood ML (Watson, 1979; O'Regan and Humbert, 1989; Wichmann and Hill, 2001a; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012; Lu and Dosher, 2013) . $\hat{\boldsymbol{\theta}}$ is defined as the value of $\boldsymbol{\theta}$ that maximimises the likelihood L defined as $L(\boldsymbol{\theta}) = f(\mathbf{k}; \boldsymbol{\theta})$.

Maximising L is equivalent to maximise $\log(L)$, which for the model in (1) is

$$\log L(\boldsymbol{\theta}) = \sum_{i=1}^{M} \left(\log \binom{n_i}{k_i} + k_i \log \psi\left(x_i; \boldsymbol{\theta}\right) + \left(n_i - k_i\right) \log\left(1 - \psi\left(x_i; \boldsymbol{\theta}\right)\right) \right). \tag{3}$$

The *confidence intervals* CI for $\boldsymbol{\theta}$ are usually estimated using parametric or non-parametric bootstrap, often using the percentile method (Wichmann and Hill, 2001b; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012). For example, for the parameter α, the bootstrap percentile interval is defined as $(\alpha^*_{(a/2)}, \alpha^*_{(1-a/2)})$ where $\alpha^*_{(a/2)}$ and $\alpha^*_{(1-a/2)}$ are the $a/2$ and the $1-a/2$ percentiles of the bootstrapped replications of $\hat{\alpha}$. The i*th* bootstrap replication α^*_i is obtained using ML, but this time for a vector of simulated responses $\mathbf{k}^*$. Each k^*_i is simulated from a binomial distribution with parameters n_i and p_i where p_i is $\psi(x_i; \hat{\theta})$ for parametric bootstrap and k_i/n_i for non-parametric bootstrap (which is equivalent to sample with replacement from the distribution of *Yes* and *No* responses). It could be demonstrated that $CI = (\alpha^*_{(a/2)}, \alpha^*_{(1-a/2)})$ is a well-defined confidence interval (Wasserman, 2013). That is, $P(\alpha \in CI) \geq 1 - a$ where P is a probability and a is often arbitrarily chosen as 0.05. The confidence intervals for the other parameters are obtained similarly.

The observer's behaviour in classifications tasks is often summarised by a *threshold* x_{th} (O'Regan and Humbert, 1989; Wichmann and Hill, 2001a; Kingdom and Prins, 2009; Knoblauch and Maloney, 2012; Lu and Dosher, 2013), which corresponds to the stimulus level that predicts an arbitrarily chosen proportion of *Yes* responses. If the proportion is 0.5, for example, then x_{th} would be the x for which $\psi(x; \hat{\boldsymbol{\theta}}) = 0.5$. The bootstrap confidence intervals for x_{th} can be obtained using the percentile method for x^*_{th}, which are the bootstrap replications of x_{th} calculated using the bootstrap replications of the parameters.

ML estimates of the parameters are often obtained using non-linear optimisation (Nash, 2014), a method that might produce unsuitable estimates when the data suffer from lapses as those describe above. Future studies might elucidate the possible problems of non-linear optimisation to fit psychometric functions with lapses and whether Bayesian approaches might be preferred (Kuss et al.,

2005).

quickpsy and similar tools

quickpsy is a package to estimate and plot the parameters, the thresholds, the bootstrap confidence intervals for the parameters and the thresholds, and the associated psychometric functions for the model in (1). **quickpsy** also allows the comparison of the parameters and the thresholds for different groups using bootstrap. It is build to easily analyse data for multiple groups, which is common in classification experiments (Gescheider, 1997; Lu and Dosher, 2013): multiple observers, for example, might be tested under different conditions, such as flashes of different size in the light detection experiments. For that purpose, **quickpsy** incorporates, for example, a function to easily create a data frame from multiple files (`quickreadfiles`) or uses the dimensions of the groups in the data to produce standard plots for the parameters, the thresholds and the associated psychometric functions.

