Table of Contents

# Data access workflow

# Full Project Workflow

# Power analysis

We conducted a power analysis through simulation using the simulateData function of the lavaan package. On each iteration, we first specified a population model (i.e., the ‘true’ model) with prespecified factor loadings and regression coefficients, and a sample model. Factor loadings in this model were randomly generated between 0.6 and 0.8 following a uniform distribution. Next, we simulated data sets based on the population model. Finally, we fitted the sample model to the simulated data and extracted the beta coefficients and corresponding *p*-values. We generated population models with beta coefficients of 0.08 and 0.1, and simulated data with sample sizes ranging from 1,500, to 8,500 with steps of 1,000. Each combination of coefficients and sample sizes was repeated 500 times, for a total of 8,000 iterations.

The results are shown in Figure S1. The simulations yield power > 90% at around *N* = 3,500 for = 0.08 and *N* = 2,500 for = 0.1. Thus, after taking out 1,500 participants for the training set, we can create two highly powered test sets of 4,250 participants. The simulations indicate that the regression paths in the training set will be underpowered. However, all models converged normally and showed good model fit (lowest CFI = 0.999; highest RMSEA = 0.005), indicating that this sample size is sufficient for initial optimization of the models.

ggplot(power, aes(n, power, group = lhs, color = Type)) +  
 geom\_point() +  
 geom\_line() +  
 geom\_hline(yintercept = 90, linetype = 'dashed') +  
 facet\_wrap(~paste0("\u03B2 = ", beta)) +  
 scale\_y\_continuous(breaks = c(30, 40, 50, 60, 70, 80, 90, 95, 100)) +  
 scale\_x\_continuous(breaks = c(1500, 2500, 3500, 4500, 5500, 6500, 7500, 8500)) +  
 scale\_color\_uchicago() +  
 theme\_classic() +  
 theme(  
 axis.text.x = element\_text(angle = 45, hjust = 1)  
 ) +  
 labs(  
 x = "\nSample size",  
 y = "Power"  
 )

|  |
| --- |
| **Figure S1.**. Results of the power simulations. The dashed line indicates 90% power. |

# Response Distributions of Cognitive Tasks

# Overview of DDM Modeling Procedure

The DDM models will be fit using the *hBayesDM* package (Ahn et al., 2017). The Stan code will be adjusted to fix the starting point to 0.5 and to estimate parameters per task condition. Each model will be fit with three Markov Chain Monte Carlo (MCMC) chains. Each chain will contain 2,000 burn-in samples and 10,000 additional samples. Of these samples, every 10th sample will be retained. Posterior samples of all three chains will be combined, resulting in a posterior sample of 3,000 samples. If a model does not converge properly with these settings, we will increase the amount of samples

Model convergence will be assessed in several ways. First, we will visually inspect the traces, which should not contain any drifts or large jumps. Second, we will calculate the Gelman-Rubin convergence statistic R^ (Gelman & Rubin (1992)), of which all values should be below 1.1. Third, we will assess whether the model provides a good fit to the participants’ data. To do this, we will simulate 50,000 trials of RT and accuracy data using the estimated DDM parameters of each participant. We will then fit the DDM to these simulated data using Kolgomorov-Smirnov estimation. We will then compute correlations between the observed and simulated scores for RTs in the 25th, 50th and 75th percentile of the RT distribution as well as for accuracies. We will consider *r* > .85 to indicate a good model fit. [*Perhaps use DIC to compare models in cases where we fit several models to the data of the same task?*]

In theory, the hierarchical Bayesian framework allows simultaneously estimating DDM parameters, latent measurement models, and the regression paths between them in a single step (e.g., Schubert et al., 2019; Vandekerckhove, 2014). An advantage of this approach is that information regarding estimation uncertainty (e.g., of the DDM parameters) gets integrated in subsequent steps. However, this approach is very computationally expensive and might even be unfeasible with the current sample size. Therefore, we opted for a two-step estimation approach.

# DDM Model Fit Assessments

*Will be presented after Stage 1 acceptance*

Ahn, W.-Y., Haines, N., & Zhang, L. (2017). Revealing neurocomputational mechanisms of reinforcement learning and decision-making with the hBayesDM package. *Computational Psychiatry*, *1*, 24–57. <https://doi.org/10.1162/CPSY_a_00002>

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 457–472. <https://doi.org/10.1214/ss/1177011136>

Schubert, A.-L., Nunez, M. D., Hagemann, D., & Vandekerckhove, J. (2019). Individual differences in cortical processing speed predict cognitive abilities: A model-based cognitive neuroscience account. *Computational Brain & Behavior*, *2*(2), 64–84.

Vandekerckhove, J. (2014). A cognitive latent variable model for the simultaneous analysis of behavioral and personality data. *Journal of Mathematical Psychology*, *60*, 58–71. <https://doi.org/10.1016/j.jmp.2014.06.004>