Supplementary Material For LMEM of Diary Data

Sungjoon Park

26 April, 2019

Introduction

This document is a supplement to the main text and provides the steps taken to construct the models and figures it contains.

Importing Libraries

```r
library(data.table)
library(tidyverse)
library(lme4)
library(lmerTest)
library(ggplot2)
library(sjPlot)
library(sjstats)
library(qdap)
library(tm)
library(jtools)
library(optimx)
```

Importing and cleaning Data

```r
# Assigning a varible to contain the diary
wisData <- fread("data/Diary.demo.csv")

#   Using the mutate() function to create new variables
wisData <- wisData %>%
    # First construct two variables that code for either positive or
    # negative recall events
  mutate(negSocEvent = case_when(experienceEventW1_2_1 == 1 |
                                   experienceEventW1_2_4 == 1 |
                                   experienceEventW1_2_5 == 1 ~ 1,
                                 TRUE ~ 0),
         posSocEvent = case_when(experienceEventW1_2_2 == 1 |
                                   experienceEventW1_2_3 == 1 ~ 1,
                                 TRUE ~ 0),
    # Next we create a new variable that codes the above and
    # includes those who experienced both types
         recallEventType = case_when(negSocEvent == 1 & posSocEvent == 1 ~ 0,
                                   posSocEvent == 1 ~ .5,
                                   negSocEvent == 1 ~ -.5),
    # Contrast coding the perspective condition
         conditionC = case_when(Condition == 1 ~ -0.5,
                                Condition == 3 ~ 0.5))

# reversing the happiness variable to ascend with happiness
wisData <- wisData %>%
  mutate(feelRightNowWk1_2_4rev = case_when(feelRightNowWk1_2_4 == 1 ~ 6,
                                                     feelRightNowWk1_2_4 == 2 ~ 5,
                                                     feelRightNowWk1_2_4 == 3 ~ 4,
                                                     feelRightNowWk1_2_4 == 4 ~ 3,
                                                     feelRightNowWk1_2_4 == 5 ~ 2,
                                                     feelRightNowWk1_2_4 == 6 ~ 1,
                                                     feelRightNowWk1_2_4 == 7 ~ 0))

# Group-mean centering the happiness variable as we are interested
# in the level-1 effect
wisData <- wisData %>%
  group_by(subjectNumber) %>%
  mutate(feelHappyC = feelRightNowWk1_2_4rev -
           mean(feelRightNowWk1_2_4rev, na.rm = TRUE))

# Setting the day data to start from zero
wisData$zeroDay <- wisData$Day_correct - 1
```

Cleaning diary entry text and extracting polarity

```
# Converting diary entry text data into utf-8 format
# This must be done for the qdap::polarity() function to read.
wisData$cleanText <- enc2utf8(wisData$Event.clean)

# Removing punctuations using the tm package.
# The qdap::polarity() function does not handle punctuations.
wisData$cleanText <- tm::removePunctuation(wisData$cleanText)

# Computing text polarity
wisData$polarity <- qdap::polarity(wisData$cleanText)$all$polarity
    # the "$all$polarity" code at the end is vital as we only wish to
    # gain the polarity value and not the other outputs.

# Summary output of polarity
summary(wisData$polarity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## -1.4849 -0.1543  0.2000  0.1683  0.4804  1.5827    1201
```

Constructing a null model

The null model only involved our dependent variable of interest (polarity) and our random effect (subjects). We can also calculate the model's ICC.

```
mod1 <- lme4::lmer(polarity ~ 1 + (1|subjectNumber), data = wisData)
summary(mod1)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: polarity ~ 1 + (1 | subjectNumber)
##    Data: wisData
##
## REML criterion at convergence: 1741.8
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3966 -0.6377  0.0520  0.6395  3.0408
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  subjectNumber (Intercept) 0.01624  0.1275
##  Residual                  0.21109  0.4594
## Number of obs: 1297, groups:  subjectNumber, 158
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  0.16430    0.01704   9.641
```

```
# Computing ICC
input = as.data.frame(summary(mod1)$varcor)
icc = input$vcov[1] / sum(input$vcov)
icc
```

```
## [1] 0.07145885
```

The ICC value is the proportion of the variance in polarity that can be explained by individual differences.

