A Tutorial on Conducting Computer Simulation for Research and Teaching

Michael Sturman

Rutgers University

The Goal of the Session

- **Simulation is a methodology with a long history**
  - In various fields since the 1940s
  - Several decades in management research
- **Not covered in most graduate programs**
  - Rarely mentioned in research methods texts
  - Only occasionally covered in research methods courses
  - Most instructional guidance is for computer programmers
- **Tutorial in *Organizational Research Methods***
  - Sturman, M. C. (2025). Real Research with Fake Data: A Tutorial on Conducting Computer Simulation for Research and Teaching. *Organizational Research Methods*, *28*(1), 76-113.
- **Goal of today's session**
  - Describe the uses of simulation
  - Provide basic instruction on how to simulate data
  - Hopefully, get you thinking about this a bit more

Target Audience

- **This session is intended for *novices* of computer simulation**
  - For those who know almost (or actually) no knowledge of or experience with computer simulation, but would like to be able to create data for research or teaching
  - This session is not about advanced computer simulation or highly technical issues
  - You don’t need to know how to program
- **Direction in three programs**
  - Original paper demonstrates this with three programs:
    - Excel
    - Mplus
    - R
  - Here, as it is CARMA, I’ll focus on R

Why Do Simulation?

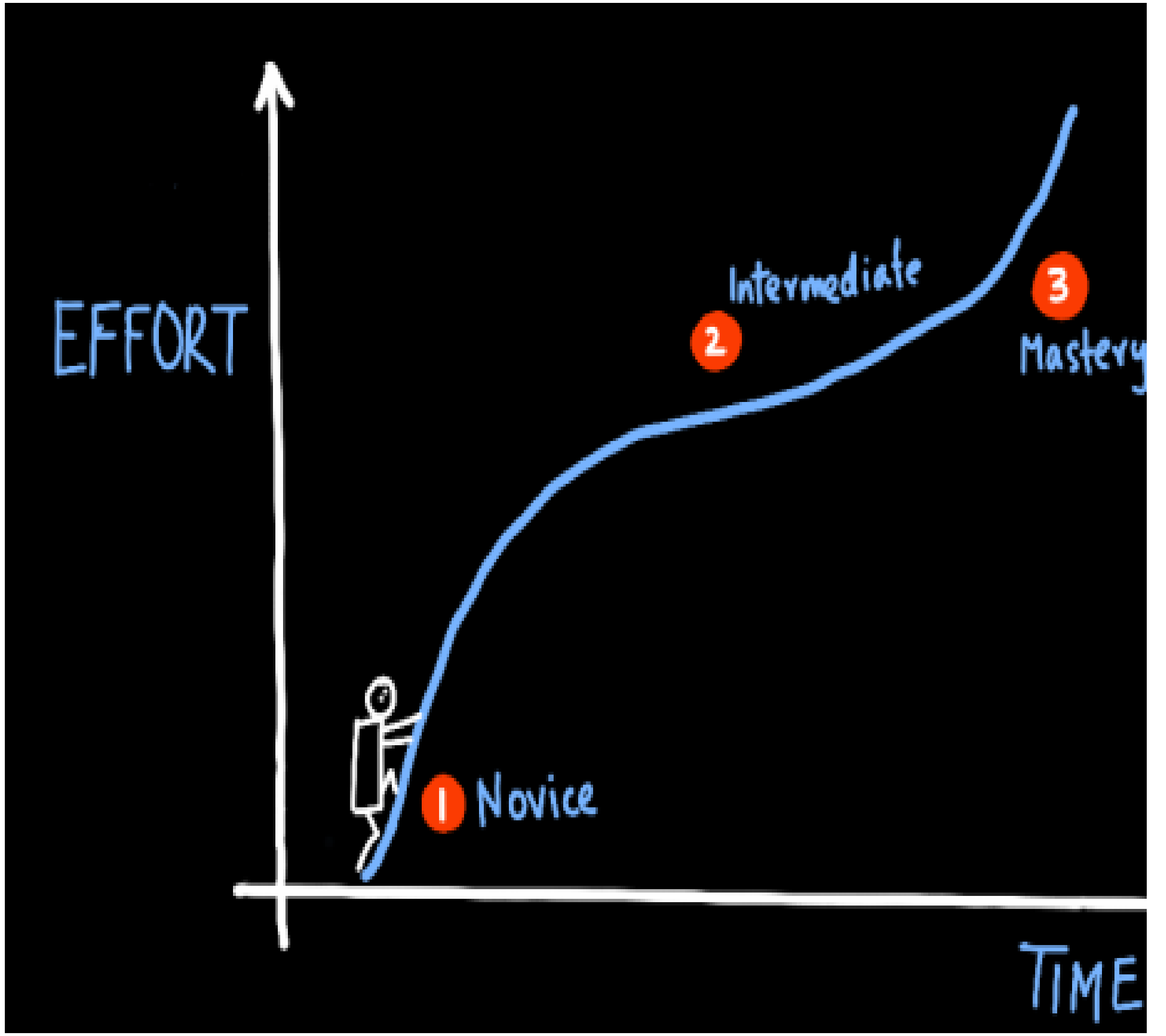
- **A research methodology with particular advantages**
  - Let’s you demonstrate the implications of given assumptions, relationships, or theories
  - You can know the “true score” of stuff
  - Math is hard; simulation is easier
  - Relatively inexpensive methodology
  - Forces precision when articulating theory
  - Can lead to counter-intuitive results
- **Other uses**
  - Develop analytical teaching cases
  - Demonstrate findings
  - Find quick answers to some questions

Types of Simulations Being Considered

- **Start with given relationships, and then test models**
  - Specify relationships among variables and constructs
  - Create the data
  - Analyze the data and see what happens
  - Examples: Common Method Variance, Misspecification, Implications of Assumptions
- **Start with a model, and provide data**
  - Develop teaching cases
  - Examples: Create datasets for a statistics class, HR Analytics Datasets and Exercises
- **Start with a model, and see what happens**
  - Specify relationships among constructs or variables, with or without error
  - “Put in the rules” and then “turn the crank”
  - Examples: Meta-Analysis and SEM, Meta-Analysis and Dynamic Simulation, Theory Testing

