PS918 Modelling Assignment: Risky Choice & CPT

2138473

Appendix: Source Code

Here I specifically transform the data into several .csv files

```
library(ggplot2)
library(cowplot)
```

```
# load the data

gambles <- read.csv("gambles.csv")
outcomes <- read.csv("outcomes.csv")
```

```
# inspect the data

head(gambles)
```

```
##   ï..choicepair A1_prob A1_payoff A2_prob A2_payoff B1_prob B1_payoff B2_prob
## 1             1    0.29        88    0.71        78    0.29        53    0.71
## 2             2    0.82        26    0.18        34    0.49        18    0.51
## 3             3    0.59        31    0.41        63    0.89        92    0.11
## 4             4    0.07        51    0.93        35    0.60         9    0.40
## 5             5    0.48        13    0.52        70    0.18        72    0.82
## 6             6    0.63        41    0.37        18    0.98        56    0.02
##   B2_payoff
## 1        91
## 2        31
## 3        20
## 4        89
## 5        60
## 6         8
```

Task 1: Estimate the Parameters

Cumulative Prospect Theory: Version 1

```
# CPT version 1

cpt_1 <- function(parms){

  alpha <- parms[1]
  lambda <- parms[2]
  tau <- parms[3]



  u <- function(x){
    ifelse(sign(x) == -1, sign(x) * lambda * abs(x) ^ alpha, x ^ alpha) # ultility function
  }
```

```r
  ua1 <- u(gambles$A1_payoff)
  ua2 <- u(gambles$A2_payoff)
  ub1 <- u(gambles$B1_payoff)
  ub2 <- u(gambles$B2_payoff)

  v <- function(u1,p1,u2,p2){
    u1*p1 + u2*p2
  }

  pa1 <- gambles$A1_prob
  pa2 <- gambles$A2_prob
  pb1 <- gambles$B1_prob
  pb2 <- gambles$B2_prob

  sva <- mapply(v, ua1,pa1,ua2,pa2)
  svb <- mapply(v, ub1,pb1,ub2,pb2)

  p <- (1 + exp(1)^(-tau*(sva-svb)))^-1 # probability function

  return(p)

}
```

```r
# fit the likelihood function for CPT - version 1

ll_cpt1 <- function(parms, choice) {

  p.A <- cpt_1(parms)

  probs <- ifelse(choice == 0, p.A, 1 - p.A)

  # in case of large negative log likelihood when prob == 0
  if (any(probs == 0)) return(1e6)
  return(-sum(log(probs)))
  }
```

```r
# try sum the ll results

logRes <- ll_cpt1(parms = c(alpha = 0.5, lambda = 2, tau = 0.5), choice = outcomes[3])

logRes
```

```
## [1] 123.7427
```

```r
# try some fitting

set.seed(88)

get_start_parms <- function(){

  c(
    alpha = runif(1, 0, 1),
    lambda = runif(1, 1, 10),
    tau = runif(1, 0, 5)
    )
```

```r
}

# use nlminb for optimisation
solution <- with(gambles, nlminb(get_start_parms(),
                                  ll_cpt1, choice = outcomes[3], lower = c(0, 1, 0)))

solution
```

```
## $par
##     alpha    lambda       tau
## 0.7204035 1.0000000 0.2221977
##
## $objective
## [1] 110.9295
##
## $convergence
## [1] 0
##
## $iterations
## [1] 27
##
## $evaluations
## function gradient
##       41       92
##
## $message
## [1] "relative convergence (4)"
```

```r
# flip the outcomes dataframe
outcomes_rev <- as.data.frame(t(outcomes))

names(outcomes_rev) <- outcomes_rev[1,]
outcomes_rev <- outcomes_rev[-1,]

outcomes_rev <- cbind(subjects = c(1:30), outcomes_rev)
rownames(outcomes_rev) <- as.factor(1:nrow(outcomes_rev))
```

```r
# multiple fitting with all 30 subjects

multifits1 <- do.call(rbind, lapply(1:30, function(y){

  # subsetting for every individual subject
  dtsub <- subset(outcomes_rev, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {
    # replicate the estimation for 5 times
    resI <- nlminb(get_start_parms(), ll_cpt1, choice = dtsub[2:181],
                   lower = c(0, 1, 0), upper = c(1, 10, 5))
                  # restrict the upper and lower bounds
    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)

    return(myres)
  })
```

```r
  res <- as.data.frame(t(res))

  # check if is empirical identifiable
  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  # remove the non-identifiable values
  mle2[, 1:3][abs(mle[, 1:3] - res[which_max[1], 1:3]) > 0.05] <- NA

  mle2

}))
```

```r
print(multifits1, row.names=FALSE)
```

```
##         alpha    lambda        tau     logLik convergence
##  0.7639738072  1.433378 0.37677187  -77.55118           0
##  0.7204033201  1.000000 0.22219783 -110.92949           0
##  0.0000000000        NA         NA -124.40860           0
##  0.2018785405 10.000000 0.04692736 -124.23226           0
##  1.0000000000  1.306770 0.06671877 -104.49288           0
##  0.9988013117  1.000000 0.14541676  -80.95304           0
##  0.8055479582  1.000000 0.27780922  -91.66223           0
##  0.9490341834  1.052095 0.15266064  -87.79810           0
##  0.8799392264  1.338184 0.18853552  -86.81109           0
##  0.0000000000  2.299076 0.42258693 -123.71802           0
##  0.8822126807  1.141587 0.29143970  -71.61531           0
##  0.0008318195        NA 0.00000000 -124.76649           0
##  0.8423711399  1.000000 0.31968148  -78.60950           0
##  0.1425543020  1.000000 0.49477298 -123.72817           0
##  0.5820450876  1.571863 1.06086204  -64.96462           0
##  0.0000000000        NA 0.00000000 -124.76649           0
##  0.0000000000        NA 0.00000000 -124.76649           0
##  0.8108142586  1.000000 0.29029970  -88.91326           0
##  1.0000000000  2.306144 0.03682532 -109.30441           0
##  1.0000000000  1.380694 0.05595248 -108.40853           0
##  0.9024683561  1.000000 0.07866512 -114.63574           0
##  1.0000000000  1.000000 0.06967292 -107.10747           0
##  1.0000000000  1.000000 0.05653522 -112.03406           0
##  0.9292943804  1.439396 0.08689039 -104.82483           0
##  1.0000000000  1.000000 0.08319428 -101.94894           0
##  1.0000000000  1.518365 0.07456964  -97.77183           0
##  0.6058113534  6.333563 0.10650236 -104.66085           0
##  1.0000000000  1.195239 0.13191585  -80.97480           0
##  0.9510353899  1.000000 0.24295069  -68.58372           0
##  0.0000000000  4.243250 0.00000000 -124.76649           0
```

```r
# simulator

decision.generator <- function(probability){
```

```r
  r.prob <- runif(1, min = 0, max = 1)
  choice <- ifelse(probability <= r.prob, 1, 0)

  return(choice)
}
```

```r
# simulations

alpha.start <- median(multifits1$alpha, na.rm = T)
lambda.start <- median(multifits1$lambda, na.rm = T)
tau.start <- median(multifits1$tau, na.rm = T)
# I use the median of estimated optimal parameters of 30 subjects

prob1 <- cpt_1(parms = c(alpha = alpha.start,
                         lambda = lambda.start,
                         tau = tau.start))
simulations1 <- as.data.frame(t(replicate(n = 30,
                                          sapply(X = prob1,
                                                 FUN = decision.generator))))
simulations1 <- cbind(subjects = c(1:30), simulations1)
```

```r
# parameters recovery

multifits1_r <- do.call(rbind, lapply(1:30, function(y){

  dtsub <- subset(simulations1, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {

    resI <- nlminb(get_start_parms(), ll_cpt1, choice = dtsub[2:181],
                   lower = c(0, 1, 0), upper = c(1, 10, 5))

    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)

    return(myres)
  })

  res <- as.data.frame(t(res))

  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  mle2[, 1:3][abs(mle[, 1:3] - res[which_max[1], 1:3]) > 0.01] <- NA

  mle2

}))
```

```r
print(multifits1_r, row.names=FALSE)
```

```
##      alpha   lambda        tau     logLik convergence
##  0.7229595 1.198677 0.17786970 -113.37035           0
```

```
##  0.7819907 1.757500 0.09460128 -115.26872          0
##  1.0000000 1.000000 0.03913818 -117.98094          0
##  0.7439143 1.000000 0.25608044 -104.42662          0
##  0.0000000 1.000000 1.80168115 -119.00558          0
##  0.8597838 1.000000 0.08640870 -116.37002          0
##  0.7757844 1.000000 0.14466217 -114.41034          0
##  0.7590900 1.000000 0.20381704 -108.77322          0
##  1.0000000 1.076713 0.05607404 -111.51903          0
##  0.8695330 1.881331 0.05565092 -116.38762          0
##  0.8381641 1.063451 0.10304731 -114.75177          0
##  0.0000000       NA         NA -120.57997          0
##  0.0000000 1.000000 1.12588200 -122.19396          0
##  0.7954429 1.013737 0.20278296 -104.11121          0
##  0.8620367 1.592001 0.04777457 -120.02084          0
##  0.7381517 1.012107 0.19273568 -112.12776          0
##  0.0000000       NA         NA -123.73192          0
##  1.0000000 1.339051 0.04363341 -114.17007          0
##  0.8780116 1.202897 0.06034807 -118.71383          0
##  0.0000000       NA         NA -123.90379          0
##  0.0000000       NA         NA -123.57376          0
##  0.6773114 1.511285 0.19325176 -112.89797          0
##  1.0000000 1.000000 0.04819054 -115.00690          0
##  0.0000000       NA         NA -124.56932          0
##  0.8270516 1.138956 0.13761516 -109.10796          0
##  0.0000000       NA         NA -120.87084          0
##  0.7598925 1.239921 0.28777400  -93.87643          0
##  1.0000000 1.104195 0.04908223 -113.91957          0
##  0.9025193 1.735387 0.06852969 -110.84684          0
##  0.0000000 3.404853 0.24324255 -124.13687          0
```

```r
# recovered parameters

alpha.rec <- median(multifits1_r$alpha, na.rm = T)
lambda.rec <- median(multifits1_r$lambda, na.rm = T)
tau.rec <- median(multifits1_r$tau, na.rm = T)

alpha.rec
```

```
## [1] 0.7678384
```

```r
lambda.rec
```

```
## [1] 1.090454
```

```r
tau.rec
```

```
## [1] 0.1203312
```