Constructing more models

```
# Addiing the day variable as a fixed effect (fixed slope model)
mod2 <- lmer(polarity ~ zeroDay + (1|subjectNumber), data=wisData)

# A model with day variable to differ between days (random slope model)
mod3 <- lmer(polarity ~ zeroDay + (zeroDay|subjectNumber), data=wisData)
```

```
## boundary (singular) fit: see ?isSingular
```

```
# comparing the two models
anova(mod2, mod3)
```

```
## refitting model(s) with ML (instead of REML)
```

```
## Data: wisData
## Models:
## mod2: polarity ~ zeroDay + (1 | subjectNumber)
## mod3: polarity ~ zeroDay + (zeroDay | subjectNumber)
##      Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod2  4 1743.1 1763.8 -867.56   1735.1
## mod3  6 1746.9 1777.9 -867.45   1734.9 0.2132      2     0.8989
```

As there is no significant difference between the two models I opted for the fixed slope model as it makes little sense to have a random slope for the day variable.

```
# Although I have changed the way I write the lmer function, it is
# identical to previous one. This is for me to see my function virtically
# instead of taking horizontal space
```

```r
# Model with the perspective condition added  as a fixed effect
mod4 <- lmer(polarity ~
               zeroDay +
               conditionC+
               (1|subjectNumber),
             data=wisData)

# Model with all fixed effect variables of interest added
mod5 <- lmer(polarity ~
               zeroDay +
               conditionC+
               recallEventType +
               feelHappyC +
               (1|subjectNumber),
             data=wisData)

# model with both happiness and recall event type variables
# added to vary across days (random-slope model)
mod6a <- lmer(polarity ~
               zeroDay +
               conditionC+
               recallEventType +
               feelHappyC +
               (feelHappyC+recallEventType|subjectNumber),
             data=wisData)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00211479
## (tol = 0.002, component 1)
```

```r
# the above model causes a warning that it failed to converge
# This can be resolved by making the lmer function try a different
# optimizer. The package "optimx" was required to do this.
mod6b <- lmer(polarity ~
               zeroDay +
               conditionC+
               recallEventType +
               feelHappyC +
               (feelHappyC+recallEventType|subjectNumber),
             data=wisData,
             control = lmerControl(optimizer = 'optimx', optCtrl=list(method='L-BFGS-B')))

# Alternatively, the code:
#    control = lmerControl(optimizer = "bobyqa",optCtrl=list(maxfun=2e5)))
# can be run with the default lme4 package. However some of my future
# models faced convergence warnings with it.
# When one optimized doesn't work it may be worth trying others.
# the optCtrl=list(method= ) portion of the code can specify the optimizer (eg.'bobyqa')

# Comparing the fixed-slope and random-slope models
anova(mod5,mod6b)
```

```
## refitting model(s) with ML (instead of REML)
## Data: wisData
## Models:
## mod5: polarity ~ zeroDay + conditionC + recallEventType + feelHappyC +
## mod5:     (1 | subjectNumber)
## mod6b: polarity ~ zeroDay + conditionC + recallEventType + feelHappyC +
## mod6b:     (feelHappyC + recallEventType | subjectNumber)
##       Df     AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod5   7 1019.56 1055.7 -502.78  1005.56
## mod6b 12  996.48 1058.3 -486.24   972.48 33.077      5  3.633e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Dealing with singular fit warning

```
# This model attempts to model the interaction between happiness and
# event recall type. However this model faces a singular fit warning
mod7a <- lmer(polarity ~
               zeroDay +
               conditionC+
               recallEventType *
               feelHappyC +
               (feelHappyC*recallEventType|subjectNumber),
             data=wisData,
             control = lmerControl(optimizer = 'optimx', optCtrl=list(method='bobyqa')))
```

```
## boundary (singular) fit: see ?isSingular
```

```
# As a result I simplified the model to only interact as fixed effect
# and not with as random slopes.
mod7b <- lmer(polarity ~
               zeroDay +
               conditionC+
               recallEventType *
               feelHappyC +
               (feelHappyC+recallEventType|subjectNumber),
             data=wisData,
             control = lmerControl(optimizer = 'optimx', optCtrl=list(method='bobyqa')))

# Comparing the two models reveal they are not significantly different
# I chose the model that does not face the singular fit warning
anova(mod7a,mod7b)
```