As an alternative to **quickpsy**, classification data can also be analysed in R using the base function `glm` and tools from **psyphy** (Knoblauch and Maloney, 2012; Knoblauch, 2014). Fitting psychometric functions could be considered a special case of fitting *generalised linear models (GLM)* (Knoblauch and Maloney, 2012; Moscatelli et al., 2012), which is done in R using `glm` (Knoblauch and Maloney, 2012). With `glm`, one can estimate the parameters of the linear predictor associated to GLM and use them to estimate the parameters of the model in (1) (Knoblauch and Maloney, 2012). Also by applying `confint` to the `glm` output (Knoblauch and Maloney, 2012), the confidence intervals can be calculated using the profile likelihood method (Venables and Ripley, 2002). To calculate the parameters and confidence intervals for multiple groups, the user needs to manually or automatically *loop* by group. Alternatively, if one is not interested in the individual values of the parameters for each group, but on fitting a single model to the multiple groups, `glm` allows to do that using advanced statistical methods (Knoblauch and Maloney, 2012; Moscatelli et al., 2012).

To estimate the parameters in (1) using `glm` is straightforward when $\gamma = 0$ and $\lambda = 0$ (to see `glm` in action for the `HSP` dataset, see Knoblauch and Maloney 2012), but becomes more complicated when γ and λ are not 0 or need to be estimated (Knoblauch and Maloney, 2012). To facilitate the application of `glm` when λ is not zero, **psyphy** provides specialised link functions. Furthermore, **psyphy** includes the `psyfun.2asym` function, which by iterating `glm` calls, allows the estimation of γ and λ.

The specific shape of F in (1) is sometimes chosen based on some theory (Green and Swets, 1966; Quick, 1974; Kingdom and Prins, 2009). Other times, especially when one is only interested in calculating the threshold, it is chosen arbitrarily. In those cases, one might prefer to fit a non-parametric model to the data, which can be done in R using **modelfree** (Zychaluk and Foster, 2009; Marin-Franch et al., 2012),

In other languages, the current available tools to calculate the parameters in (1) and its confidence intervals are **psignifit** (Fründ et al., 2011) for Python (free), **psycophysica** (Watson and Solomon, 1997) for Mathematica (commercial) and **palamedes** (Kingdom and Prins, 2009) for MATLAB (commercial). From them, **palamedes** can also fit models for multiple groups. Furthermore, **palamedes** and **psignifit** can fit psychometric functions using Bayesian methods, which is something not implemented in **quickpsy**.

Examples of usage

Light detection

A classic experiment on light detection (similar to the one first described in the Introduction) was conducted by Hecht et al. (1942). The data from the experiment is available at **MPDiR**, a package that includes material from the book *Modeling Psychophysical Data in R* (Knoblauch and Maloney, 2012).

The data is included in the data frame `HSP`, which has a *tidy* structure (Wickham, 2014), that is, each column corresponds to a variable and each row is an observational unit. The variables in `HSP` are the intensity level measured in quanta `Q` (what we named stimulus level x_i), the number of times that each intensity was presented `N` (what we named n_i) and two variables identifying the groups: the identifier of the observer `Obs` and the run for each observer `Run`. In the original dataset, the number of *Yes* responses for each stimulus level k_i was not given—the probability `p` of responding *Yes* was given instead. But, calculating the number of *Yes* responses from `p` is trivial:

```
library(MPDiR)
library(dplyr)
library(quickpsy)
HSP <- HSP %>% mutate(k = round(p * N / 100)) # adding a column with the number of Yes
```

