

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

Online Supplementary Material

BUGS code for TAX (independent estimate of a single individual)

```

model {
  #error parameter
  theta.pre~dbeta(1,1)
  theta <- theta.pre*5 # range of theta from 0 to 5

  #set TAX parameter
  n <- 2

  d.pre ~ dbeta(1,1) #delta (configural weight)
  d <- 4*d.pre-2     #range of d scaled from -2 to 2

  g.pre ~ dbeta(1,1) #gamma (probability weighting)
  g <- g.pre * 5     #range scaled from 0 to 5

  b.pre ~ dbeta(1,1) #beta (utility)
  b <- b.pre * 5     #range scaled from 0 to 5

  # calculate subjective probability
  pSubMaxA <- pow(probs[,1],g)
  pSubMinA <- pow(probs[,3],g)
  pSubMaxB <- pow(probs[,2],g)
  pSubMinB <- pow(probs[,4],g)

  #calculate weights
  weightAmax <- pSubMaxA - d/(n+1) * pSubMinA #weight for high outcome is reduced
  weightAmin <- pSubMinA + d/(n+1) * pSubMaxA #weight for low outcome is increased

  weightBmax <- pSubMaxB - d/(n+1) * pSubMinB
  weightBmin <- pSubMinB + d/(n+1) * pSubMaxB

  #calculate utilities
  utilMaxA <- signPayoffs[,1]*pow(abs(payoffs[,1]),b)
  utilMinA <- signPayoffs[,3]*pow(abs(payoffs[,3]),b)
  utilMaxB <- signPayoffs[,2]*pow(abs(payoffs[,2]),b)
  utilMinB <- signPayoffs[,4]*pow(abs(payoffs[,4]),b)

  #obtain tax utilities for both options
  taxA <- (weightAmax*utilMaxA+weightAmin*utilMinA)/(pSubMaxA+pSubMinA)
  taxB <- (weightBmax*utilMaxB+weightBmin*utilMinB)/(pSubMaxB+pSubMinB)

  #diffrence between tax utilities
  taxDiff <- taxA-taxB

  for (choice in 1:nChoices) {

    #Luce choice rule
    pA[choice] <- 1/(1+exp(-1*theta*taxDiff[choice]))

    #Likelihood function
    choiceA[choice]~dbern(pA[choice])
  }
}

```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

Data Provided from outside BUGS:

probs 138 x 4 matrix containing the probabilities for each pair of gambles A and B.

payoffs 138 x 4 matrix containing the payoffs for each pair of gambles A and B.

signPayoff 138 x 4 matrix indicating positive (1) and negative (-1) payoffs

nChoices 138

choiceA vector containing the observed choices for all 138 pairs of gambles (coded 0 and 1)

Columns for probs and payoffs are sorted by the maximum payoff: maximum A, maximum B, minimum A, and maximum B

BUGS code for TAX (hierarchical estimate for a group as a whole)

```

model {

  #set TAX parameter
  n <- 2

  #set group-level parameters
  mu.phi.d ~ dnorm(0,1)
  tau.phi.d <- pow(sigma.phi.d, -2)
  sigma.phi.d ~ dunif(0,10)

  mu.phi.g ~ dnorm(0,1)
  tau.phi.g <- pow(sigma.phi.g, -2)
  sigma.phi.g ~ dunif(0,10)

  mu.phi.b ~ dnorm(0,1)
  tau.phi.b <- pow(sigma.phi.b, -2)
  sigma.phi.b ~ dunif(0,10)

  mu.phi.theta ~ dnorm(0,1)
  tau.phi.theta <- pow(sigma.phi.theta, -2)
  sigma.phi.theta ~ dunif(0,10)

  for (i in 1:nVpn) {

    #error parameter
    theta.probit[i]~dnorm(mu.phi.theta,tau.phi.theta)
    theta.pre[i]<-phi(theta.probit[i])
    theta[i]<-theta.pre[i]*5

    d.probit[i] ~ dnorm(mu.phi.d,tau.phi.d) #delta (configural weight)
    d.pre[i] <- phi(d.probit[i])

    d[i] <- 4*d.pre[i]-2 #scale d from -2 to 2

    g.probit[i] ~ dnorm(mu.phi.g, tau.phi.g)
    g.pre[i] <- phi(g.probit[i]) #gamma (probability weighting)
    g[i] <- g.pre[i] * 5 #scale from 0 to 5

    b.probit[i] ~ dnorm(mu.phi.b, tau.phi.b)
    b.pre[i] <- phi(b.probit[i]) #beta (utility)
    b[i] <- b.pre[i] * 5 #scale from 0 to 5

    for (choice in 1:nChoices[i]) {

      # calculate subjective probability
      pSubMaxA[i,choice] <- pow(probs[i,choice,1],g[i])
      pSubMinA[i,choice] <- pow(probs[i,choice,3],g[i])
      pSubMaxB[i,choice] <- pow(probs[i,choice,2],g[i])
      pSubMinB[i,choice] <- pow(probs[i,choice,4],g[i])
    }
  }
}

