Age x Income: Analysis

1 Preliminaries

In this section, the RStudio workspace and console panes are cleared of old output, variables, and other miscellaneous debris. Be sure that nothing to be removed is needed. Once the decks are cleared, get required packages and data files.

1.1 Options

```
# Set some global options
options(replace.assign = TRUE, width = 65, digits = 4, scipen = 4, fig.width = 4,
    fig.height = 4)
# Clear the workspace and console
rm(list = ls(all.names = TRUE))
cat("\f")
```

```
# Start timing
how_long <- Sys.time()
# Set some additional formatting options
library(knitr)
library(formatR)
knitr::opts_chunk$set(message = FALSE, tidy.opts = list(width.cutoff = 60), tidy = TRUE)
thm <- knit_theme$get("biogoo")
knit_theme$set(thm)
```

1.2 Packages

```
library(psych)
library(rprojroot)
library(minpack.lm)
library(lme4)
library(glmmTMB)
library(multcomp)
library(modelr)
```

```r
library(here)
library(brms)
library(cmdstanr)
library(rstan)
library(emmeans)
library(tidybayes)
library(bayestestR)
library(tidyverse)
```

1.3 Data and Functions

```r
setwd(here("manuscript", "Resubmission2", "Analysis"))
# Load the data and functions for data analyses
source(here("manuscript", "Resubmission2", "Analysis", "DataTrim.R"))
source(here("manuscript", "Resubmission2", "Analysis", "Function.R"))
```

2 Discounting Measures

The following analyses were conducted to establish the representativeness of the current discounting data: For each discounting procedure (MCQ and Adj-Amt), we examined whether the results reflected the systematic changes in preference usually observed as the amount of reward and/or the delay to a reward increase.

2.1 Model Fit and Magnitude Effect

```
# MCQ: Logistic Growth Regression Model Fit
group_by(MCQ_grp_df, Group, Amount) |>
  do(# Intercept of the logistic growth function
     Intercept = coef(nlsLM(Prop ~ 1 / (1 + exp( - (log_k - a) * (r) )),
                      data = ., start = list(a = 1, r = .01),
                      control = list(maxiter = 1000)))[1],
     # Slope of the logistic growth function
     Slope = coef(nlsLM(Prop ~ 1 / (1 + exp( - (log_k - a) * (r) )),
                        data = ., start = list(a = 1, r = .01),
                        control = list(maxiter = 1000)))[2],
     # Model fit
     R2 = rsquare(nlsLM(Prop ~ 1 / (1 + exp( - (log_k - a) * (r) )),
                        data = ., start = list(a = 1, r = .01),
                        control = list(maxiter = 1000)), data =.)) |>
  as.data.frame() |> print(digits = 3)

##                      Group  Amount Intercept Slope    R2
## 1   Younger, Lower-Income $75-$85     -4.99 0.946 0.995
## 2   Younger, Lower-Income $50-$60     -4.62 0.883 0.991
## 3   Younger, Lower-Income $25-$35     -3.91  1.01 0.994
## 4     Older, Lower-Income $75-$85     -5.71 0.872 0.994
## 5     Older, Lower-Income $50-$60     -5.29  0.86 0.996
## 6     Older, Lower-Income $25-$35     -4.78 0.892 0.997
## 7  Younger, Higher-Income $75-$85     -5.58 0.798 0.994
## 8  Younger, Higher-Income $50-$60     -5.35 0.875 0.987
## 9  Younger, Higher-Income $25-$35     -4.56 0.849 0.994
## 10   Older, Higher-Income $75-$85     -5.82 0.908 0.986
## 11   Older, Higher-Income $50-$60      -5.5 0.934 0.991
## 12   Older, Higher-Income $25-$35     -4.75 0.968 0.996

# MCQ: Magnitude Effect (Multilevel Logistic Regression)
group_by(MCQ_ID_df, Group) |>
  summarise(pvalue = summary(
```

```r
    # Linear contrast was conducted using the glht() function from the multcomp package
    glht(
      # Multilevel logistic regression model was conducted using the glmer() function
      # from the lme4 package
      glmer(cbind(Num_Choice,9-Num_Choice) ~ -1+Amount+(1|ID), family=binomial()),
      linfct=matrix(c(contr.poly(3)[,1]), nc=3), alternative="two.sided", rhs=0)
    )$test$pvalues)

## # A tibble: 4 x 2
##   Group      pvalue
##   <dbl>       <dbl>
## 1     1 0.000000323
## 2     2 0.000000130
## 3     3 0.0000224
## 4     4 0.00000480

# Adj-Amt: Hyperboloid Function Model Fit
group_by(AdjAmt_grp_df, Amount, Group) |>
  do(# k of the hyperboloid discounting function
     k = coef(nlsLM(Mean_RSV ~ 1 / (1 + (k) * Delay)^(s),
                    data = ., start = list(k = .1, s = 1),
                    control = list(maxiter = 1000)))[1],
     # s of the hyperboloid discounting function
     s = coef(nlsLM(Mean_RSV ~ 1 / (1 + (k) * Delay)^(s),
                    data = ., start = list(k = .1, s = 1),
                    control = list(maxiter = 1000)))[2],
     # Model fit
     R2 = rsquare(nlsLM(Mean_RSV ~ 1 / (1 + (k) * Delay)^(s),
                        data = ., start = list(k = .1, s = 1),
                        control = list(maxiter = 1000)),
                  data =.)) |>
  as.data.frame() |>
  print(digits = 3)

##    Amount                  Group       k     s    R2
## 1    $500   Younger, Lower-Income   0.015   0.4 0.976
## 2    $500     Older, Lower-Income  0.0179 0.311  0.98
## 3    $500  Younger, Higher-Income 0.00452 0.602 0.967
## 4    $500    Older, Higher-Income  0.0098 0.386 0.952
## 5     $80   Younger, Lower-Income   0.073 0.271  0.99
## 6     $80     Older, Lower-Income   0.116 0.158 0.985
## 7     $80 Younger, Higher-Income   0.121 0.155 0.988
```

```
## 8     $80   Older, Higher-Income   0.142 0.141 0.952
## 9     $30  Younger, Lower-Income   0.181  0.23 0.997
## 10    $30    Older, Lower-Income   0.123 0.191 0.944
## 11    $30 Younger, Higher-Income   0.217 0.169 0.997
## 12    $30   Older, Higher-Income   0.131  0.18 0.967
```

```
# Adj-Amt: Magnitude Effect (Multilevel Beta Regression)
group_by(AdjAmt_ID_df, Group) |>
  summarise(pvalue = summary(
    # Linear contrast was conducted using the glht() function from the multcomp package
    glht(
      # Multilevel logistic regression model was conducted using the glmmTMB() function
      # from the glmmTMB package
      glmmTMB(AuC ~ -1+as.factor(Amount)+(1|ID), family=beta_family()),
      linfct=matrix(c(contr.poly(3)[,1]), nc = 3), alternative="two.sided", rhs=0)
    )$test$pvalues[1])
```

```
## # A tibble: 4 x 2
##   Group pvalue
##   <dbl>  <dbl>
## 1     1      0
## 2     2      0
## 3     3      0
## 4     4      0
```

