TWCF Intrepid Experiment 2a staircase simulation

Clement Abbatecola

2025-06-17

The context of this document is this project: https://www.templetonworldcharity.org/projects-database/0646

See the pre-registration with more details here: https://osf.io/4rn85

Simulation function

This function is mostly a direct translation of a behavioural staircase with a simulated observer for the distance task in experiment 2a.

1. For each trial, define the actual difference in distance between target and foil ("dif").
2. Add random Gaussian noise to this value to simulate an imperfect observer ("perceived_dif"). By default this noise is centered at 0 with a standard deviation of 1, you can change these values to simulate a bias or precision effect.
3. Compare this perceived difference to 0. If the perceived difference is negative, it means that our simulated observer perceived the foil as smaller than the target and therefore chooses the foil. If the perceived distance is positive, it means that our simulated observer perceives the foil as bigger than the target and therefore chooses the target.
4. At the end of the trial increment staircase.

The function's output is a data frame with two columns, "foil_targ_dif" is the (actual) difference in distance for all trials, "targ_chosen" is TRUE when the target was chosen and FALSE when the foil was chosen.

```
simulate_staircase <- function(
        noise_mean = 0, # change bias (e.g. 1 to perceive targets as smaller)
        noise_sd = 1   # change in precision
        )
{
    nRevs   <- 20
    nTrials <- 60
    intervals <- c(3.5,3, 2.5, 2, 1.5, 1, .5, 0, -.5, -1, -1.5, -2, -2.5, -3, -3.5)
    direction <- c(1, -1)
    foil_type <- c(1, -1)
    revs <- c(0, 0)
    trial <- c(0, 0)
    cur_int <- c(1, 1)
    resps <- list(c(TRUE), c(TRUE))
    stairs_ongoing <- c(TRUE, TRUE)

    targ_chosen <- c()
    foil_targ_dif <- c()
```

```
    trials = 0
    while(any(stairs_ongoing)) {
        trials = trials + 1
        which_stair <- ifelse(stairs_ongoing[1], 1, 2)

        # simulate trial
        dif <- intervals[cur_int[which_stair]] * foil_type[which_stair]
        perceived_dif <- dif + rnorm(1, noise_mean, noise_sd)
        foil_targ_dif <- c(foil_targ_dif, dif)
        targ_chosen <- c(targ_chosen, perceived_dif > 0)

        # update staircase (which direction, is there a reversal?)
        resps[[which_stair]] <- c(resps[[which_stair]], perceived_dif > 0)

        if (tail(resps[[which_stair]], 2)[1] != tail(resps[[which_stair]], 2)[2]) {
            revs[which_stair] <- revs[which_stair] + 1
            direction[which_stair] <- direction[which_stair] * -1
        }

        # increment/update
        cur_int[which_stair] <- max(min(cur_int[which_stair] + direction[which_stair],
                                        length(intervals)), 1)
        trial[which_stair] <- trial[which_stair] + 1
        stairs_ongoing[which_stair] <- (revs[which_stair] <= nRevs) |
                                           (trial[which_stair] < nTrials)
    }

    return(data.frame(foil_targ_dif = foil_targ_dif, targ_chosen = targ_chosen))
}
```

Dataset aggregation function

This function takes a raw dataset and converts it to an analysis-friendly format.

It only requires a dataset with two columns (targ_chosen and foil_targ_dif) to work, but it will also add columns with additional relevant information (ID, targ_in_BS, ipsi_eye) if it's present. **Be careful that it will only look at the first row for this information, i.e. this function is only intended to work on datasets from a single condition.**

```
to_agg <- function(data.df)
{
    agg_data.df <- data.frame(
        targ_chosen = with(data.df, tapply(targ_chosen, foil_targ_dif, sum)),
        foil_chosen = with(data.df, tapply(!targ_chosen, foil_targ_dif, sum))
        )
    agg_data.df$foil_targ_dif <- as.numeric(rownames(agg_data.df))

    rownames(agg_data.df) <- c()

    exp_names <- seq(-3.5, 3.5,.5)
    for(x in exp_names) {
        if (!(x %in% agg_data.df$foil_targ_dif)) {
            agg_data.df <- rbind(agg_data.df,
```

```
                                matrix(c(0, 0, x), ncol = 3,
                                       dimnames = list(NULL, colnames(agg_data.df))))
        }
    }

    agg_data.df <- agg_data.df[order(agg_data.df$foil_targ_dif),]

    agg_data.df$ID         <- data.df[1,]$ID
    agg_data.df$targ_in_BS <- data.df[1,]$targ_in_BS
    agg_data.df$ipsi_eye   <- data.df[1,]$ipsi_eye
    return(agg_data.df)
}
```

