In health services research, experiments are often conducted at the provider or site level rather than the patient level. However, we might still be interested in the outcome at the patient level. For example, we could be interested in understanding the effect of a training program for physicians on their patients. It would be very difficult to randomize patients to be exposed or not to the training if a group of patients all see the same doctor. So the experiment is set up so that only some doctors get the training and others serve as the control; we still compare the outcome at the patient level.

Typically, when conducting an experiment we assume that individual outcomes are not related to each other (other than the common effect of the exposure). With site-level randomization, we can’t make that assumption – groups of patients are all being treated by the same doctor. In general, even before the intervention, there might be variation across physicians. At the same time, patients within a practice will vary. So, we have two sources of variation: *between* practice and *within* practice variation that explain overall variation.

I touched on this when I discussed issues related to Gamma distributed clustered data. A key concept is the intra-class coefficient or ICC, which is a measure of how *between* variation relates to overall variation. The ICC ranges from 0 (where there is no *between* variation – all site averages are the same) to 1 (where there is no variation within a site – all patients within the site have the same outcomes). Take a look at the earlier post for a bit more detail.

Gamma distributed clustered data

Way back when I was studying algebra and wrestling with one word problem after another (I think now they call them story problems), I complained to my father. He laughed and told me to get used to it. “Life is one big word problem,” is how he put it. Well, maybe one could say any statistical analysis is really just some form of multilevel data analysis, whether we treat it that way or not.

A key feature of the multilevel model is the ability to explicitly untangle the variation that occurs at different levels. Variation of individuals within a sub-group, variation across sub-groups, variation across groups of sub-groups, and so on. The intra-class coefficient (ICC) is one summarizing statistic that attempts to characterize the relative variability across the different levels.

The amount of clustering as measured by the ICC has implications for study design, because it communicates how much information is available at different levels of the hierarchy. We may have thousands of individuals that fall into ten or twenty clusters, and think we have a lot of information. But if most of the variation is at the cluster/group level (and not across individuals within a cluster), we don’t have thousands of observations, but more like ten or twenty. This has important implications for our measures of uncertainty.

Recently, a researcher was trying to use simstudy to generate cost and quality-of-life measurements to simulate clustered data for a cost-effectiveness analysis. (They wanted the cost and quality measurements to correlate within individuals, but I am going to ignore that aspect here.) Cost data are typically right skewed with most values falling on the lower end, but with some extremely high values on the upper end. (These dollar values cannot be negative.)

Because of this characteristic shape, cost data are often modeled using a Gamma distribution. The challenge here was that in simulating the data, the researcher wanted to control the group level variation relative to the individual-level variation. If the data were normally distributed, it would be natural to talk about that control in terms of the ICC. But, with the Gamma distribution, it is not as obvious how to partition the variation.

As most of my posts do, this one provides simulation and plots to illuminate some of these issues.

### Gamma distribtution

The Gamma distribution is a continuous probability distribution that includes all non-negative numbers. The probability density function is typically written as a function of two parameters - the shape \alpha*α* and the rate \beta*β*:

f(x) = \frac{\beta ^ \alpha}{\Gamma(\alpha)} x^{\alpha - 1} e^{-\beta x},*f*(*x*)=Γ(*α*)*βα*​*xα*−1*e*−*βx*,

with \text{E}(x) = \alpha / \betaE(*x*)=*α*/*β*, and \text{Var}(x)=\alpha / \beta^2Var(*x*)=*α*/*β*2. \Gamma(.)Γ(.) is the continuous Gamma function, which lends its name to the distribution. (When \alpha*α* is a positive integer, \Gamma(\alpha)=(\alpha - 1 )!Γ(*α*)=(*α*−1)!) In simstudy, I decided to parameterize the pdf using \mu*μ* to represent the mean and a dispersion parameter \nu*ν*, where \text{Var}(x) = \nu\mu^2Var(*x*)=*νμ*2. In this parameterization, shape \alpha = \frac{1}{\nu}*α*=*ν*1​ and rate \beta = \frac{1}{\nu\mu}*β*=*νμ*1​. (There is a simstudy function gammaGetShapeRate that maps \mu*μ* and \nu*ν* to \alpha*α* and \beta*β*.) With this parameterization, it is clear that the variance of a Gamma distributed random variable is a function of the (square) of the mean.

Simulating data gives a sense of the shape of the distribution and also makes clear that the variance depends on the mean (which is not the case for the normal distribution):

mu <- 20

nu <- 1.2

# theoretical mean and variance

c(mean = mu, variance = mu^2 \* nu)

## mean variance

## 20 480

**library**(simstudy)

(ab <- gammaGetShapeRate(mu, nu))

## $shape

## [1] 0.8333333

##

## $rate

## [1] 0.04166667

# simulate data using R function

set.seed(1)

g.rfunc <- rgamma(100000, ab$shape, ab$rate)

round(c(mean(g.rfunc), var(g.rfunc)), 2)

## [1] 19.97 479.52

# simulate data using simstudy function - no difference

set.seed(1)

defg <- defData(varname = "g.sim", formula = mu, variance = nu,

dist = "gamma")

dt.g1 <- genData(100000, defg)

dt.g1[, .(round(mean(g.sim),2), round(var(g.sim),2))]