2.2 Within-Procedure Correlation

The following analyses were conducted to evaluate the correlations among Amounts within each discounting procedure.

```
# MCQ
MCQ_cor <- ungroup(MCQ_ID_df) |>
    pivot_wider(names_from = Amount, values_from = Num_Choice) |>
    select(`$25-$35`, `$50-$60`, `$75-$85`)
print(cor(MCQ_cor), digits = 2)
```

```
##         $25-$35 $50-$60 $75-$85
## $25-$35    1.00    0.92    0.90
## $50-$60    0.92    1.00    0.94
## $75-$85    0.90    0.94    1.00
```

```
# Adj-Amt
```

```r
AdjAmt_cor <- ungroup(AdjAmt_ID_df) |>
    pivot_wider(names_from = Amount, values_from = AuC) |>
    select(`30`, `80`, `500`) |>
    rename(`$30` = "30", `$80` = "80", `$500` = "500")
print(cor(AdjAmt_cor), digits = 2)
```

```
##       $30  $80 $500
## $30  1.00 0.91 0.77
## $80  0.91 1.00 0.84
## $500 0.77 0.84 1.00
```

2.3 Between-Procedure Correlation

The following analyses were conducted to evaluate the intercorrelations among Amounts and the two discounting procedures.

```r
print(cor(MCQ_cor$`$25-$35`, AdjAmt_cor$`$30`), digits = 2)  # $30
```

```
## [1] 0.84
```

```r
print(cor(MCQ_cor$`$75-$85`, AdjAmt_cor$`$80`), digits = 2)  # $80
```

```
## [1] 0.86
```

```r
print(cor(cbind(MCQ_cor, AdjAmt_cor)), digits = 2)  # All Amounts
```

```
##         $25-$35 $50-$60 $75-$85  $30  $80 $500
## $25-$35    1.00    0.92    0.90 0.84 0.81 0.70
## $50-$60    0.92    1.00    0.94 0.86 0.86 0.76
## $75-$85    0.90    0.94    1.00 0.85 0.86 0.77
## $30        0.84    0.86    0.85 1.00 0.91 0.77
## $80        0.81    0.86    0.86 0.91 1.00 0.84
## $500       0.70    0.76    0.77 0.77 0.84 1.00
```

3 Effects of Age and Distress on Discounting

Focused contrasts were conducted using multilevel generalized linear models to examine the effect of Age at each level of Income for each discounting procedure using data from the two common amounts (i.e., $30, $80).