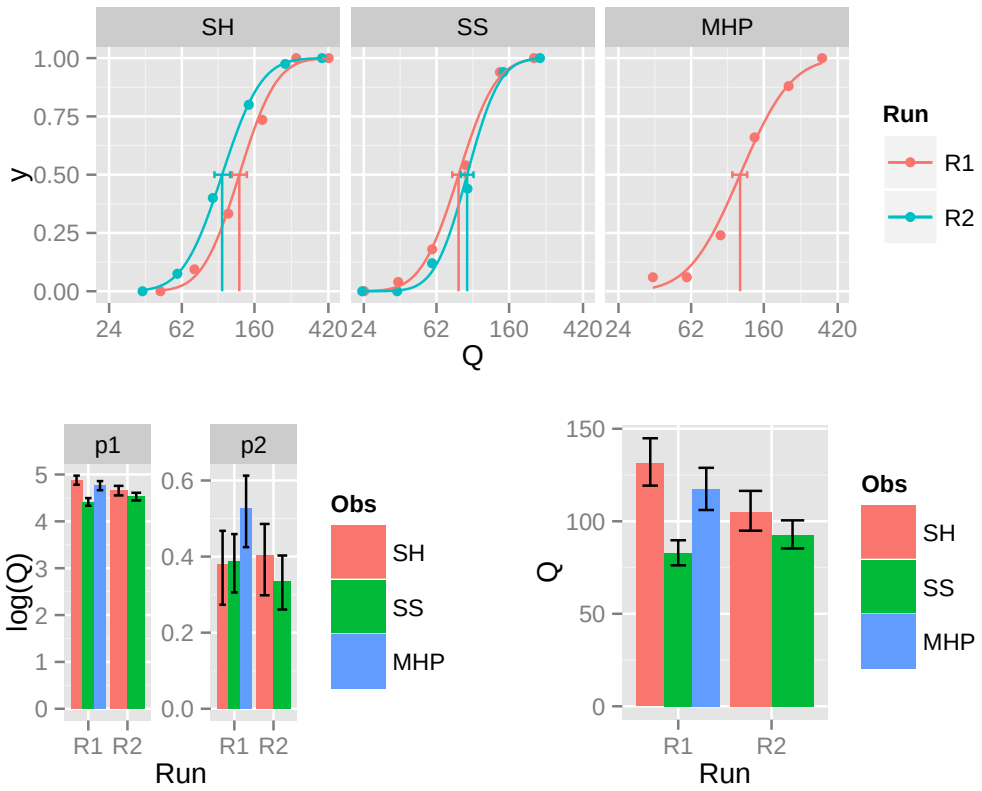

Figure 1: Psychometric functions, parameters and thresholds (including confidence intervals) resulting from fitting the model in (1) to the `HSP` data.

We used the `round` function because, curiously, there was some error in the original data set (see Knoblauch and Maloney, 2012).

To fit the model in (1) to the `HSP` dataset, we call the `quickpsy` function from **quickpsy**

```
fit <- quickpsy(HSP, Q, k, N, grouping = .(Run, Obs), B = 1000)
```

where `B` indicates the number of bootstrap samples (which can be reduced to shorten the computation time). In this call to `quickpsy`, many arguments were not explicitly specified but left to the default values: γ and λ fixed to 0: `guess = 0` and `lapses = 0`; F set to the cumulative normal distribution: `fun = cum_normal_fun`; the probability to calculate the threshold set to 0.5: `prob = guess + 0.5 * (1 -guess)`; the confidence intervals calculated using parametric bootstrap: `bootstrap = 'parametric'`.

`quickpsy` returns a list with all the information from the fitting. Most elements of the list are dataframes. For example, the parameters α and β, which for the cumulative normal distribution correspond, respectively, to the mean and the standard deviation, can be found in the dataframe `fit$par` (where `p1` corresponds to α and `p2` corresponds to β). The confidence intervals for the parameters are located in `fit$parci`. The thresholds and the confidence intervals for the thresholds are located in `fit$thresholds` and `fit$thresholdsci`.

`quickpsy` also returns the dataframe `fit$parcomparisons`, which includes for each parameter, pair comparisons between groups for all possible pairs of groups using bootstrap (Efron and Tibshirani, 1994). To compare two given groups, the difference between the bootstrap estimations of the parameter is calculated for all samples and from the distribution of differences, given the significance level set by the user (default: .95), the percentile confidence intervals are calculated. It is considered that the parameter differs between the two groups if the confidence intervals include the zero. Pair comparisons for the thresholds performed using the same method are available in `fit$thresholdcomparisons`.

To plot the fitted psychometric functions, we call the **quickpsy** function `plotcurves` including the fitted model as an argument

```
plotcurves(fit)
```

To plot the parameters and the thresholds, we use the **quickpsy** functions `plotpar(fit)` and `plotthresholds(fit)`, respectively.

Hecht, Shlaer and Pirenne, indeed, used $\log Q$, instead of Q as stimulus level. To easily fit and plot the model using the logarithm of the stimulus level, we can call `quickpsy` with `log = TRUE`

```
fit <- quickpsy(HSP, Q, k, N, grouping = .(Run, Obs), B = 1000, log = TRUE)
```

The plots associated to the fit above (Figure 1), using **gridExtra** (Auguie, 2015) for the plot arrangements, can be obtained as follows

```
library(gridExtra)
grid.arrange(plotcurves(fit), arrangeGrob(plotpar(fit),plotthresholds(fit), ncol = 2))
```

`HSP` is a dataframe that contains summarised data: the counts of *Yes* responses. `quickpsy`, however, can fit the dataframes containing more raw data in which each row corresponds to the result of one classification. In that case, the dataframe should contain a response column with `1`s indicating *Yes* responses and `0`s or `-1`s indicating *No* responses and `quickpsy` should be called with the name of the response column as the `k` argument (without the argument `n` corresponding to the number of trials).