```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

```
#calculate weights
weightAmax[i,choice] <- pSubMaxA[i,choice]
- d[i]/(n+1) * pSubMinA[i,choice] #weight for high outcome reduced

weightAmin[i,choice] <- pSubMinA[i,choice]
+ d[i]/(n+1) * pSubMaxA[i,choice] #weight for low outcome increased

weightBmax[i,choice] <- pSubMaxB[i,choice]
- d[i]/(n+1) * pSubMinB[i,choice]

weightBmin[i,choice] <- pSubMinB[i,choice]
+ d[i]/(n+1) * pSubMaxB[i,choice]

#calculate utilities
utilMaxA[i,choice] <-
signPayoffs[i,choice,1]*pow(abs(payoffs[i,choice,1]),b[i])

utilMinA[i,choice] <-
signPayoffs[i,choice,3]*pow(abs(payoffs[i,choice,3]),b[i])

utilMaxB[i,choice] <-
signPayoffs[i,choice,2]*pow(abs(payoffs[i,choice,2]),b[i])

utilMinB[i,choice] <-
signPayoffs[i,choice,4]*pow(abs(payoffs[i,choice,4]),b[i])

#obtain tax utilities for both options
taxA[i,choice] <-
(weightAmax[i,choice]*utilMaxA[i,choice]+weightAmin[i,choice]
*utilMinA[i,choice])/(pSubMaxA[i,choice]+pSubMinA[i,choice])

taxB[i,choice] <-
(weightBmax[i,choice]*utilMaxB[i,choice]+weightBmin[i,choice]
*utilMinB[i,choice])/(pSubMaxB[i,choice]+pSubMinB[i,choice])

#difference between tax utilities
taxDiff[i,choice] <- taxA[i,choice]-taxB[i,choice]

#Luce choice rule
pA[i,choice] <- 1/(1+exp(-1*theta[i]*taxDiff[i,choice]))

#Likelihood function
choiceA[i,choice]~dbern(pA[i,choice])
}
}
}
```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

Data Provided from outside BUGS:

probs 64 x 138 x 4 array containing the probabilities for each pair of gambles A and B for each participant

payoffs 64 x 138 x 4 array containing the payoffs for each pair of gambles A and B for each participant.

signPayoff 64 x 138 x 4 array indicating positive (1) and negative (-1) payoffs

nChoices Vector containing the number of choices (i.e. 138) for each participant.

choiceA Vector containing the observed choices for all 138 pairs of gambles (coded 1 for choosing option A and 0 for choosing option B)

nVpn 64

Columns for probs and payoffs are sorted by the maximum payoff: maximum A, maximum B, minimum A, and maximum B

