Testing Adaptive Toolbox Models: A Bayesian Hierarchical Approach

Online Supplemental Material

Benjamin Scheibehenne, Jörg Rieskamp, & Eric-Jan Wagenmakers

Details of the BUGS Sampling

The computational efficiency of the BUGS (Bayesian Analysis Using Gibbs Sampling) code used for the simulation, and in most cases also for the analysis of the empirical data, was increased through the use of pseudo-priors and by setting a prior on the model indicator variable such that both models were evaluated about equally often (for details of this procedure see Carlin & Chib, 1995; Kruschke, 2011; Lodewyckx, Lee, & Wagenmakers, 2011; Ntzoufras, 2009). For each (simulated) set of data, this required several calls to BUGS. We first estimated the posterior parameter distributions of each choice model separately to obtain the pseudo-priors. Next, both models were estimated simultaneously to obtain a prior estimate on the model indicator variable. In a final third step, the Bayes factor was estimated from the model indicator.

During each of these steps, four independent Monte Carlo Markov chain processes were run. We obtained a total of 100,000 representative samples from the (joint) posterior distributions for most analyses except for the analyses of the exemplar model, where only 50,000 samples were drawn to reduce the sampling time. Chains were stored after an initial "burn-in" of 6,000 steps to ensure that the obtained estimates were unaffected by the random starting values of each chain. To reduce memory load, only every 10th sample was stored for most analyses, so the actual chain length before this thinning procedure was 1,000,000. For the exemplar model a thinning of 5 was used, and for the group-level simulation, a thinning of 100 was used.

For all estimated parameters, convergence to stationary sample distributions was confirmed based on visual inspections of the trace plots, the autocorrelation within the chains, and the Gelman–Rubin statistic (Gelman & Rubin, 1992), which indicated that the whole range of the posterior parameter distributions was explored and that the sampling procedure in BUGS was efficient. To further ensure that the model estimation in BUGS worked well, we checked if the posterior estimates of the parameters accurately recovered the true parameter values used when simulating the data. As can be seen from the respective parameter recovery plots below, this was the case for the individual as well as the group-level data.

Implementation of a Single Strategy A_ε and a Toolbox $TB_{A,B}$ in BUGS

The following code shows the schematic implementation of a single strategy and a toolbox consisting of two strategies in BUGS.

```
model {

   #prior on epsilon (application error)
   #epsilon ranges from 0 (deterministic choice) to .5 (random choice)

   epsilon2.tb~dbeta(1,1)    #prior on epsilon (application error)
   epsilon.tb<-epsilon2.tb/2


   #prior on beta (probability of A over B)
   beta.tb~dbeta(1,1)

   for (cc in 1:nOptions) {
      p.B[cc] <- predB[cc] * (1-epsilon.tb) + (1-predB[cc]) * epsilon.tb
      p.A[cc] <- predA[cc] * (1-epsilon.tb) + (1-predA[cc]) * epsilon.tb
      p.TB[cc]<- pA.A[cc] * beta.tb + pA.B[cc] * (1-beta.tb)

      choices[cc] ~ dbern(pA.TB[cc])
   }
}

#Note: Model reduces to "only" strategy A if beta.tb is set to 1.
#For beta.tb=0, the model reduces to "only" strategy B
```

Data supplied from outside BUGS:

- nOptions = number of pairwise choices (choice of option X coded as 1, choice of option Y coded as 0)
- predB = deterministic predictions of strategy B [1 × nOptions]
- predA = deterministic predictions ofstrategy A [1 × nOptions]
- choices = actual choices of one individual [1 × nOptions]

Note that for most applications, the Bernoulli distribution in the BUGS code was replaced with a binomial distribution to increase sampling efficiency. See following code for details.

BUGS Code for Comparison Between $A_\varepsilon/B_\varepsilon$ and $TB_{A,B}$ for a Single Individual

The following code provides the implementation of the comparison between a single model against a toolbox in BUGS.

```
model {

   modelIdx ~ dcat(p[])
   p[1] <- .5   #1 = prediction TB
   p[2] <- .5   #2 = prediction A

   epsilon2.tb ~ dbeta(1,1) #prior on epsilon (application error)
   epsilon.tb <- epsilon2.tb/2

   beta.tb    ~ dbeta(1,1)  #prior on beta (probability of A over B)
   epsilon2.A ~ dbeta(1,1) #prior on epsilon for strategy A
   epsilon.A <- epsilon2.A/2

   #probabilities for the four possible cases of pA.TB
   pX[1,1] <- (1-epsilon.tb)*beta.tb+(1-epsilon.tb)*(1-beta.tb) #A & B
   pX[1,2] <- (1-epsilon.tb)*beta.tb+epsilon.tb    *(1-beta.tb) #A only
   pX[1,3] <- epsilon.tb    *beta.tb+(1-epsilon.tb)*(1-beta.tb) #B only
   pX[1,4] <- epsilon.tb    *beta.tb+epsilon.tb    *(1-beta.tb) #not A nor B

   #probabilities for the four possible cases of pA.TTB
   pX[2,1] <- (1-epsilon.A) #A & B
   pX[2,2] <- (1-epsilon.A) #A only
   pX[2,3] <- epsilon.A     #B only
   pX[2,4] <- epsilon.A     #neither A nor B

   for (cc in 1:4) {
    choices[cc] ~ dbin(pX[modelIdx,cc],trials[cc])
   }
}
```

Data supplied from outside BUGS:

- choices[1] = number of X choices that match the deterministic predictions of A and B
- choices[2] = number of X choices that match the deterministic predictions of A only
- choices[3] = number of X choices that match the deterministic predictions of B only
- choices[4] = number of X choices that match the deterministic predictions of neither B nor A
- trials[1:4] = total number of choices for each of the four possible cases

Recovery of Parameter Values Used in the Simulation (Individual-Level Data)

Figure A below plots the parameters used in the choice simulation (*x*-axes) against the parameter estimates obtained from the BUGS model (*y*-axes). Error bars indicate the region of highest posterior density (HPD_{95}) of the posterior estimates. The figure shows that the true parameter values are accurately recovered. If ε is close to 1 (indicating random choice), the error margins around the estimated β parameters become very wide. This seems plausible as random choices can be described with strategy A_ε and B_ε alike.

