Eyewitness identification data analysis

Kym McCormick

15 May 2019

setwd("C:/Users/mccormick/GitHub/PhD-Thesis")  
  
data <- read.csv("data/Experiment 3/experiment3data.csv")  
  
  
din <- data %>%  
 select(uid,  
 condition,   
 Test\_T1\_suspectIdentified,   
 confidence\_rating,   
 demographics\_age,   
 demographics\_gender,   
 demographics\_country)%>%  
 filter(demographics\_country == "USA", !is.na(confidence\_rating)) %>%  
 separate(condition,c("memory","expectation","target")) %>%  
 mutate(CID = if\_else(target == "P" & Test\_T1\_suspectIdentified == "F68", 1, 0),  
 Miss = if\_else(target == "P" & Test\_T1\_suspectIdentified == "Silhouette", 1, 0),  
 TPFoilID = if\_else(target == "P" & Test\_T1\_suspectIdentified != "F68" & Test\_T1\_suspectIdentified != "Silhouette", 1, 0),  
 CR = if\_else(target == "A" & Test\_T1\_suspectIdentified == "Silhouette", 1, 0),  
 TAFoilID = if\_else(target == "A" & Test\_T1\_suspectIdentified != "Silhouette", 1, 0)  
 )

## There are twice the number of target present lineups than target absent lineups. To rebalance this in a way that preserves the groupings of confidence intervals, I have chosen to double the Target Absent data before grouping into equal sized confidence rating bins  
  
## Separate data into target present and absent dataframes ---  
Ratings\_Absent <- din %>%  
 filter(target == "A")  
  
Ratings\_Present <- din %>%  
 filter(target == "P")  
  
## Bind back together, doubling up on the target absent group ---  
## NOTE THAT DUPLICATES MAY BE REMOVED USING THE UID NUMBERS ---  
Ratings <- rbind(Ratings\_Absent,Ratings\_Absent,Ratings\_Present)  
  
## Group chooser confidence into decile confidence rating bins  
din\_mod\_choose <- Ratings %>%  
 filter(Test\_T1\_suspectIdentified != "Silhouette") %>%  
 mutate(decile\_cr = ntile(confidence\_rating,10)  
 )  
  
## Group non-chooser confidence into a single zero confidence rating bin  
din\_mod\_nochoose <- Ratings %>%  
 filter(Test\_T1\_suspectIdentified == "Silhouette") %>%  
 mutate(decile\_cr = 0)  
   
## Bind chooser and non-chooser data back together  
din\_mod <- rbind(din\_mod\_choose, din\_mod\_nochoose)

## Check the grouping of the confidence rating deciles to ensure evenness  
Rating\_prop <- din\_mod %>%  
 select(decile\_cr) %>%  
 map(~prop.table(table(.))  
 )  
Rating\_prop

## $decile\_cr  
## .  
## 0 1 2 3 4 5   
## 0.24442765 0.07555723 0.07555723 0.07555723 0.07555723 0.07555723   
## 6 7 8 9 10   
## 0.07555723 0.07555723 0.07555723 0.07555723 0.07555723

## Demographics

#length(unique(din$uid))  
  
describe(din$demographics\_age)

## vars n mean sd median trimmed mad min max range skew kurtosis  
## X1 1 2012 37.15 11.42 35 35.91 10.38 18 81 63 0.92 0.26  
## se  
## X1 0.25

demo <- din %>%   
 select(demographics\_gender) %>%   
 map(~prop.table(table(.)))  
  
demo

## $demographics\_gender  
## .  
## female male other   
## 0.000000000 0.509443340 0.484095427 0.006461233

## Identification Data

Number of Correct and False IDs per confidence decile

ID\_dat <- din\_mod %>%   
 group\_by(decile\_cr) %>%  
 summarise(  
 Miss = sum(Miss),  
 TPFID = sum(TPFoilID),  
 Correct\_ID = sum(CID),  
 CR = sum(CR),  
 TAFID = sum(TAFoilID)\*7/8,  
 False\_ID = sum(TAFoilID)/8,  
 n = n()  
 )   
  