Cumulative Prospect Theory: Version 2

```r
# CPT version 2

cpt_2 <- function(parms){

  alpha <- parms[1]
  lambda <- parms[2]
  tau <- parms[3]
  beta <- parms[4]
```

```r
  u <- function(x){
    ifelse(sign(x) == -1, sign(x) * lambda * abs(x) ^ beta, x ^ alpha)
  }

  ua1 <- u(gambles$A1_payoff)
  ua2 <- u(gambles$A2_payoff)
  ub1 <- u(gambles$B1_payoff)
  ub2 <- u(gambles$B2_payoff)

  v <- function(u1,p1,u2,p2){
    u1*p1 + u2*p2
  }

  pa1 <- gambles$A1_prob
  pa2 <- gambles$A2_prob
  pb1 <- gambles$B1_prob
  pb2 <- gambles$B2_prob

  sva <- mapply(v, ua1,pa1,ua2,pa2)
  svb <- mapply(v, ub1,pb1,ub2,pb2)

  p <- (1 + exp(1)^(-tau*(sva-svb)))^-1

  return(p)

}
```

```r
# fit the likelihood function for CPT - version 2

ll_cpt2 <- function(parms, choice) {

  p.A <- cpt_2(parms)

  probs <- ifelse(choice == 0, p.A, 1 - p.A)

  if (any(probs == 0)) return(1e6)
  return(-sum(log(probs)))
  }
```

```r
# try sum the ll results

logRes2 <- ll_cpt2(parms = c(alpha = 0.5, lambda = 2,
                            tau = 0.5, beta = 0.8), choice = outcomes[3])

logRes2
```

```
## [1] 260.5062
```

```r
# try some fitting

set.seed(3)

get_start_parms_2 <- function(){

  c(
```

```r
    alpha = runif(1, 0, 1),
    lambda = runif(1, 1, 10),
    tau = runif(1, 0, 5),
    beta = runif(1, 0, 1)
    )

}

solution2 <- with(gambles, nlminb(get_start_parms_2(),
                                  ll_cpt2,
                                  choice = outcomes[4],
                                  lower = c(0, 1, 0, 0)))

solution2
```

```
## $par
##     alpha    lambda       tau      beta
## 0.9029240 1.0000000 0.1973634 0.9269873
##
## $objective
## [1] 84.8792
##
## $convergence
## [1] 0
##
## $iterations
## [1] 42
##
## $evaluations
## function gradient
##       69      189
##
## $message
## [1] "relative convergence (4)"
```

```r
# multiple fitting with all 30 subjects

multifits2 <- do.call(rbind, lapply(1:30, function(y){

  dtsub <- subset(outcomes_rev, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {

    resI <- nlminb(get_start_parms_2(), ll_cpt2, choice = dtsub[2:181],
                   lower = c(0, 1, 0, 0), upper = c(1, 10, 5, 1))

    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)

    return(myres)
  })

  res <- as.data.frame(t(res))

  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
```

```r
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  mle2[, 1:4][abs(mle[, 1:4] - res[which_max[1], 1:4]) > 0.01] <- NA

  mle2

}))
```

```r
print(multifits2, row.names=FALSE)
```

```
##      alpha    lambda        tau      beta        logLik convergence
##  0.8438639  2.794600 0.25253500 0.7106283 -7.742095e+01           0
##  0.6666357  1.000000 0.37084268 0.5263867 -1.090680e+02           0
##  0.9029241  1.000000 0.19736331 0.9269874 -8.487920e+01           0
##  0.0047883 10.000000 0.04962034 0.1913765 -1.242293e+02           0
##  1.0000000  1.306771 0.06671877 1.0000000 -1.044929e+02           0
##  1.0000000  1.000000 0.16783072 0.9384222 -7.904579e+01           0
##  0.8898850  2.689813 0.29435398 0.5313303 -8.255373e+01           0
##  0.9457845  1.000000 0.15219082 0.9660299 -8.767121e+01           0
##  0.8438257  1.000000 0.22262875 0.9116112 -8.630243e+01           0
##  0.8684348  1.000000 0.21662966 0.9130216 -8.547053e+01           0
##  0.8679748  1.000000 0.31042660 0.9003328 -7.138194e+01           0
##  1.0000000  1.000000 0.03539875 0.8426066 -1.204744e+02           0
##  0.8424992  1.000000 0.31662904 0.8467031 -7.859939e+01           0
##  0.2128146  1.000000 0.57106429 0.0000000 -1.235344e+02           0
##  0.5298985  1.000000 1.39951196 0.6178008 -6.456148e+01           0
##  0.6227399  1.000000 0.05642197 0.0000000 -1.244548e+02           0
##  1.0000000  1.000000 0.14489319 0.9153823 -8.654180e+01           0
##  0.9476953  3.598867 0.16530519 0.6453887 -8.762822e+01           0
##  0.8323374  1.000000 0.08601438 1.0000000 -1.088535e+02           0
##  1.0000000  1.000000 0.03171212 0.0000000 -1.210326e+02           0
##  0.8950535  1.000000 0.09708500 0.8021302 -1.138156e+02           0
##  1.0000000  1.000000 0.09129001 0.8676057 -1.042486e+02           0
##  0.0000000  5.053041 0.00000000 0.0000000 -1.247665e+02           0
##  0.8848152  1.000000 0.10618896 0.9697656 -1.044516e+02           0
##  1.0000000  1.000000 0.08418906 0.9942356 -1.019408e+02           0
##  0.9108578  1.000000 0.11504774 1.0000000 -9.735034e+01           0
##  0.7315031 10.000000 0.06953180 0.6011951 -1.045205e+02           0
##  1.0000000  1.195239 0.13191583 1.0000000 -8.097480e+01           0
##  1.0000000  1.848267 0.24368593 0.7898407 -6.362696e+01           0
##         NA        NA         NA        NA -1.000000e+06           0
```

```r
# simulations

alpha.start2 <- median(multifits2$alpha, na.rm = T)
lambda.start2 <- median(multifits2$lambda, na.rm = T)
tau.start2 <- median(multifits2$tau, na.rm = T)
beta.start2 <- median(multifits2$beta, na.rm = T)

prob2 <- cpt_2(parms = c(alpha = alpha.start2,
                         lambda = lambda.start2,
```

```r
                          tau = tau.start2,
                          beta = beta.start2))
simulations2 <- as.data.frame(t(replicate(n = 30,
                                          sapply(X = prob2,
                                                 FUN = decision.generator))))
simulations2 <- cbind(subjects = c(1:30), simulations2)
```

```r
# parameters recovery

multifits2_r <- do.call(rbind, lapply(1:30, function(y){

  dtsub <- subset(simulations2, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {

    resI <- nlminb(get_start_parms_2(), ll_cpt2, choice = dtsub[2:181],
                   lower = c(0, 1, 0, 0), upper = c(1, 10, 5, 1))

    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)

    return(myres)
  })

  res <- as.data.frame(t(res))

  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  mle2[, 1:3][abs(mle[, 1:3] - res[which_max[1], 1:3]) > 0.01] <- NA

  mle2

}))
```

```r
print(multifits2_r, row.names=FALSE)
```

```
##      alpha    lambda        tau      beta        logLik convergence
##  0.8695103  1.000000 0.16800795 0.8483410 -1.006406e+02           0
##  1.0000000  3.867835 0.10837361 0.6814712 -9.458403e+01           0
##  1.0000000  1.080233 0.06778337 1.0000000 -1.069124e+02           0
##  1.0000000  1.000000 0.10258403 0.9810378 -9.610193e+01           0
##  0.8277767  1.000000 0.18232490 0.6844350 -1.079804e+02           0
##  1.0000000 10.000000 0.07725802 0.4370710 -1.066564e+02           0
##  1.0000000  1.000000 0.09531807 0.9643801 -9.964663e+01           0
##  0.8766738  1.169106 0.12404087 0.8839388 -1.047575e+02           0
##  1.0000000  1.866518 0.07595525 0.7454131 -1.088178e+02           0
##  0.0000000 10.000000 0.00741835 1.0000000 -1.124725e+02           0
##  1.0000000  3.363097 0.10240003 0.7434864 -9.445350e+01           0
##  1.0000000  1.000000 0.11025933 0.9665756 -9.438662e+01           0
##  0.8118822  1.242568 0.22445895 0.6639791 -1.034568e+02           0
##  1.0000000  1.000000 0.05172105 0.9788952 -1.144573e+02           0
```

```
##          NA         NA          NA 0.8824427 -1.000000e+06           0
##   0.8556108  1.000000 0.19862242 0.6435721 -1.029304e+02           0
##   0.0000000  1.000000 0.03715463 1.0000000 -1.209789e+02           0
##   1.0000000  1.000000 0.07280721 0.9756878 -1.071399e+02           0
##   0.8642219  1.000000 0.12349281 0.8581297 -1.093316e+02           0
##   0.9920888 10.000000 0.07190038 0.4782288 -1.076998e+02           0
##   0.8378248  1.000000 0.23657877 0.8144242 -9.391630e+01           0
##   1.0000000  1.000000 0.06502020 0.9796035 -1.097873e+02           0
##   0.0000000  2.292116 0.78256432 0.0000000 -1.214943e+02           0
##   0.8754173  2.231748 0.16813245 0.6346504 -1.018135e+02           0
##   1.0000000  7.927300 0.09653129 0.5298607 -9.829760e+01           0
##   1.0000000  1.000000 0.06605390 0.9418944 -1.107001e+02           0
##   1.0000000  1.000000 0.07202600 0.9651588 -1.078649e+02           0
##   0.8728599  1.000000 0.23156620 0.7517333 -9.354491e+01           0
##   1.0000000  1.000000 0.11647081 0.9930192 -9.043924e+01           0
##   0.7253204  1.000000 0.29353849 0.6915760 -1.048290e+02           0
```