We hope that this example has illustrated that **quickpsy** requires little coding to perform typical fits and plots in a classification task with multiple groups.

Criterion-independent light detection

Following the second example in the Introduction, consider some hypothetical data in which an observer needs to decide on which from two intervals a flash of light was presented

```
Q <- c(80, 160, 240, 320, 400) # luminance
n <- 100 # number of classifications per stimulus level
k <- c(59, 56, 69, 84, 96) # number of correct classifications
dat2IFC <- data.frame(Q, k, n)
```

To fit the model in (1) to this data, we call `quickpsy` with γ set to chance level (`guess = 0.5`)

```
fit <- quickpsy(dat2IFC, Q, k, n, guess = .5)
```

Light detection with lapses

Consider some hypothetical data from a light detection experiment, in which the observer commits a lapse (third example of the Introduction)

```
Q <- seq(0,420,60)
n <- 20
k <- c(0, 0, 4, 18, 20, 20, 19, 20) # lapse in the second to last intensity
datLapse <- data.frame(Q, k, n)
```

Suppose that we suspect that the observer might have committed lapses. Accordingly, we fit the model in (1) allowing λ to vary (`lapses = TRUE`)

```
fit <- quickpsy(datLapse, Q, k, n, lapses = TRUE, prob = .75)
```

The fitted psychometric function obtained with `plotcurves(fit)` is showed in Figure 2.

Typically, we are interested in the values of α, β or the threshold, which are related to the classification mechanisms (Wichmann and Hill, 2001a,b), but not in the specific value of λ. λ is allowed to vary, however, because fixing it when lapses occur can bias the estimation of α, β or the threshold (Wichmann and Hill, 2001a,b). To illustrate it, we fit the previous data with λ fixed to zero

```
fit <- quickpsy(datLapse, Q, k, n, lapses = 0, prob = .75)
```

Figure 3 shows that β and the threshold are biased.

Allowing λ to vary, however, not always *fixes* the fit. Prins (2012) has shown that, indeed, the simulations used by Wichmann and Hill (2001a,b) to illustrate how using variable λ eliminates the bias do cause a bias in threshold estimation. The following code uses **quickpsy** to replicate the results of Prins. Wichmann and Hill created 7 sampling schemes for stimulus presentation that were created specifying the values of ψ, which was a Weibull function with parameters 10 and 3

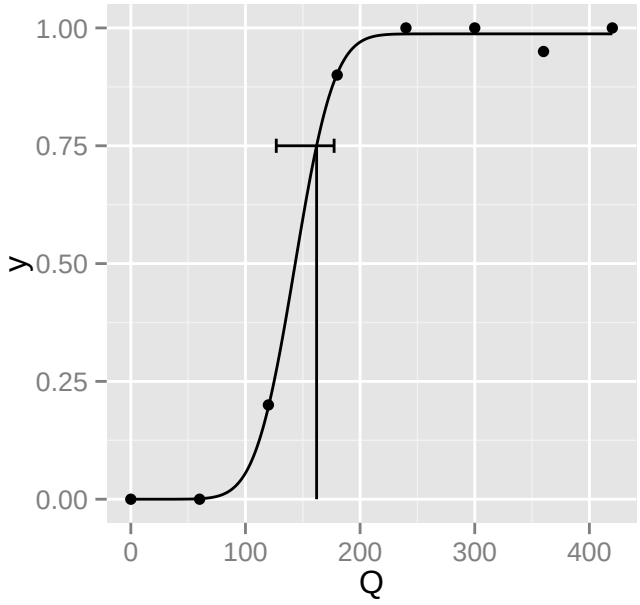

Figure 2: Psychometric function for the light detection experiment with lapses allowing λ to vary.