3.1 Discounting = fn[Age]

```
# Note: See Function.R file for more information on
# binomial_brm(), beta_brm(), and brm_summary() functions

# Lower Income
MCQ_LowInc_mod1 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + (1 | ID)), Data = subset(MCQ_mod1_df, Income_grp ==
    0))  # MCQ

## Running MCMC with 10 parallel chains...
##
## Chain 2 finished in 37.0 seconds.
## Chain 10 finished in 36.8 seconds.
## Chain 6 finished in 37.8 seconds.
## Chain 7 finished in 38.8 seconds.
## Chain 1 finished in 41.8 seconds.
## Chain 4 finished in 45.1 seconds.
## Chain 3 finished in 45.4 seconds.
## Chain 9 finished in 45.2 seconds.
## Chain 5 finished in 45.6 seconds.
## Chain 8 finished in 45.5 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 41.9 seconds.
## Total execution time: 45.8 seconds.

brm_summary(MCQ_LowInc_mod1)

##     rowname      b 95% CI_l 95% CI_u    pd signif
## 1 Intercept -0.092   -0.252    0.062 0.877
## 2   Age_grp  0.388    0.076    0.705 0.992      *

AdjAmt_LowInc_mod1 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    (1 | ID)), Data = subset(AdjAmt_mod1_df, Income_grp == 0))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 8 finished in 72.7 seconds.
## Chain 7 finished in 73.8 seconds.
## Chain 1 finished in 77.7 seconds.
## Chain 6 finished in 78.3 seconds.
## Chain 9 finished in 78.7 seconds.
## Chain 5 finished in 79.3 seconds.
## Chain 4 finished in 79.5 seconds.
## Chain 10 finished in 79.5 seconds.
## Chain 3 finished in 80.0 seconds.
## Chain 2 finished in 80.2 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 78.0 seconds.
## Total execution time: 80.3 seconds.
```

```
brm_summary(AdjAmt_LowInc_mod1)
```

```
##     rowname     b 95% CI_l 95% CI_u    pd signif
## 1 Intercept 0.526    0.349    0.706 1.000      *
## 2   Age_grp 0.485    0.127    0.833 0.997      *
```

```
# Higher Income
MCQ_HighInc_mod1 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + (1 | ID)), Data = subset(MCQ_mod1_df, Income_grp ==
    1))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 33.4 seconds.
## Chain 3 finished in 33.8 seconds.
## Chain 8 finished in 33.9 seconds.
## Chain 7 finished in 34.3 seconds.
## Chain 9 finished in 36.1 seconds.
## Chain 4 finished in 36.7 seconds.
## Chain 10 finished in 40.4 seconds.
## Chain 2 finished in 41.5 seconds.
## Chain 5 finished in 41.4 seconds.
## Chain 6 finished in 43.5 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 37.5 seconds.
```

```
## Total execution time: 43.8 seconds.

brm_summary(MCQ_HighInc_mod1)

##     rowname     b 95% CI_l 95% CI_u    pd signif
## 1 Intercept 0.034   -0.115    0.188 0.669
## 2   Age_grp 0.037   -0.266    0.340 0.595

AdjAmt_HighInc_mod1 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    (1 | ID)), Data = subset(AdjAmt_mod1_df, Income_grp == 1))  # Adj-Amt

## Running MCMC with 10 parallel chains...
##
## Chain 2 finished in 65.8 seconds.
## Chain 9 finished in 70.8 seconds.
## Chain 7 finished in 74.7 seconds.
## Chain 6 finished in 76.4 seconds.
## Chain 4 finished in 78.3 seconds.
## Chain 5 finished in 78.4 seconds.
## Chain 1 finished in 79.1 seconds.
## Chain 3 finished in 79.4 seconds.
## Chain 8 finished in 79.5 seconds.
## Chain 10 finished in 79.5 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 76.2 seconds.
## Total execution time: 80.1 seconds.

brm_summary(AdjAmt_HighInc_mod1)

##     rowname     b 95% CI_l 95% CI_u    pd signif
## 1 Intercept 0.768    0.599    0.940 1.000      *
## 2   Age_grp 0.077   -0.250    0.424 0.674
```

3.2 Discounting = fn[Age, Education, Gender]

```
# Lower Income
MCQ_LowInc_mod2 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + Education_grp_c + Gender_c + (1 | ID)), Data = subset(MCQ_mod2_df,
    Income_grp == 0))  # MCQ

## Running MCMC with 10 parallel chains...
```

```
##
## Chain 2 finished in 33.6 seconds.
## Chain 8 finished in 33.4 seconds.
## Chain 3 finished in 33.8 seconds.
## Chain 10 finished in 33.9 seconds.
## Chain 7 finished in 36.9 seconds.
## Chain 5 finished in 38.5 seconds.
## Chain 6 finished in 40.0 seconds.
## Chain 4 finished in 41.2 seconds.
## Chain 1 finished in 42.1 seconds.
## Chain 9 finished in 42.0 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 37.5 seconds.
## Total execution time: 42.5 seconds.
```

```
brm_summary(MCQ_LowInc_mod2)
```

```
##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept -0.027   -0.194    0.141 0.622
## 2       Age_grp  0.325    0.019    0.647 0.979      *
## 3 Education_grp  0.333    0.019    0.657 0.980      *
## 4        Gender -0.210   -0.514    0.105 0.908
```

```
AdjAmt_LowInc_mod2 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    Education_grp_c + Gender_c + (1 | ID)), Data = subset(AdjAmt_mod2_df,
    Income_grp == 0))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 81.9 seconds.
## Chain 2 finished in 82.2 seconds.
## Chain 10 finished in 82.1 seconds.
## Chain 5 finished in 82.5 seconds.
## Chain 3 finished in 82.8 seconds.
## Chain 4 finished in 82.8 seconds.
## Chain 8 finished in 82.7 seconds.
## Chain 7 finished in 83.1 seconds.
## Chain 6 finished in 83.4 seconds.
## Chain 9 finished in 84.3 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 82.8 seconds.
```

```
## Total execution time: 84.8 seconds.

brm_summary(AdjAmt_LowInc_mod2)

##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept  0.626    0.437    0.811 1.000      *
## 2       Age_grp  0.384    0.036    0.737 0.984      *
## 3 Education_grp  0.500    0.153    0.862 0.997      *
## 4        Gender -0.400   -0.752   -0.048 0.987      *

# Higher Income
MCQ_HighInc_mod2 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + Education_grp_c + Gender_c + (1 | ID)), Data = subset(MCQ_mod2_df,
    Income_grp == 1))  # MCQ

## Running MCMC with 10 parallel chains...
##
## Chain 3 finished in 35.4 seconds.
## Chain 8 finished in 37.7 seconds.
## Chain 5 finished in 38.0 seconds.
## Chain 4 finished in 39.7 seconds.
## Chain 1 finished in 42.7 seconds.
## Chain 9 finished in 44.8 seconds.
## Chain 10 finished in 44.8 seconds.
## Chain 6 finished in 45.3 seconds.
## Chain 7 finished in 45.5 seconds.
## Chain 2 finished in 46.3 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 42.0 seconds.
## Total execution time: 46.5 seconds.

brm_summary(MCQ_HighInc_mod2)

##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept -0.052   -0.224    0.123 0.722
## 2       Age_grp  0.010   -0.300    0.312 0.526
## 3 Education_grp  0.457    0.043    0.891 0.983      *
## 4        Gender  0.130   -0.173    0.447 0.795

AdjAmt_HighInc_mod2 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    Education_grp_c + Gender_c + (1 | ID)), Data = subset(AdjAmt_mod2_df,
    Income_grp == 1))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 4 finished in 81.1 seconds.
## Chain 6 finished in 83.1 seconds.
## Chain 1 finished in 84.0 seconds.
## Chain 5 finished in 83.8 seconds.
## Chain 9 finished in 83.5 seconds.
## Chain 2 finished in 84.0 seconds.
## Chain 10 finished in 83.5 seconds.
## Chain 3 finished in 84.0 seconds.
## Chain 7 finished in 84.3 seconds.
## Chain 8 finished in 84.3 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 83.6 seconds.
## Total execution time: 84.8 seconds.
```

```
brm_summary(AdjAmt_HighInc_mod2)
```

```
##         rowname     b 95% CI_l 95% CI_u    pd signif
## 1     Intercept 0.668    0.476    0.855 1.000      *
## 2       Age_grp 0.042   -0.291    0.381 0.600
## 3 Education_grp 0.545    0.083    1.015 0.990      *
## 4        Gender 0.158   -0.188    0.494 0.816
```