Cumulative Prospect Theory: Version 3

```r
# CPT version 3

cpt_3 <- function(parms){

  alpha <- parms[1]
  lambda <- parms[2]
  tau <- parms[3]
  beta <- parms[4]
  gamma <- parms[5]

  u <- function(x){
    ifelse(sign(x) == -1, sign(x) * lambda * abs(x) ^ beta, x ^ alpha)
  }

  ua1 <- u(gambles$A1_payoff)
  ua2 <- u(gambles$A2_payoff)
  ub1 <- u(gambles$B1_payoff)
  ub2 <- u(gambles$B2_payoff)

  p <- function(x){
    x ^ gamma
  }

  pa1 <- p(gambles$A1_prob)
  pa2 <- p(gambles$A2_prob)
  pb1 <- p(gambles$B1_prob)
  pb2 <- p(gambles$B2_prob)

  v <- function(u1,p1,u2,p2){
    u1*p1 + u2*p2
  }

  sva <- mapply(v, ua1,pa1,ua2,pa2)
  svb <- mapply(v, ub1,pb1,ub2,pb2)

  p <- (1 + exp(1)^(-tau*(sva-svb)))^-1
```

```r
  return(p)

}
```

```r
# fit the likelihood function for CPT - version 3

ll_cpt3 <- function(parms, choice) {

  p.A <- cpt_3(parms)

  probs <- ifelse(choice == 0, p.A, 1 - p.A)

  if (any(probs == 0)) return(1e6)
  return(-sum(log(probs)))
  }
```

```r
# try sum the ll results

logRes3 <- ll_cpt3(parms = c(alpha = 0.5, lambda = 2,
                             tau = 0.5, beta = 0.8,
                             gamma = 0.7), choice = outcomes[3])

logRes3
```

```
## [1] 281.5448
```

```r
# try some fitting

set.seed(88)

get_start_parms_3 <- function(){

  c(
    alpha = runif(1, 0, 1),
    lambda = runif(1, 1, 10),
    tau = runif(1, 0, 5),
    beta = runif(1, 0, 1),
    gamma = runif(1, 0, 1)
    )

}

solution3 <- nlminb(get_start_parms_3(), ll_cpt3,
                    choice = outcomes[3], lower = c(0, 1, 0, 0, 0))

solution3
```

```
## $par
##     alpha    lambda       tau      beta     gamma
## 0.5733933 1.0000000 0.5824589 0.0000000 1.3256074
##
## $objective
## [1] 109.6906
##
## $convergence
```

```
## [1] 0
##
## $iterations
## [1] 16
##
## $evaluations
## function gradient
##       20       94
##
## $message
## [1] "relative convergence (4)"
```

```r
# multiple fitting with all 30 subjects

multifits3 <- do.call(rbind, lapply(1:30, function(y){

  dtsub <- subset(outcomes_rev, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {

    resI <- nlminb(get_start_parms_3(), ll_cpt3, choice = dtsub[2:181],
                   lower = c(0, 1, 0, 0, 0), upper = c(1, 10, 5, 1, 1))

    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)

    return(myres)
  })

  res <- as.data.frame(t(res))

  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  mle2[, 1:5][abs(mle[, 1:5] - res[which_max[1], 1:5]) > 0.01] <- NA

  mle2

}))
```

```r
print(multifits3, row.names=FALSE)
```

```
##       alpha   lambda        tau       beta        gamma     logLik convergence
##  0.00000000 1.000000 0.78146848 0.60288435 1.000000e+00 -100.14879           0
##  0.66142697 1.000000 0.37694760 0.54345176 9.134575e-01 -108.96199           0
##  0.81678329 1.734649 0.36104388 0.73828887 6.633113e-01  -71.52458           0
##  0.02846925 1.000000 1.94045637 0.04477896 6.877771e-02 -121.62149           0
##  1.00000000 1.376562 0.07385245 1.00000000 4.767556e-01  -91.31470           0
##  1.00000000 1.000000 0.16783071 0.93842225 1.000000e+00  -79.04579           0
##  0.88995339 3.250996 0.28470979 0.50978895 9.272733e-01  -82.29830           0
##  0.89499695 1.000000 0.19239890 0.92269625 8.858359e-01  -86.85257           0
##  0.72168073 1.000000 0.49991623 0.77226545 6.977206e-01  -76.73365           0
##  0.84961406 1.000000 0.23745291 0.89406916 9.515565e-01  -85.32770           0
```

```
## 0.86797485  1.000000 0.31042657 0.90033286 1.000000e+00  -71.38194           0
## 0.35157388 10.000000 0.22035481 0.02071084 0.000000e+00 -122.43762           0
## 0.81345398  1.000000 0.36599131 0.82025856 9.015521e-01  -77.86732           0
## 0.10449485  1.000000 1.64599870 0.00000000 1.966975e-01 -120.65455           0
## 0.52989851  1.000000 1.39951200 0.61780084 1.000000e+00  -64.56148           0
## 0.35858122  1.000000 0.37490993 0.00000000 2.513947e-01 -122.55584           0
## 1.00000000  1.000000 0.14192754 0.92385129 9.288495e-01  -86.23553           0
## 0.91915258  3.855042 0.18768112 0.61066537 9.018371e-01  -87.00078           0
## 0.78879406  1.000000 0.12871112 0.95099082 3.260596e-01  -90.77299           0
## 0.92668656  1.135000 0.08728074 1.00000000 2.088285e-01  -81.65155           0
## 0.76725052  1.000000 0.17988870 0.69751542 4.386876e-01 -107.30256           0
## 1.00000000  1.401171 0.08796728 0.81994329 7.948135e-01 -103.16482           0
## 0.93861845 10.000000 0.08933295 0.50607072 1.797212e-01  -82.67805           0
## 0.73103542  1.000000 0.29746333 0.78755039 5.076624e-01  -90.59463           0
## 1.00000000  1.839102 0.08418128 0.88278460 7.348191e-01  -98.82203           0
## 0.00000000  1.000000 0.02063830 1.00000000 5.138662e-07 -118.88802           0
## 0.73150305 10.000000 0.06953187 0.60119489 1.000000e+00 -104.52045           0
## 0.96590460  1.040444 0.15631299 1.00000000 8.497430e-01  -79.13440           0
## 1.00000000  2.439288 0.23700197 0.74334502 9.274020e-01  -63.09497           0
## 1.00000000  1.000000 0.08684431 1.00000000 5.325863e-01  -92.69294           0
```