ID\_dat

## # A tibble: 11 x 8  
## decile\_cr Miss TPFID Correct\_ID CR TAFID False\_ID n  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 0 261 0 0 386 0 0 647  
## 2 1 0 60 26 0 99.8 14.2 200  
## 3 2 0 66 50 0 73.5 10.5 200  
## 4 3 0 55 43 0 89.2 12.8 200  
## 5 4 0 50 38 0 98 14 200  
## 6 5 0 70 55 0 65.6 9.38 200  
## 7 6 0 43 49 0 94.5 13.5 200  
## 8 7 0 58 58 0 73.5 10.5 200  
## 9 8 0 70 62 0 59.5 8.5 200  
## 10 9 0 40 91 0 60.4 8.62 200  
## 11 10 0 32 100 0 59.5 8.5 200

Vector of Correct ID and False ID counts for each confidence decile. Note that False ID counts are the number of identifications from target absent lineups divided by the number of items within the lineup (n = 8) ##Overall responses

TP <- count(din\_mod,target == "P")  
TP

## # A tibble: 2 x 2  
## `target == "P"` n  
## <lgl> <int>  
## 1 FALSE 1270  
## 2 TRUE 1377

ID\_dvector <- din\_mod %>%   
 group\_by(decile\_cr) %>%  
 summarise(  
 Miss = sum(Miss)/n(),  
 TPFID = sum(TPFoilID)/n(),  
 Correct\_ID = sum(CID)/n(),  
 CR = sum(CR)/n(),  
 TAFID = sum(TAFoilID)\*7/8/n(),  
 False\_ID = sum(TAFoilID)/8/n(),  
 )   
  
ID\_dvector

## # A tibble: 11 x 7  
## decile\_cr Miss TPFID Correct\_ID CR TAFID False\_ID  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0 0.403 0 0 0.597 0 0   
## 2 1 0 0.3 0.13 0 0.499 0.0712  
## 3 2 0 0.33 0.25 0 0.368 0.0525  
## 4 3 0 0.275 0.215 0 0.446 0.0638  
## 5 4 0 0.25 0.19 0 0.49 0.07   
## 6 5 0 0.35 0.275 0 0.328 0.0469  
## 7 6 0 0.215 0.245 0 0.472 0.0675  
## 8 7 0 0.290 0.290 0 0.368 0.0525  
## 9 8 0 0.35 0.31 0 0.298 0.0425  
## 10 9 0 0.2 0.455 0 0.302 0.0431  
## 11 10 0 0.16 0.5 0 0.298 0.0425

HIT\_FA\_prop <- ID\_dat %>%  
 filter(decile\_cr != 0) %>%  
 select(Correct\_ID, False\_ID) %>%  
 transform(Correct\_ID = Correct\_ID/1270,  
 False\_ID = False\_ID/1377)   
  
HIT\_FA\_prop <- HIT\_FA\_prop[10:1,]  
  
HIT\_FA\_prop

## Correct\_ID False\_ID  
## 10 0.07874016 0.006172840  
## 9 0.07165354 0.006263617  
## 8 0.04881890 0.006172840  
## 7 0.04566929 0.007625272  
## 6 0.03858268 0.009803922  
## 5 0.04330709 0.006808279  
## 4 0.02992126 0.010167030  
## 3 0.03385827 0.009259259  
## 2 0.03937008 0.007625272  
## 1 0.02047244 0.010348584

Vector of CIDs and FIDs for each confidence decile. Note that FIDs are calculated using Wixted & Mickes method, as per above.

## Creating HIT/FA pairs for ROC  
TP <- count(din\_mod,target == "P")  
TP

## # A tibble: 2 x 2  
## `target == "P"` n  
## <lgl> <int>  
## 1 FALSE 1270  
## 2 TRUE 1377

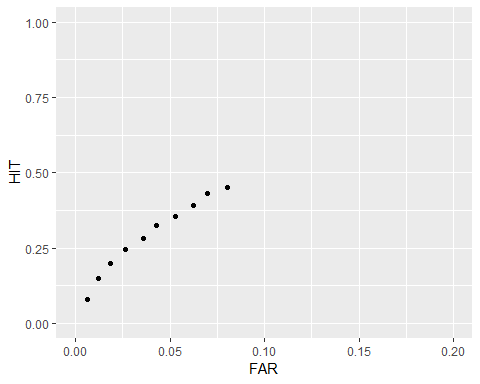
HIT\_FA\_prop <- ID\_dat %>%  
 filter(decile\_cr != 0) %>%  
 select(Correct\_ID, False\_ID) %>%  
 transform(Correct\_ID = Correct\_ID/1270,  
 False\_ID = False\_ID/1377)   
  