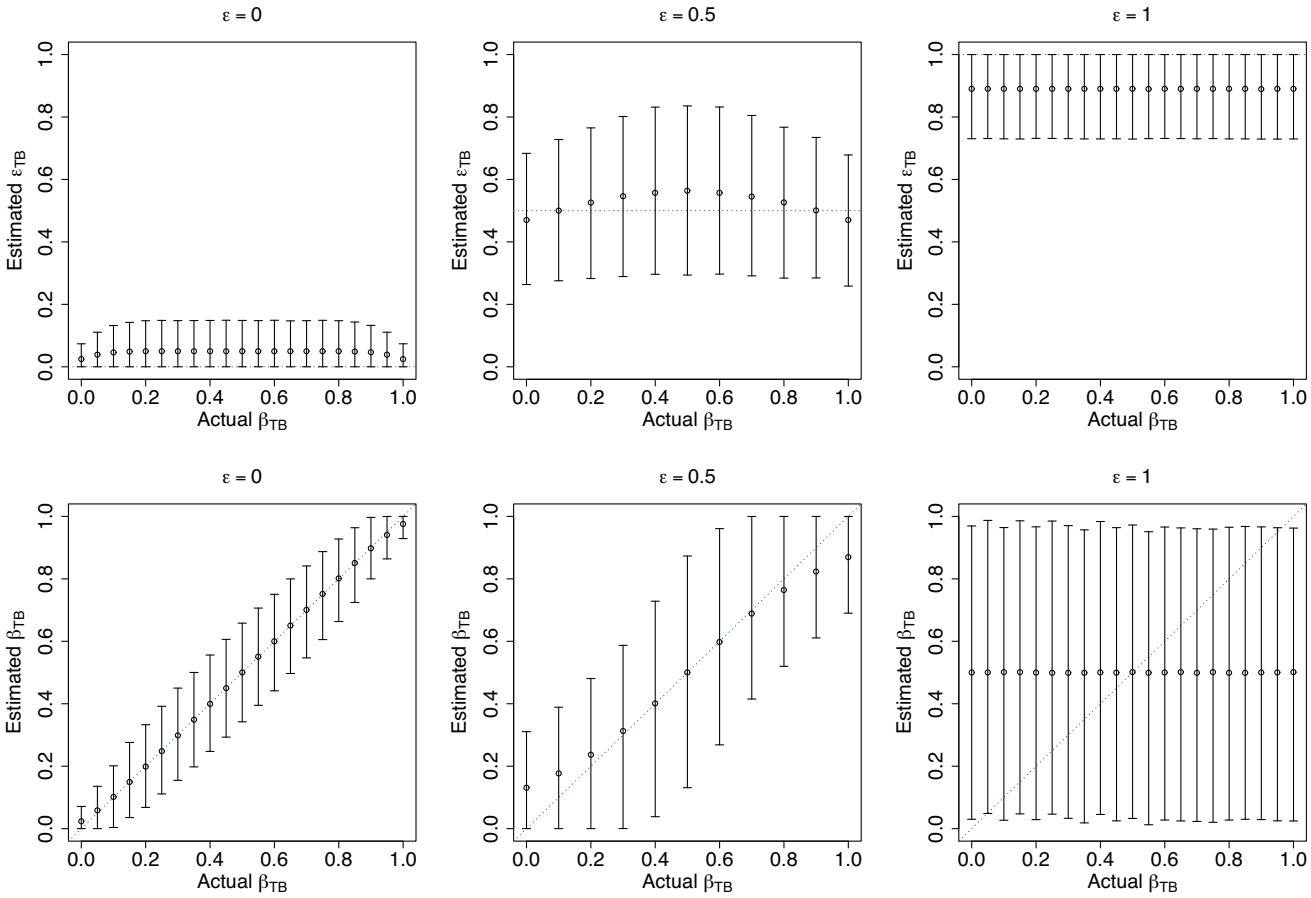

Figure A. Parameter recovery for individual-level simulation.

Parameter Estimates for the Data in Rieskamp and Otto (2006)

Beta Parameter

Figure B shows the estimate of β, the probability of the take-the-best strategy (TTB_ε) in the toolbox, for each individual decision maker in Rieskamp and Otto (2006), ordered by the Bayes factor. The filled triangles indicate cases for which the probability of the toolbox was higher than the probability of the reinforced model (weighted additive [WADD] in the compensatory condition and TTB in the noncompensatory condition). Error bars indicate the region of highest posterior density (HPD_{95}) of the posterior estimates.

These parameter estimates seem plausible in the light of the model comparison. For example, in the left figure, individuals with low β estimates (indicating predominantly WADD choices) were better described by a WADD-only model as compared to a toolbox.

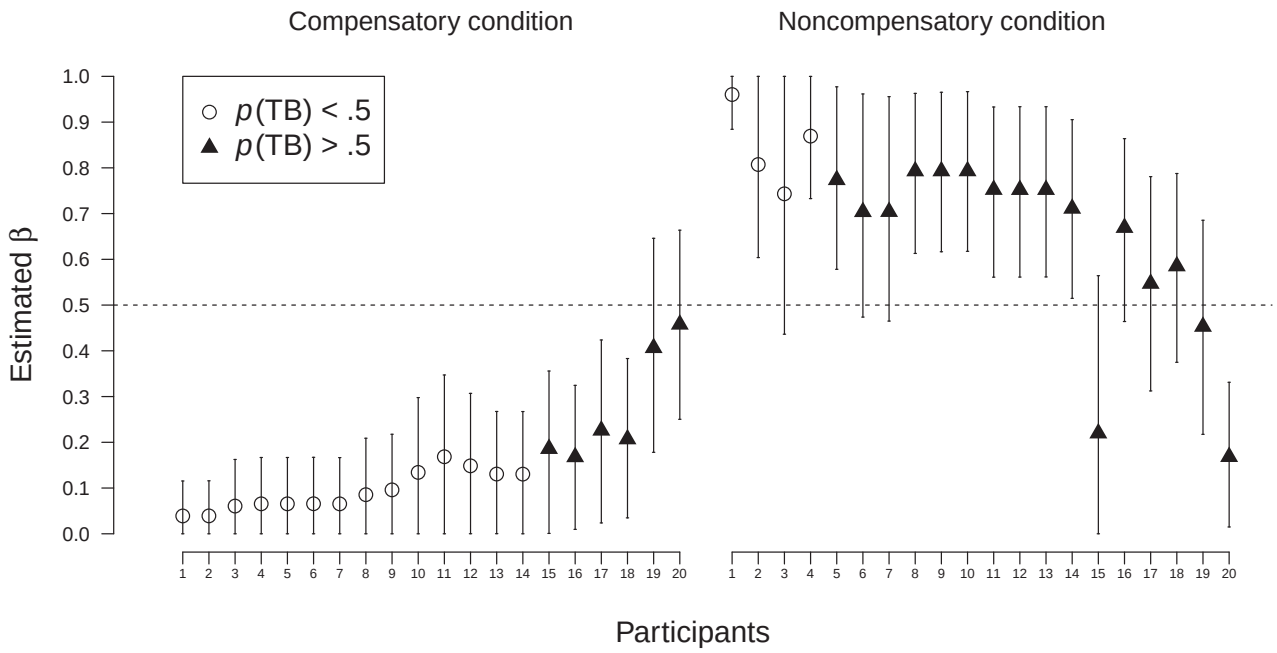

Figure B. Estimates for individual β parameters in Rieskamp and Otto (2006).

Error Parameter

Figure C shows the estimated application error ε for each individual decision maker in Rieskamp and Otto (2006), ordered by the Bayes factor. Triangles indicate the toolbox error, circles the error for the reinforced model (weighted additive [WADD] in the compensatory condition and take-the-best [TTB] in the noncompensatory condition). Error bars indicate the region of highest posterior density (HPD_{95}) of the posterior estimates. The figure shows that, for example, individuals who are best described by a toolbox in the noncompensatory condition are characterized by high application errors for the TTB model. This seems plausible as TTB could not describe their choices very well. The figure also indicates that a few individuals might not be well described by a toolbox either. For example, participant 15 in the noncompensatory condition still has a rather high implementation error.

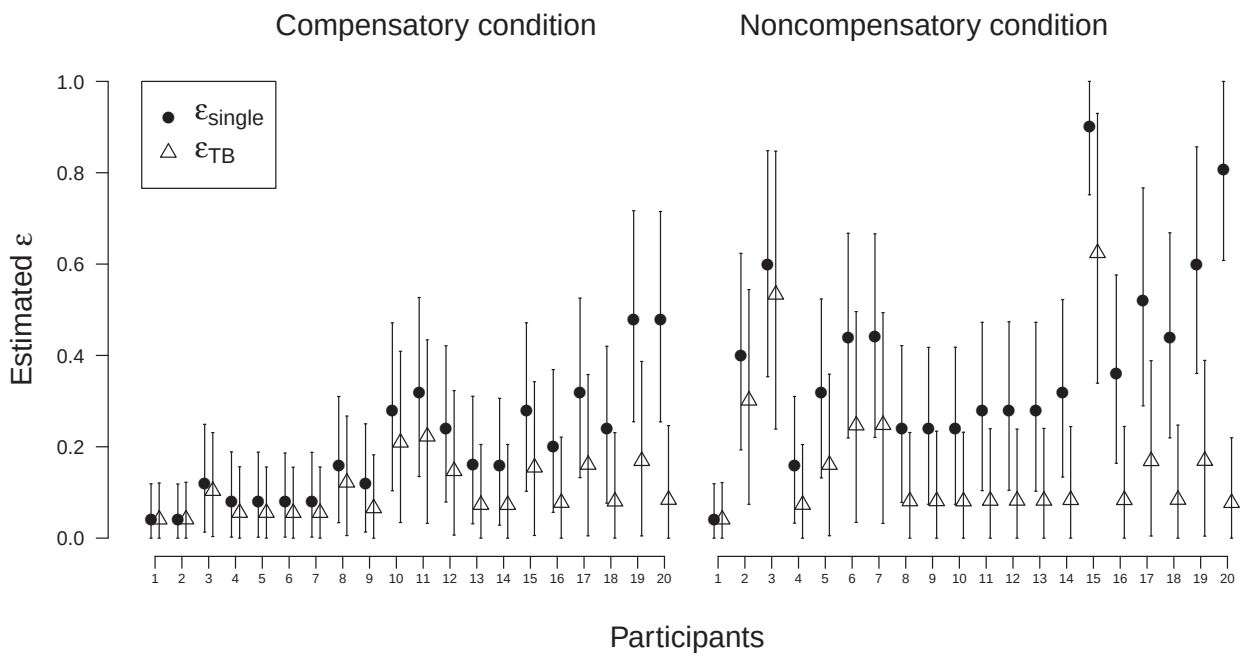

Figure C. Estimates for individual ε parameters in Rieskamp and Otto (2006).