3.3 Discounting = fn[Age, Education, Gender, HADS]

```
# Lower Income
MCQ_LowInc_mod3 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + Education_grp_c + Gender_c + HADS_c + (1 | ID)),
    Data = subset(MCQ_mod3_df, Income_grp == 0))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 37.4 seconds.
## Chain 2 finished in 38.0 seconds.
## Chain 3 finished in 39.1 seconds.
## Chain 4 finished in 41.4 seconds.
## Chain 7 finished in 46.1 seconds.
## Chain 6 finished in 46.9 seconds.
## Chain 10 finished in 46.7 seconds.
```

```
## Chain 8 finished in 46.9 seconds.
## Chain 9 finished in 47.5 seconds.
## Chain 5 finished in 48.9 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 43.9 seconds.
## Total execution time: 49.3 seconds.
```

```
brm_summary(MCQ_LowInc_mod3)
```

```
##          rowname      b 95% CI_l 95% CI_u    pd signif
## 1      Intercept  0.004   -0.167    0.175 0.521
## 2        Age_grp  0.236   -0.103    0.580 0.914
## 3 Education_grp  0.322   -0.000    0.645 0.975
## 4         Gender -0.190   -0.510    0.118 0.884
## 5           HADS -0.239   -0.553    0.074 0.933
```

```
AdjAmt_LowInc_mod3 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    Education_grp_c + Gender_c + HADS_c + (1 | ID)), Data = subset(AdjAmt_mod3_df,
    Income_grp == 0))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 81.6 seconds.
## Chain 5 finished in 81.7 seconds.
## Chain 6 finished in 81.6 seconds.
## Chain 4 finished in 82.3 seconds.
## Chain 8 finished in 82.1 seconds.
## Chain 9 finished in 82.1 seconds.
## Chain 3 finished in 83.3 seconds.
## Chain 7 finished in 83.5 seconds.
## Chain 2 finished in 84.1 seconds.
## Chain 10 finished in 84.5 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 82.7 seconds.
## Total execution time: 85.0 seconds.
```

```
brm_summary(AdjAmt_LowInc_mod3)
```

```
##          rowname      b 95% CI_l 95% CI_u    pd signif
## 1      Intercept  0.666    0.471    0.857 1.000      *
## 2        Age_grp  0.270   -0.105    0.651 0.919
```

```
## 3 Education_grp  0.505    0.145    0.874 0.996      *
## 4        Gender -0.373   -0.724   -0.014 0.980      *
## 5          HADS -0.287   -0.634    0.050 0.950

# Higher Income
MCQ_HighInc_mod3 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Age_grp_c + Education_grp_c + Gender_c + HADS_c + (1 | ID)),
    Data = subset(MCQ_mod3_df, Income_grp == 1))  # MCQ

## Running MCMC with 10 parallel chains...
##
## Chain 5 finished in 36.7 seconds.
## Chain 2 finished in 37.5 seconds.
## Chain 9 finished in 37.8 seconds.
## Chain 8 finished in 38.4 seconds.
## Chain 10 finished in 39.2 seconds.
## Chain 1 finished in 46.0 seconds.
## Chain 7 finished in 45.7 seconds.
## Chain 4 finished in 46.6 seconds.
## Chain 6 finished in 47.2 seconds.
## Chain 3 finished in 47.9 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 42.3 seconds.
## Total execution time: 48.2 seconds.

brm_summary(MCQ_HighInc_mod3)

##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept -0.039   -0.214    0.130 0.676
## 2       Age_grp  0.090   -0.220    0.398 0.721
## 3 Education_grp  0.426    0.008    0.842 0.978      *
## 4        Gender  0.136   -0.166    0.449 0.812
## 5          HADS  0.232   -0.117    0.581 0.905

AdjAmt_HighInc_mod3 <- beta_brm(Formula = bf(AuC ~ Age_grp_c +
    Education_grp_c + Gender_c + HADS_c + (1 | ID)), Data = subset(AdjAmt_mod3_df,
    Income_grp == 1))  # Adj-Amt

## Running MCMC with 10 parallel chains...
##
## Chain 3 finished in 86.7 seconds.
## Chain 7 finished in 86.5 seconds.
```

```
## Chain 10 finished in 86.4 seconds.
## Chain 2 finished in 86.9 seconds.
## Chain 1 finished in 87.3 seconds.
## Chain 4 finished in 87.2 seconds.
## Chain 6 finished in 87.1 seconds.
## Chain 8 finished in 87.0 seconds.
## Chain 9 finished in 87.0 seconds.
## Chain 5 finished in 89.6 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 87.2 seconds.
## Total execution time: 90.0 seconds.
```

```
brm_summary(AdjAmt_HighInc_mod3)
```

```
##         rowname     b 95% CI_l 95% CI_u    pd signif
## 1     Intercept 0.683    0.485    0.876 1.000      *
## 2       Age_grp 0.105   -0.251    0.454 0.719
## 3 Education_grp 0.520    0.070    0.996 0.986      *
## 4        Gender 0.147   -0.205    0.490 0.796
## 5          HADS 0.223   -0.178    0.628 0.862
```