HIT\_FA\_prop <- HIT\_FA\_prop[10:1,]  
  
HIT\_FA\_prop

## Correct\_ID False\_ID  
## 10 0.07874016 0.006172840  
## 9 0.07165354 0.006263617  
## 8 0.04881890 0.006172840  
## 7 0.04566929 0.007625272  
## 6 0.03858268 0.009803922  
## 5 0.04330709 0.006808279  
## 4 0.02992126 0.010167030  
## 3 0.03385827 0.009259259  
## 2 0.03937008 0.007625272  
## 1 0.02047244 0.010348584

ROC <- HIT\_FA\_prop %>%  
 transform(HIT = cumsum(Correct\_ID),  
 FAR = cumsum(False\_ID)  
 ) %>%  
 select(HIT,FAR)  
  
ROC

## HIT FAR  
## 10 0.07874016 0.00617284  
## 9 0.15039370 0.01243646  
## 8 0.19921260 0.01860930  
## 7 0.24488189 0.02623457  
## 6 0.28346457 0.03603849  
## 5 0.32677165 0.04284677  
## 4 0.35669291 0.05301380  
## 3 0.39055118 0.06227306  
## 2 0.42992126 0.06989833  
## 1 0.45039370 0.08024691

library(ggplot2)  
# Basic scatter plot  
ggplot(ROC, aes(x=FAR, y=HIT)) +   
 geom\_point() +  
 xlim(0,.2) +  
 ylim(0,1)



## By manipulation group responses

ID\_dat <- din\_mod %>%   
 group\_by(decile\_cr, memory,expectation) %>%  
 summarise(  
 Miss = sum(Miss),  
 TPFID = sum(TPFoilID),  
 Correct\_ID = sum(CID),  
 CR = sum(CR),  
 TAFID = sum(TAFoilID)\*7/8,  
 False\_ID = sum(TAFoilID)/8,  
 TP = sum(Miss, TPFID, Correct\_ID),  
 TA = sum(CR, TAFID,False\_ID),  
 n = n()  
 )  
ID\_dat

## # A tibble: 44 x 12  
## # Groups: decile\_cr, memory [22]  
## decile\_cr memory expectation Miss TPFID Correct\_ID CR TAFID False\_ID  
## <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0 S H 35 0 0 94 0 0   
## 2 0 S L 74 0 0 104 0 0   
## 3 0 W H 64 0 0 70 0 0   
## 4 0 W L 88 0 0 118 0 0   
## 5 1 S H 0 15 6 0 26.2 3.75  
## 6 1 S L 0 14 5 0 14.9 2.12  
## 7 1 W H 0 17 9 0 28 4   
## 8 1 W L 0 14 6 0 30.6 4.38  
## 9 2 S H 0 12 17 0 27.1 3.88  
## 10 2 S L 0 14 9 0 21 3   
## # ... with 34 more rows, and 3 more variables: TP <dbl>, TA <dbl>, n <int>

HIT\_FA\_WL <- ID\_dat %>%  
 filter(decile\_cr != 0, memory == "W", expectation == "L")%>%  
 select(decile\_cr, memory, expectation, Correct\_ID, False\_ID,TP,TA) %>%  
 group\_by(decile\_cr)%>%  
 summarise(Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TP=sum(TP),  
 TA=sum(TA))  
  
HIT\_FA\_WL <- HIT\_FA\_WL[10:1,]  
  
HIT\_FA\_WL<- HIT\_FA\_WL %>%  
 transform (Correct\_ID = Correct\_ID/sum(TP), False\_ID = False\_ID/sum(TA) )%>%  
 transform(HIT = cumsum(Correct\_ID),  
 FAR = cumsum(False\_ID)  
 ) %>%  
 select(HIT,FAR)  
  