Lewandowsky, Kalish, and Ngang (2002) – "Quadratic" Experts

Figure D shows the same analysis as in Figure 6 in the manuscript except that here, the two partitioning functions are implemented as quadratic rather than linear regressions. Here, the evidence in favor of knowledge partitioning (gray bars) in the systematic context condition.is stronger as compared to the case of liner experts This suggests that participants did partition their knowledge but that they might have used two separate quadratic rules rather than linear rules.

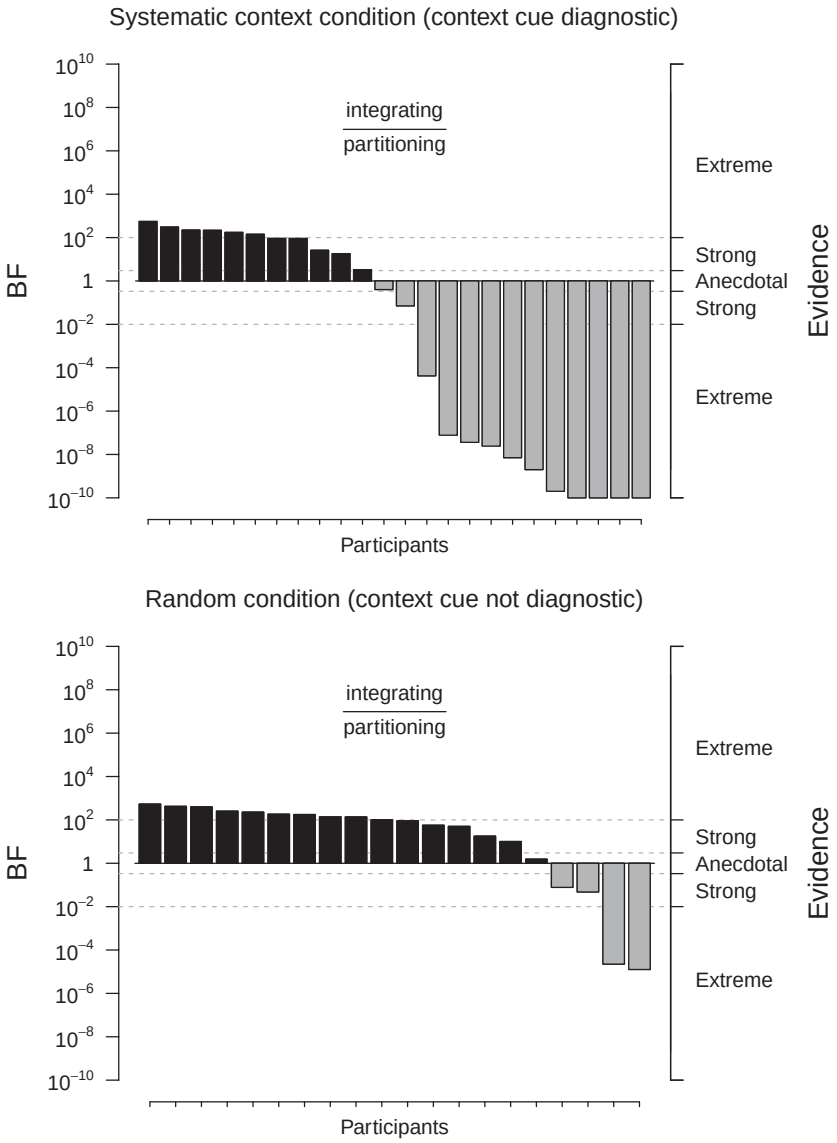

Figure D. Reanalysis of Lewandowsky, Kalish, and Ngang (2002) with "quadratic" rather than linear experts.

BUGS Code for the Exemplar Model

The following code describes the implementation of the exemplar model on the individual level in BUGS for the analysis of Rieskamp and Otto (2006).

```
model {

     #prior on alpha parameter
     alpha ~ dbeta(1,1) # denoted "s" parameter in the text

     for (cc in 1:nOptions) {

        #loop to compare option on hand with previous options
        for(dd in 1:143) { pow.alpha[cc,dd] <- pow(alpha,sim.matrix[cc,dd]) }

       denominator[cc] <- inprod(pow.alpha[cc,],valid.matrix[cc,])
       nominator[cc]   <- inprod(pow.alpha[cc,],kBase.matrix[cc,])

       pExemplar[cc] <- nominator[cc]/denominator[cc]

       choices[cc] ~ dbern(pExemplar[cc])
     }
}
```

Data supplied from outside BUGS:

- nOptions = number of pairwise choices
- choices = actual choices of one individual [1 × nOptions]
- sim.matrix = matrix [nOptions × 143] = similarity of the last 48 with the previous 143 options.

Individual Parameter Estimates for the Exemplar Model

Figure E shows the estimated attention weight parameter *s* in the exemplar model for each individual decision maker in Rieskamp and Otto (2006), ordered by the Bayes factor. The triangles indicate cases for which the probability of the toolbox (TB) was higher than the probability of the exemplar model. Error bars indicate the region of highest posterior density (HPD_{95}) of the posterior estimates.

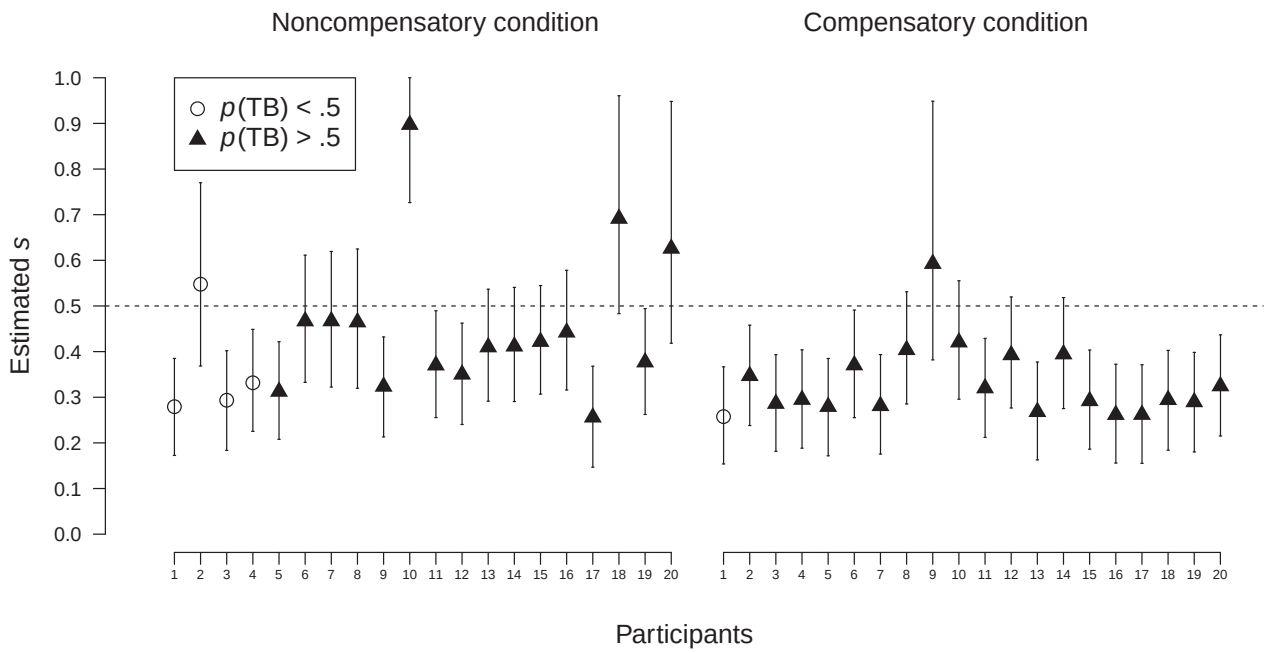

Figure E. Estimates for individual s parameters (exemplar model) in Rieskamp and Otto (2006).