Simulation Learning Curve

- **Start very simple**
  - To understand the principles,
  - Not extremely helpful (at first)
  - Actually, can be helpful later
- **Consider multivariate situation**
  - Good for methods papers
- **Consider model**
  - Good for theory testing or teaching
- **Add complexity**
  - Good for methods, theory, or teaching
- **Programming**
  - Necessary for sophisticated methods, theory, and teaching

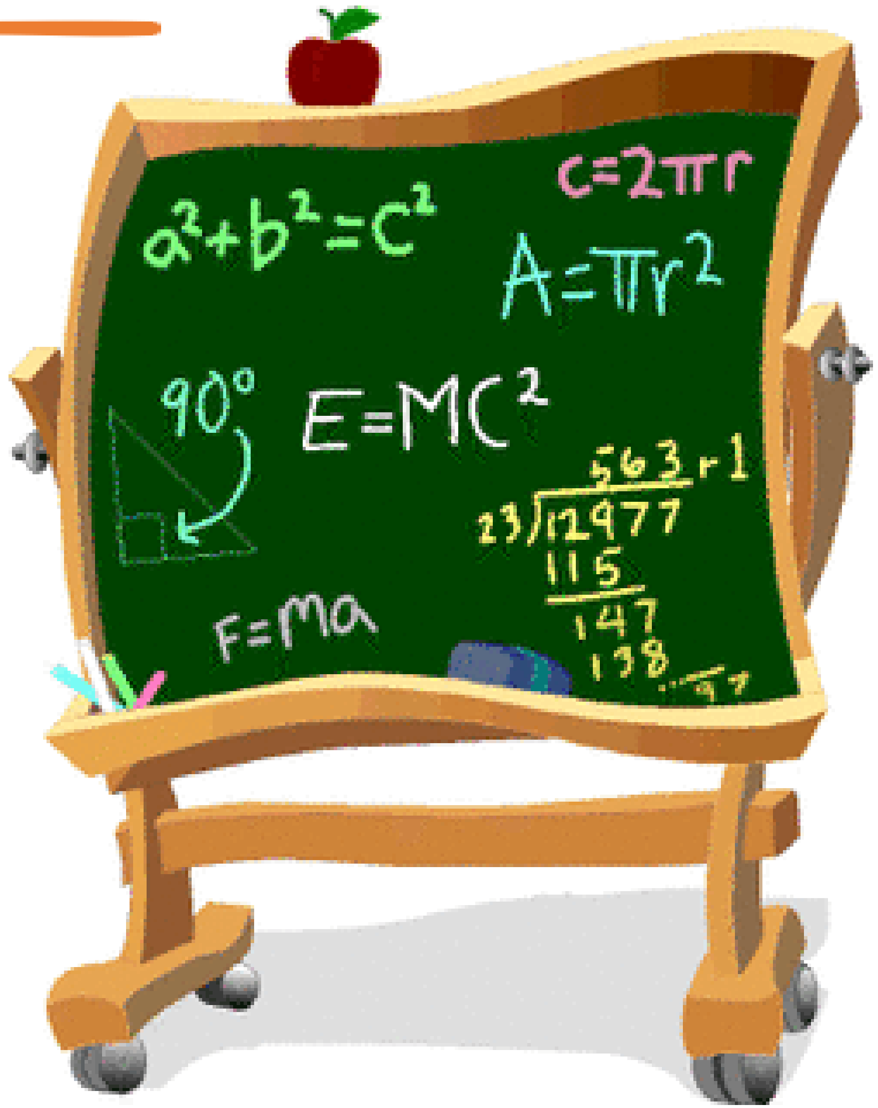


Progression of Examples

- **Creating two correlated variables**
- **Generating multivariate data**
  - Based on a correlation matrix
  - Based on a model
- **Giving the data more "character"**
  - Dichotomization and categorical variables
  - Adding skew and kurtosis
  - Creating observed items for a latent construct
  - Creating non-linear and moderated relationships

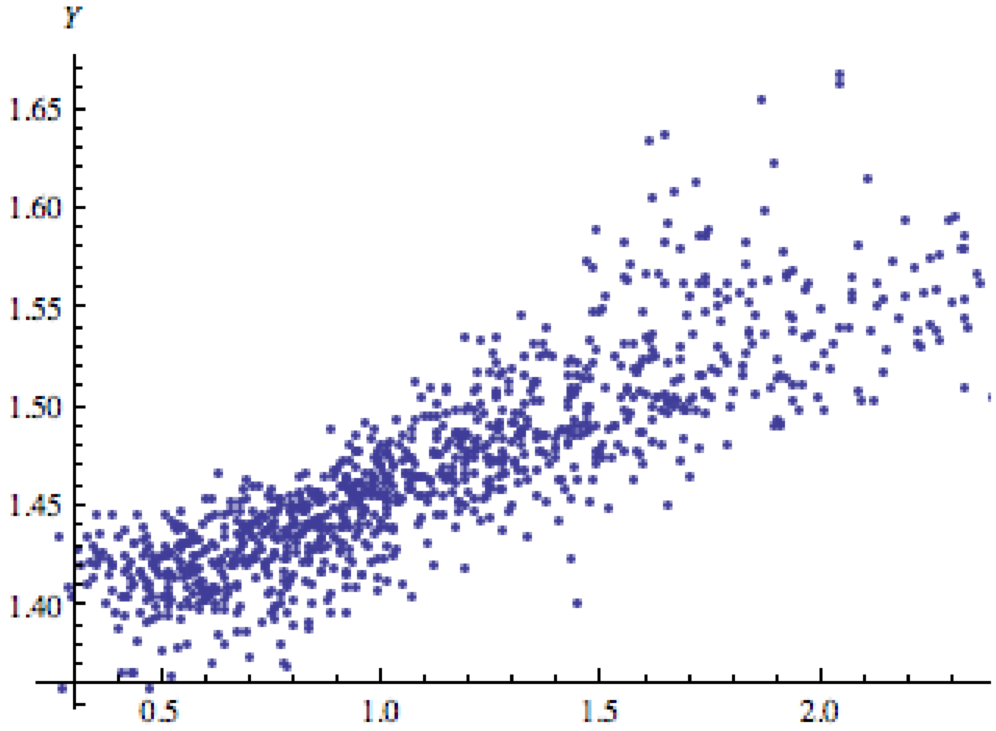
Some Quick (Mathematic) Background

- **All the math comes from two general places**
  - Linear combination and multiple correlations
    - Nunnally, J. C., and Bernstein, I. H. (1994). *Psychometric Theory (3rd edition)*. McGraw-Hill: New York, NY. Chapter 5.
  - Multiple regression
    - $B = (X^T X)^{-1}(X^T Y)$
    - $R^2 = (X^T Y)^T B$
    - If R^2 is percent variance explained, then $(1- R^2)$ is the percent variance unexplained
- **The mathematics is not absolutely necessary**
  - But it does help understand what is going on and why
  - More important if trying to do this manually, such as in Excel or if making your own procedures in R