BUGS code for CPT (independent estimate for a single individual)

```

model
{
  for (j in 1:64)# Subject-loop
  {
    alpha[j] ~ dbeta(1,1) #range 0 to 1

    luce.pre[j] ~dbeta(1,1)      luce[j]<-luce.pre[j]*5
    lambda.pre[j] ~dbeta(1,1)
    lambda[j]<-lambda.pre[j]*5

    #probability weighting
    gamma.gain[j] ~dbeta(1,1) #range 0 to 1
    gamma.loss[j] ~dbeta(1,1) #range 0 to 1

    delta.gain.pre[j] ~dbeta(1,1)
    delta.loss.pre[j] ~dbeta(1,1)

    delta.gain[j]<-delta.gain.pre[j]*5
    delta.loss[j]<-delta.loss.pre[j]*5

    for (i in 1:70)# Item-Loop - positive gambles
    {
      #-----
      #positive gambles,gamble A
      v.x.a[i,j] <- pow(prospects.a[j,i,1],alpha[j])
      v.y.a[i,j] <- pow(prospects.a[j,i,3],alpha[j])

      #2parameter weighting function
      w.x.a[i,j] <- delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])/
      (delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])+pow((1-
      prospects.a[j,i,2]),gamma.gain[j]))
      w.y.a[i,j] <- 1-w.x.a[i,j] #delta[j]*pow(prospects.a[j,i,4],gamma[j])/
      (delta[j]*pow(prospects.a[j,i,4],gamma[j])+pow((1-prospects.a[j,i,4]),gamma[j]))
      Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

      #-----
      #positive gambles,gamble B

      v.x.b[i,j] <- pow(prospects.b[j,i,1],alpha[j])
      v.y.b[i,j] <- pow(prospects.b[j,i,3],alpha[j])

      #2parameter weighting function
      w.x.b[i,j] <- delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])/
      (delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])+pow((1-
      prospects.b[j,i,2]),gamma.gain[j]))
    }
  }
}

```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

```

w.y.b[i,j] <- 1-w.x.b[i,j] #delta[j]*pow(prospects.b[j,i,4],gamma[j])/
(delta[j]*pow(prospects.b[j,i,4],gamma[j])+pow((1-prospects.b[j,i,4]),gamma[j]))

Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]

#-----
# positive gambles,choice-rule

binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))
rawdata[i,j] ~ dbern(binval[i,j])
}

for (i in 71:100)# Item-Loop, negative gambles,gamble A
{
v.x.a[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.a[j,i,1]),alpha[j])
v.y.a[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.a[j,i,3]),alpha[j])

#2parameter weighting function
w.x.a[i,j] <- 1-(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])+pow(prospects.a[j,i,2],gamma.loss[
j])))
w.y.a[i,j] <- 1-w.x.a[i,j]

Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

#-----
# Item-Loop, negative gambles,gamble B

v.x.b[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.b[j,i,1]),alpha[j])
v.y.b[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.b[j,i,3]),alpha[j])

#2parameter weighting function
w.x.b[i,j] <- 1-(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])+pow(prospects.b[j,i,2],gamma.loss[
j])))
w.y.b[i,j] <- 1-w.x.b[i,j]

Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]

#-----
# Item-Loop, negative gambles,choice-rule

binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))
rawdata[i,j] ~ dbern(binval[i,j])
}

for (i in 101:138)# Item-Loop, mixed gambles,gamble A
{
v.x.a[i,j] <- pow(prospects.a[j,i,1],alpha[j])

```


HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

```

v.y.a[i,j] <- (-1 * lambda[j]) * pow((-1 * prospects.a[j,i,3]),alpha[j])

#2parameter weighting function
w.x.a[i,j] <- delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])/
(delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])+pow(prospects.a[j,i,4],gamma.gain[j]))
w.y.a[i,j] <- delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])+pow(prospects.a[j,i,2],gamma.loss[j]))

Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

#-----
# Item-Loop, mixed gambles,gamble B

v.x.b[i,j] <- pow(prospects.b[j,i,1],alpha[j])
v.y.b[i,j] <- (-1 * lambda[j]) * pow((-1 * prospects.b[j,i,3]),alpha[j])

#2parameter weighting function
w.x.b[i,j] <- delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])/
(delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])+pow(prospects.b[j,i,4],gamma.gain[j]))
w.y.b[i,j] <- delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])+pow(prospects.b[j,i,2],gamma.loss[j]))

Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]

#-----
# Item-Loop, mixed gambles,choice-rule

binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))
rawdata[i,j] ~ dbern(binval[i,j])
}
}
}

```

Data Provided from Outside BUGS:

prospects.a 64 x 138 x 4 array containing the payoffs and probabilities for gamble A

prospects.b 64 x 138 x 4 array containing the payoffs and probabilities for gamble B

rawdata 138 x 64 matrix containing the observed choices for all 138 pairs of gambles
(0 for choosing option A and 1 for choosing option B)