```
## refitting model(s) with ML (instead of REML)
```

```
## Data: wisData
## Models:
## mod7b: polarity ~ zeroDay + conditionC + recallEventType * feelHappyC +
## mod7b:     (feelHappyC + recallEventType | subjectNumber)
## mod7a: polarity ~ zeroDay + conditionC + recallEventType * feelHappyC +
## mod7a:     (feelHappyC * recallEventType | subjectNumber)
##       Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod7b 13 984.58 1051.6 -479.29   958.58
## mod7a 17 983.70 1071.3 -474.85   949.70 8.8857      4    0.06402 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Display summary of the full model
summary(mod7b)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: polarity ~ zeroDay + conditionC + recallEventType * feelHappyC +
##     (feelHappyC + recallEventType | subjectNumber)
##    Data: wisData
## Control:
## lmerControl(optimizer = "optimx", optCtrl = list(method = "bobyqa"))
##
## REML criterion at convergence: 1001.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.2715 -0.6023 -0.0588  0.6046  2.9006
##
## Random effects:
##  Groups        Name            Variance Std.Dev. Corr
##  subjectNumber (Intercept)     0.009550 0.09773
##                feelHappyC      0.001835 0.04284  -0.61
##                recallEventType 0.043182 0.20780   0.13 -0.18
##  Residual                      0.107601 0.32803
## Number of obs: 1281, groups:  subjectNumber, 157
##
## Fixed effects:
##                              Estimate Std. Error         df t value
## (Intercept)                 3.740e-02  2.012e-02  4.895e+02   1.859
## zeroDay                    -4.005e-04  1.181e-03  1.221e+03  -0.339
## conditionC                 -4.615e-02  2.642e-02  1.281e+02  -1.747
## recallEventType             5.865e-01  3.291e-02  9.072e+01  17.820
## feelHappyC                  4.442e-02  8.647e-03  8.792e+01   5.137
## recallEventType:feelHappyC  6.231e-02  1.653e-02  6.494e+02   3.770
##                            Pr(>|t|)
## (Intercept)                0.063633 .
## zeroDay                    0.734679
## conditionC                 0.083041 .
## recallEventType             < 2e-16 ***
## feelHappyC                 1.67e-06 ***
## recallEventType:feelHappyC 0.000178 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) zeroDy cndtnC rcllET flHppC
## zeroDay     -0.625
## conditionC  -0.052 -0.036
## rcllEvntTyp -0.188  0.009  0.002
## feelHappyC   0.023 -0.049  0.008 -0.475
## rcllEvnT:HC -0.361 -0.032  0.023  0.080 -0.028
```

Modeling interactions with factors

The next model attempts to model the recall event type as factors. I did this because I was worried that my assumption that event types will has a nice step-wise relation where recalling only negative event is significantly worse than recalling both positive and negative events.

```
mod8 <- lmer(polarity ~
               zeroDay +
               conditionC+
               as.factor(recallEventType) *
               feelHappyC +
               (feelHappyC+as.factor(recallEventType)|subjectNumber),
             data=wisData,
             control = lmerControl(optimizer = 'optimx', optCtrl=list(method='L-BFGS-B')))

# Comparing the two models
anova(mod7b, mod8)
```

```
## refitting model(s) with ML (instead of REML)

## Data: wisData
## Models:
## mod7b: polarity ~ zeroDay + conditionC + recallEventType * feelHappyC +
## mod7b:     (feelHappyC + recallEventType | subjectNumber)
## mod8: polarity ~ zeroDay + conditionC + as.factor(recallEventType) *
## mod8:     feelHappyC + (feelHappyC + as.factor(recallEventType) | subjectNumber)
##       Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod7b 13 984.58 1051.6 -479.29   958.58
## mod8  19 935.02 1033.0 -448.51   897.02 61.559      6  2.169e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The comparison demonstrates that modeling event recall type as factor results in a significantly different and better model (indicated by its smaller AIC).