```r
# simulations

alpha.start3 <- median(multifits3$alpha, na.rm = T)
lambda.start3 <- median(multifits3$lambda, na.rm = T)
tau.start3 <- median(multifits3$tau, na.rm = T)
beta.start3 <- median(multifits3$beta, na.rm = T)
gamma.start3 <- median(multifits3$gamma, na.rm = T)

prob3 <- cpt_3(parms = c(alpha = alpha.start3,
                         lambda = lambda.start3,
                         tau = tau.start3,
                         beta = beta.start3,
                         gamma = gamma.start3))

#set.seed(33)

simulations3 <- as.data.frame(t(replicate(n = 30,
                                          sapply(X = prob3,
                                                 FUN = decision.generator))))

simulations3 <- cbind(subjects = c(1:30), simulations3)
```

```r
# parameters recovery

multifits3_r <- do.call(rbind, lapply(1:30, function(y){

  dtsub <- subset(simulations3, subjects==y)

  res <- replicate(n = 5, simplify = TRUE, {

    resI <- nlminb(get_start_parms_3(), ll_cpt3, choice = dtsub[2:181],
                   lower = c(0, 1, 0, 0, 0), upper = c(1, 10, 5, 1, 1))

    myres <- c(resI$par, logLik = -resI$objective, convergence = resI$convergence)
```

```r
    return(myres)
  })

  res <- as.data.frame(t(res))

  which_max <- which(round(max(res$logLik), 3) == round(res$logLik, 3))
  which_max <- which_max[which_max != which.max(res$logLik)]

  mle <- res[which.max(res$logLik),]

  mle2 <- mle
  mle2[, 1:3][abs(mle[, 1:3] - res[which_max[1], 1:3]) > 0.01] <- NA

  mle2

}))
```

```r
print(multifits3_r, row.names=FALSE)
```

```
##      alpha   lambda        tau      beta     gamma     logLik convergence
##  0.7067825 1.000000 0.42835953 0.6648776 0.7624774  -94.92716           0
##  0.8400709 1.000000 0.17747038 0.7932980 0.8741679 -104.37493           0
##  1.0000000 3.281562 0.09872317 0.7023360 0.6748874  -95.16836           0
##  0.8368545 1.000000 0.17383035 0.6954097 0.8077687 -107.16412           0
##  0.8670184 1.000000 0.24389161 0.8037744 0.7358642  -86.78331           0
##  1.0000000 2.611499 0.07501748 0.7964539 0.8302572 -104.52730           0
##  0.8277464 1.220356 0.22393264 0.7684845 0.7730583  -96.21616           0
##  0.7382158 1.000000 0.36121078 0.6424822 0.7408373  -97.60605           0
##  0.7357802 1.000000 0.28902539 0.6960856 0.7718634 -103.76018           0
##  0.7185709 1.000000 0.43386905 0.6762844 0.6891598  -91.02772           0
##  1.0000000 1.000000 0.08130630 0.9972539 0.8034009 -101.65015           0
##  0.8282663 1.000000 0.18087607 0.7242897 0.9063580 -107.25665           0
##  0.7794174 1.000000 0.21763296 0.7334868 0.8698337 -106.77161           0
##  0.7110176 1.000000 0.31418172 0.6746038 0.6694133 -103.44165           0
##  0.9216057 1.000000 0.17949206 0.8261141 0.8473537  -91.82902           0
##  0.8174649 1.000000 0.22070227 0.8070188 0.7444488  -97.79760           0
##  0.8891094 1.000000 0.15195146 0.8124293 0.6246970  -98.91794           0
##  0.9117054 5.125170 0.22181392 0.5254858 0.6142768  -76.28735           0
##  0.7744341 1.000000 0.21655442 0.7579772 0.6358381 -103.87371           0
##  0.7778341 1.848396 0.20880666 0.6078741 0.6700809 -106.06136           0
##  1.0000000 6.702850 0.06982286 0.6045386 0.7185099 -105.18252           0
##  0.9165020 1.000000 0.10295939 1.0000000 0.7855182  -99.77511           0
##  0.8205369 1.000000 0.21897892 0.7351082 0.8231773 -101.77251           0
##  0.8298483 1.179923 0.29479944 0.7875161 0.8813080  -85.01294           0
##  0.7753470 1.000000 0.26848642 0.7757565 0.7269903  -97.10189           0
##  0.7935023 1.000000 0.23409908 0.6824253 0.8359889 -104.50804           0
##  1.0000000 1.600155 0.09667719 0.8609293 0.9255392  -99.06568           0
##  0.6969985 1.000000 0.27842309 0.6591929 0.6853995 -109.03110           0
##  0.8201218 2.822490 0.22944070 0.5464138 0.7395700  -97.84969           0
##  0.9194517 2.700255 0.13642091 0.6225728 0.7776842 -100.98084           0
```