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

BUGS code for CPT (hierarchical estimate for a group of individuals)

```
model
{

  for (j in 1:64)
  {

    alpha.phi[j] ~ dnorm(mu.phi.alpha,tau.phi.alpha) T(-5, 5)
    alpha[j] <- phi(alpha.phi[j]) #range 0 to 1

    gamma.gain.phi[j] ~ dnorm(mu.phi.gamma.gain,tau.phi.gamma.gain) T(-5, 5)
    gamma.loss.phi[j] ~ dnorm(mu.phi.gamma.loss,tau.phi.gamma.loss) T(-5, 5)

    gamma.gain[j] <- phi(gamma.gain.phi[j]) #range 0 to 1
    gamma.loss[j] <- phi(gamma.loss.phi[j]) #range 0 to 1

    delta.gain.phi[j] ~ dnorm(mu.phi.delta.gain,tau.phi.delta.gain) T(-5,5)
    delta.gain.pre[j] <- phi(delta.gain.phi[j])
    delta.gain[j] <- delta.gain.pre[j]*5

    delta.loss.phi[j] ~ dnorm(mu.phi.delta.loss,tau.phi.delta.loss) T(-5,5)
    delta.loss.pre[j] <- phi(delta.loss.phi[j])
    delta.loss[j] <- delta.loss.pre[j]*5

    luce.phi[j] ~ dnorm(mu.phi.luce, tau.phi.luce) T(-5,5)
    luce.pre[j] <- phi(luce.phi[j])
    luce[j] <- luce.pre[j]*5

    lambda.phi[j] ~ dnorm(mu.phi.lambda, tau.phi.lambda) T(-5,5)
    lambda.pre[j] <- phi(lambda.phi[j])
    lambda[j] <- lambda.pre[j]*5

  }

  mu.phi.alpha ~ dnorm(0,1)
  tau.phi.alpha <- pow(sigma.phi.alpha,-2)
  sigma.phi.alpha ~ dunif(0,10)

  #probability weighing
  mu.phi.gamma.gain ~ dnorm(0,1)
  tau.phi.gamma.gain <- pow(sigma.phi.gamma.gain,-2)
  sigma.phi.gamma.gain ~ dunif(0,10)

  mu.phi.gamma.loss ~ dnorm(0,1)
  tau.phi.gamma.loss <- pow(sigma.phi.gamma.loss,-2)
  sigma.phi.gamma.loss ~ dunif(0,10)

  mu.phi.delta.gain ~ dnorm(0,1)
  tau.phi.delta.gain <- pow(sigma.phi.delta.gain,-2)
  sigma.phi.delta.gain ~ dunif(0,10)

  mu.phi.delta.loss ~ dnorm(0,1)
  tau.phi.delta.loss <- pow(sigma.phi.delta.loss,-2)
}
```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

```

sigma.phi.delta.loss ~ dunif(0,10)

mu.phi.lambda ~ dnorm(0,1)
tau.phi.lambda <- pow(sigma.phi.lambda,-2)
sigma.phi.lambda ~ dunif(0,10)

mu.phi.luce ~ dnorm(0,1)
tau.phi.luce <- pow(sigma.phi.luce,-2)
sigma.phi.luce ~ dunif(0,10)

for (j in 1:64)# Subject-loop
{
  for (i in 1:70)# Item-Loop - positive gambles
  {
    #-----
    #positive gambles,gamble A

    v.x.a[i,j] <- pow(prospects.a[j,i,1],alpha[j])
    v.y.a[i,j] <- pow(prospects.a[j,i,3],alpha[j])

    #2parameter weighting function
    w.x.a[i,j] <- delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])/
(delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])+pow((1-
prospects.a[j,i,2]),gamma.gain[j]))
    w.y.a[i,j] <- 1-w.x.a[i,j]

    Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

    #-----
    #positive gambles,gamble B

    v.x.b[i,j] <- pow(prospects.b[j,i,1],alpha[j])
    v.y.b[i,j] <- pow(prospects.b[j,i,3],alpha[j])

    #2parameter weighting function
    w.x.b[i,j] <- delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])/
(delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])+pow((1-
prospects.b[j,i,2]),gamma.gain[j]))
    w.y.b[i,j] <- 1-w.x.b[i,j] #delta[j]*pow(prospects.b[j,i,4],gamma[j])/
(delta[j]*pow(prospects.b[j,i,4],gamma[j])+pow((1-prospects.b[j,i,4]),gamma[j]))

    Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]

    #-----
    # positive gambles,choice-rule

    binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))
    rawdata[i,j] ~ dbern(binval[i,j])
  }
}