## V1 V2

## 1: 19.97 479.52

# doubling dispersion factor

defg <- updateDef(defg, changevar = "g.sim", newvariance = nu \* 2)

dt.g0 <- genData(100000, defg)

dt.g0[, .(round(mean(g.sim),2), round(var(g.sim),2))]

## V1 V2

## 1: 20.09 983.01

# halving dispersion factor

defg <- updateDef(defg, changevar = "g.sim", newvariance = nu \* 0.5)

dt.g2 <- genData(100000, defg)

dt.g2[, .(round(mean(g.sim),2), round(var(g.sim),2))]

## V1 V2

## 1: 19.98 240.16

Generating data sets with the same mean but decreasing levels of dispersion makes it appear as if the distribution is “moving” to the right: the peak shifts to the right and variance decreases …

**library**(ggplot2)

dt.g0[, nugrp := 0]

dt.g1[, nugrp := 1]

dt.g2[, nugrp := 2]

dt.g <- rbind(dt.g0, dt.g1, dt.g2)

ggplot(data = dt.g, aes(x=g.sim), group = nugrp) +

geom\_density(aes(fill=factor(nugrp)), alpha = .5) +

scale\_fill\_manual(values = c("#226ab2","#b22222","#22b26a"),

labels = c(nu\*2, nu, nu\*0.5),

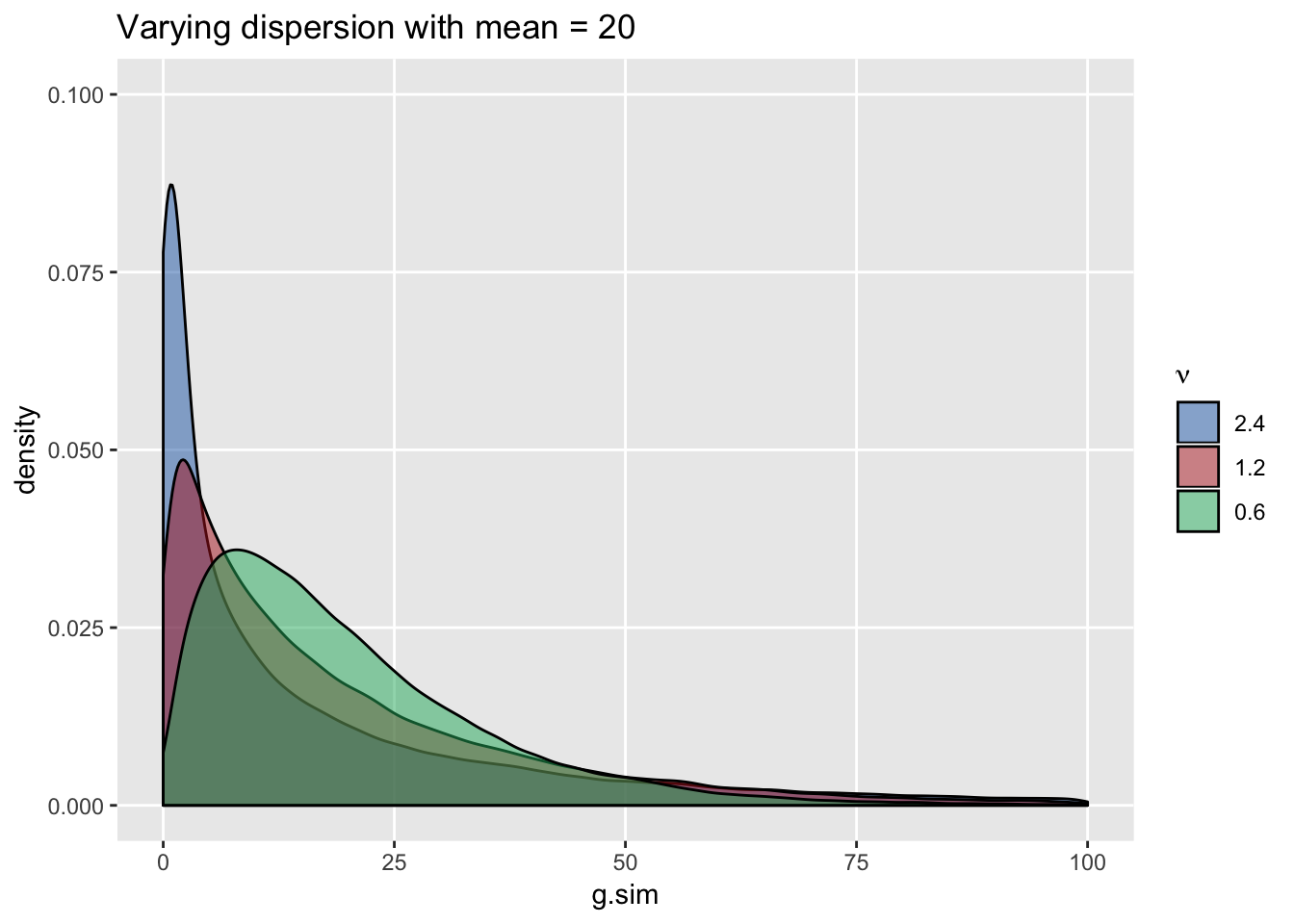
name = bquote(nu)) +

scale\_y\_continuous(limits = c(0, 0.10)) +

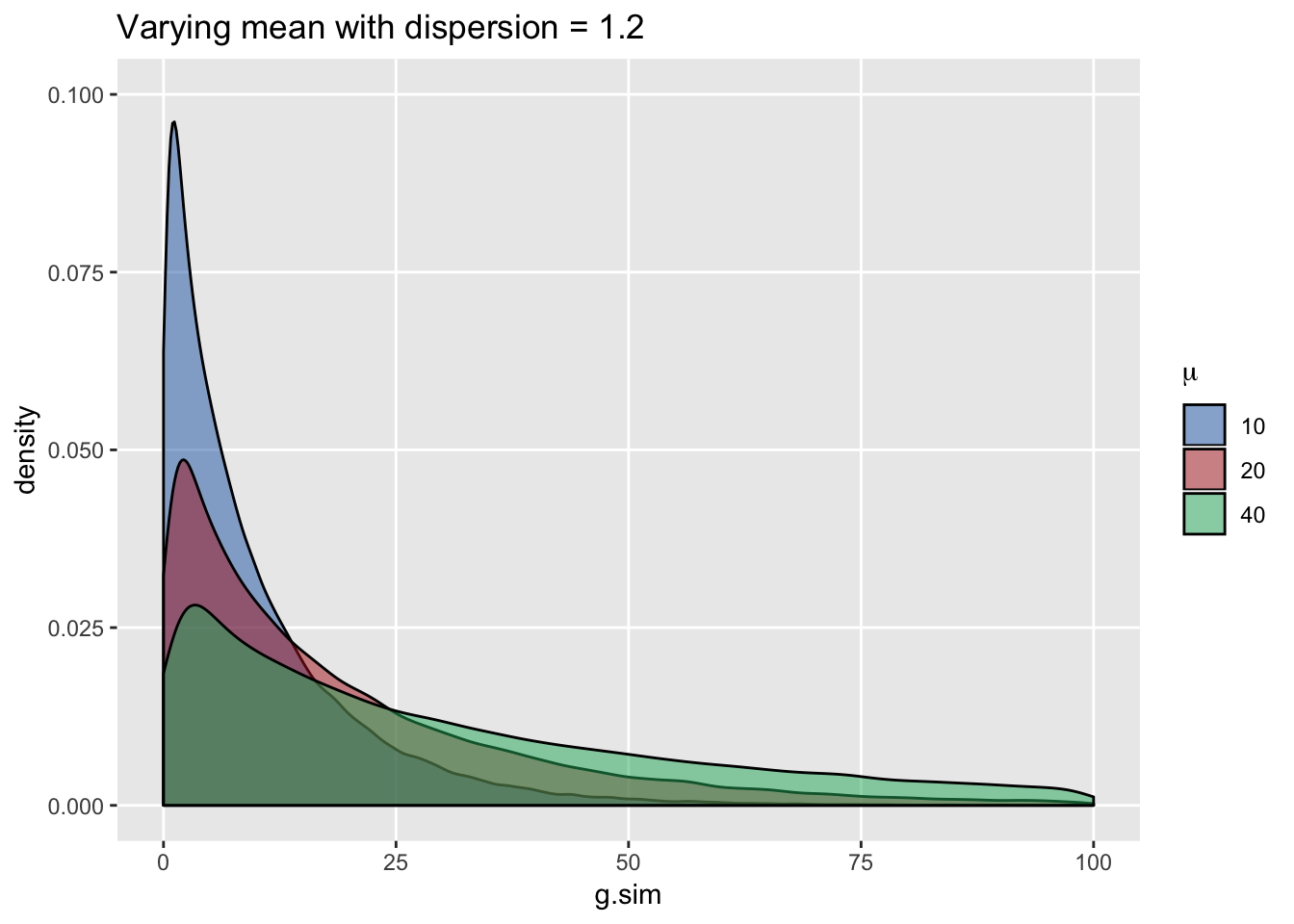
scale\_x\_continuous(limits = c(0, 100)) +

theme(panel.grid.minor = element\_blank()) +

ggtitle(paste0("Varying dispersion with mean = ", mu))