Task 2-4: Model Comparison, Parameter Visualisation and Model Recovery

```r
# AIC calculator

AIC <- function(lL, K){
  aic <- 2 * (K - lL)
  return(aic)
}
```

```r
# Calculating AIC values - V1

aic1 <- with(multifits1, AIC(logLik, 3 ))

aic1.mean <- mean(aic1)
aic1.mean
```

```
## [1] 209.3139
```

```r
aic1.std <- sd(aic1)
aic1.std
```

```
## [1] 38.34794
```

```r
aic1.min <- min(aic1)
aic1.min
```

```
## [1] 135.9292
```

```r
# calcucate AIC values - V2

aic2 <- with(multifits2, AIC(logLik, 4 ))

aic2.mean <- mean(aic2)
aic2.mean
```

```
## [1] 66861.59
```

```r
aic2.std <- sd(aic2)
aic2.std
```

```
## [1] 365113.1
```

```r
aic2.min <- min(aic2)
aic2.min
```

```
## [1] 135.2539
```

```r
# calculate AIC values - V3

aic3 <- with(multifits3, AIC(logLik, 5 ))

aic3.mean <- mean(aic3)
aic3.mean
```

```
## [1] 194.6561
```

```r
aic3.std <- sd(aic3)
aic3.std
```

```
## [1] 34.86966
```

```r
aic3.min <- min(aic3)
aic3.min
```

```
## [1] 136.1899
```

```r
# AIC differences for model 1 vs 2

aic.diff_13 <- aic1 - rep(194.6561, 30)

mean(aic.diff_13)
```

```
## [1] 14.65783
```

```r
# AIC differences for model 1 vs 3

aic.diff_23 <- aic2 - rep(194.6561, 30)

mean(aic.diff_23)
```

```
## [1] 66666.94
```

```r
# AIC differences for model 1 vs 3

aic.diff_3 <- aic3 - rep(194.6561, 30)

mean(aic.diff_3)
```

```
## [1] 3.52605e-05
```

```r
# LHR calculator

LHR <- function(lLs, lLg){
  lhr <- -2 * (lLs - lLg)
  return(lhr)
}
```

```r
# LRT for model 1 VS 2

lhr.1v2 <- LHR(multifits1$logLik, multifits2$logLik)

mean(lhr.1v2)
```

```
## [1] -66650.28
```

```r
sd(lhr.1v2)
```

```
## [1] 365104.3
```

```r
pchisq(mean(lhr.1v2), 1, lower.tail = F)
```