Example 1: Generating Two Correlated Variables

- **Work with standardized data**
  - You can always "unstandardized it" when you are done
    - $Y = Z(Y) \ * SD_Y + \bar{Y}$
  - This is much easier to think about
  - Think about correlations
- **Simple math**
  - Creating two standardized variables (Y and X)
  - Each can have its own mean and standard deviation
  - Correlated r (or alternatively, the standardized beta is r)
  - Y = (X * r) + (C * Error)
- **What you may need to do**
  - Create random X and Error
  - Calculate C = $\sqrt{(1 - r^2)}$



Example 1 Scenario

- **Mean(X) = 3.5**
- **SD(X) = 1.2**
- **Mean(Y) = 12.4**
- **SD(Y) = 4.4**
- **r(X,Y) = 0.40**

Example 1 in *R*

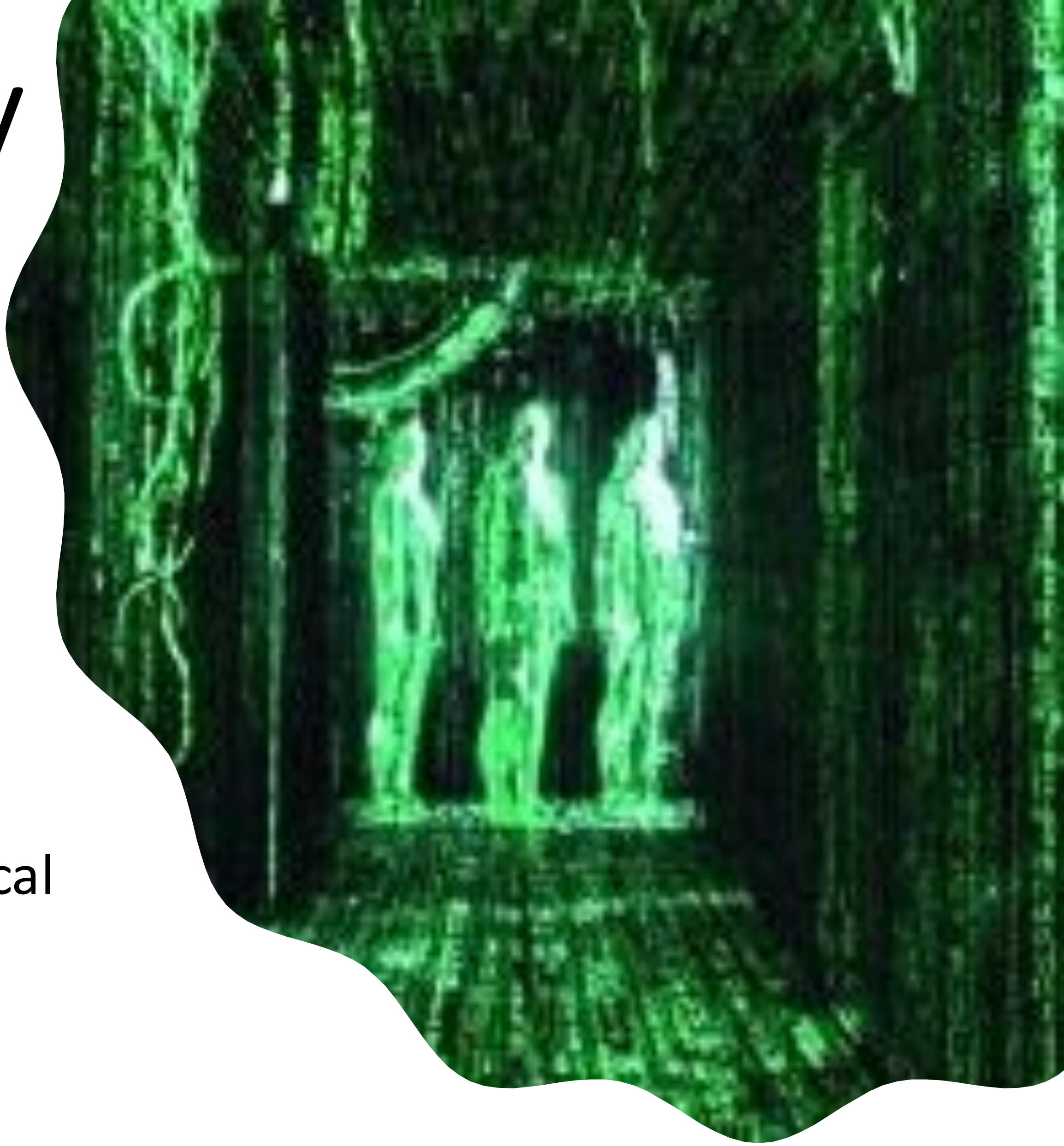
- **R is complex, but has good flexibility**
- **As with many things in R, there are multiple ways to do this**
- **Set up your parameters:**
  - N <- 5
  - MeanX <- 3.5
  - SDX <- 1.2
  - MeanY <- 12.4
  - SDY <- 4.4
  - rXY <- 0.40