Simplifying the model

Since the full model does not indicate that perspective condition or days are significant variables. I decided to remove them from this model to simplify it.

```
mod9 <- lmer(polarity ~
               as.factor(recallEventType) *
               feelHappyC +
               (feelHappyC+as.factor(recallEventType)|subjectNumber),
             data=wisData,
             control = lmerControl(optimizer = 'optimx', optCtrl=list(method='L-BFGS-B')))

# Comparing simple model with the full model
anova(mod8,mod9)
```

```
## refitting model(s) with ML (instead of REML)

## Data: wisData
## Models:
## mod9: polarity ~ as.factor(recallEventType) * feelHappyC + (feelHappyC +
## mod9:     as.factor(recallEventType) | subjectNumber)
## mod8: polarity ~ zeroDay + conditionC + as.factor(recallEventType) *
## mod8:     feelHappyC + (feelHappyC + as.factor(recallEventType) | subjectNumber)
##      Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod9 17 934.44 1022.1 -450.22   900.44
## mod8 19 935.02 1033.0 -448.51   897.02 3.4227      2     0.1806
```

The comparison shows that the simple model is not significantly different from the more complicated full model. In addition, it boasts a marginally smaller AIC and BIC, but it does lose two units of degrees of freedom.

Visualization of model

These codes are those used to create the figures on the main text.

Figure 1.

This code plots the interaction between the two fixed effects

```
sjPlot::plot_model(mod7b, "int")
```

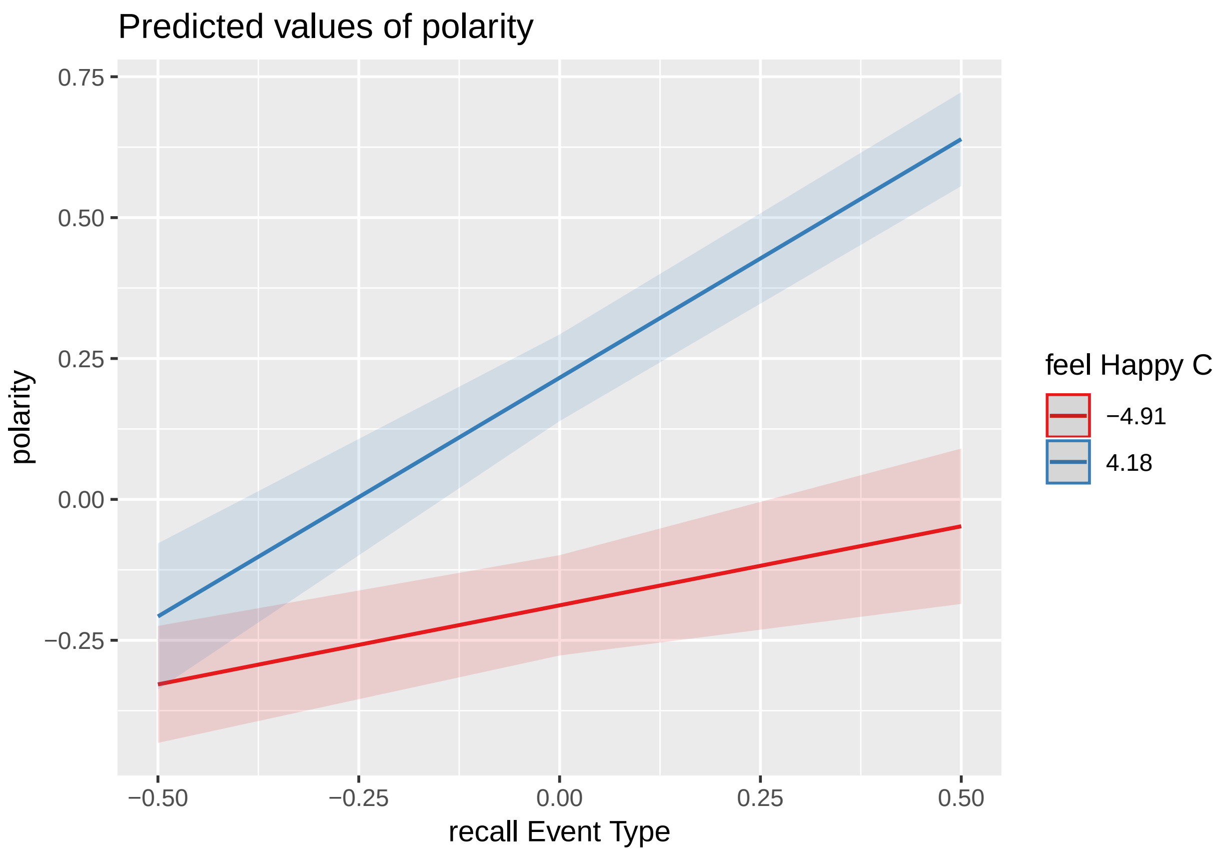


Figure 2.