HIT\_FA\_WL

## HIT FAR  
## 1 0.09663866 0.01250000  
## 2 0.13865546 0.02569444  
## 3 0.19327731 0.03333333  
## 4 0.24369748 0.04375000  
## 5 0.27310924 0.05902778  
## 6 0.33193277 0.06944444  
## 7 0.36974790 0.08194444  
## 8 0.39495798 0.09305556  
## 9 0.44957983 0.10069444  
## 10 0.47478992 0.12500000

HIT\_FA\_WH <- ID\_dat %>%  
 filter(decile\_cr != 0, memory == "W", expectation == "H")%>%  
 select(decile\_cr, memory, expectation, Correct\_ID, False\_ID,TP,TA) %>%  
 group\_by(decile\_cr)%>%  
 summarise(Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TP=sum(TP),  
 TA=sum(TA))   
  
HIT\_FA\_WH <- HIT\_FA\_WH[10:1,]  
  
HIT\_FA\_WH<- HIT\_FA\_WH %>%  
 transform (Correct\_ID = Correct\_ID/sum(TP), False\_ID = False\_ID/sum(TA) )%>%  
 transform(HIT = cumsum(Correct\_ID),  
 FAR = cumsum(False\_ID)  
 ) %>%  
 select(HIT,FAR)  
  
HIT\_FA\_WH

## HIT FAR  
## 1 0.1106870 0.00625000  
## 2 0.1870229 0.01770833  
## 3 0.2404580 0.02760417  
## 4 0.2671756 0.03958333  
## 5 0.3129771 0.05572917  
## 6 0.3473282 0.06770833  
## 7 0.3664122 0.08333333  
## 8 0.3931298 0.09895833  
## 9 0.4351145 0.10833333  
## 10 0.4694656 0.12500000

HIT\_FA\_SH <- ID\_dat %>%  
 filter(decile\_cr != 0, memory == "S", expectation == "H")%>%  
 select(decile\_cr, memory, expectation, Correct\_ID, False\_ID,TP,TA) %>%  
 group\_by(decile\_cr)%>%  
 summarise(Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TP=sum(TP),  
 TA=sum(TA))   
  
HIT\_FA\_SH <- HIT\_FA\_SH[10:1,]  
  
HIT\_FA\_SH<- HIT\_FA\_SH %>%  
 transform (Correct\_ID = Correct\_ID/sum(TP), False\_ID = False\_ID/sum(TA) )%>%  
 transform(HIT = cumsum(Correct\_ID),  
 FAR = cumsum(False\_ID)  
 ) %>%  
 select(HIT,FAR)  
  
HIT\_FA\_SH

## HIT FAR  
## 1 0.08011869 0.01128472  
## 2 0.17804154 0.01909722  
## 3 0.24035608 0.03081597  
## 4 0.31157270 0.04210069  
## 5 0.35905045 0.05729167  
## 6 0.40356083 0.06944444  
## 7 0.43916914 0.08506944  
## 8 0.48664688 0.09852431  
## 9 0.53709199 0.11197917  
## 10 0.55489614 0.12500000

HIT\_FA\_SL <- ID\_dat %>%  
 filter(decile\_cr != 0, memory == "S", expectation == "L")%>%  
 select(decile\_cr, memory, expectation, Correct\_ID, False\_ID,TP,TA) %>%  
 group\_by(decile\_cr)%>%  
 summarise(Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TP=sum(TP),  
 TA=sum(TA))  
  
HIT\_FA\_SL <- HIT\_FA\_SL[10:1,]  
  
HIT\_FA\_SL<- HIT\_FA\_SL%>%  
 transform (Correct\_ID = Correct\_ID/sum(TP), False\_ID = False\_ID/sum(TA) ) %>%  
 transform(HIT = cumsum(Correct\_ID),  
 FAR = cumsum(False\_ID)  
 ) %>%  
 select(HIT,FAR)  
  