BUGS Model for the Group-Level Toolbox

Graphical Model

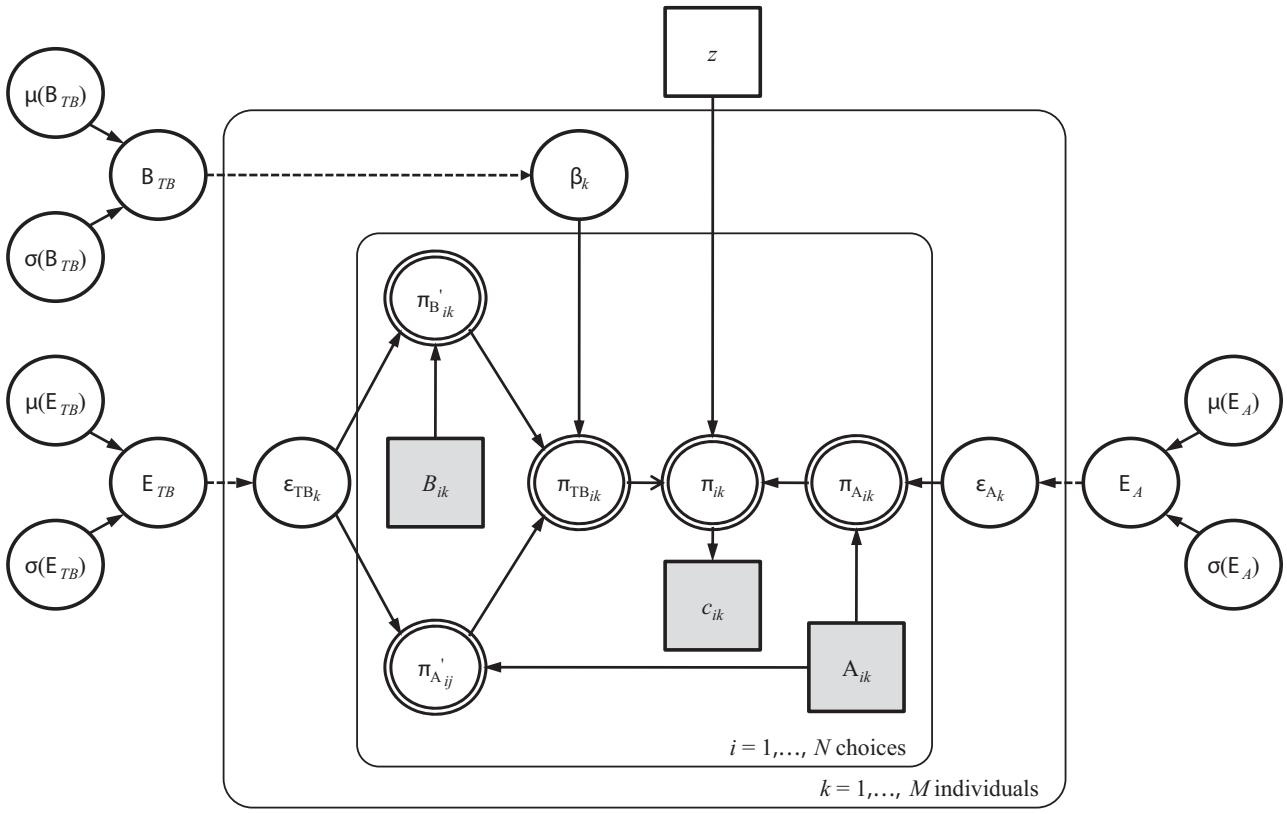

Figure F. Extension of the graphical model for single decision makers (Figure 1) to the group level. All individual toolbox (TB) and single model (A) parameters stem from normally distributed group-level parameters (indicated by capital Greek letters E and B) that are defined by its respective means (μ) and variances (σ). Group-level parameters mapped to individual-level rate scale (ranging from 0 to 1) through an inverse probit transformation (indicated by dashed arrows).

BUGS model

The following code provides the BUGS code for the group-level comparison between a single strategy and a toolbox consisting of two strategies. Here, the individual parameters stem from a group-level distribution.

```
model {

   #Model Index
   modelIdx ~ dcat(p[])
   p[1] <- pTB   # probability of Toolbox (idx = 1)
   p[2] <- 1-pTB


      #Group-level for epsilon
       prior.e.mean     <- 0          #prior on group-mean
       prior.e.mean.tau <- pow(1,-2)  #prior on precision of group mean
       group.e.mean      ~ dnorm(prior.e.mean,prior.e.mean.tau

       group.e.sd01 ~ dbeta(1,1)       #prior on sd for epsilon
       group.e.sd  <- group.e.sd01 * 3.999 + .001
       group.e.tau <- pow(group.e.sd,-2)

      #Group-level for beta (on probit scale)
       prior.mean.beta <- 0        #prior on mean of beta
       prior.tau.beta  <- pow(1,-2)
       group.beta.mean ~ dnorm(prior.mean.beta, prior.tau.beta)

       group.beta.sd01 ~ dbeta(1,1) #sd = 1 -> flat prior on the 0-1 scale
       group.beta.sd  <- group.beta.sd01 * 3.999 + .001
       group.beta.tau <- pow(group.beta.sd,-2)

      #Group-level for epsilon (TTB)
       prior.e.ttb.mean     <- 0  #prior on group-mean
       prior.e.ttb.mean.tau <- pow(1,-2)  #prior on precision of group mean
       group.e.ttb.mean      ~ dnorm(prior.e.ttb.mean,prior.e.ttb.mean.tau)

       group.e.ttb.sd01 ~ dbeta(1,1)  #prior on sd for epsilon
       group.e.ttb.sd  <- group.e.ttb.sd01 * 3.999 + .001
       group.e.ttb.tau <- pow(group.e.ttb.sd,-2)
```

```
for (vpn in 1:nParticipants) {

   #individual for epsilon TB
    epsilon2.norm[vpn] ~ dnorm(group.e.mean ,group.e.tau) I(-4,4)
    epsilon2.tb[vpn]  <- phi(epsilon.norm[vpn])
    epsilon.tb[vpn]   <- epsilon2.tb[vpn]/2

   #individual for beta
    beta.norm[vpn] ~ dnorm(group.beta.mean,group.beta.tau)  I(-4,4)
    beta.tb[vpn]  <- phi(beta.norm[vpn])

   #individual for epsilon TTB
    epsilon.ttb.norm[vpn] ~ dnorm(group.e.ttb.mean ,group.e.ttb.tau) I(-4,4)
    epsilon2.ttb[vpn]    <- phi(epsilon.ttb.norm[vpn])
    epsilon.ttb[vpn]     <- epsilon2.ttb[vpn]/2


   #probabilities for the four possible cases of pA.TB

   pA[1,vpn,1]<- (1-epsilon.tb[vpn])*beta.tb[vpn]
                +(1-epsilon.tb[vpn])*(1-beta.tb[vpn]) #TTB & WADD prediction

   pA[1,vpn,2]<- (1-epsilon.tb[vpn])*beta.tb[vpn]
                +epsilon.tb[vpn]*(1-beta.tb[vpn])     #TTB only prediction

   pA[1,vpn,3]<- epsilon.tb[vpn]*beta.tb[vpn]
                +(1-epsilon.tb[vpn])*(1-beta.tb[vpn]) #WADD only prediction

   pA[1,vpn,4]<- epsilon.tb[vpn]*beta.tb[vpn]
                +epsilon.tb[vpn]*(1-beta.tb[vpn])     #neither TTB nor WADD

   #probabilities for the four possible cases of pA.TTB
    pA[2,vpn,1] <- (1-epsilon.ttb[vpn]) #TTB & WADD prediction
    pA[2,vpn,2] <- (1-epsilon.ttb[vpn]) #TTB only prediction
    pA[2,vpn,3] <- epsilon.ttb[vpn]     #WADD only prediction
    pA[2,vpn,4] <- epsilon.ttb[vpn]     #neither TTB nor WADD prediction


   #1 -- TB   2 -- TTB
     for (cc in 1:4) {
      choicesA[vpn,cc] ~ dbin(pA[modelIdx,vpn,cc],trials[cc])
     }
  }
}
```

Parameter Recovery of the Group-Level Simulation

Group-level distributions

Figure G shows the results of the parameter recovery for the group-level simulation (part III of the manuscript), for the group-level parameters E_{TB} and B_{TB} (*x*-axes), separately for $\varphi_A = 1$ and $\varphi_A = .9$. Error bars indicate the region of highest posterior density (HPD_{95}) of the posterior estimates (*y*-axis). The figure indicates that the group-level estimates are very close to the actual parameters used in the simulation. If E is close to 1 (indicating random choice), the error margins around the estimated B parameters become very wide. This is plausible as the individual β parameters are not identified for random choices.