Conversely, generating data with constant dispersion but increasing the mean does not shift the location but makes the distribution appear less “peaked”. In this case, variance increases with higher means (we can see that longer tails are associated with higher means) …



### ICC for clustered data where within-group observations have a Gaussian (normal) distribution

In a 2-level world, with multiple groups each containing individuals, a normally distributed continuous outcome can be described by this simple model:Y\_{ij} = \mu + a\_j + e\_{ij},*Yij*​=*μ*+*aj*​+*eij*​,where Y\_{ij}*Yij*​ is the outcome for individual i*i* who is a member of group j*j*. \mu*μ* is the average across all groups and individuals. a\_j*aj*​ is the group level effect and is typically assumed to be normally distributed as N(0, \sigma^2\_a)*N*(0,*σa*2​), and e\_{ij}*eij*​ is the individual level effect that is N(0, \sigma^2\_e)*N*(0,*σe*2​). The variance of Y\_{ij}*Yij*​ is \text{Var}(a\_j + e\_{ij}) = \text{Var}(a\_j) + \text{Var}(e\_{ij}) = \sigma^2\_a + \sigma^2\_eVar(*aj*​+*eij*​)=Var(*aj*​)+Var(*eij*​)=*σa*2​+*σe*2​. The ICC is the proportion of total variation of Y*Y* explained by the group variation:ICC = \frac{\sigma^2\_a}{\sigma^2\_a+\sigma^2\_e}*ICC*=*σa*2​+*σe*2​*σa*2​​If individual level variation is relatively low or variation across groups is relatively high, then the ICC will be higher. Conversely, higher individual variation or lower variation between groups implies a smaller ICC.