HIT\_FA\_SL

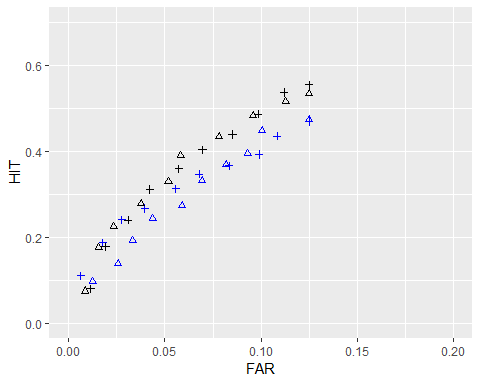
## HIT FAR  
## 1 0.07526882 0.008522727  
## 2 0.17562724 0.015625000  
## 3 0.22580645 0.023437500  
## 4 0.27956989 0.037642045  
## 5 0.32974910 0.051846591  
## 6 0.39068100 0.058238636  
## 7 0.43369176 0.078125000  
## 8 0.48387097 0.095880682  
## 9 0.51612903 0.112926136  
## 10 0.53405018 0.125000000

## Estimating UV-SDT model parameters

ROC <- data.frame(HIT\_FA\_WL, HIT\_FA\_WH,HIT\_FA\_SH, HIT\_FA\_SL)  
ROC

## HIT FAR HIT.1 FAR.1 HIT.2 FAR.2  
## 1 0.09663866 0.01250000 0.1106870 0.00625000 0.08011869 0.01128472  
## 2 0.13865546 0.02569444 0.1870229 0.01770833 0.17804154 0.01909722  
## 3 0.19327731 0.03333333 0.2404580 0.02760417 0.24035608 0.03081597  
## 4 0.24369748 0.04375000 0.2671756 0.03958333 0.31157270 0.04210069  
## 5 0.27310924 0.05902778 0.3129771 0.05572917 0.35905045 0.05729167  
## 6 0.33193277 0.06944444 0.3473282 0.06770833 0.40356083 0.06944444  
## 7 0.36974790 0.08194444 0.3664122 0.08333333 0.43916914 0.08506944  
## 8 0.39495798 0.09305556 0.3931298 0.09895833 0.48664688 0.09852431  
## 9 0.44957983 0.10069444 0.4351145 0.10833333 0.53709199 0.11197917  
## 10 0.47478992 0.12500000 0.4694656 0.12500000 0.55489614 0.12500000  
## HIT.3 FAR.3  
## 1 0.07526882 0.008522727  
## 2 0.17562724 0.015625000  
## 3 0.22580645 0.023437500  
## 4 0.27956989 0.037642045  
## 5 0.32974910 0.051846591  
## 6 0.39068100 0.058238636  
## 7 0.43369176 0.078125000  
## 8 0.48387097 0.095880682  
## 9 0.51612903 0.112926136  
## 10 0.53405018 0.125000000

ggplot(HIT\_FA\_WL, aes(x=FAR, y=HIT)) +   
 geom\_point(data = HIT\_FA\_WL, colour = 'blue', shape = 2) +  
 geom\_point(data = HIT\_FA\_WH, colour = 'blue', shape = 3) +  
 geom\_point(data = HIT\_FA\_SH, colour = 'black', shape = 3) +  
 geom\_point(data = HIT\_FA\_SL, colour = 'black', shape = 2) +  
 xlim(0,.2) +  
 ylim(0,.7)+   
 theme(legend.position="top")



IndObvSDT\_eyewit <- function(Q, data, param.names, n.params, tmp.env){

n <- 8 mean <- Q[1] sd <- Q[2] cr <- c(Q[3:12]) #confidence criterion

v <- vector()

for (i in 1:length(cr)) { TID <- integrate( f = function(x){ dnorm(x,mean,sd)*(pnorm(x,0,1)^(n-1)) }, lower = cr[i], upper = Inf )value ) FA <- (1-pnorm(cr[i])^n)/n #TD <- 1-(pnorm((mean-cr[i])/sd)*(pnorm(cr[i])^(n-1)))

I <- c(  
 t(  
 cbind(  
 (1-(FID+TID)),#Miss (TP)  
 FID, #Foil ID (TP)  
 TID, #Target ID  
 1-FA, #Correct Rejection (TA)  
 FA\*(n-1)/n, #Foil ID (TA)  
 FA #False ID (TA)  
 )  
 )  
)

v <- cbind(v,I)

}

Outcomes <- c(t(v)) return(Outcomes)

}

IndObvMLE <- function(Q, data, param.names, n.params, tmp.env, lower.bound, upper.bound){

e <- IndObvSDT\_eyewit(Q, param.names, n.params, tmp.env)  
LL <- -sum(data[data!=0]\*log(e[data!=0]))  
  
  
  
return(LL)