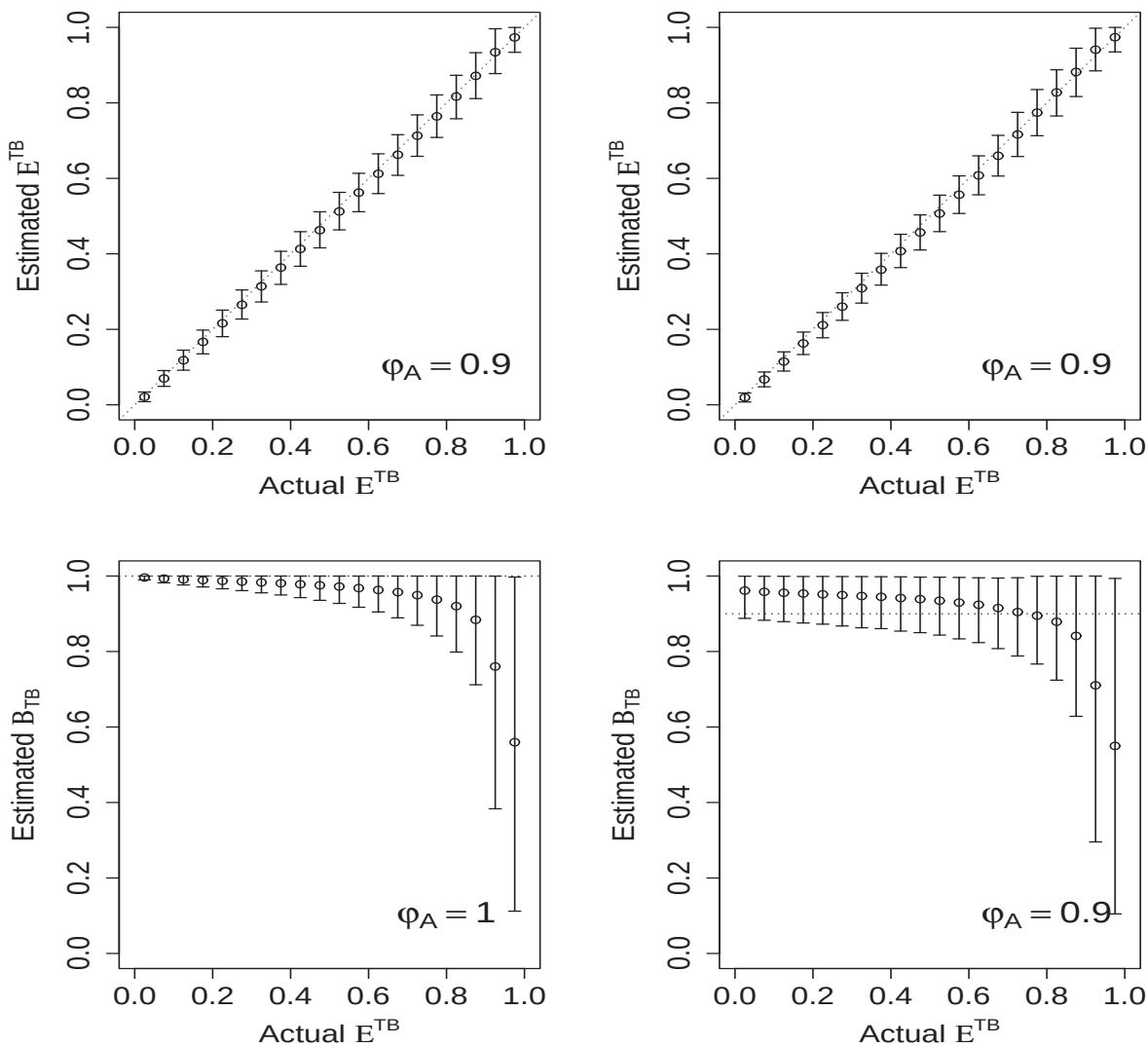

Figure G. Recovery for group-level parameters in the group-level simulation.

Individual-level distribution of the error parameter

Figures H and I show the individual parameter recovery for the toolbox error parameter ε^{TB} in the group-level simulation for each of the 20 synthetic individuals. The recovery is split-up for different simulation values for the mean group-level application errors E across the panels. Figure H shows the cases where each of the 20 individuals in the group used A_ε (i.e., $\varphi_A = 1$). Figure I shows the cases where only 18 of the 20 individuals in the group used A_ε (i.e., $\varphi_A = .9$). As can be seen from the figures, the error parameters are accurately recovered on the individual level.

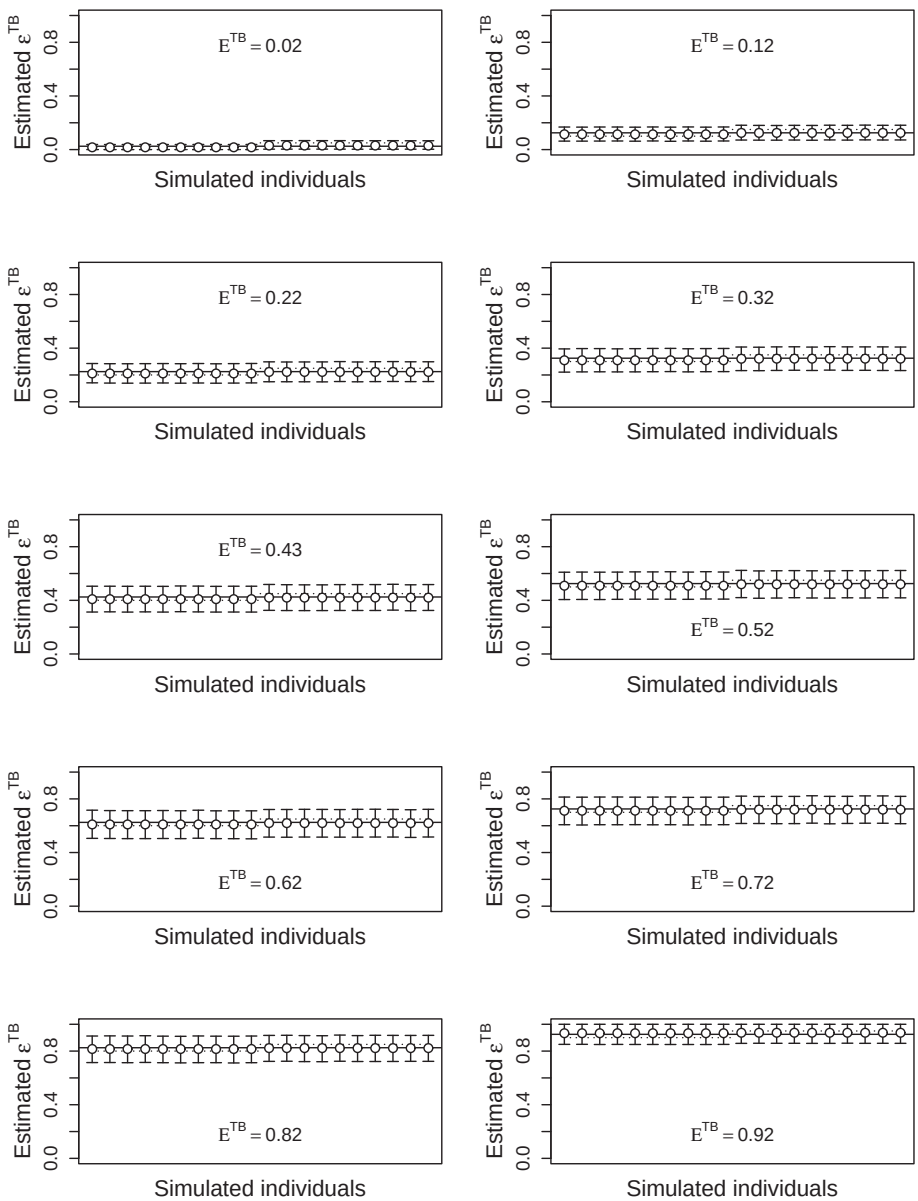

Figure H. Recovery for individual-level ε^{TB} parameters in the group-level simulation, $\varphi_A = 1$.