4 Effect of Income on Discounting

Focused contrasts were conducted using multilevel generalized linear models to examine the effects of Income at each level of Age using data from the two common amounts (i.e., $30, $80).

4.1 Discounting = fn[Income]

```
# Younger
MCQ_Young_mod1 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Income_grp_c + (1 | ID)), Data = subset(MCQ_mod1_df, Age_grp ==
    0))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 3 finished in 34.5 seconds.
## Chain 8 finished in 34.9 seconds.
## Chain 6 finished in 35.5 seconds.
## Chain 10 finished in 35.3 seconds.
```

```
## Chain 9 finished in 36.5 seconds.
## Chain 4 finished in 39.8 seconds.
## Chain 1 finished in 42.7 seconds.
## Chain 5 finished in 43.0 seconds.
## Chain 7 finished in 42.9 seconds.
## Chain 2 finished in 44.6 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 39.0 seconds.
## Total execution time: 44.7 seconds.

brm_summary(MCQ_Young_mod1)

##      rowname      b 95% CI_l 95% CI_u    pd signif
## 1  Intercept -0.129   -0.282    0.026 0.949
## 2 Income_grp  0.296   -0.018    0.606 0.969

AdjAmt_Young_mod1 <- beta_brm(Formula = bf(AuC ~ Income_grp_c +
    (1 | ID)), Data = subset(AdjAmt_mod1_df, Age_grp == 0))  # Adj-Amt

## Running MCMC with 10 parallel chains...
##
## Chain 8 finished in 75.4 seconds.
## Chain 1 finished in 85.2 seconds.
## Chain 2 finished in 85.4 seconds.
## Chain 9 finished in 85.0 seconds.
## Chain 5 finished in 85.4 seconds.
## Chain 7 finished in 85.5 seconds.
## Chain 4 finished in 85.8 seconds.
## Chain 6 finished in 85.7 seconds.
## Chain 3 finished in 86.0 seconds.
## Chain 10 finished in 85.6 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 84.5 seconds.
## Total execution time: 86.3 seconds.

brm_summary(AdjAmt_Young_mod1)

##      rowname     b 95% CI_l 95% CI_u    pd signif
## 1  Intercept 0.515    0.338    0.685 1.000      *
## 2 Income_grp 0.433    0.089    0.774 0.993      *
```

```r
# Older
MCQ_Old_mod1 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Income_grp_c + (1 | ID)), Data = subset(MCQ_mod1_df, Age_grp ==
    1))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 2 finished in 31.0 seconds.
## Chain 8 finished in 31.5 seconds.
## Chain 4 finished in 31.8 seconds.
## Chain 1 finished in 32.1 seconds.
## Chain 6 finished in 31.8 seconds.
## Chain 10 finished in 31.5 seconds.
## Chain 3 finished in 32.1 seconds.
## Chain 7 finished in 33.7 seconds.
## Chain 5 finished in 34.0 seconds.
## Chain 9 finished in 37.8 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 32.7 seconds.
## Total execution time: 38.5 seconds.
```

```r
brm_summary(MCQ_Old_mod1)
```

```
##      rowname      b 95% CI_l 95% CI_u    pd signif
## 1  Intercept  0.081   -0.073    0.234 0.849
## 2 Income_grp -0.053   -0.365    0.253 0.632
```

```r
AdjAmt_Old_mod1 <- beta_brm(Formula = bf(AuC ~ Income_grp_c +
    (1 | ID)), Data = subset(AdjAmt_mod1_df, Age_grp == 1))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 66.3 seconds.
## Chain 10 finished in 76.8 seconds.
## Chain 4 finished in 78.0 seconds.
## Chain 9 finished in 77.6 seconds.
## Chain 8 finished in 77.8 seconds.
## Chain 3 finished in 78.6 seconds.
## Chain 7 finished in 78.4 seconds.
## Chain 5 finished in 78.8 seconds.
## Chain 2 finished in 79.2 seconds.
## Chain 6 finished in 79.1 seconds.
```

```
##
## All 10 chains finished successfully.
## Mean chain execution time: 77.1 seconds.
## Total execution time: 79.6 seconds.
```

```r
brm_summary(AdjAmt_Old_mod1)
```

```
##       rowname     b 95% CI_l 95% CI_u    pd signif
## 1  Intercept 0.798    0.624    0.977 1.000      *
## 2 Income_grp 0.033   -0.311    0.390 0.573
```