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

```r
# LRT for model 1 VS 3

lhr.1v3 <- LHR(multifits1$logLik, multifits3$logLik)

mean(lhr.1v3)
```

```
## [1] 18.65779
```

```r
sd(lhr.1v3)
```

```
## [1] 32.73328
```

```
pchisq(mean(lhr.1v3), 2, lower.tail = F)
```

```
## [1] 8.882021e-05
```

```
# LRT for model 2 VS 3

lhr.2v3 <- LHR(multifits2$logLik, multifits3$logLik)

mean(lhr.2v3)
```

```
## [1] 66668.94
```

```
sd(lhr.2v3)
```

```
## [1] 365112.9
```

```
pchisq(mean(lhr.2v3), 1, lower.tail = F)
```

```
## [1] 0
```

Plots and Graphics

```
subjects <- as.factor(1:30)

r3.alpha <- as.data.frame(cbind(subjects, alpha.diff = multifits3_r$alpha) - multifits3$alpha)
r3.lambda <- as.data.frame(cbind(subjects, lambda.diff = multifits3_r$lambda - multifits3$lambda))
r3.beta <- as.data.frame(cbind(subjects, beta.diff = multifits3_r$beta - multifits3$beta))
r3.tau <- as.data.frame(cbind(subjects, tau.diff = multifits3_r$tau - multifits3$tau))
r3.gamma <- as.data.frame(cbind(subjects, gamma.diff = multifits3_r$gamma - multifits3$gamma))
```

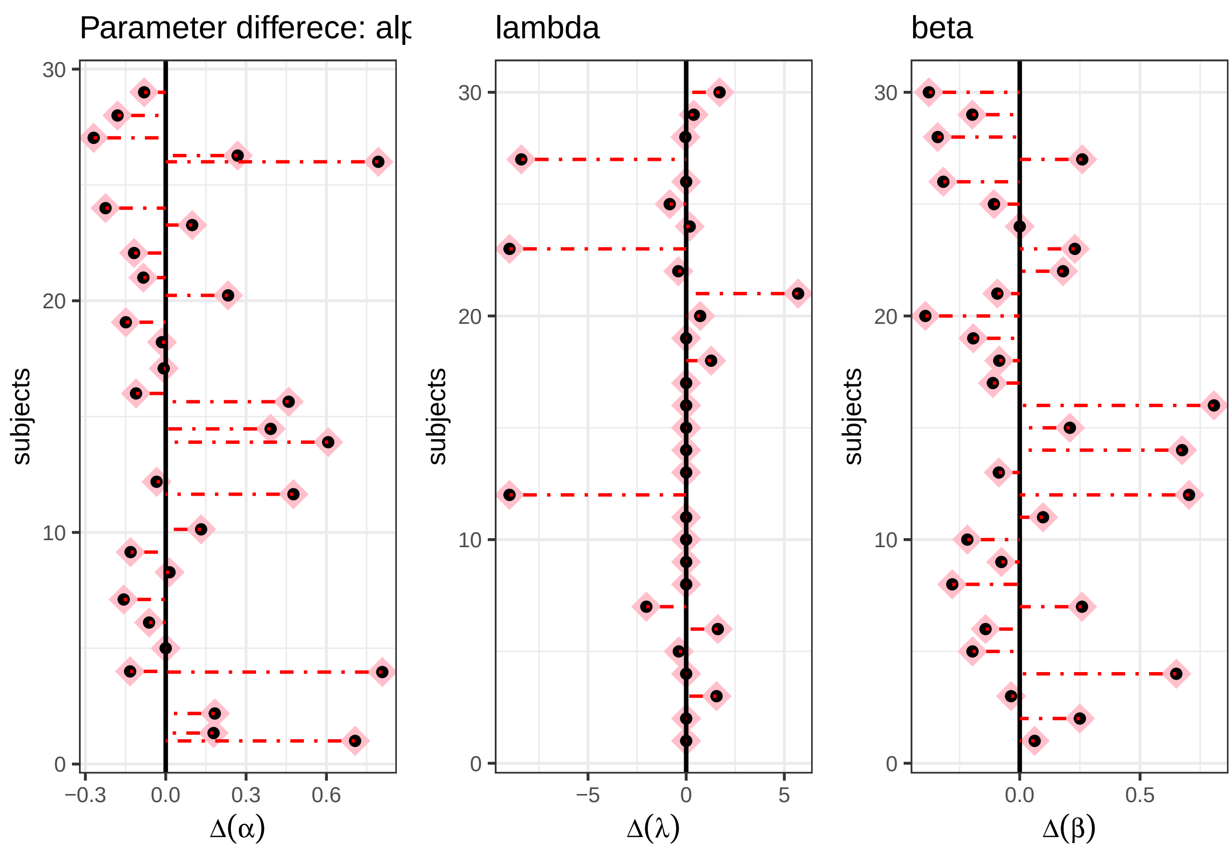
```
pa3 <- ggplot(r3.alpha, aes(x = subjects, y = alpha.diff))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_hline(yintercept = 0, size = 0.8)+
  geom_segment(aes(x = subjects, y = alpha.diff, xend = subjects, yend = 0),
               linetype = "dotdash", colour = "red", size = 0.6)+
  theme_bw()+
  theme(text=element_text(size=10)) +
  labs(x='subjects',y = expression(Delta(alpha)), title = "Parameter differece: alpha")+
  coord_flip()

pl3 <- ggplot(r3.lambda, aes(x = subjects, y = lambda.diff))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_hline(yintercept = 0, size = 0.8)+
  geom_segment(aes(x = subjects, y = lambda.diff, xend = subjects, yend = 0),
               linetype = "dotdash", colour = "red", size = 0.6)+
  theme_bw()+
  theme(text=element_text(size=10)) +
  labs(x='subjects',y = expression(Delta(lambda)), title = "lambda")+
  coord_flip()

pb3 <- ggplot(r3.beta, aes(x = subjects, y = beta.diff))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_hline(yintercept = 0, size = 0.8)+
  geom_segment(aes(x = subjects, y = beta.diff, xend = subjects, yend = 0),
```

```
                linetype = "dotdash", colour = "red", size = 0.6)+
  theme_bw()+
  theme(text=element_text(size=10)) +
  labs(x='subjects',y = expression(Delta(beta)), title = "beta")+
  coord_flip()

palb.rec.3 <- plot_grid(pa3, pl3, pb3, nrow = 1)
palb.rec.3
```