Here is a simulation of data for 50 groups, where each group has 250 individuals. The ICC is 0.10:

# define the group level data

defgrp <- defData(varname = "a", formula = 0,

variance = 2.8, dist = "normal", id = "cid")

defgrp <- defData(defgrp, varname = "n", formula = 250,

dist = "nonrandom")

# define the individual level data

defind <- defDataAdd(varname = "ynorm", formula = "30 + a",

variance = 25.2, dist = "normal")

# generate the group and individual level data

set.seed(3017)

dt <- genData(50, defgrp)

dc <- genCluster(dt, "cid", "n", "id")

dc <- addColumns(defind, dc)

dc

## cid a n id ynorm

## 1: 1 -2.133488 250 1 30.78689

## 2: 1 -2.133488 250 2 25.48245

## 3: 1 -2.133488 250 3 22.48975

## 4: 1 -2.133488 250 4 30.61370

## 5: 1 -2.133488 250 5 22.51571

## ---

## 12496: 50 -1.294690 250 12496 25.26879

## 12497: 50 -1.294690 250 12497 27.12190

## 12498: 50 -1.294690 250 12498 34.82744

## 12499: 50 -1.294690 250 12499 27.93607

## 12500: 50 -1.294690 250 12500 32.33438

# mean Y by group

davg <- dc[, .(avgy = mean(ynorm)), keyby = cid]

# variance of group means

(between.var <- davg[, var(avgy)])

## [1] 2.70381

# overall (marginal) mean and var of Y

gavg <- dc[, mean(ynorm)]

gvar <- dc[, var(ynorm)]

# individual variance within each group

dvar <- dc[, .(vary = var(ynorm)), keyby = cid]

(within.var <- dvar[, mean(vary)])

## [1] 25.08481

# estimate of ICC

(ICCest <- between.var/(between.var + within.var))

## [1] 0.09729918

ggplot(data=dc, aes(y = ynorm, x = factor(cid))) +

geom\_jitter(size = .5, color = "grey50", width = 0.2) +

geom\_point(data = davg, aes(y = avgy, x = factor(cid)),

shape = 21, fill = "firebrick3", size = 3) +

theme(panel.grid.major.y = element\_blank(),

panel.grid.minor.y = element\_blank(),

axis.ticks.x = element\_blank(),

axis.text.x = element\_blank(),

axis.text.y = element\_text(size = 12),

axis.title = element\_text(size = 14)

) +

xlab("Group") +

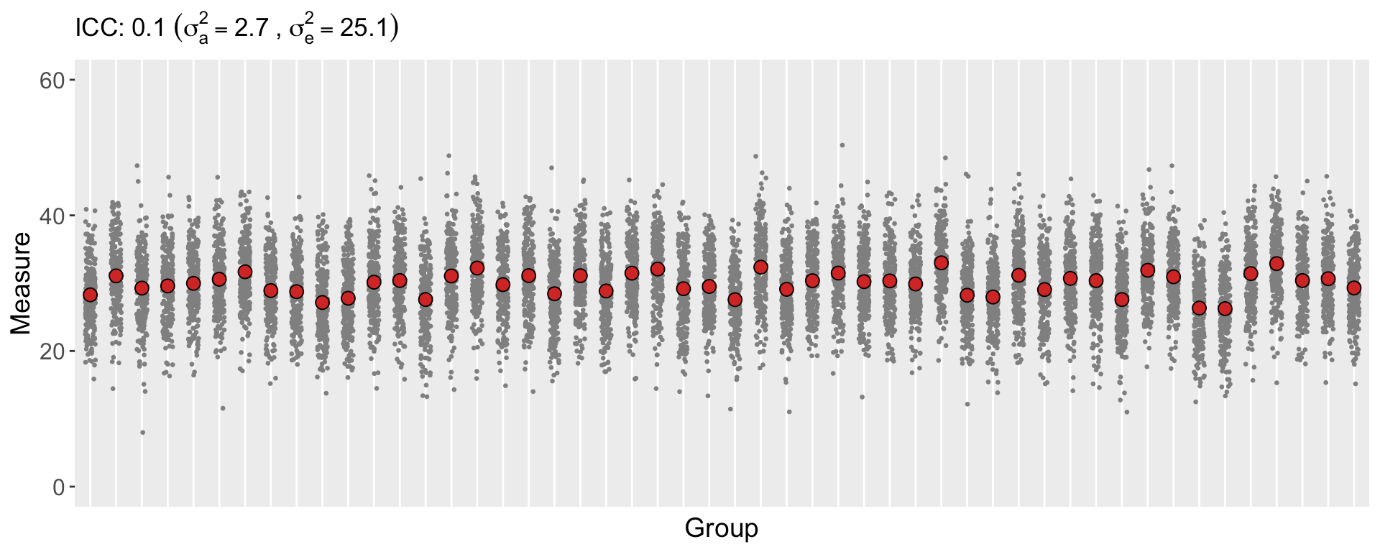
scale\_y\_continuous(limits = c(0, 60), name = "Measure") +