Three Ways to do this in *R*

- **Method 1**
  - Uses rnorm(N) to generate N random normal variables
    - ZX <- rnorm(N)
    - Error <- rnorm(N)
    - ZY <- (ZX * rXY) + sqrt(1-rXY^2)*Error
    - X <- (ZX*SDX) + MeanX
    - Y <- (ZY*SDY) + MeanY
- **Method 2**
  - Uses the “faux” package, which is designed for simulating correlated data
    - install.packages(“faux”)
    - library(“faux”)
    - DataFile <- rnrom_multi(n=N, mu=c(MeanX, MeanY), sd=c(SDX,SDY), r=rXY, varnames = c(“X”,”Y”)
- **Method 3**
  - Uses the “MASS” library
    - Covar <- matrix(c(SDX^2, rXY*SDX*SDY, rXY*SDX*SDY, SDY^2), nrow=2, ncol=2)
    - Datafile2 <- as.data.frame(mvnorm(n=N, mu = c(MeanZ, MeanY), Sigma = Covar)
    - Colnames(Datafile2) <- c(“X”, “Y”)

Let’s Add Some Complexity

- **Generating multivariate Data**
- **Two approaches**
  - Based the data on a correlation matrix
  - Based the data on a model
- **Correlation matrix approach**
  - Useful to examine implications of a pattern of correlations
  - Useful for evaluating the efficacy of an analytical method
- **Model-based approach**
  - Useful for generating a specific desired model
  - Requires you to specify relationships within the data, even if they are not the focus of your model



Correlation Approach

- **Specify a full correlation matrix**
  - Must provide all correlations
  - Provide means and standard deviations if desired
- **Easy in R**
  - Well…fairly easy, if you already know how to use R in general
  - You can ask ChatGPT to do this
    - "Give me R code to create data from a specified 5x5 correlation matrix"
    - Then just put in the desired correlations, means, and SD
    - ChatGPT does this well
  - Easy-enough to do it on your own

Example 2 Scenario

Variable	Mean	SD	A	B	C	D	E
A	3.5	1.2	1				
B	12.4	4.4	0.40	1			
C	2.6	0.8	0.30	0.10	1		
D	2.7	1.0	0.20	0.15	0.25	1	
E	3.8	1.5	0.40	0.30	0.20	0.20	1

Example 2 in *R*

- **I use the "faux" library for this, as it is *much* easier**
- **I don't like "*MASS*" library**
  - *mvnorm* function requires a covariance matrix rather than a correlation matrix
  - Although the function does provide an option to create exact values rather than a sample
- **I use a method very similar to Method 2 of Example 1**
  - Instead of giving 1 correlation coefficient, you provide a list
- **Use the *rnorm_multi()* function**

```
DataFile <- rnorm_multi(n=1000, mu=c(3.5,12.4,2.6,2.7,3.8),
                        sd=c(1.2,4.4,0.8,1.0,1.5),
                        r = c(0.4,0.3,0.2,0.4,0.1,0.15,0.3,0.25,0.2,0.2),
                        varnames=c("A","B","C","D","E"))
```

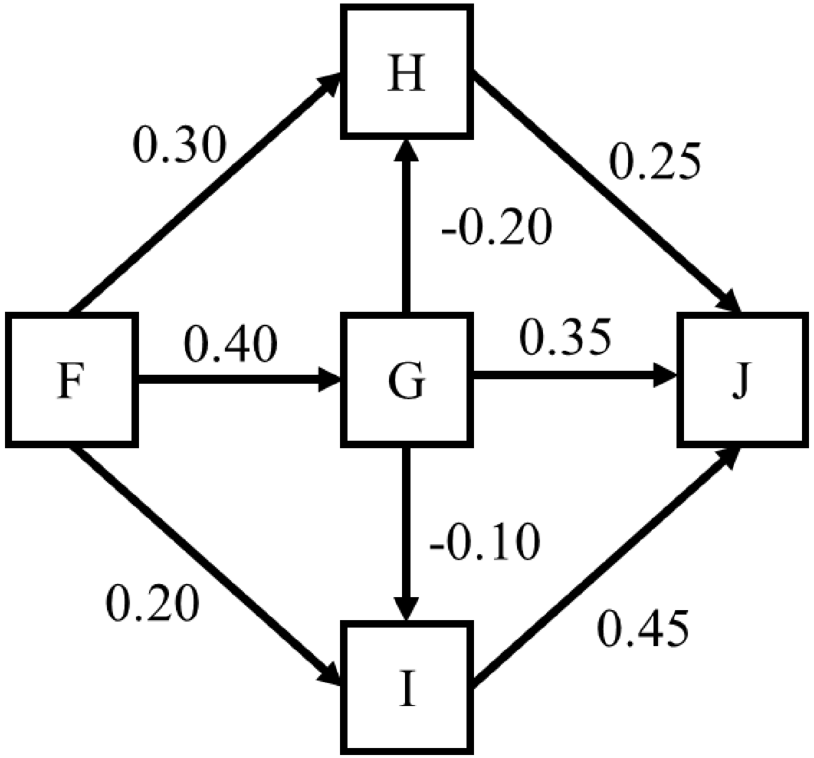
Warning

- **Non-positive definite matrices**
  - Computers will try to do whatever you tell them to do
  - Sometimes, however, you may ask for the impossible
- **Definition**
  - Formal: A matrix C is said to be positive definite if C is symmetric and $\mathbf{v}^T\mathbf{C}\mathbf{v} > 0$
  - Informal (for us): A matrix of correlations that cannot exist in reality
- **This happens often when “making up” correlation matrices**
  - In “matrixcalc” library, use *is.positive.definite* function
- **How it will be shown**
  - R: Error in cormat(r, vars) : correlation matrix not positive definite

Generating Data Based on a Path Model

- **Advantages**
  - Often, this is more intuitive
  - Draw it, then simulate it
  - Makes simulating mediation very easy
- **Disadvantages**
  - You need to specify more than just one set of betas
- **Determining Error**
  - Because you don't have correlations, you cannot use the matrix equation to know the R-square in your model
  - To determine the R-square, remember R-square is percent of variance explained
  - Given we are creating normal data with a mean of 0 and SD of 1 the variance of predicted values *IS* the percent of variance explained
  - Of course *E(var(e))* does not always equal *1-var(x)*, so greater precision may be needed
    - *lavaan* can help with this