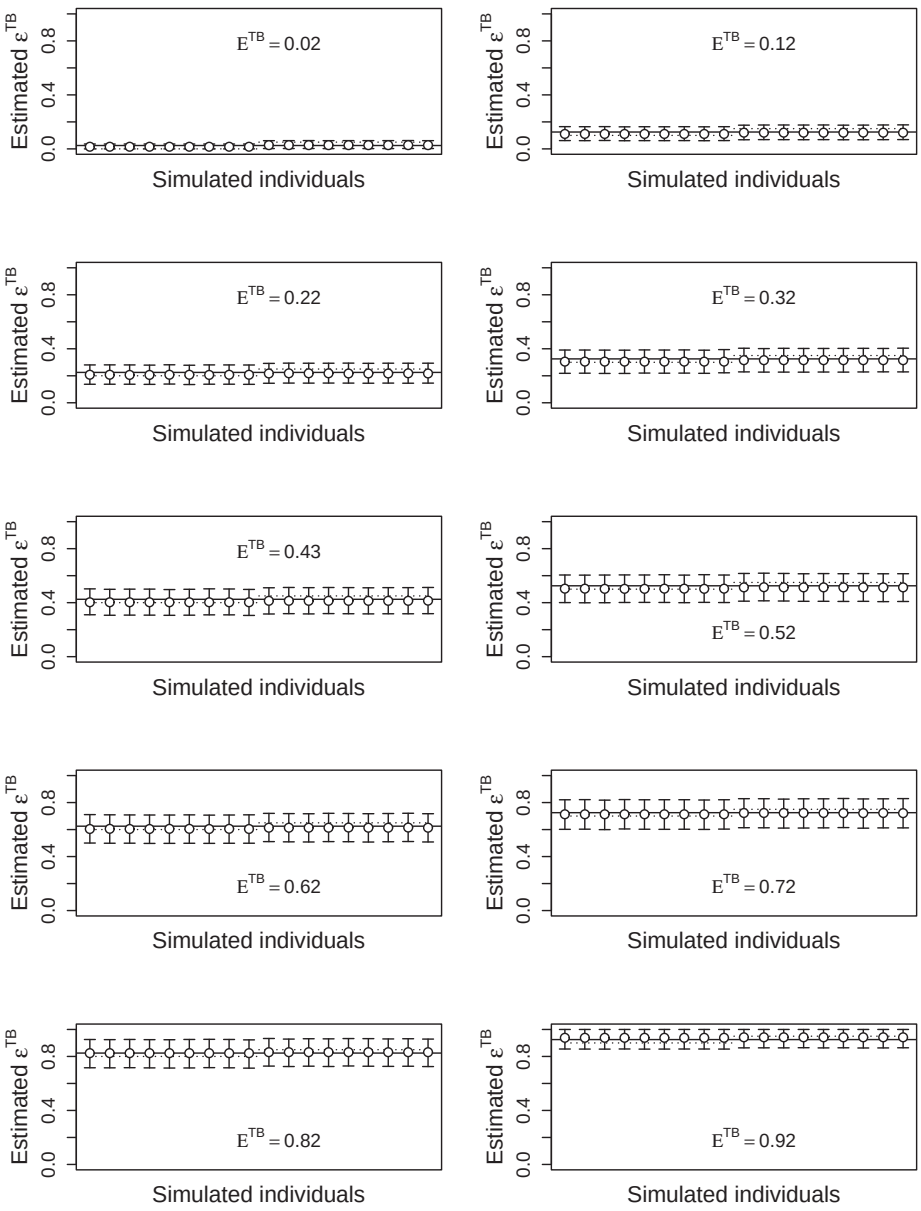

Figure I. Recovery for individual-level ε^{TB} parameters in the group-level simulation, $\varphi_A = .9$.

Individual-level distribution of the beta parameter

Figure J and K show the parameter recovery for individual β parameters for different values of mean group-level application errors E (across panels). For Figure J, all of the 20 individuals in the simulation used A_ε (i.e., $\varphi_A = 1$, indicated by the horizontal line). For Figure K, 18 out of the 20 used A_ε and 2 used B_ε (i.e., $\varphi_A = .9$). In both cases, the individual β parameters get accurately recovered. Note that if $\varphi_A = .9$, the individual estimates accurately distinguished B_ε and A_ε users as long as the application error was not too high, which is not obvious from the estimates on the group-level outlined above .If the application error is close to 1 (indicating random choice), the error margins around the estimated β parameters become very wide, which seems plausible, and the individual estimates "shrink" towards the group-mean because the genuine number of B_ε choices now became very small.

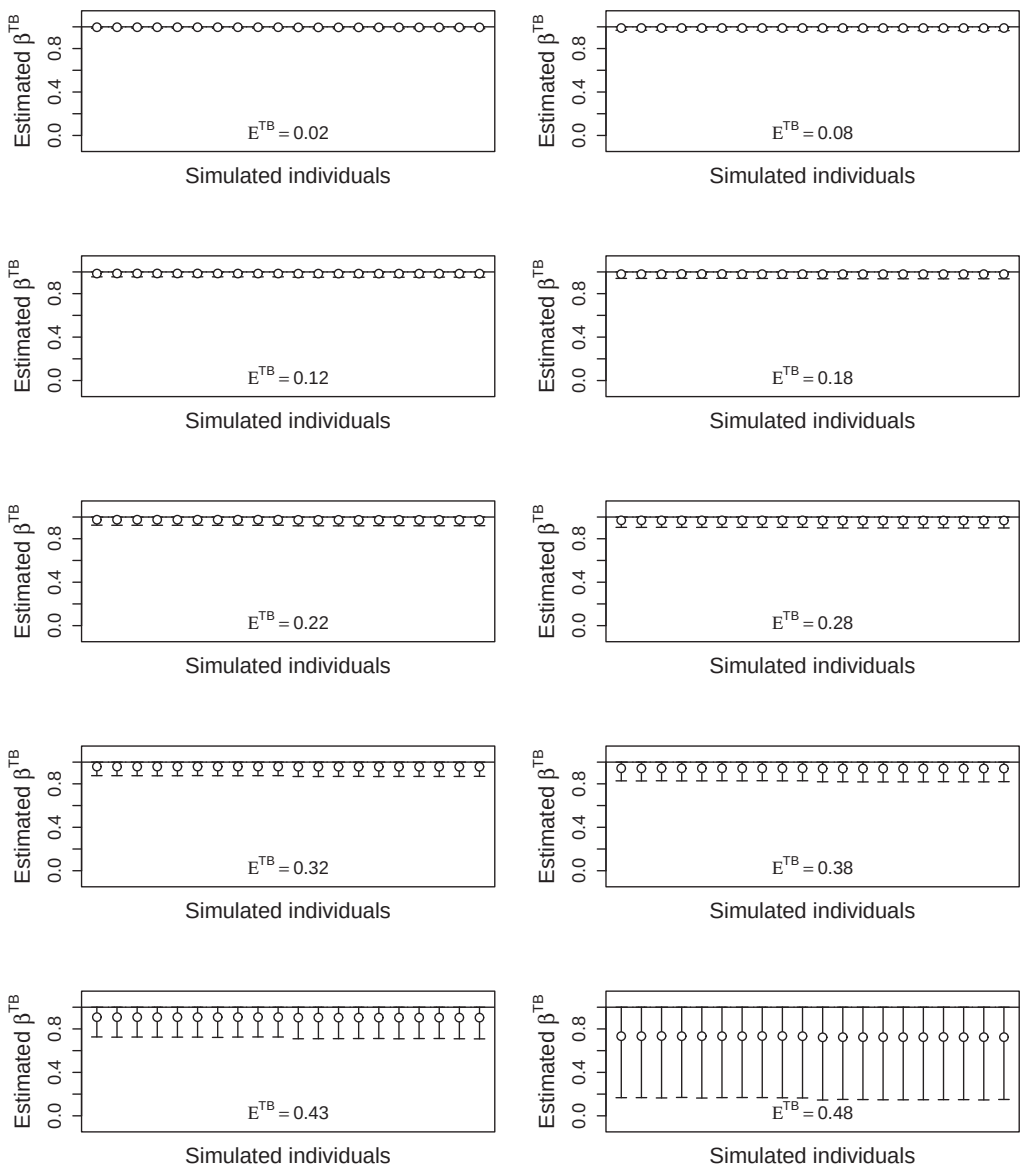

Figure J. Recovery for individual-level β^{TB} parameters in the group-level simulation, $\varphi_A = 1$.