4.2 Discounting = fn[Income, Education, Gender]

```r
# Younger
MCQ_Young_mod2 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Income_grp_c + Education_grp_c + Gender_c + (1 | ID)), Data = subset(MCQ_mod2_df,
    Age_grp == 0))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 4 finished in 37.9 seconds.
## Chain 6 finished in 38.6 seconds.
## Chain 2 finished in 41.2 seconds.
## Chain 1 finished in 49.7 seconds.
## Chain 10 finished in 49.4 seconds.
## Chain 8 finished in 49.7 seconds.
## Chain 9 finished in 49.7 seconds.
## Chain 3 finished in 50.3 seconds.
## Chain 7 finished in 50.2 seconds.
## Chain 5 finished in 51.3 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 46.8 seconds.
## Total execution time: 51.6 seconds.
```

```r
brm_summary(MCQ_Young_mod2)
```

```
##          rowname      b 95% CI_l 95% CI_u    pd signif
## 1      Intercept -0.100   -0.259    0.052 0.896
## 2     Income_grp  0.097   -0.245    0.431 0.715
## 3 Education_grp  0.472    0.129    0.824 0.996      *
## 4         Gender -0.101   -0.406    0.208 0.740
```

```r
AdjAmt_Young_mod2 <- beta_brm(Formula = bf(AuC ~ Income_grp_c +
    Education_grp_c + Gender_c + (1 | ID)), Data = subset(AdjAmt_mod2_df,
    Age_grp == 0))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 3 finished in 89.0 seconds.
## Chain 9 finished in 89.0 seconds.
## Chain 2 finished in 89.7 seconds.
## Chain 5 finished in 90.2 seconds.
## Chain 6 finished in 90.0 seconds.
## Chain 10 finished in 89.8 seconds.
## Chain 1 finished in 90.5 seconds.
## Chain 8 finished in 90.2 seconds.
## Chain 7 finished in 90.5 seconds.
## Chain 4 finished in 91.2 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 90.0 seconds.
## Total execution time: 91.5 seconds.
```

```r
brm_summary(AdjAmt_Young_mod2)
```

```
##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept  0.546    0.372    0.713 1.000      *
## 2    Income_grp  0.218   -0.159    0.599 0.874
## 3 Education_grp  0.524    0.132    0.919 0.995      *
## 4        Gender -0.085   -0.422    0.250 0.692
```

```r
# Older
MCQ_Old_mod2 <- binomial_brm(Formula = bf(Num_Choice | trials(9) ~
    Income_grp_c + Education_grp_c + Gender_c + (1 | ID)), Data = subset(MCQ_mod2_df,
    Age_grp == 1))  # MCQ
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 2 finished in 35.8 seconds.
## Chain 3 finished in 37.1 seconds.
## Chain 6 finished in 42.4 seconds.
## Chain 1 finished in 44.6 seconds.
## Chain 9 finished in 45.6 seconds.
## Chain 8 finished in 46.3 seconds.
## Chain 4 finished in 47.1 seconds.
```

```
## Chain 10 finished in 46.6 seconds.
## Chain 5 finished in 47.4 seconds.
## Chain 7 finished in 48.2 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 44.1 seconds.
## Total execution time: 48.8 seconds.
```

```
brm_summary(MCQ_Old_mod2)
```

```
##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept  0.069   -0.087    0.225 0.809
## 2    Income_grp -0.133   -0.468    0.202 0.784
## 3 Education_grp  0.220   -0.149    0.576 0.885
## 4        Gender -0.010   -0.327    0.304 0.526
```

```
AdjAmt_Old_mod2 <- beta_brm(Formula = bf(AuC ~ Income_grp_c +
    Education_grp_c + Gender_c + (1 | ID)), Data = subset(AdjAmt_mod2_df,
    Age_grp == 1))  # Adj-Amt
```

```
## Running MCMC with 10 parallel chains...
##
## Chain 1 finished in 82.5 seconds.
## Chain 4 finished in 82.6 seconds.
## Chain 2 finished in 83.0 seconds.
## Chain 10 finished in 82.4 seconds.
## Chain 9 finished in 82.9 seconds.
## Chain 3 finished in 84.9 seconds.
## Chain 6 finished in 85.8 seconds.
## Chain 5 finished in 86.4 seconds.
## Chain 7 finished in 96.6 seconds.
## Chain 8 finished in 98.7 seconds.
##
## All 10 chains finished successfully.
## Mean chain execution time: 86.6 seconds.
## Total execution time: 99.3 seconds.
```

```
brm_summary(AdjAmt_Old_mod2)
```

```
##         rowname      b 95% CI_l 95% CI_u    pd signif
## 1     Intercept  0.769    0.590    0.947 1.000      *
## 2    Income_grp -0.103   -0.488    0.273 0.705
## 3 Education_grp  0.408   -0.018    0.810 0.972
## 4        Gender -0.167   -0.526    0.188 0.826
```