Simulating a participant

With a bias effect of BS and a precision effect of ipsi vs contra (to illustrate both).

This is illustrated with plots.

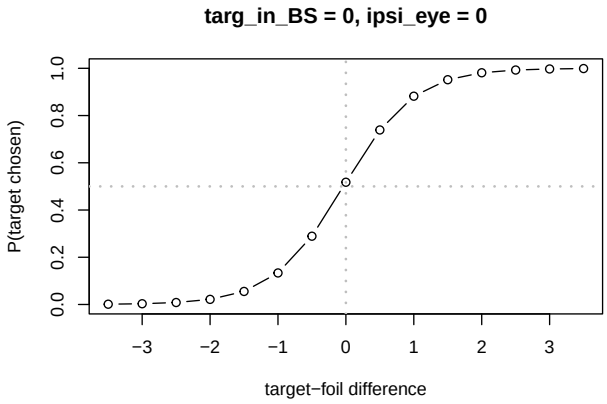
```
set.seed(0)

sim_out_contra.df <- simulate_staircase(0,1)
sim_out_contra.df$targ_in_BS <- FALSE
sim_out_contra.df$ipsi_eye <- FALSE

sim_in_contra.df <- simulate_staircase(1,1)
sim_in_contra.df$targ_in_BS <- TRUE
sim_in_contra.df$ipsi_eye <- FALSE

sim_out_ipsi.df <- simulate_staircase(0,2)
sim_out_ipsi.df$targ_in_BS <- FALSE
sim_out_ipsi.df$ipsi_eye <- TRUE

sim_in_ipsi.df <- simulate_staircase(1,2)
sim_in_ipsi.df$targ_in_BS <- TRUE
sim_in_ipsi.df$ipsi_eye <- TRUE

sim.df <- rbind(
    to_agg(sim_out_contra.df),
    to_agg(sim_in_contra.df),
    to_agg(sim_out_ipsi.df),
    to_agg(sim_in_ipsi.df)
    )

head(sim_out_contra.df)
```

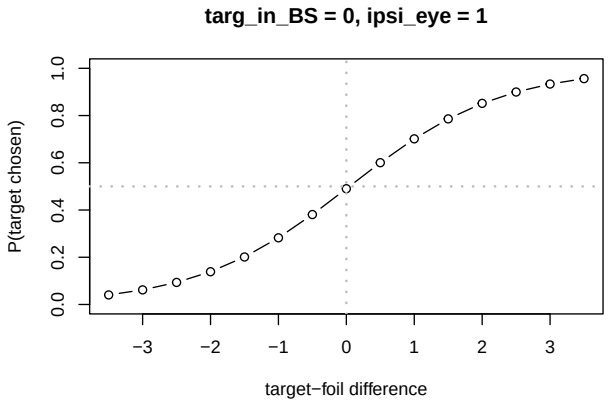
```
##   foil_targ_dif targ_chosen targ_in_BS ipsi_eye
## 1           3.5        TRUE      FALSE    FALSE
## 2           3.0        TRUE      FALSE    FALSE
## 3           2.5        TRUE      FALSE    FALSE
## 4           2.0        TRUE      FALSE    FALSE
## 5           1.5        TRUE      FALSE    FALSE
## 6           1.0       FALSE      FALSE    FALSE
```