This code plots the bar graph with the perspective condition side by side with the recall event type as the x-axis and polarity as the y-axis and plots a linear regression line for the perspective conditions.

```r
# This data.table, square bracket, syntax includes rows where the recall event not is NA
wisData[!is.na(wisData$recallEventType),] %>%
  ggplot(aes(y=polarity, x=as.factor(recallEventType),color=as.factor(conditionC)))+
  geom_boxplot()+
      # se=FALSE command removes the standard error shading
  geom_smooth(aes(group=as.factor(conditionC)), method = "lm",se=FALSE)+
      # pos = "top" command places the color legend above the plot
  legend_style(pos = "top")+
      # command below changes the label of the color legend
  labs(color="Perspective Condition")+
      # command below changes the label of the x-axis factors
  scale_x_discrete(labels=c("Negative","Mixed","Positive"))+
      # commands below re-names the axis labels
  xlab("Recall Event Type")+
  ylab("Polarity")
```

```
## Warning: Removed 24 rows containing non-finite values (stat_boxplot).

## Warning: Removed 24 rows containing non-finite values (stat_smooth).
```

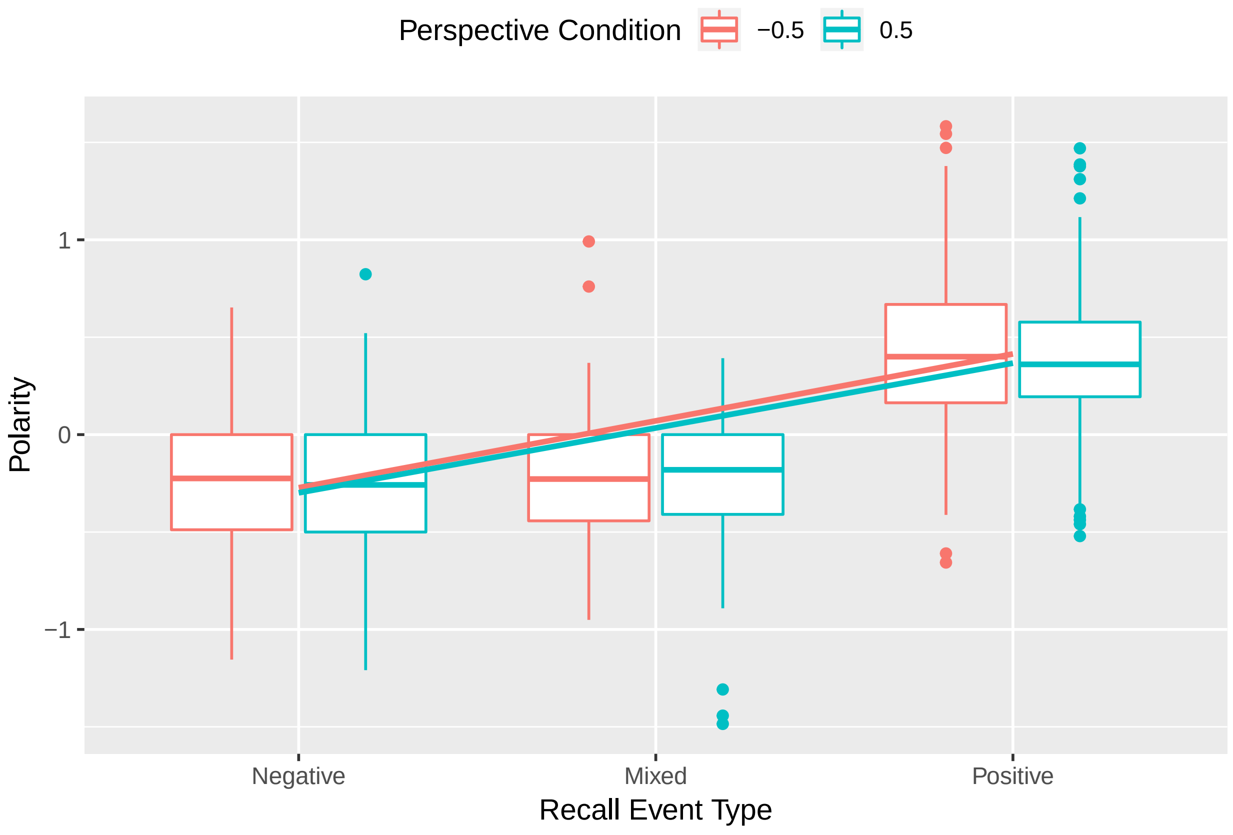


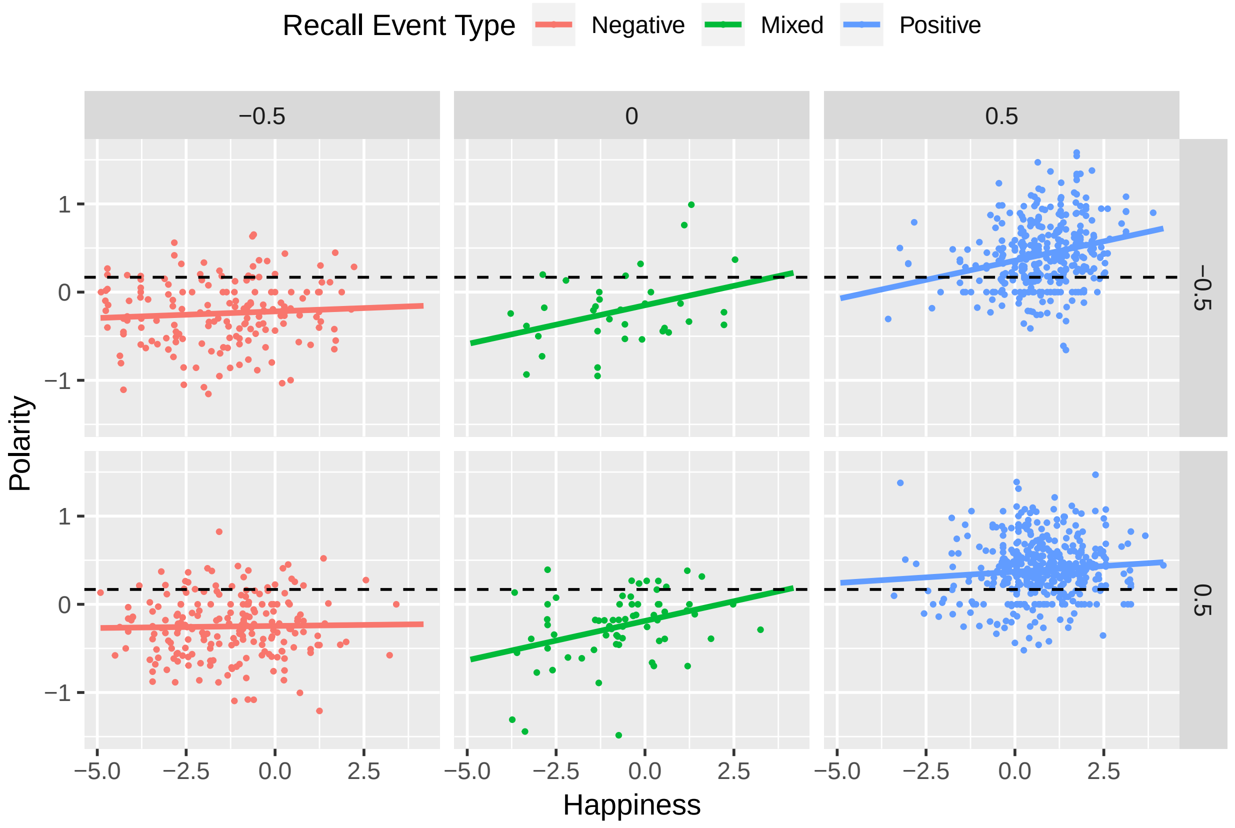
Figure 3

This code plots a scatter plot with happiness as the x-axis, polarity as the y-axis. It also divides horizontally so the top row represents the immersed condition and the bottom row represents the distanced condition. And it is divided vertically so that each column represents the respective recall type.

```r
# This data.table, square bracket, syntax include rows where the recall event is not NA
wisData[!is.na(wisData$recallEventType),] %>%
  ggplot(aes(y=polarity, x=feelHappyC, color = as.factor(recallEventType)))+
  geom_point(size= 0.5)+
      # fullrange=TRUE command extends the regression line
  geom_smooth(aes(group=as.factor(recallEventType)),method = "lm",se=FALSE,fullrange=TRUE)+
      # adding a dashed line representing the mean polarity