4.3 The Association of Age/Income with the Composite Discounting Measure

The following analyses used a composite discounting measure (i.e., the participant's mean z-scores on the delayed choices and AuC measures) to evaluate the correlations between Age Group and z-scores for each Income Group and between Income Group and z-scores for each Age Group.

```
# Correlation between Age Group and z-score for each Income
# Group
cor.test(subset(z_df, Income_grp == 0)$z_score, subset(z_df,
    Income_grp == 0)$Age_grp)  # Lower Income

##
##  Pearson's product-moment correlation
##
## data:  subset(z_df, Income_grp == 0)$z_score and subset(z_df, Income_grp == 0)$Age_grp
## t = 2.7, df = 174, p-value = 0.007
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.05582 0.33983
## sample estimates:
##    cor
## 0.2021

cor.test(subset(z_df, Income_grp == 1)$z_score, subset(z_df,
    Income_grp == 1)$Age_grp)  # Higher Income

##
##  Pearson's product-moment correlation
##
## data:  subset(z_df, Income_grp == 1)$z_score and subset(z_df, Income_grp == 1)$Age_grp
## t = 0.55, df = 181, p-value = 0.6
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1046  0.1851
## sample estimates:
##     cor
## 0.04112

# Correlation between Income Group and z-score for each Age
# Group
cor.test(subset(z_df, Age_grp == 0)$z_score, subset(z_df, Age_grp ==
    0)$Income_grp)  # Younger
```

```
##
##  Pearson's product-moment correlation
##
## data:  subset(z_df, Age_grp == 0)$z_score and subset(z_df, Age_grp == 0)$Income_grp
## t = 2.2, df = 185, p-value = 0.03
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.01966 0.29920
## sample estimates:
##    cor
## 0.1627
```

```
cor.test(subset(z_df, Age_grp == 1)$z_score, subset(z_df, Age_grp ==
    1)$Income_grp)  # Older
```

```
##
##  Pearson's product-moment correlation
##
## data:  subset(z_df, Age_grp == 1)$z_score and subset(z_df, Age_grp == 1)$Income_grp
## t = 0.045, df = 170, p-value = 1
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1463  0.1530
## sample estimates:
##      cor
## 0.003426
```

5 The Magnitude of the Age Difference

The following analyses used the composite discounting measure to examine the effects of Age and Income on discounting.

```
# Mean z_score for each group
group_by(z_df, Age_grp, Income_grp) |>
    summarise(mean_zscore = mean(z_score)) |>
    mutate(Age_grp = ifelse(Age_grp == 0, "Younger", "Older"),
        Income_grp = ifelse(Income_grp == 0, "Lower-Income",
            "Higher-Income"))
```

```
## # A tibble: 4 x 3
## # Groups:   Age_grp [2]
```

```
##   Age_grp Income_grp    mean_zscore
##   <chr>   <chr>              <dbl>
## 1 Younger Lower-Income      -0.266
## 2 Younger Higher-Income      0.0425
## 3 Older   Lower-Income       0.112
## 4 Older   Higher-Income      0.118
```

```
# z-score = fn[Age, Income, Age x Income]
summary(lm(z_score ~ (Age_grp_c + Income_grp_c)^2, data = z_df))
```

```
##
## Call:
## lm(formula = z_score ~ (Age_grp_c + Income_grp_c)^2, data = z_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4354 -0.6807  0.0689  0.7513  1.9832
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -0.00141    0.04860   -0.03    0.977
## Age_grp_c               0.22360    0.09729    2.30    0.022 *
## Income_grp_c            0.16343    0.09722    1.68    0.094 .
## Age_grp_c:Income_grp_c -0.30196    0.19460   -1.55    0.122
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.921 on 355 degrees of freedom
## Multiple R-squared:  0.0284,Adjusted R-squared:  0.0202
## F-statistic: 3.46 on 3 and 355 DF,  p-value: 0.0166
```

```
# Age effect in each of the two income groups (Discounting
# = fn[Continuous Age]
summary(lm(z_score ~ Age, data = subset(z_df, Income_grp == 0)))  # Lower Income
```

```
##
## Call:
## lm(formula = z_score ~ Age, data = subset(z_df, Income_grp ==
##     0))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3842 -0.6577  0.0021  0.6979  2.0178
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.71330    0.26398   -2.70   0.0076 **
## Age          0.01147    0.00462    2.48   0.0140 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.923 on 174 degrees of freedom
## Multiple R-squared:  0.0342,Adjusted R-squared:  0.0287
## F-statistic: 6.16 on 1 and 174 DF,  p-value: 0.014
```

```r
summary(lm(z_score ~ Age, data = subset(z_df, Income_grp == 1)))  # Higher Income
```

```
##
## Call:
## lm(formula = z_score ~ Age, data = subset(z_df, Income_grp ==
##     1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4444 -0.6585  0.0795  0.7869  1.6923
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.09942    0.27267   -0.36     0.72
## Age          0.00328    0.00488    0.67     0.50
##
## Residual standard error: 0.921 on 181 degrees of freedom
## Multiple R-squared:  0.00249,Adjusted R-squared:  -0.00302
## F-statistic: 0.452 on 1 and 181 DF,  p-value: 0.502
```

```r
# Get system details.
S <- benchmarkme::get_sys_details()
GB <- memuse::Sys.meminfo()
```