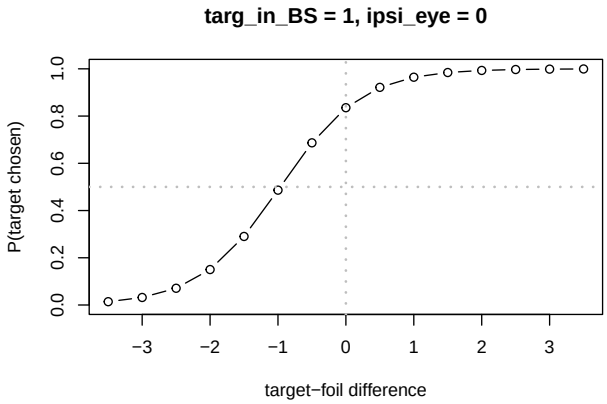
```r
head(sim.df)
```

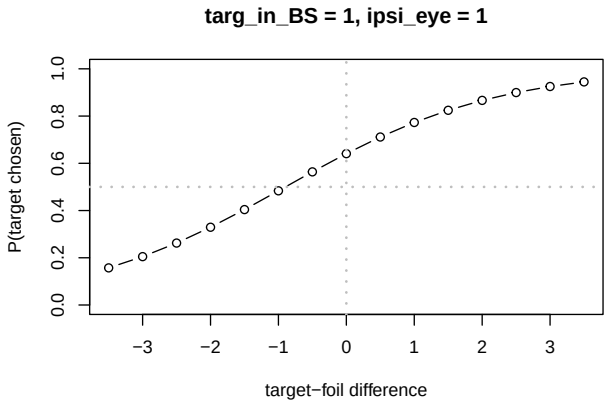
```
##   targ_chosen foil_chosen foil_targ_dif targ_in_BS ipsi_eye
## 1           0           1          -3.5      FALSE    FALSE
## 2           0           1          -3.0      FALSE    FALSE
## 3           0           1          -2.5      FALSE    FALSE
## 4           0           1          -2.0      FALSE    FALSE
## 5           0           1          -1.5      FALSE    FALSE
## 6           0           9          -1.0      FALSE    FALSE
```

```r
plot_cond <- function(agg.df, ...)
{
    mod <- glm(cbind(targ_chosen, foil_chosen) ~ foil_targ_dif,
               data = agg.df, family = binomial)

    plot(predict(mod, type = 'response') ~ agg.df$foil_targ_dif,
         type = 'b',
         xlim = c(-3.5, 3.5), ylim = c(0,1),
         xlab = "target-foil difference",
         ylab = "P(target chosen)",
         ...)

    abline(h = .5, lwd = 2, lty = 3, col = 'gray')
    abline(v = 0, lwd = 2, lty = 3, col = 'gray')
}

opar <- par(mfrow = c(2,2))

plot_cond(sim.df[!sim.df$targ_in_BS&!sim.df$ipsi_eye,],main="targ_in_BS = 0, ipsi_eye = 0")
plot_cond(sim.df[!sim.df$targ_in_BS& sim.df$ipsi_eye,],main="targ_in_BS = 0, ipsi_eye = 1")
plot_cond(sim.df[ sim.df$targ_in_BS&!sim.df$ipsi_eye,],main="targ_in_BS = 1, ipsi_eye = 0")
plot_cond(sim.df[ sim.df$targ_in_BS& sim.df$ipsi_eye,],main="targ_in_BS = 1, ipsi_eye = 1")
```


targ_in_BS = 0, ipsi_eye = 0
P(target chosen)
0.0
0.2
0.4
0.6
0.8
1.0
−3
−2
−1
0
1
2
3
target−foil difference


targ_in_BS = 0, ipsi_eye = 1
P(target chosen)
0.0
0.2
0.4
0.6
0.8
1.0
−3
−2
−1
0
1
2
3
target−foil difference


targ_in_BS = 1, ipsi_eye = 0
P(target chosen)
0.0
0.2
0.4
0.6
0.8
1.0
−3
−2
−1
0
1
2
3
target−foil difference


targ_in_BS = 1, ipsi_eye = 1
P(target chosen)
0.0
0.2
0.4
0.6
0.8
1.0
−3
−2
−1
0
1
2
3
target−foil difference