Example 3 Scenario



Example using *lavaan*

```
library("lavaan")
Population.model <- '     G =~ 0.40*F
                          H =~ 0.30*F + -0.20*G
                          I =~ 0.20*F + -0.10*G
                          J =~ 0.35*G + 0.25*H + 0.45*I  '
myData <- simulateData(Population.model, sample.nobs = 1000)
```

Giving the Data More Character

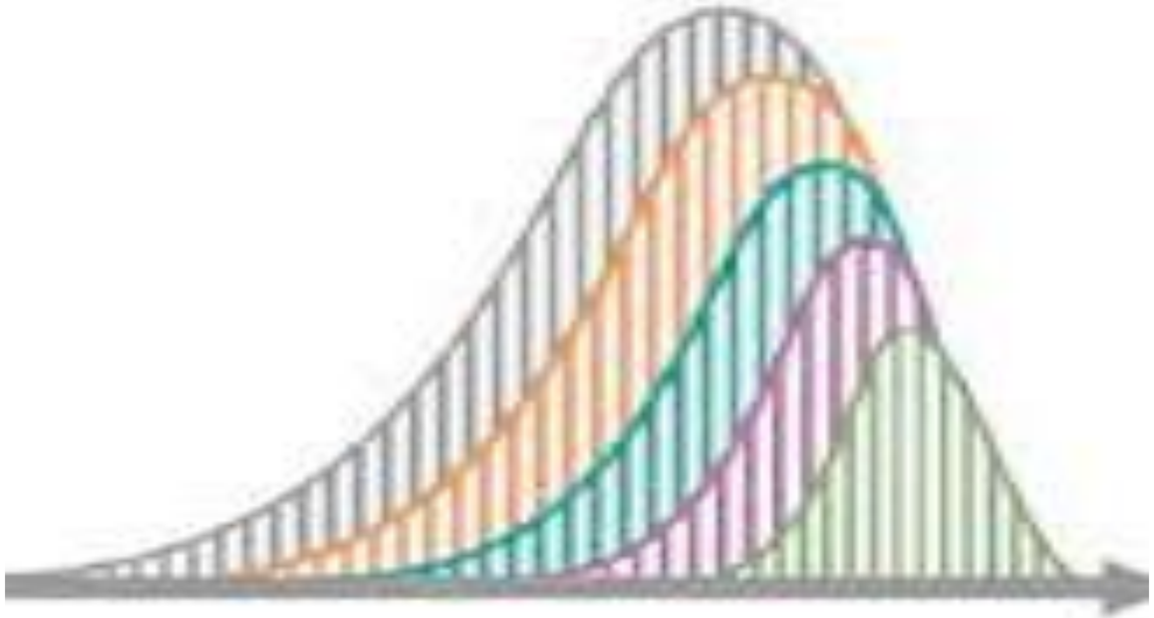
- **So far**
  - Every variable we created is continuous
  - Every variable we created is normally distributed
  - All variables are represented by a single number
  - All relationships are linear
- **This does not look like actual data**
  - And you may want “real” looking data
  - May want data that is
    - Binary or categorical
    - Is non-normal
    - Operates like a multi-item measure of an observed construct
    - Had moderated or non-linear relationships in the data
- **Our recommended approach**
  - Create data, based on correlation matrix or model, based on the methods already shown
  - Then transform the data to give it the desired characteristics

Dichotomized and Categorical Variables

- **Dichotomized and categorical variables**
  - Use IF statements from the original multivariate normal data to create the desired split or categories
- **More categories is more complexity**
  - But this is mostly of an issue about checking your code
  - R is most flexible with respect to IF statements but requires coding knowledge
  - R doesn’t “like” looping so much
  - More efficient methods require better programming

Adding Skew and Kurtosis

- **Ways to generate non-normal data**
  - Method 1: Add error terms that are non-normal
  - Method 2: Create a normal variable and transform it
  - I recommend the use Method 2, and specifically with the G-and-H distribution
- **G-and-H distribution**
  - Transforms a uniform normal into a non-normal shape
  - Had three parameters
    - G: which affects the skew
    - H: which affects the kurtosis
    - A: which indicates the distribution's median
    - B: a parameter that influences the variance
- **Formulas**
  - Where g≠0, $Y = A + B\left(\frac{e^{gZ}-1}{g} e^{(hZ^2)/2}\right)$
  - If g=0, $Y = A + \mathrm{B}(Ze^{(hZ^2)/2})$



Operationalizing the G-and-H Distribution

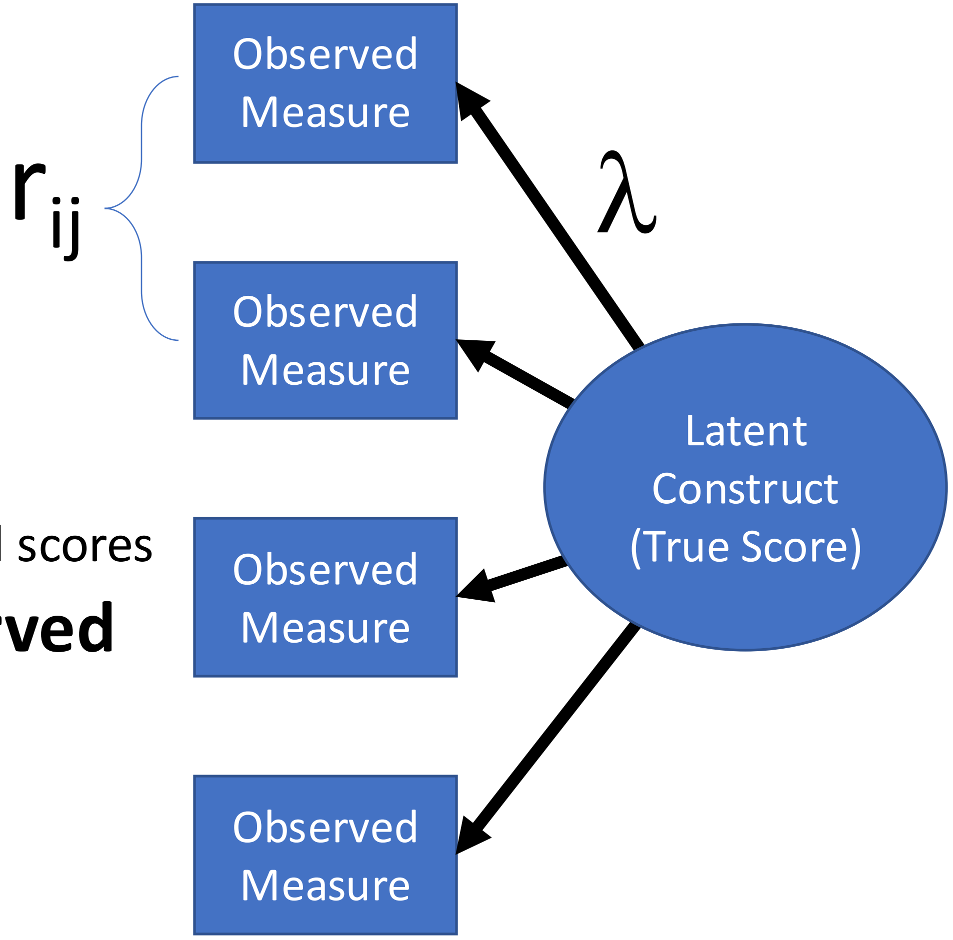
- **Choosing your parameters**
  - Pick numbers until it looks right
  - Have a desired skew and kurtosis and find values that approximate that
- **I provide an Excel tool to help you determine the G, H, A, and B parameters**
  - Available with the slides
  - Quick demonstration
- **Alternatively, you can try other distributional transformations and just see if they seem to work**
  - Yuck