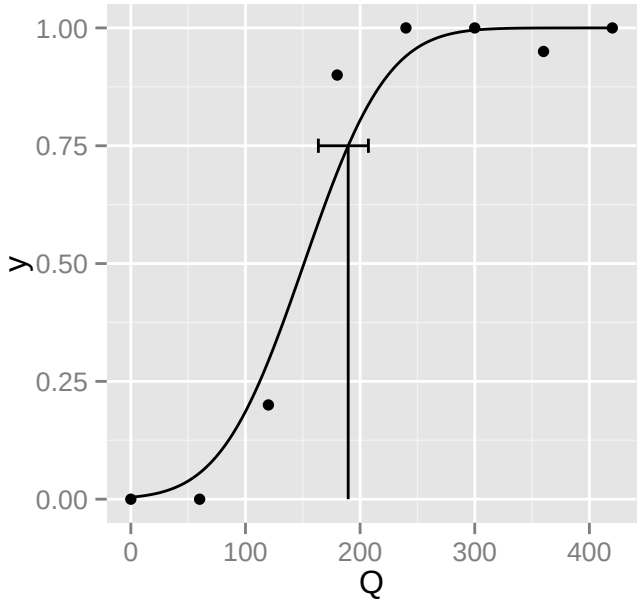

Figure 3: Psychometric function for the light detection experiment with lapses fixing λ.

```
parweibull <- c(10, 3)
create_xs <- function(i, f) data.frame(scheme = i,  y = f,
                                       x = inv_weibull_fun(f, parweibull))
s <- list()
s[[1]] <- create_xs(1, c(.3, .4, .48, .52, .6, .7))
s[[2]] <- create_xs(2, c(.1, .3, .4, .6, .7, .9))
s[[3]] <- create_xs(3, c(.3, .44, .7, .8, .9, .98))
s[[4]] <- create_xs(4, c(.1, .2, .3, .4, .5, .6))
s[[5]] <- create_xs(5, c(.08, .18, .28, .7, .85, .99))
s[[6]] <- create_xs(6, c(.3, .4, .5, .6, .7, .99))
s[[7]] <- create_xs(7, c(.34, .44, .54, .8, .9, .98))
s <- do.call('rbind', s)
```

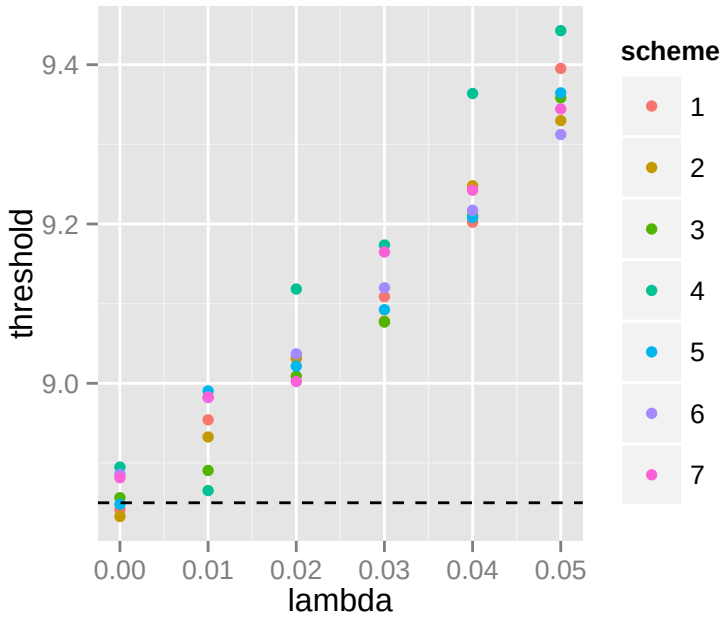

Figure 4: Threshold estimation for several sampling scheme simulating data using different values for λ. The horizontal dotted line shows the values of the unbiased threshold.

```
s$scheme <- factor(s$scheme)
```

Next, Wichmann and Hill simulated binomial responses using the Weibull function with several possible values for λ

```
create_sim_dat <- function(d) {
  psychometric_fun <- create_psy_fun(weibull_fun, .5, d$lambda)
  ypred <- psychometric_fun(d$x, parweibull)
  k <- rbinom(length(d$x), d$n, ypred)
  data.frame(x = d$x, k = k, n = d$n , y = k/d$n)
}
library(dplyr)
simdat <- merge(s, expand.grid(n = 160, sample = 1:100, lambda = seq(0,.05, .01))) %>%
  group_by(scheme, n, sample, lambda) %>% do(create_sim_dat(.))
```