  geom_hline(yintercept=summary(wisData$polarity)[[4]],linetype = "dashed") +
      # splitting the plot by perspective and recall event type
  facet_grid(conditionC~recallEventType)+
  legend_style(pos = "top")+
  labs(color="Recall Event Type")+
  scale_color_discrete(labels=c("Negative","Mixed","Positive"))+
  xlab("Happiness")+
  ylab("Polarity")
```

```
## Warning: Removed 32 rows containing non-finite values (stat_smooth).

## Warning: Removed 32 rows containing missing values (geom_point).
```



Useful packages and functions

Here I will demonstrate some of the useful packages and functions I used to extract information from the models.

```r
# This function displays a well formatted summary of a model
# I used it to extract information about the random effects, group variables,
# and model fit. (Of concern is the discrepancy of the AIC and BIC in this output
# and others)
jtools::summ(mod9)
```

```
## MODEL INFO:
## Observations: 1281
## Dependent Variable: polarity
## Type: Mixed effects linear regression
##
## MODEL FIT:
## AIC = 971.17, BIC = 1058.81
## Pseudo-R² (fixed effects) = 0.44
## Pseudo-R² (total) = 0.54
##
## FIXED EFFECTS:
## ----------------------------------------------------------------------
##                                                  Est.   S.E.   t val.
## ---------------------------------------------- ------- ------ --------
## (Intercept)                                     -0.22   0.03    -8.68
## as.factor(recallEventType)0                      0.07   0.04     1.55
## as.factor(recallEventType)0.5                    0.58   0.03    17.29
## feelHappyC                                       0.02   0.01     1.35
## as.factor(recallEventType)0:feelHappyC           0.07   0.02     2.79
## as.factor(recallEventType)0.5:feelHappyC         0.04   0.02     2.30
## ----------------------------------------------------------------------
##
## --------------------------------------------------------------
##                                                   d.f.      p
## ---------------------------------------------- --------- ------