}

fit\_kafc <- fit.mptinr(

data = ID\_cum\_vector,  
  
objective = IndObvMLE,  
  
param.names = c("mu", "sigma", "cr1", "cr2", "cr3", "cr4", "cr5", "cr6", "cr7", "cr8", "cr9", "cr10"),  
  
categories.per.type = c(6,6,6,6,6,6,6,6,6,6),  
  
prediction = IndObvSDT\_eyewit,  
  
lower.bound = c(0,0.1,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf),  
  
upper.bound = Inf,  
  
n.optim = 5,  
  
#starting.values = c(1,1,1,1,1,1,1,1,1,1,1,1,1,1),  
  
show.messages = FALSE

)

fit\_kafc$goodness.of.fit

fit\_kafc$parameters

UVSDT\_eyewit <- function(Q, data, param.names, n.params, tmp.env){

n <- 8 mean <- Q[1] sd <- Q[2] cr <- c(Q[3:12]) #confidence criterion

v <- vector()

for (i in 1:length(cr)) { CID <- integrate( f = function(x){ dnorm(x,mean,sd)\*(pnorm(x,0,1)^(n-1)) }, lower = cr[i], upper = Inf )$value

FA <- (1-pnorm(cr[i])^n)/n  
  
I <- c(  
 t(  
 cbind( (1-CID), CID, 1-(FA), FA )  
 )  
)

v <- cbind(v,I) #print(v)

}

Outcomes <- c(t(v)) return(Outcomes)

}

IndObvMLE2 <- function(Q, data, param.names, n.params, tmp.env, lower.bound, upper.bound){

e <- UVSDT\_eyewit(Q, param.names, n.params, tmp.env)  
LL <- -sum(data[data!=0]\*log(e[data!=0]))  
  
  
  
return(LL)

}

fit\_kafc <- fit.mptinr(

data = ID\_cum\_vector2,  
  
objective = IndObvMLE2,  
  
param.names = c("mu", "sigma", "cr1", "cr2", "cr3", "cr4", "cr5", "cr6", "cr7", "cr8", "cr9", "cr10"),  
  
categories.per.type = c(4,4,4,4,4,4,4,4,4,4),  
  
prediction = UVSDT\_eyewit,  
  
lower.bound = c(0,0.1,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf),  
  
upper.bound = Inf,  
  
n.optim = 5,  
  
starting.values = c(0,0,0,0,0,0,0,0,0,0,0,0),  
  
show.messages = FALSE

)

fit\_kafc$goodness.of.fit

fit\_kafc$parameters

n <- 8 UVSDT <-" pnorm((cr1 - mu)/ss) pnorm((cr1+cr2 - mu)/ss) - pnorm((cr1 - mu)/ss) pnorm((cr1+cr2+cr3 - mu)/ss) - pnorm((cr1+cr2 - mu)/ss) pnorm((cr1+cr2+cr3+cr4 - mu)/ss) - pnorm((cr1+cr2+cr3 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5+cr6 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4+cr5 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8 - mu)/ss) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9+cr10 - mu)/ss) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9 - mu)/ss)

1 - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9+cr10 - mu)/ss)

pnorm((cr1)) pnorm((cr1+cr2)) - pnorm((cr1)) pnorm((cr1+cr2+cr3)) - pnorm((cr1+cr2)) pnorm((cr1+cr2+cr3+cr4)) - pnorm((cr1+cr2+cr3)) pnorm((cr1+cr2+cr3+cr4+cr5)) - pnorm((cr1+cr2+cr3+cr4)) pnorm((cr1+cr2+cr3+cr4+cr5+cr6)) - pnorm((cr1+cr2+cr3+cr4+cr5)) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7)) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6)) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8)) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7)) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9)) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8)) pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9+cr10)) - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9))

1 - pnorm((cr1+cr2+cr3+cr4+cr5+cr6+cr7+cr8+cr9+cr10)) "

fit\_UVSDT <- fit.model(ID\_dvector,textConnection(UVSDT),lower.bound=c(-Inf,rep(0,9),0,0.1),upper.bound=rep(Inf,12), n.optim=5, show.messages = FALSE)

fit\_UVSDT$goodness.of.fit # goodness of fit fit\_UVSDT$parameters # parameters ```