To fit the simulated data, we use `quickpsy` bounding the possible values of λ to $[0, 0.6]$ as Wichmann and Hill did

```
fit_lapses <- quickpsy(simdat, x, k, n, within = .(scheme, lambda, sample),
                       fun = weibull_fun, bootstrap = 'none', guess =.5, lapses = TRUE,
                       parini = list(c(1,30), c(1,10), c(0,.06)))
```

Then, we average the threshold estimation across simulations

```
thre_lapses <- fit_lapses$thresholds %>% group_by(scheme, lambda) %>%
  summarise (threshold = mean(thre))
```

and plot them including the non-biased threshold for comparison

```
real_threshold <- inv_weibull_fun((.75 - .5) / (1 - .5 - 0), parweibull)
library(ggplot2)
ggplot(thre_lapses) +
  geom_point(aes(x = lambda, y = threshold, color = scheme)) +
  geom_hline(yintercept = real_threshold, lty = 2)
```

Figure 4 shows, consistent with Prins (2012), that the thresholds are biased for all of the sampling schemes and that the bias increases with the λ used to simulate the data.

Appearance-based procedures

In the previous sections, we exemplified the use of **quickpsy** for *performance-based procedures* (light detection), but fitting psychometric functions is also common for *appearance-based procedures* (Kingdom and Prins, 2009). For example, psychometric functions are often fitted to data measuring visual illusions such as the flash-lag (e.g. López-Moliner and Linares, 2006). In those cases, γ and λ are usually fixed to 0 and the probability summarising the behaviour of the classifier to 0.5. The stimulus level that predicts 0.5 proportion of *Yes* responses is called the *point of subjective equality (PSE)*.

Implementation details

Non-standard evaluation and grouping

`quickpsy`, the main function of **quickpsy**, is a *non-standard evaluation NSE* function and, as such, the name of the arguments and not only their values can be accessed (Wickham, 2014). NSE functions, which are common in R, are useful for example to label the axes of a plot using the name of the arguments (Wickham, 2014). As calling NSE functions from other functions is difficult (Wickham, 2014), **quickpsy** also incorporates `quickspsy_` which is the *standard evaluation SE* version of `quickpsy`. To call SE functions, some of the arguments need to be quoted. The following code exemplifies the use of `quickspsy_` to fit the HSP dataset from the first example of usage

```
fit <- quickpsy_(HSP, 'Q', 'k', 'N', grouping = c('Run', 'Obs'))
```

The NSE high-level plotting functions of **quickpsy** `plotcurves`, `plotpar` and `plotthresholds` also have associated SE versions: `plotcurves_`, `plotpar_` and `plotthresholds_`.

One of the main features of **quickpsy** is the possibility of fitting multiple groups. To handle the data analysis and plotting for multiple groups, **quickpsy** relies on **dplyr** (Wickham and Francois, 2015) and **ggplot2** (Wickham, 2009) respectively. In particular, **quickpsy** makes extensive use of the function `do` from **dplyr**, which splits input data frames by group, applies a function to each group and returns an output data frame. `quickpsy`, for example, calls the functions `curves` and `thresholds`, which basically contain a `do` function that, in turn, calls `one_curve` and `one_threshold`, which are the functions that calculate the psychometric curve and the threshold for a group of data (this method of function `X` calling function `one_X`, which performs the computations for a group of data, is used for some other functions called from `quickpsy`).

Closures

Closures or *function factories* are functions written by functions (Wickham, 2014). When estimating the maximum likelihood parameters for the model in (1), two procedures can be executed naturally using closures. One is the creation of ψ from F (and the parameters γ and λ), which is implemented in the `create_psy_fun` closure. The other is the creation of the negative log likelihood function from ψ (and the data), which is implemented in the `create_nll` closure.

Round-off errors in the log likelihood

Consider that we want to *manually* fit a cumulative normal psychometric function to the following data

```
x <- c( -0.056,  0.137,  0.331,  0.525,  0.719,  0.912,  1.100)
k <- c( 0, 5, 11, 12, 12, 12, 12)
n <- c( 12, 12, 12, 12, 12, 12, 12)
```

by direct minimisation of the negative log likelihood. We build the negative log likelihood

```
nll <- function(p) {
  phi <- pnorm(x, p[1], p[2])
  -sum(k * log(phi) + (n - k) * log(1 - phi))
}
```

and use `optim` to find the parameters that minimise it

```
optim(c(0.5, 0.1), nll)
```

`optim` requires initial values for the parameters that we arbitrarily chose as `c(0.5,0.1)`. But suppose that we choose another set of initial parameters

```
optim(c(0.1, 0.1), nll)
```

This time `optim` returns an error: `Error in optim(c(0.1,0.1),nll) : function cannot be evaluated at initial parameters`. The problem is that for the `1.100` stimulus level, `pnorm(1.100,0.1,0.1)` is rounded to `1` and therefore the term `log(1-phi)` in the negative log likelihood is `-Inf`.