ggtitle(bquote("ICC:" ~ .(round(ICCest, 2)) ~

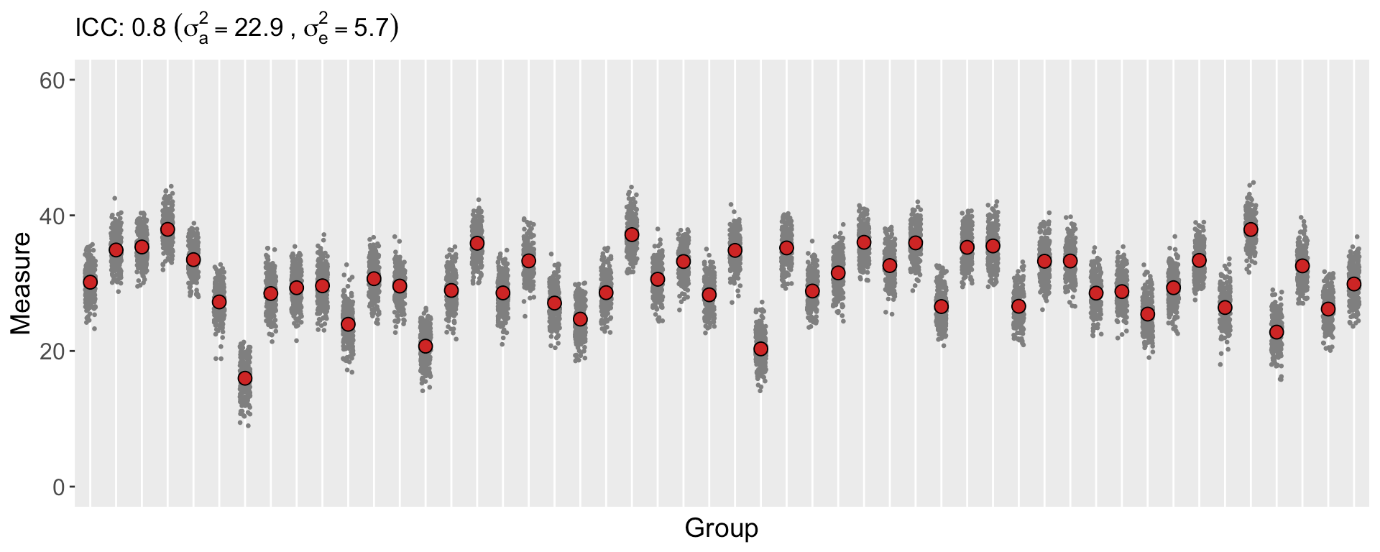
(sigma[a]^2 == .(round(between.var, 1)) ~ "," ~

sigma[e]^2 == .(round(within.var, 1)))

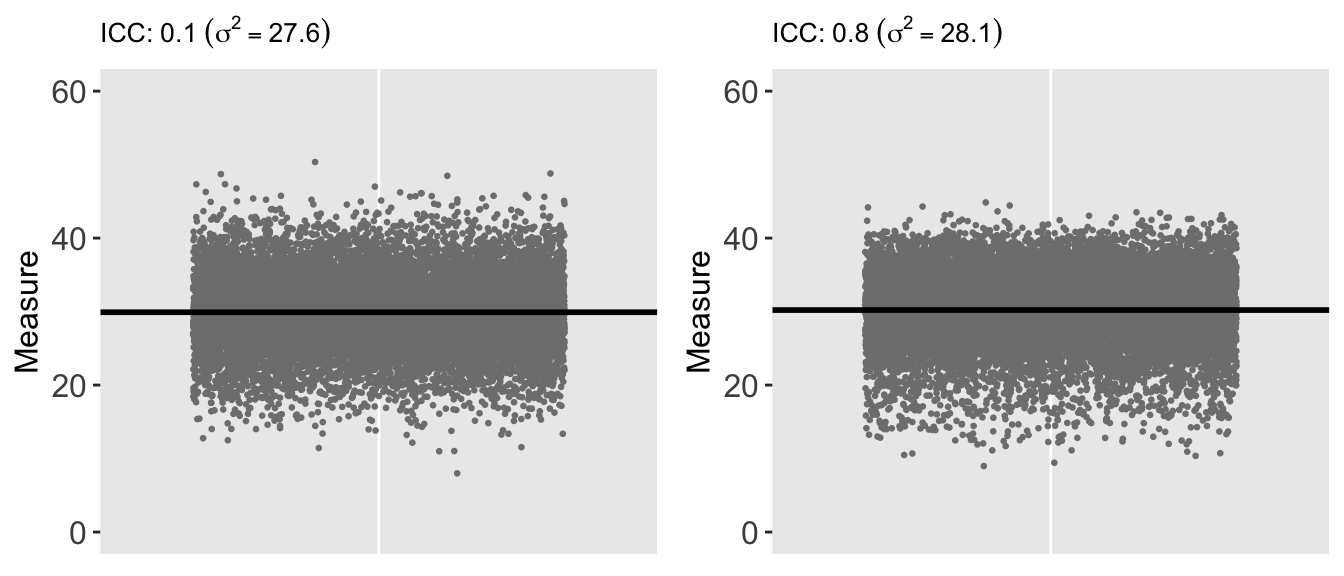
))