```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

```

for (i in 71:100)# Item-Loop, negative gambles,gamble A
{
  v.x.a[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.a[j,i,1]),alpha[j])
  v.y.a[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.a[j,i,3]),alpha[j])

  #2parameter weighting function
  w.x.a[i,j] <- 1-(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])+pow(prospects.a[j,i,2],gamma.loss[
j])))

  w.y.a[i,j] <- 1-w.x.a[i,j]#delta[j]*pow(prospects.a[j,i,4],gamma[j])/
(delta[j]*pow(prospects.a[j,i,4],gamma[j])+pow((1-prospects.a[j,i,4]),gamma[j]))

  Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

  #-----
  # Item-Loop, negative gambles,gamble B
  v.x.b[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.b[j,i,1]),alpha[j])
  v.y.b[i,j] <- lambda[j]*(-1) * pow((-1 * prospects.b[j,i,3]),alpha[j])

  #2parameter weighting function
  w.x.b[i,j] <- 1-(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])+pow(prospects.b[j,i,2],gamma.loss[
j])))

  w.y.b[i,j] <- 1-w.x.b[i,j]#delta[j]*pow(prospects.b[j,i,4],gamma[j])/
(delta[j]*pow(prospects.b[j,i,4],gamma[j])+pow((1-prospects.b[j,i,4]),gamma[j]))

  Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]

  #-----
  # Item-Loop, negative gambles,choice-rule
  binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))
  rawdata[i,j] ~ dbern(binval[i,j])
}

for (i in 101:138)# Item-Loop, mixed gambles,gamble A
{
  v.x.a[i,j] <- pow(prospects.a[j,i,1],alpha[j])
  v.y.a[i,j] <- (-1 * lambda[j]) * pow((-1 * prospects.a[j,i,3]),alpha[j])

  #2parameter weighting function
  w.x.a[i,j] <- delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])/
(delta.gain[j]*pow(prospects.a[j,i,2],gamma.gain[j])+pow((prospects.a[j,i,4]),gamma.gai
n[j]))

  w.y.a[i,j] <- delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])/
(delta.loss[j]*pow(prospects.a[j,i,4],gamma.loss[j])+pow((prospects.a[j,i,2]),gamma.los
s[j]))

  Vf.a[i,j] <- w.x.a[i,j] * v.x.a[i,j] + w.y.a[i,j] * v.y.a[i,j]