To avoid this problem, the coding of the negative log likelihood in **quickpsy** incorporates the following lines

```
phi[phi < .Machine$double.eps] <- .Machine$double.eps
phi[phi > (1 - .Machine$double.eps)] <- 1 - .Machine$double.eps
```

That is, values that are smaller than the smallest positive floating-point number (`.Machine$double.eps`) are replaced by the smallest positive floating-point number and values that are larger than `1 - .Machine$double.eps` are replaced by `1 - .Machine$double.eps`. We can verify that `quickpsy` does not produce errors when using the *problematic* initial values with the following code

```
dat <- data.frame(x, k, n)
fit <- quickpsy(dat, x,  k, n, parini = c(0.1, 0.1))
```

Optimisation and initial parameters

By default, `quickpsy` searches the minimum of the negative log likelihood using `optim`, which requires initial values for the parameters. To free the user from the necessity of providing initial parameters, `quickpsy` uses as them the parameters obtained by linear modelling probit-transformed data (McKee et al., 1985; Gescheider, 1997). Linear modelling probit-transformed data is a poor technique to estimate the parameters of the psychometric function (McKee et al., 1985), but our tests suggest that they are good enough to be used as initial parameters.

The user can overwrite the initial parameters calculated using probit-transformed data by providing a vector of initial parameters to the argument `parini` of `quickpsy`. A list of vectors can also be fed when the user wants to set up some bounds for the parameters (see the last example of usage).

Acknowledgments

We thank Kenneth Knoblauch for providing helpful comments on a draft version of the manuscript and helping us with the problem associated to the rounding-off errors in the calculation of the likelihood. The research group is supported by Grant 2014SGR79 from the Catalan government. J. López-Moliner was supported by an ICREA Academic Distinguished Professorship award and grant PSI2013-41568-P from MINECO.

Bibliography

B. Auguie. *gridExtra: Miscellaneous Functions for 'grid' Graphics*, 2015. URL http://CRAN.R-project.org/package=gridExtra. R package version 2.0.0. [p5]

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. CRC Press, 1994. [p4]

J. Fischer and D. Whitney. Serial dependence in visual perception. *Nature Neuroscience*, 17(5):738–743, May 2014. [p1]

I. Fründ, N. V. Haenel, and F. A. Wichmann. Inference for psychometric functions in the presence of nonstationary behavior. *Journal of Vision*, 11(6), 2011. [p1, 3]

G. A. Gescheider. *Psychophysics*. The Fundamentals. Psychology Press, June 1997. [p2, 3, 9]

J. I. Gold and L. Ding. How mechanisms of perceptual decision-making affect the psychometric function. *Progress in Neurobiology*, 103:98–114, Apr. 2013. [p1]

J. I. Gold and M. N. Shadlen. The neural basis of decision making. *Annual Review of Neuroscience*, 30: 535–574, 2007. [p1]

D. M. Green and J. A. Swets. *Signal Detection Theory and Psychophysics*. Wiley, 1966. [p2, 3]

S. Hecht, S. Shlaer, and M. H. Pirenne. Energy, quanta, and vision. *The Journal of General Physiology*, 25 (6):819–840, July 1942. [p3]

F. A. A. Kingdom and N. Prins. *Psychophysics*. A Practical Introduction. Academic Press, Sept. 2009. [p1, 2, 3, 8]

S. A. Klein. Measuring, estimating, and understanding the psychometric function: A commentary. *Perception and Psychophysics*, 63(8):1421–1455, Nov. 2001. [p1]

K. Knoblauch. *psyphy: Functions for Analyzing Psychophysical Data in R*, 2014. URL http://CRAN.R-project.org/package=psyphy. R package version 0.1-9. [p3]