## (Intercept)                                     109.23   0.00
## as.factor(recallEventType)0                     331.86   0.12
## as.factor(recallEventType)0.5                   106.67   0.00
## feelHappyC                                      164.32   0.18
## as.factor(recallEventType)0:feelHappyC          544.65   0.01
## as.factor(recallEventType)0.5:feelHappyC        615.00   0.02
## --------------------------------------------------------------
##
## p values calculated using Satterthwaite d.f.
##
## RANDOM EFFECTS:
## -----------------------------------------------------------
##      Group                 Parameter             Std. Dev.
## --------------- ------------------------------- -----------
##  subjectNumber            (Intercept)              0.13
##  subjectNumber            feelHappyC               0.04
##  subjectNumber    as.factor(recallEventType)0      0.05
##  subjectNumber   as.factor(recallEventType)0.5     0.22
##    Residual                                        0.32
## -----------------------------------------------------------
##
## Grouping variables:
## ---------------------------------
##      Group       # groups   ICC
## ---------------- ---------- ------
##  subjectNumber     157      0.15
## ---------------------------------
```

```
# This function displays the standardized beta estimate, errors and confidence intervals
sjstats::std_beta(mod9)
```

```
##                                        term std.estimate  std.error
## 1              as.factor(recallEventType)0   0.03892664 0.02513857
## 2            as.factor(recallEventType)0.5   0.58692242 0.03395452
## 3                               feelHappyC   0.05604857 0.04140590
## 4   as.factor(recallEventType)0:feelHappyC   0.06954980 0.02490164
## 5 as.factor(recallEventType)0.5:feelHappyC   0.07621451 0.03308582
##      conf.low  conf.high
## 1 -0.01034405 0.08819733
## 2  0.52037279 0.65347205
## 3 -0.02510549 0.13720264
## 4  0.02074348 0.11835612
## 5  0.01136748 0.14106153
```