Latent Constructed and Observed Measures

- **Can use *lavaan***
    - But harder to get "true construct scores"
    - Probably an R expert could do this
- **Alternatively, first generate latent scores**
- **Then, make the measures**
    - The simulated score is the latent construct
    - Need to know how many measures (K)
    - Need to know the desired reliability (α)
    - This is why it is helpful to know how to generate two correlated scores
- **While "normally" we calculate alpha give observed measures, in simulation we do the opposite**

$$r_{ij} = \frac{\alpha}{k - \alpha k + \alpha}$$

$$\lambda = \sqrt{r_{ij}}$$



- **Scenario**
  - 4-item measure; Desired alpha of 0.85
  - $r_{ij} = \frac{\alpha}{k - \alpha k + \alpha}$
  - $r_{ij} = \frac{0.85}{4 - 0.85 * 4 + 0.85}$
  - $r_{ij} = 0.586207$
  - $\lambda = 0.765641$
- **R code**

```
L = 0.765641
Population.model2 <- ‘
          # Factors
          F =~ L*x1 + L*x2 + L*x3 + L*x4
          G =~ L*x5 + L*x6 + L*x7 + L*x8
          H =~ L*x9 + L*x10 + L*x11 + L*x12
          I =~ L*x13 + L*x14 + L*x15 + L*x16
          J =~ L*x17 + L*x18 + L*x19 + L*x20
          # Regressors
          G ~ 0.40*F
          H ~ 0.30*F + -0.20*G
          I ~ 0.20*F + -0.10*G
          J ~ 0.35*G + 0.25*H + 0.45*I  ’
myData <- simulateData(Population.model2, sample.nobs=1000)
```

Adjusting for Coarse Measures

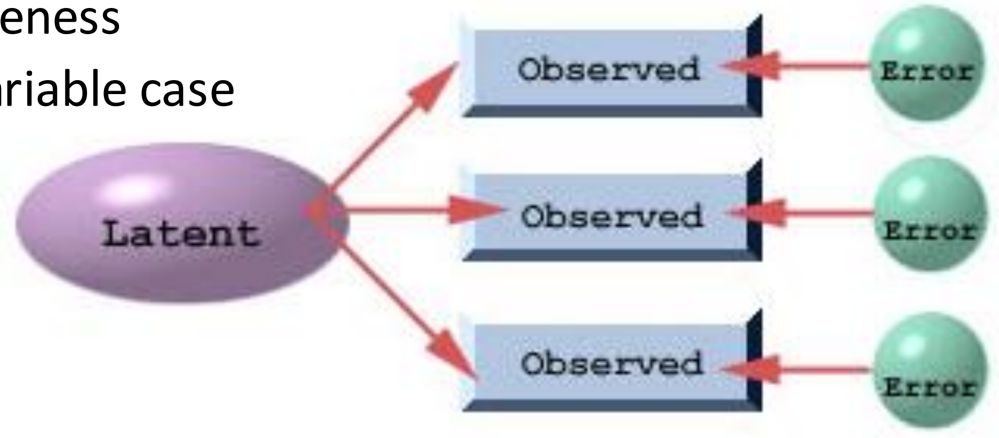
- **When measures have a limited number of possible outcomes**
  - This attenuates observed correlations
  - Described in Aguinis, Pierce, & Culpepper, *ORM,* 2009
  - They provide correction values
  - Divide the desired correlation by the correction value squared
    - You are correcting for both variables, hence the squared term
- **Most relevant for computing alpha**
  - Coarseness can affect all variables
  - But with multiple items, measures are typically not very coarse
    - Five 5-point scales means you can have 21 levels (correction factor is essentially 1)
  - For measuring reliability, effect can be bigger
    - You correct for each item
    - If each scale item has 5 levels, correction factor is r/0.889

- **Scenario**
  - 5-item measure
  - Desired alpha of 0.80
  - 5-poiont scale
  - $r_{ij} = \frac{\alpha}{k-\alpha k+\alpha}$
  - $r_{ij} = \frac{0.80}{5-0.80*5+0.80}$
  - $r_{ij} = 0.44444$
- **Correct for coarseness**
  - Correction factor for 5 scale points is 0.943
  - $\widetilde{r_{ij}} = \frac{0.44444}{(.943)^2}$
  - $\widetilde{r_{ij}} = 0.499797$
  - $\lambda = 0.706963$
- **Generate your data**
  - Which will be continuous
- **Round to 5-point scale**
  - You need to know the variance and mean of observed variable

```
ObsX <- round((X*1.2)+3,0)
ObsX[ObsX<1] <- 1
ObsX[ObsX>5] <- 5
```