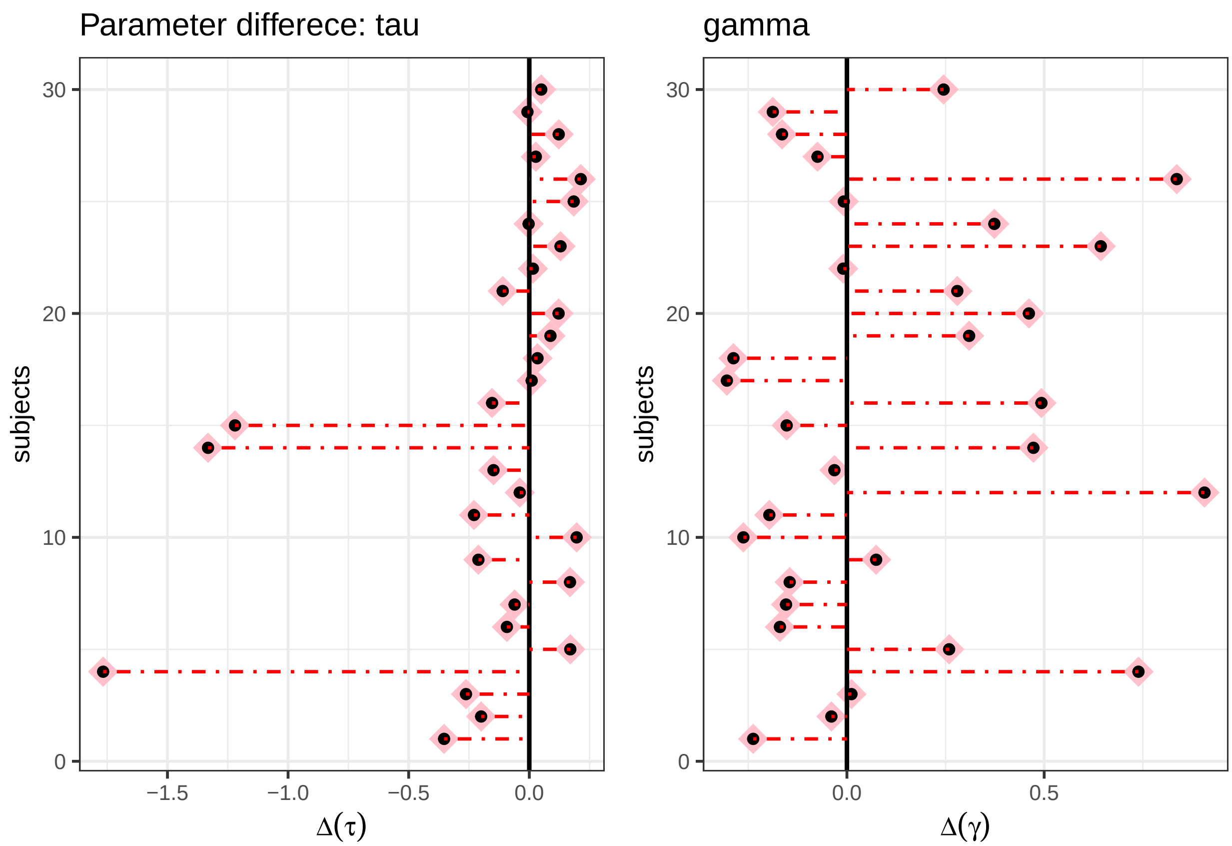



```
pt3 <- ggplot(r3.tau, aes(x = subjects, y = tau.diff))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_hline(yintercept = 0, size = 0.8)+
  geom_segment(aes(x = subjects, y = tau.diff, xend = subjects, yend = 0),
                linetype = "dotdash", colour = "red", size = 0.6)+
  theme_bw()+
  theme(text=element_text(size=10)) +
  labs(x='subjects',y = expression(Delta(tau)), title = "Parameter differece: tau")+
  coord_flip()

pg3 <- ggplot(r3.gamma, aes(x = subjects, y = gamma.diff))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_hline(yintercept = 0, size = 0.8)+
  geom_segment(aes(x = subjects, y = gamma.diff, xend = subjects, yend = 0),
                linetype = "dotdash", colour = "red", size = 0.6)+
```

```
  theme_bw()+
  theme(text=element_text(size=10)) +
  labs(x='subjects',y = expression(Delta(gamma)), title = "gamma")+
  coord_flip()

ptg.rec.3 <- plot_grid(pt3, pg3, nrow = 1)
ptg.rec.3
```




```
# plot alpha and beta against gamma in version 3

p1 <- ggplot(data = NULL, mapping = aes(x = multifits3$alpha, y = multifits3$gamma))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_smooth(method = 'lm', colour = "gray") +
  theme_bw() +
  theme(text=element_text(size=12)) +
  labs(x=expression(alpha),y=expression(gamma), title = "Scatter plot of parameter pairs")



p2 <- ggplot(data = NULL, mapping = aes(x = multifits3$beta, y = multifits3$gamma))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_smooth(method = 'lm', colour = "gray") +
  theme_bw() +
  theme(text=element_text(size=12)) +
```