Here is a plot of data generated using the same overall variance of 28, but based on a much higher ICC of 0.80. Almost all of the variation in the data is driven by the clusters rather than the individuals. This has implications for a study, because (in contrast to the first data set generated above) the individual-level data is not providing as much information or insight into the variation of Y*Y*. The most useful information (from this extreme example) can be derived from the difference between the groups (so we really have more like 50 data points rather than 125K).



Of course, if we look at the individual-level data for each of the two data sets while ignoring the group membership, the two data sets are indistinguishable. That is, the marginal (or population level) distributions are both normally distributed with mean 30 and variance 28:



### ICC for clustered data with Gamma distribution

Now, back to the original question … how do we think about the ICC with clustered data that is Gamma distributed? The model (and data generating process) for these type of data can be described as:

Y\_{ij} \sim \text{gamma}(\mu\_{j}, \nu),*Yij*​∼gamma(*μj*​,*ν*),where \text{E}(Y\_{j}) = \mu\_jE(*Yj*​)=*μj*​ and \text{Var}(Y\_j) = \nu\mu\_j^2Var(*Yj*​)=*νμj*2​. In addition, the mean of each group is often modeled as:

\text{log}(\mu\_j) = \beta + a\_j,log(*μj*​)=*β*+*aj*​,where \beta*β* is log of the mean for the group whose group effect is 0, and a\_j \sim N(0, \sigma^2\_a)*aj*​∼*N*(0,*σa*2​). So, the group means are normally distributed on the log scale (or are lognormal) with variance \sigma^2\_a*σa*2​. (Although the individual observations within each cluster are Gamma-distributed, the means of the groups are not themselves Gamma-distributed.)

But what is the within group (individual) variation, which is Gamma-distributed? It is not so clear, as the variance within each group depends on both the group mean \mu\_j*μj*​ and the dispersion factor \nu*ν*. A paper by Nakagawa et al shows that \sigma^2\_e*σe*2​ on the log scale is also lognormal and can be estimated using the trigamma function (the 2nd derivative of the gamma function) of the dispersion factor. So, the ICC of clustered Gamma observations can be defined on the the log scale:

\text{ICC}\_\text{gamma-log} = \frac{\sigma^2\_a}{\sigma^2\_a + \psi\_1 \left( \frac{1}{\nu}\right)}ICCgamma-log​=*σa*2​+*ψ*1​(*ν*1​)*σa*2​​\psi\_1*ψ*1​ is the trigamma function. I’m quoting from the paper here: “the variance of a gamma-distributed variable on the log scale is equal to \psi\_1 (\frac{1}{\nu})*ψ*1​(*ν*1​), where \frac{1}{\nu}*ν*1​ is the shape parameter of the gamma distribution and hence \sigma^2\_e*σe*2​ is \psi\_1 (\frac{1}{\nu})*ψ*1​(*ν*1​).” (The formula I have written here is slightly different, as I define the dispersion factor as the reciprocal of the the dispersion factor used in the paper.)

sigma2a <- 0.8

nuval <- 2.5

(sigma2e <- trigamma(1/nuval))

## [1] 7.275357

# Theoretical ICC on log scale

(ICC <- sigma2a/(sigma2a + sigma2e))

## [1] 0.09906683

# generate clustered gamma data

def <- defData(varname = "a", formula = 0, variance = sigma2a,

dist = "normal")

def <- defData(def, varname = "n", formula = 250, dist = "nonrandom")

defc <- defDataAdd(varname = "g", formula = "2 + a",

variance = nuval, dist = "gamma", link = "log")

dt <- genData(1000, def)

dc <- genCluster(dt, "id", "n", "id1")

dc <- addColumns(defc, dc)

dc

## id a n id1 g

## 1: 1 0.6629489 250 1 4.115116e+00

## 2: 1 0.6629489 250 2 6.464886e+01

## 3: 1 0.6629489 250 3 3.365173e+00

## 4: 1 0.6629489 250 4 3.624267e+01

## 5: 1 0.6629489 250 5 6.021529e-08

## ---

## 249996: 1000 0.3535922 250 249996 1.835999e+00

## 249997: 1000 0.3535922 250 249997 2.923195e+01

## 249998: 1000 0.3535922 250 249998 1.708895e+00

## 249999: 1000 0.3535922 250 249999 1.298296e+00

## 250000: 1000 0.3535922 250 250000 1.212823e+01

Here is an estimation of the ICC on the log scale using the raw data …

dc[, lg := log(g)]

davg <- dc[, .(avgg = mean(lg)), keyby = id]