The current machine uses the following CPU: Apple M1, with 8 cores and 16.000 GiB of RAM.

```r
sessionInfo()

## R version 4.2.1 (2022-06-23)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS 14.3
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods
## [7] base
##
## other attached packages:
##  [1] benchmarkme_1.0.8  readxl_1.4.3       lubridate_1.9.3
##  [4] forcats_1.0.0      stringr_1.5.1      dplyr_1.1.4
##  [7] purrr_1.0.2        readr_2.1.5        tidyr_1.3.1
## [10] tibble_3.2.1       ggplot2_3.4.4      tidyverse_2.0.0
## [13] bayestestR_0.13.1  tidybayes_3.0.6    emmeans_1.10.0
## [16] rstan_2.32.5       StanHeaders_2.32.5 cmdstanr_0.5.3
## [19] brms_2.20.4        Rcpp_1.0.12        here_1.0.1
## [22] modelr_0.1.11      multcomp_1.4-25    TH.data_1.1-2
## [25] MASS_7.3-60.0.1    survival_3.5-7     mvtnorm_1.2-4
## [28] glmmTMB_1.1.8      lme4_1.1-35.1      Matrix_1.6-5
## [31] minpack.lm_1.2-4   rprojroot_2.0.4    psych_2.4.1
## [34] formatR_1.14       knitr_1.45
##
## loaded via a namespace (and not attached):
##   [1] backports_1.4.1       plyr_1.8.9
##   [3] igraph_2.0.1.1        TMB_1.9.10
##   [5] splines_4.2.1         svUnit_1.0.6
```

```
##   [7] crosstalk_1.2.1       rstantools_2.4.0
##   [9] inline_0.3.19         digest_0.6.34
##  [11] foreach_1.5.2         htmltools_0.5.7
##  [13] fansi_1.0.6           magrittr_2.0.3
##  [15] checkmate_2.3.1       doParallel_1.0.17
##  [17] tzdb_0.4.0            RcppParallel_5.1.7
##  [19] matrixStats_1.2.0     xts_0.13.2
##  [21] sandwich_3.1-0        timechange_0.3.0
##  [23] colorspace_2.1-0      ggdist_3.3.1
##  [25] xfun_0.41             jsonlite_1.8.8
##  [27] iterators_1.0.14      zoo_1.8-12
##  [29] glue_1.7.0            gtable_0.3.4
##  [31] V8_4.4.1              distributional_0.3.2
##  [33] pkgbuild_1.4.3        abind_1.4-5
##  [35] scales_1.3.0          miniUI_0.1.1.1
##  [37] xtable_1.8-4          stats4_4.2.1
##  [39] DT_0.31               httr_1.4.7
##  [41] datawizard_0.9.1      htmlwidgets_1.6.4
##  [43] threejs_0.3.3         arrayhelpers_1.1-0
##  [45] posterior_1.5.0       ellipsis_0.3.2
##  [47] pkgconfig_2.0.3       loo_2.6.0
##  [49] farver_2.1.1          utf8_1.2.4
##  [51] tidyselect_1.2.0      rlang_1.1.3
##  [53] reshape2_1.4.4        later_1.3.2
##  [55] munsell_0.5.0         cellranger_1.1.0
##  [57] tools_4.2.1           cli_3.6.2
##  [59] generics_0.1.3        broom_1.0.5
##  [61] evaluate_0.23         fastmap_1.1.1
##  [63] processx_3.8.3        nlme_3.1-164
##  [65] mime_0.12             compiler_4.2.1
##  [67] bayesplot_1.11.0      shinythemes_1.2.0
##  [69] rstudioapi_0.15.0     curl_5.2.0
##  [71] stringi_1.8.3         ps_1.7.6
##  [73] highr_0.10            Brobdingnag_1.2-9
##  [75] memuse_4.2-3          lattice_0.22-5
##  [77] nloptr_2.0.3          markdown_1.12
##  [79] shinyjs_2.1.0         tensorA_0.36.2.1
##  [81] vctrs_0.6.5           pillar_1.9.0
##  [83] lifecycle_1.0.4       bridgesampling_1.1-2
##  [85] estimability_1.4.1    data.table_1.15.0
##  [87] insight_0.19.8        httpuv_1.6.14
```

```
##  [89] QuickJSR_1.1.3        R6_2.5.1
##  [91] promises_1.2.1        gridExtra_2.3
##  [93] codetools_0.2-19      benchmarkmeData_1.0.4
##  [95] boot_1.3-28.1         colourpicker_1.3.0
##  [97] gtools_3.9.5          withr_3.0.0
##  [99] shinystan_2.6.0       mnormt_2.1.1
## [101] mgcv_1.9-1            parallel_4.2.1
## [103] hms_1.1.3             grid_4.2.1
## [105] coda_0.19-4.1         minqa_1.2.6
## [107] numDeriv_2016.8-1.1   shiny_1.8.0
## [109] base64enc_0.1-3       dygraphs_1.1.1.6
```

```
Sys.time() - how_long
```

```
## Time difference of 24.72 mins
```