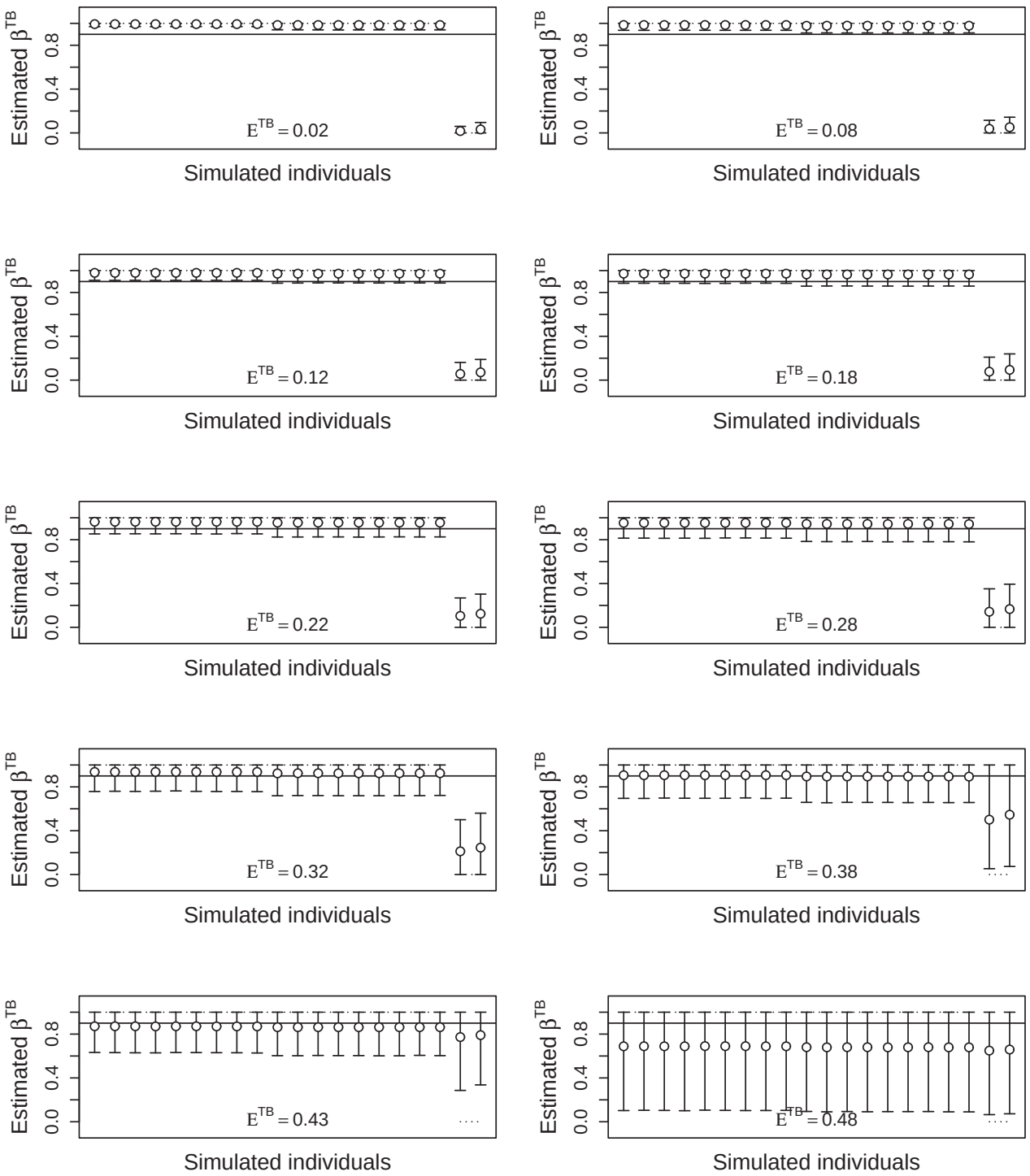

Figure K. Recovery for individual-level β^{TB} parameters in the group-level simulation, $\varphi_A = .9$.

Comparing the Bayesian Information Criterion (BIC) and the Bayes Factor (BF)

Toolbox With Two Independent Errors

The following BUGS code introduces a toolbox with an independent error for each tool. This case extends the toolbox model used in the previous analyses where a common error was assumed for all tools. Together with the code outlined next, this model provided the basis of comparing BIC and BF.

```
model {

# e.TB.R1 and e.TB.R2 are independent

    #Model indicator
     modelProb[1]<-pR1    #index 1 = probability for simple model (only R1)
     modelProb[2]<-1-pR1  #index 2 = probability for toolbox (R1 and R2)
     idx ~ dcat(modelProb[])

       #Prior on R1 error
        e.R1 ~ dunif(0,.5)

       #Prior on error in the toolbox
        e.TB.R1 ~ dunif(0,.5)     #error of R1
        e.TB.R2 ~ dunif(0,.5)     #error of R2 independent of R1

       #probability of using R1 in the toolbox
        beta ~  dbeta(1,1)

     for (cc in 1:nItems) {

     #Toolbox model
      pA.R1.TB[cc]<-predR1[cc]*(1-e.TB.R1)+(1-predR1[cc])*e.TB.R1
      pA.R2.TB[cc]<-predR2[cc]*(1-e.TB.R2)+(1-predR2[cc])*e.TB.R2
      pA.TB[cc]<-pA.R1.TB[cc]*beta + pA.R2.TB[cc]*(1-beta)

     #R1 model
      pA.R1[cc]<-predR1[cc]*(1-e.R1)+(1-predR1[cc])*e.R1

      pA[1,cc] <- pA.R1[cc] #index 1 = probability for only R1
      pA[2,cc] <- pA.TB[cc] #index 2 = probability for TB
      answers[cc] ~ dbern(pA[idx,cc])
     }
}
```

BUGS Model of Toolbox With Dependent Errors

The following BUGS code introduces a toolbox where the error of one tool (R1) must always be higher than the error of the other tool (R2), thus introducing a change in the functional form as compared to the case of independent errors outline above.

```
model {

# e.TB.R2 >= e.TB.R1

    #Model indicator
     modelProb[1]<-pR1    #index 1 = probability for simple model (only R1)
     modelProb[2]<-1-pR1  #index 2 = probability for toolbox (R1 and R2)
     idx ~ dcat(modelProb[])

       #Prior on R1 error
        e.R1 ~ dunif(0,.5)

       #Prior on error in the toolbox
        e.TB.R1 ~ dunif(0,.5)   #error of R1
        min.e.TB.R2 <- e.TB.R1
        e.TB.R2 ~ dunif(min.e.TB.R2,.5) #e.TB.R2 >= e.TB.R1

       #probability of using R1 in the toolbox
        beta ~  dbeta(1,1)

     for (cc in 1:nItems) {

     #Toolbox model
      pA.R1.TB[cc]<-predR1[cc]*(1-e.TB.R1)+(1-predR1[cc])*e.TB.R1
      pA.R2.TB[cc]<-predR2[cc]*(1-e.TB.R2)+(1-predR2[cc])*e.TB.R2
      pA.TB[cc]<-pA.R1.TB[cc]*beta + pA.R2.TB[cc]*(1-beta)

     #R1 model
      pA.R1[cc]<-predR1[cc]*(1-e.R1)+(1-predR1[cc])*e.R1

      pA[1,cc] <- pA.R1[cc] #index 1 = probability for only R1
      pA[2,cc] <- pA.TB[cc] #index 2 = probability for TB
      answers[cc] ~ dbern(pA[idx,cc])
      }
}
```

Difference Between BIC and BF in the Model Comparison

Table A1 and Figure L provide an overview of the Bayes Factors obtained using BUGS, and its approximation using BIC for changes in functional form. Although the BIC is affected by the number of observations, in contrast to BUGS it is not sensitive to changes in functional form (i.e. the difference between the independent and the dependent model implementation).