(between <- davg[, var(avgg)])

## [1] 0.8137816

dvar <- dc[, .(varg = var(lg)), keyby = id]

(within <- dvar[, mean(varg)])

## [1] 7.20502

(ICCest <- between/(between + within))

## [1] 0.1014842

Here is an estimation of the ICC (on the log scale) based on the estimated variance of the random effects using a generalized mixed effects model. The between-group variance is a ratio of the intercept variance and the residual variance. An estimate of \nu*ν* is just the residual variance …

**library**(lme4)

glmerfit <- glmer(g ~ 1 + (1|id),

family = Gamma(link="log"), data= dc)

summary(glmerfit)

## Generalized linear mixed model fit by maximum likelihood (Laplace

## Approximation) [glmerMod]

## Family: Gamma ( log )

## Formula: g ~ 1 + (1 | id)

## Data: dc

##

## AIC BIC logLik deviance df.resid

## 1328004.4 1328035.7 -663999.2 1327998.4 249997

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -0.6394 -0.6009 -0.4061 0.1755 14.0254

##

## Random effects:

## Groups Name Variance Std.Dev.

## id (Intercept) 1.909 1.382

## Residual 2.446 1.564

## Number of obs: 250000, groups: id, 1000

##

## Fixed effects:

## Estimate Std. Error t value Pr(>|z|)

## (Intercept) 2.03127 0.02803 72.47 <2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

estnu <- as.data.table(VarCorr(glmerfit))[2,4]

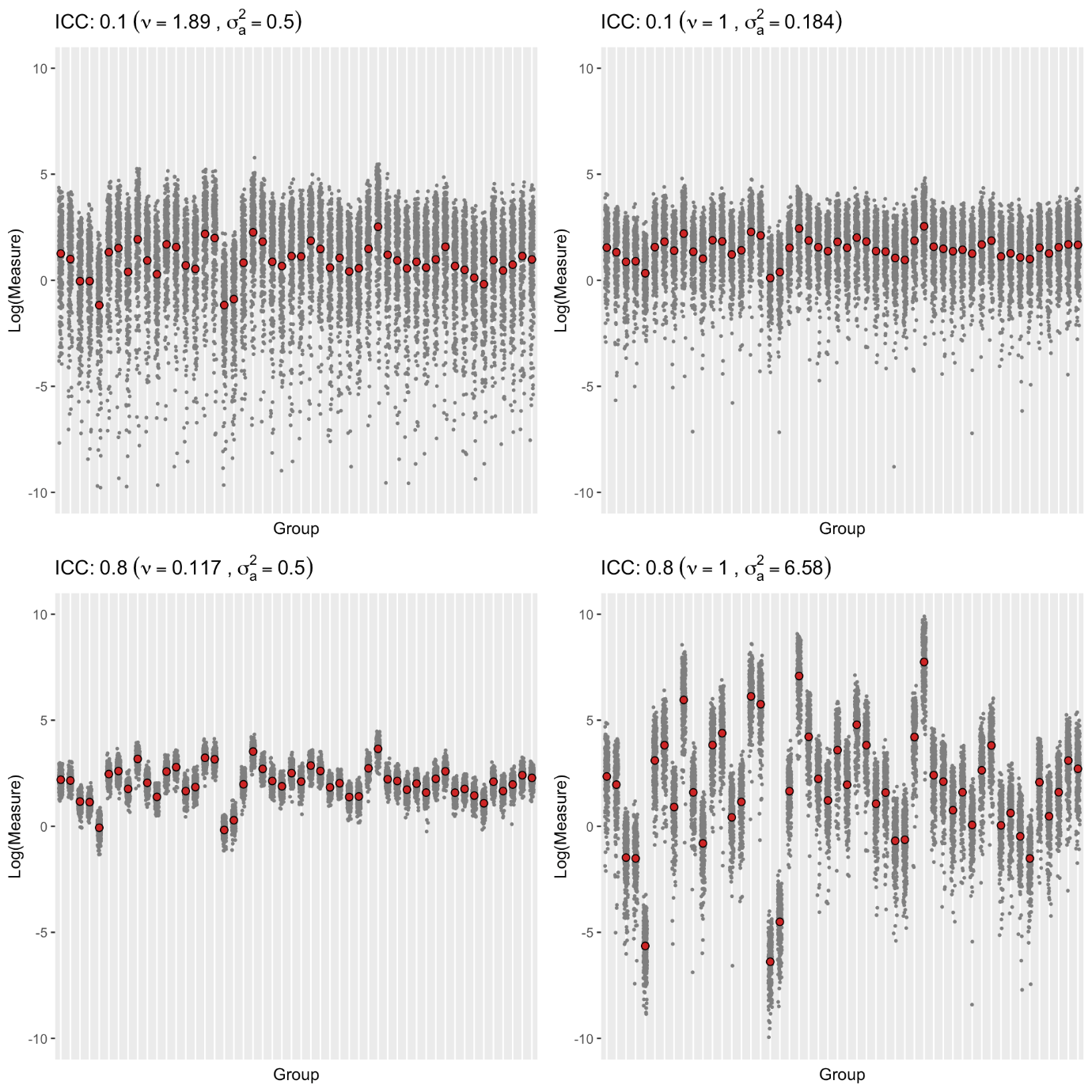
estsig <- as.data.table(VarCorr(glmerfit))[1,4] / estnu

estsig/(estsig + trigamma(1/estnu))