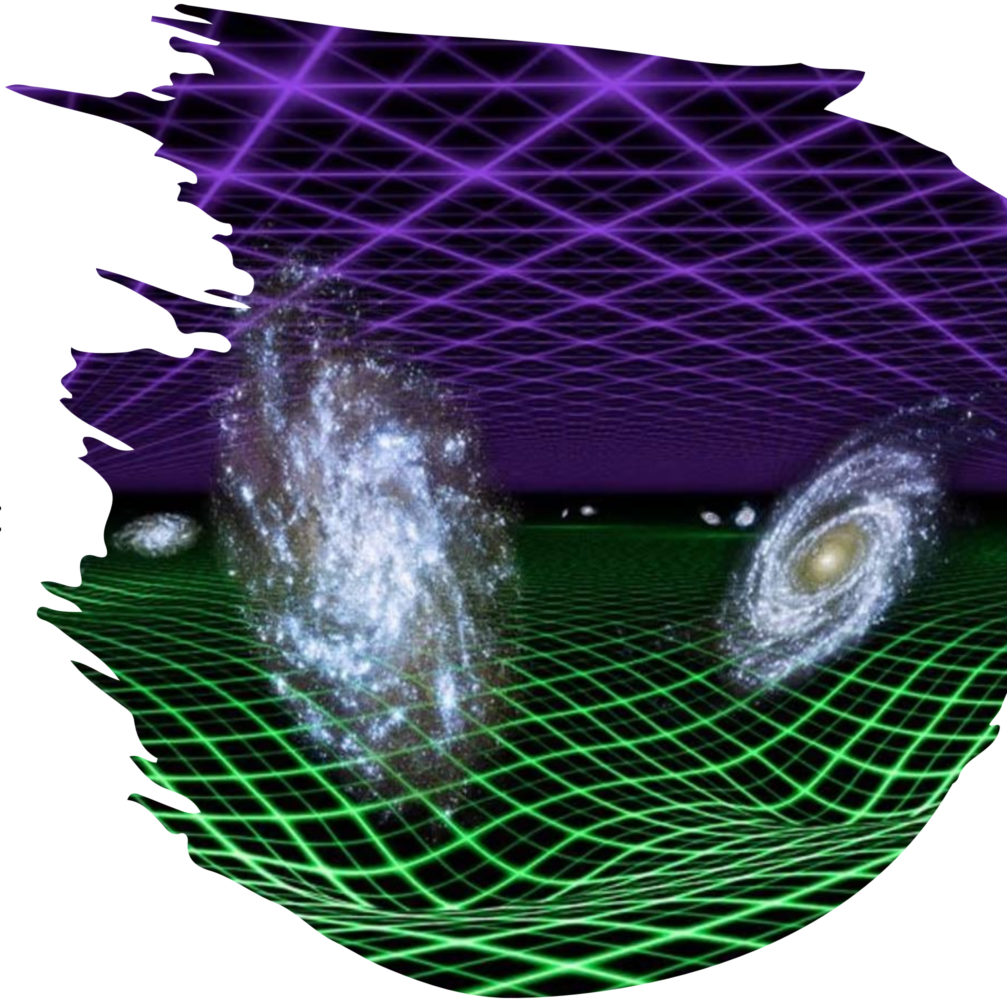
Making Observed Measures

- **If you don't need "true" scores**
  - Use lavaan method
- **If you need both "true" and observed scores**
  - Generate all your "true" scores first
  - Determine r_{ij} based on your desired k, α, and coarseness
  - Create each measure just like in the 2 correlated variable case
    - Your X is the true score
    - Your r is the calculated r_{ij}
    - Repeat for all K measures

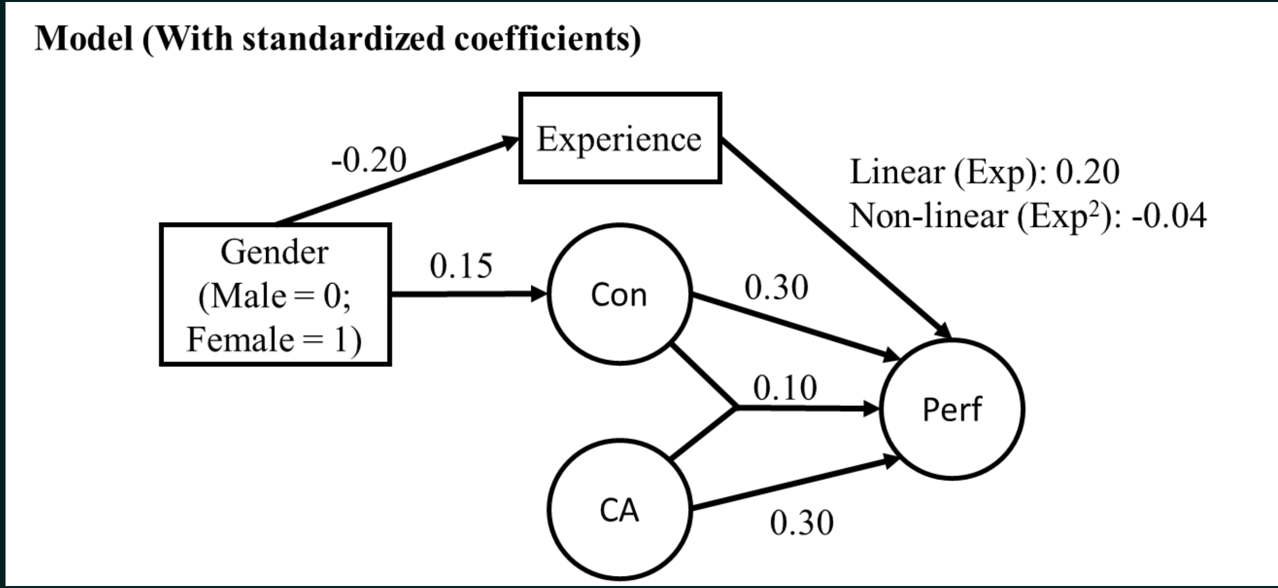


Non-Linear and Moderated Relationships

- **Can be tricky**
  - Nonlinear and product terms have constrained and difficult to calculate relationships with other variables
- **My advice**
  - Create everything with linear relationships first
  - Use Correlational or Model-based approach
  - Then, use the Model-based approach to create the dependent variable that is influenced by the nonlinear terms
- **Example**
  - J ~ 0.35*G + 0.25*H + 0.45*I + 0.10*G*H