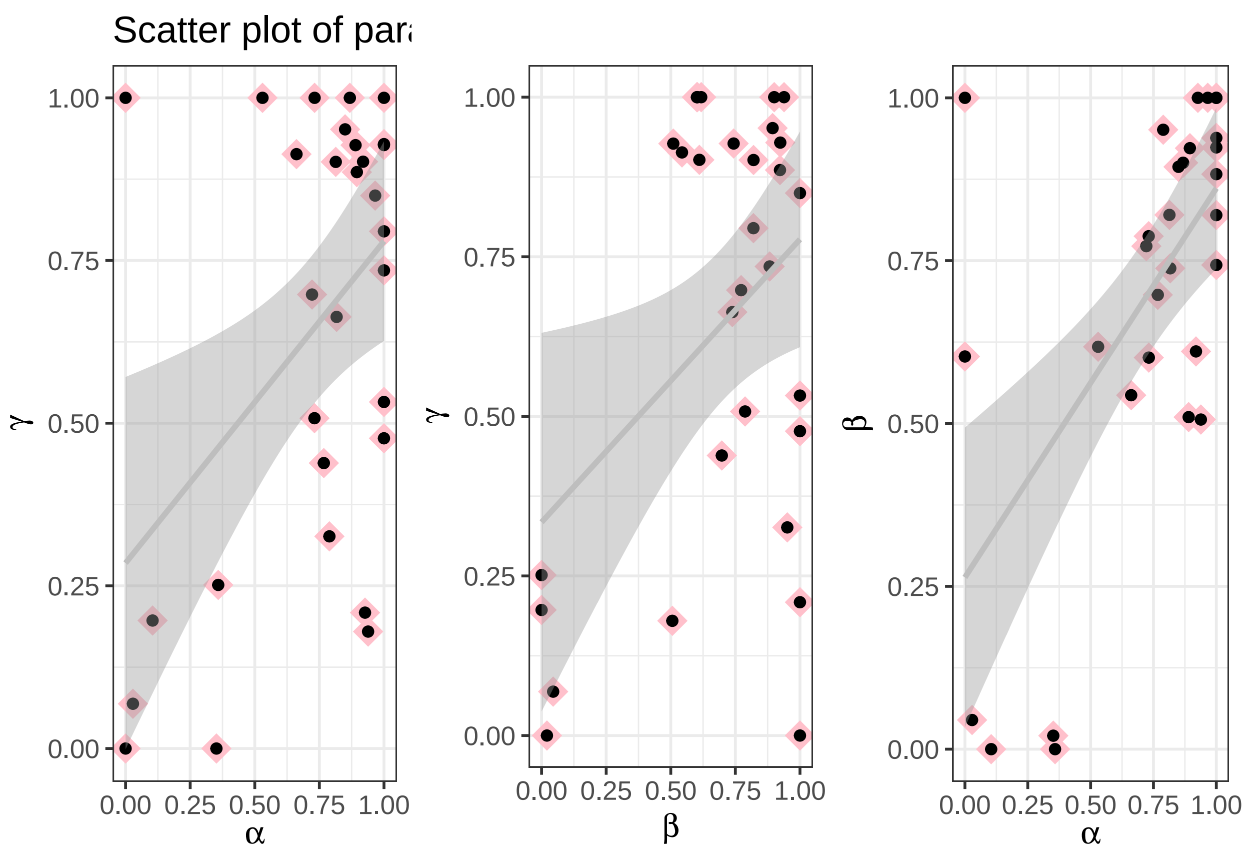
```
  labs(x=expression(beta),y=expression(gamma), title = "")

p3 <- ggplot(data = NULL, mapping = aes(x = multifits3$alpha, y = multifits3$beta))+
  geom_point(colour="pink", shape=18, size = 5)+
  geom_point(colour = "black", size = 1.5) +
  geom_smooth(method = 'lm', colour = "gray") +
  theme_bw() +
  theme(text=element_text(size=12)) +
  labs(x=expression(alpha),y=expression(beta), title = "")


pv3 <- plot_grid(p1, p2, p3, ncol = 3)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

```
pv3
```




```
pl1 <- ggplot(multifits1, aes(lambda)) +
  geom_histogram(binwidth = 0.05, colour = "maroon", fill = "maroon") +
  theme_bw() +
  theme(text=element_text(size=14)) +
  labs(x=expression(lambda), title = "Distribution of lambda")

pl2 <- ggplot(multifits2, aes(lambda)) +
  geom_histogram(binwidth = 0.05, colour = "maroon", fill = "maroon") +
```

```
  theme_bw() +
  theme(text=element_text(size=14)) +
  labs(x=expression(lambda), y = "", title = "")

pl3 <-  ggplot(multifits2, aes(lambda)) +
  geom_histogram(binwidth = 0.05, colour = "maroon", fill = "maroon") +
  theme_bw() +
  theme(text=element_text(size=14)) +
  labs(x=expression(lambda), y = "",title = "")

pll <- plot_grid(pl1,pl2,pl3, nrow = 1)
```

```
## Warning: Removed 4 rows containing non-finite values (stat_bin).

## Warning: Removed 1 rows containing non-finite values (stat_bin).

## Warning: Removed 1 rows containing non-finite values (stat_bin).
```

```
pll
```