K. Knoblauch and L. T. Maloney. *Modeling Psychophysical Data in R*. Springer, New York, NY, Sept. 2012. [p1, 2, 3, 4]

M. Kuss, F. Jäkel, and F. A. Wichmann. Bayesian inference for psychometric functions. *Journal of Vision*, 5(5):478–492, 2005. [p2]

J. López-Moliner and D. Linares. The flash-lag effect is reduced when the flash is perceived as a sensory consequence of our action. *Vision Research*, 46(13):2122–2129, June 2006. [p8]

Z.-L. Lu and B. Dosher. *Visual Psychophysics*. From Laboratory to Theory. MIT Press, Oct. 2013. [p1, 2, 3]

N. A. Macmillan and C. D. Creelman. *Detection Theory*. A User's Guide. Psychology Press, Sept. 2004. [p1, 2]

I. Marin-Franch, K. Zychaluk, and D. H. Foster. *modelfree: Model-Free Estimation of a Psychometric Function*, 2012. URL http://CRAN.R-project.org/package=modelfree. R package version 1.1-1. [p3]

S. P. McKee, S. A. Klein, and D. Y. Teller. Statistical properties of forced-choice psychometric functions: Implications of probit analysis. *Perception and Psychophysics*, 37(4):286–298, 1985. [p9]

A. Moscatelli, M. Mezzetti, and F. Lacquaniti. Modeling psychophysical data at the population-level: The generalized linear mixed model. *Journal of Vision*, 12(11), 2012. [p3]

J. C. Nash. *Nonlinear Parameter Optimization Using R Tools*. John Wiley & Sons, New York, 2014. [p2]

J. K. O'Regan and R. Humbert. Estimating psychometric functions in forced-choice situations: significant biases found in threshold and slope estimations when small samples are used. *Perception and Psychophysics*, 46(5):434–442, Nov. 1989. [p1, 2]

N. Prins. The psychometric function: The lapse rate revisited. *Journal of Vision*, 12(6), 2012. [p5, 7]

R. F. Quick. A vector-magnitude model of contrast detection. *Kybernetik*, 16(2):65–67, 1974. [p3]

C. Summerfield and K. Tsetsos. Do humans make good decisions? *Trends in Cognitive Sciences*, 19(1): 27–34, Jan. 2015. [p1]

E. Van der Burg, D. Alais, and J. Cass. Rapid recalibration to audiovisual asynchrony. *Journal of Neuroscience*, 33(37):14633–14637, Sept. 2013. [p1]

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Statistics and Computing. Springer, New York, NY, 2002. [p3]

L. Wasserman. *All of Statistics*. A Concise Course in Statistical Inference. Springer, New York, NY, Dec. 2013. [p2]

A. B. Watson. Probability summation over time. *Vision Research*, 19(5):515–522, 1979. [p1, 2]

A. B. Watson and J. A. Solomon. Psychophysica: Mathematica notebooks for psychophysical experiments. *Spatial Vision*, 10(4):447–466, 1997. [p3]

F. A. Wichmann and N. J. Hill. The psychometric function: I. Fitting, sampling, and goodness of fit. *Perception and Psychophysics*, 63(8):1293–1313, Nov. 2001a. [p1, 2, 5]

F. A. Wichmann and N. J. Hill. The psychometric function: II. Bootstrap-based confidence intervals and sampling. *Perception and Psychophysics*, 63(8):1314–1329, Nov. 2001b. [p2, 5]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer, New York, 2009. URL http://had.co.nz/ggplot2/book. [p8]

H. Wickham. *Advanced R*. CRC Press, Sept. 2014. [p3, 8]

H. Wickham and R. Francois. *dplyr: A Grammar of Data Manipulation*, 2015. URL http://CRAN.R-project.org/package=dplyr. R package version 0.4.3. [p8]

K. Zychaluk and D. H. Foster. Model-free estimation of the psychometric function. *Attention, Perception & Psychophysics*, 71(6):1414–1425, Aug. 2009. [p3]

Daniel Linares
Vision and Control of Action Group, Departament de Psicologia Bàsica, Universitat de Barcelona
Passeig Vall d' Hebron 171, Barcelona 08035
Spain
danilinares@gmail.com

Joan López-Moliner
Vision and Control of Action Group, Departament de Psicologia Bàsica, Universitat de Barcelona
Institute for Brain, Cognition and Behaviour (IR3C), Barcelona
Passeig Vall d' Hebron 171, Barcelona 08035
Spain
j.lopezmoliner@ub.edu