Data			BF				log(BF)			
			Dependent		Independent		Dependent		Independent	
N	*p*TTB	e	BIC	BF dep	BIC	BF indep	BIC	BF dep	BIC	BF indep
20	0.8	0	0.22	1.22	0.22	0.61	-1.5	0.2	-1.5	-0.5
40	0.8	0	0.50	2.23	0.50	1.00	-0.7	0.8	-0.7	0
60	0.8	0	1.49	4.95	1.49	2.23	0.4	1.6	0.4	0.8
80	0.8	0	4.95	13.46	4.95	6.05	1.6	2.6	1.6	1.8

Table A1: Comparison for dependent and independent error terms

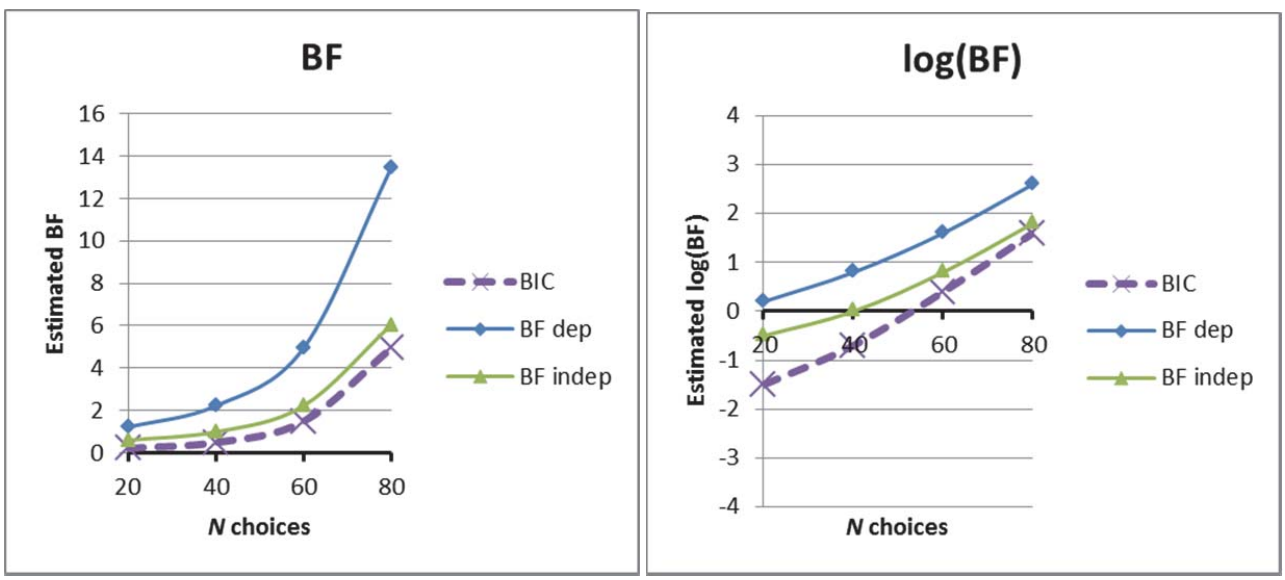

Figure L. Plot of the Bayes factors (BF) in Table A1. Note that BF < 1 and log(BF) < 0 indicates evidence for R1 over the toolbox. BIC: Bayesian information criterion; indep: independent; dep: dependent.

Comparison Between BIC and BF for Different Priors on β

Table A2 and Figure M provide an overview of the Bayes Factors obtained using BUGS, and its approximation using BIC for changes in functional form due to different priors. Although the BIC is affected by the number of observations, in contrast to BUGS it is not sensitive to changes in the prior distributions (i.e. the difference between a uniform or a slightly skewed prior distribution).

Data			BF				log(BF)			
			BF_BIC	BF_BUGS			BF_BIC	BF_BUGS		
N	*p*TTB	e		β(1,1)	β(1,4)	β(1,5)		β(1,1)	β(1,4)	β(1,5)
80	0.8	0	45	19.8	0.9	0.27	3.8	3	-0.1	-1.3
60	0.8	0	11.5	5.7	0.3	0.09	2.4	1.7	-1.3	-2.4
40	0.8	0	3.2	2	0.1	0.04	1.2	0.7	-2.2	-3.1

Table A2: Comparison for different priors

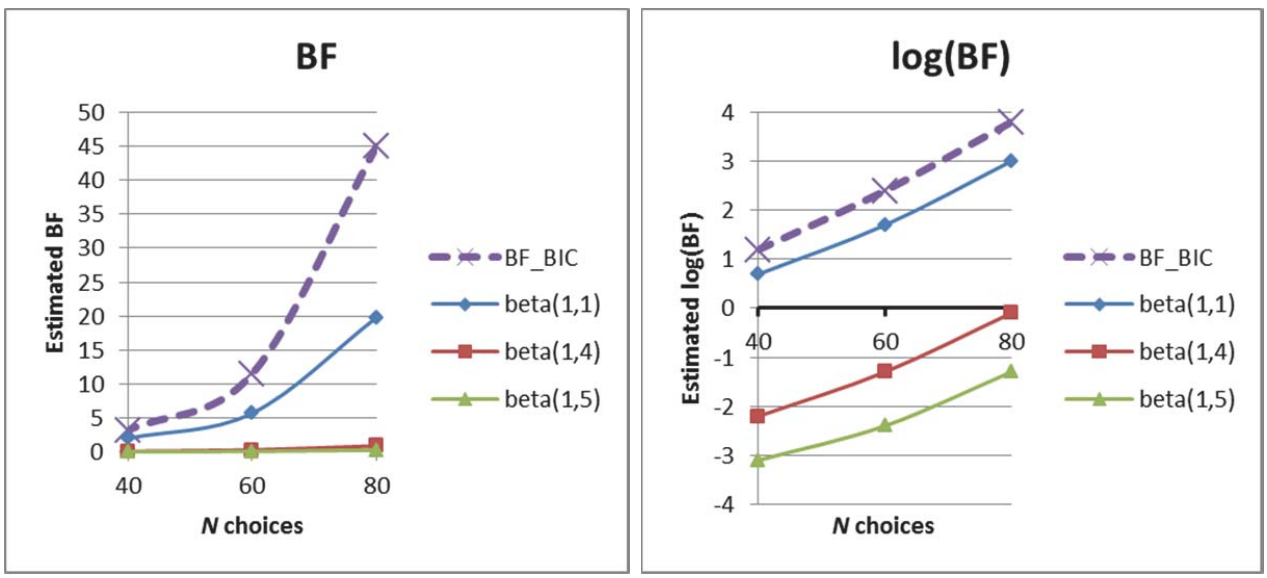

Figure M. Plot of the Bayes factors (BF) in Table A1. Note that BF < 1 and log(BF) < 0 indicates evidence for R1 over the toolbox. BIC: Bayesian information criterion; indep: independent; dep: dependent.

References

Carlin, B. P., & Chib, S. (1995). Bayesian model choice via Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 57*, 473-484. doi: 10.2307/2346151

Gelman, A. & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences, *Statistical Science, 7*, 457-511.

Godsill, S. J. (2001). On the relationship between Markov chain Monte Carlo methods for model uncertainty. *Journal of Computational and Graphical Statistics, 10*, 230–248.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika, 82*, 711–732.

Lodewyckx, T., Kim, W., Tuerlinckx, F., Kuppens, P., Lee, M. D., & Wagenmakers, E.-J. (2011). A tutorial on Bayes factor estimation with the product space method. *Journal of Mathematical Psychology, 55*, 331-347.

Lunn, D. J., Best, N., & Whittaker, J. C. (2009). Generic reversible jump MCMC using graphical models. *Statistics and Computing, 19*, 395–408.

Kruschke, J. K. (2011). *Doing Bayesian Data Analysis. A Tutorial with R and BUGS*. Oxford, UK: Elsevier Academic Press.

Ntzoufras, I. (2009). *Bayesian modeling using WinBUGS*. Hoboken, NJ: Wiley.