```

HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

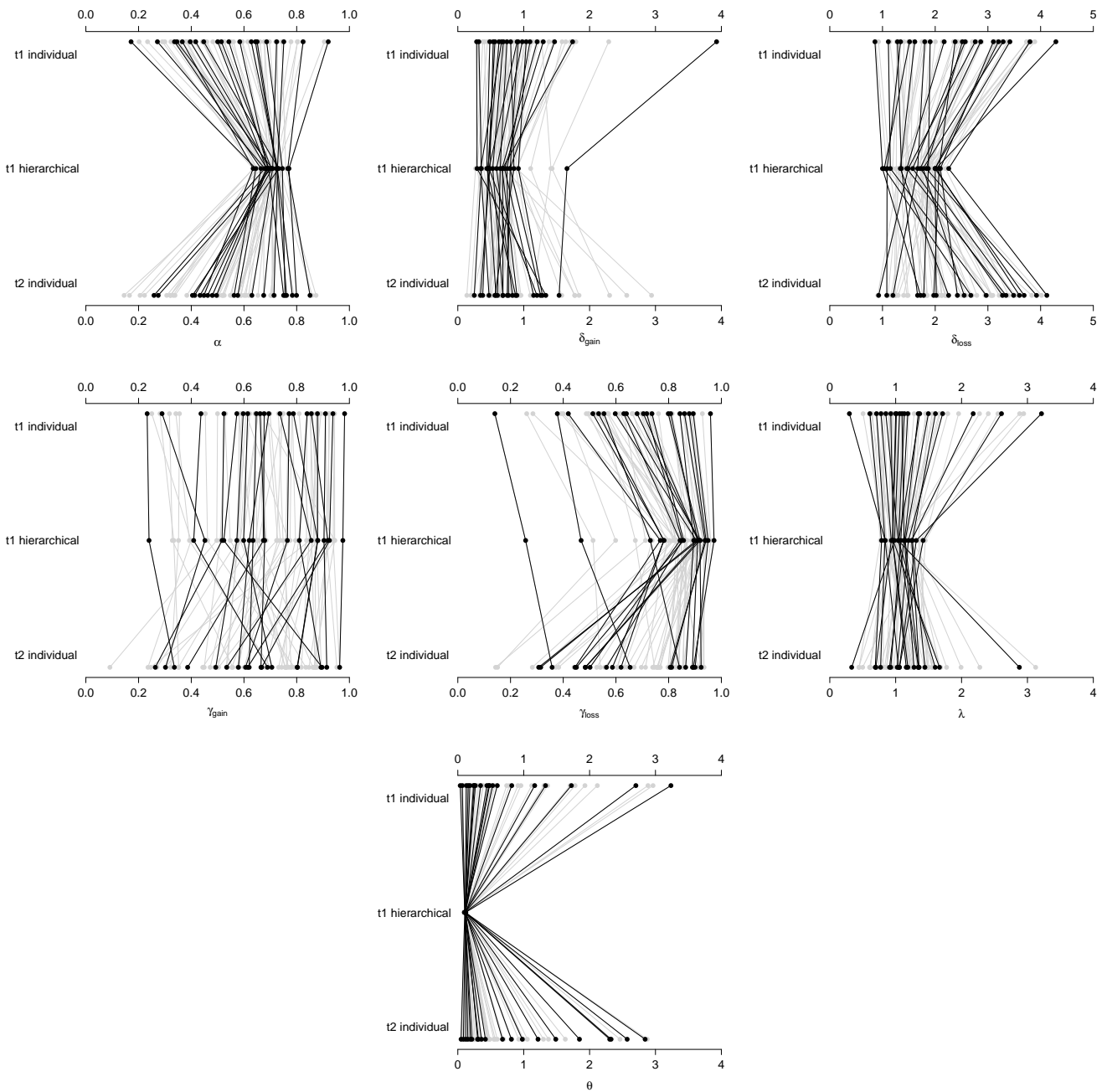
```
#-----  
# Item-Loop, mixed gambles, gamble B  
  
v.x.b[i,j] <- pow(prospects.b[j,i,1],alpha[j])  
v.y.b[i,j] <- (-1 * lambda[j]) * pow((-1 * prospects.b[j,i,3]),alpha[j])  
  
#2parameter weighting function  
w.x.b[i,j] <- delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])/  
(delta.gain[j]*pow(prospects.b[j,i,2],gamma.gain[j])+pow(prospects.b[j,i,4],gamma.gain[j]))  
w.y.b[i,j] <- delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])/  
(delta.loss[j]*pow(prospects.b[j,i,4],gamma.loss[j])+pow(prospects.b[j,i,2],gamma.loss[j]))  
  
Vf.b[i,j] <- w.x.b[i,j] * v.x.b[i,j] + w.y.b[i,j] * v.y.b[i,j]  
  
#-----  
# Item-Loop, mixed gambles, choice-rule  
  
binval[i,j] <- (1)/(1+exp((-1*luce[j])*(Vf.b[i,j]-Vf.a[i,j])))  
rawdata[i,j] ~ dbern(binval[i,j])  
}  
}
```

Data Provided from Outside BUGS:

same as for the independent code

Shrinkage for CPT Parameters

In resemblance to Figure 4, the plots display mean posterior estimates of the free parameters in CPT separately for each individual at t1 and t2 (upper and lower row) and the hierarchically estimated parameters at t1 (middle row). For illustrative purposes, the points indicate a subset of 20 participants spaced out across the whole data range.



HIERARCHICAL BAYESIAN MODELING – SUPPLEMENTARY MATERIAL

Shrinkage for TAX parameters

In resemblance to Figure 4, the plots display mean posterior estimates of the free parameters in TAX separately for each individual at t1 and t2 (upper and lower row) and the hierarchically estimated parameters at t1 (middle row). For illustrative purposes, the points indicate a subset of 20 participants spaced out across the whole data range.