Comprehensive Example



Variable Characteristics

Variable	Min	Max	Mean	SD	Skew	Kurtosis	Number of items	Alpha	Scale points	Calculated Values
Gender	0	1	0.40	–	–	–	1	–	–	Cutoff = 0.25335
Experience	0	12	3.0	1.07	0.69	1.20	1	–	–	A = 2.8952; B = 1.0015 g = 0.2000; h = 0.0225
Conscientiousness	1	5	2.70	0.40	0	0	4	0.85	5	r = 0.6592
Cognitive Ability	70	130	100	10	0	0	5	0.90	>15	r = 0.6429
Performance	1	5	3.20	0.80	0	0	3	0.80	5	r = 0.6426

Possible Next Steps (But Not Today)

- **Multilevel data**
  - Option 1:
    - Use *lavaan*
  - Option 2:
    - Treat the lower-level data as group-centered
    - Create the higher-level data first, per normal
    - Match it to the lower level, and then create as per normal
    - Then just add up the levels when done
- **Range restriction, selection effects, and missing data**
  - Use of IF statements and deleting or changing lines as necessary

Other Possible Steps

- **Varying parameters**
  - Requires some programming
  - Easily done in R
  - Some methods are more efficient than others
    - R isn't great at "loops"
- **Simulating longitudinal data**
  - Same as what we've done so far, just different logic
  - Need to have a correlation matrix with time 1 and time 2
  - Generate time 1, then create time 2
  - Time 2 then becomes the next Time 1
  - Repeat

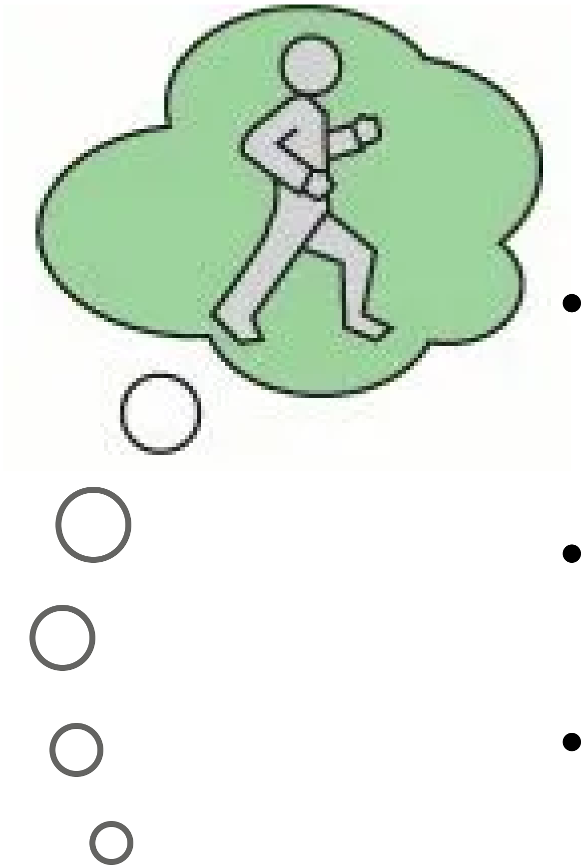
If You Really Want to Get Good

- **Better understand the math**
  - Again, Chapter 5 of Nunnally & Bernstein may be the most helpful
  - A little matrix algebra can be helpful
- **You will eventually need to do programming**
  - Loops are almost always essential for complex simulations
  - Not very hard in R, but takes some programming basics
  - Greater efficiency requires better programming
- **Practice, Practice, Practice**

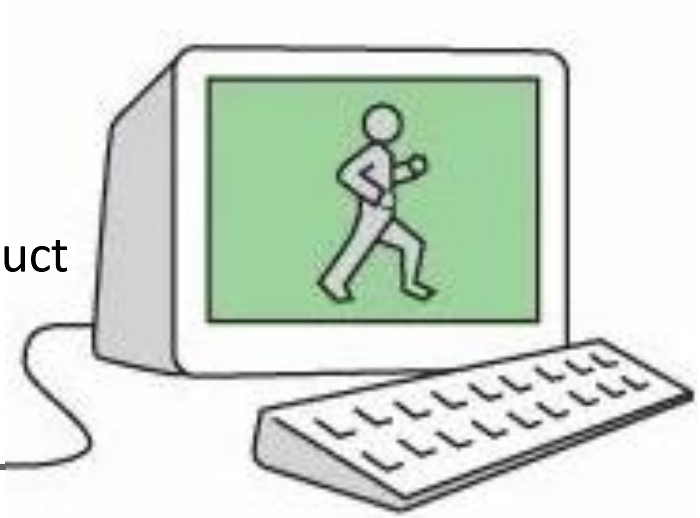
How to Practice

- **Simulation is great for creating teaching cases**
  - Create datasets with known relationships
  - Great for PhD Methods exams
  - I've used them in intro HR classes
  - With a loop (or enough time) you can create unique datasets for each student
    - Same true underlying relationships or model
    - But sampling error will give everyone slightly different answers
- **Research methods questions**
  - Generate data based on a correlation matrix to check the efficacy of different modeling approaches
  - Generate data based on a model to examine the effect of incorrect analyses
  - These can be interesting, but require that you do not have "too many" parameters

Final Thoughts



- **Simulation is a very useful methodology**
  - Lets you ask questions that can't be answered with other methods
  - Lets you create data for teaching and research purposes
- **But instruction in simulation is missing from most research methods texts or PhD program curriculums**
- **Hopefully, after today you can**
  - Create data based on a correlation matrix
  - Create data based on a model
  - Give created data more "character" by changing its distribution
  - Make observed scores of an underlying latent construct




THANK YOU!
P100-LOAD-TABLE.
OPEN INPUT PART-M
MOVE 0 TO WS-SUB.