## vcov

## 1: 0.1003386

Finally, here are some plots of the generated observations and the group means on the log scale. The plots in each row have the same ICC but different underlying mean and dispersion parameters. I find these plots interesting because looking across the columns or up and down the two rows, they provide some insight to the interplay of group means and dispersion on the ICC …



My goal here is to highlight a little function recently added to simstudy (v0.1.9, now available on CRAN). In the course of exploring study designs for cluster randomized trials, it is often useful to understand what happens (to sample size requirements, for example) when the ICC changes. When generating the data, it is difficult to control the ICC directly – we do this by controlling the variation. With normally distributed data, the ICC is an obvious function of the variances used to generate the data, so the connection is pretty clear. But, when the outcomes have binary, Poisson, or Gamma distributions (or anything else really), the connection between variation and the ICC is not always so obvious. Figuring out how to specify the data to generate a particular ICC might require quite a bit of trial and error.

The new function, iccRE (short for ICC random effects), allows users to specify target ICCs for a desired distribution (along with relevant parameters). The function returns the corresponding random effect variances that would be specified at the cluster level to generate the desired ICC(s).

Here’s an example for three possible ICCs in the context of the normal distribution:

library(simstudy)

targetICC <- c(0.05, 0.075, 0.10)

setVars <- iccRE(ICC = targetICC, dist = "normal", varWithin = 4)

round(setVars, 4)

## [1] 0.2105 0.3243 0.4444

In the case when the target ICC is 0.075:

\[ ICC = \frac{\sigma\_b^2}{\sigma\_b ^2 + \sigma\_w ^2} = \frac{0.324}{0.324 + 4} \approx 0.075\]

**Simulating from the normal distribution**

If we specify the variance for the site-level random effect to be 0.2105 in conjunction with the individual-level (within) variance of 4, the observed ICC from the simulated data will be approximately 0.05:

set.seed(73632)

# specify between site variation

d <- defData(varname = "a", formula = 0, variance = 0.2105, id = "grp")

d <- defData(d, varname = "size", formula = 1000, dist = "nonrandom")

a <- defDataAdd(varname = "y1", formula = "30 + a",

variance = 4, dist = "normal")

dT <- genData(10000, d)

# add patient level data

dCn05 <- genCluster(dtClust = dT, cLevelVar = "grp",

numIndsVar = "size", level1ID = "id")

dCn05 <- addColumns(a, dCn05)

dCn05

## grp a size id y1

## 1: 1 -0.3255465 1000 1 32.08492

## 2: 1 -0.3255465 1000 2 27.21180

## 3: 1 -0.3255465 1000 3 28.37411

## 4: 1 -0.3255465 1000 4 27.70485

## 5: 1 -0.3255465 1000 5 32.11814

## ---

## 9999996: 10000 0.3191311 1000 9999996 30.15837

## 9999997: 10000 0.3191311 1000 9999997 32.66302

## 9999998: 10000 0.3191311 1000 9999998 28.34583

## 9999999: 10000 0.3191311 1000 9999999 28.56443

## 10000000: 10000 0.3191311 1000 10000000 30.06957

The *between* variance can be roughly estimated as the variance of the group means, and the *within* variance can be estimated as the average of the variances calculated for each group (this works well here, because we have so many clusters and patients per cluster):

between <- dCn05[, mean(y1), keyby = grp][, var(V1)]

within <- dCn05[, var(y1), keyby = grp][, mean(V1)]

total <- dCn05[, var(y1)]

round(c(between, within, total), 3)

## [1] 0.212 3.996 4.203

The ICC is the ratio of the *between* variance to the *total*, which is also the sum of the two component variances:

round(between/(total), 3)

## [1] 0.05

round(between/(between + within), 3)

## [1] 0.05

Setting the site-level variance at 0.4444 gives us the ICC of 0.10:

d <- defData(varname = "a", formula = 0, variance = 0.4444, id = "grp")

d <- defData(d, varname = "size", formula = 1000, dist = "nonrandom")

a <- defDataAdd(varname = "y1", formula = "30 + a",

variance = 4, dist = "normal")

dT <- genData(10000, d)

dCn10 <- genCluster(dtClust = dT, cLevelVar = "grp",

numIndsVar = "size", level1ID = "id")

dCn10 <- addColumns(a, dCn10)

between <- dCn10[, mean(y1), keyby = grp][, var(V1)]

within <- dCn10[, var(y1), keyby = grp][, mean(V1)]

round(between / (between + within), 3)

## [1] 0.102

**Other distributions**

The ICC is a bit more difficult to interpret using other distributions where the variance is a function of the mean, such as with the binomial, Poisson, or Gamma distributions. However, we can still use the notion of *between* and *within*, but it may need to be transformed to another scale.

In the case of **binary** outcomes, we have to imagine an underlying or latent continuous process that takes place on the logistic scale.

Ordinal Regression

I was thinking a lot about proportional-odds cumulative logit models last fall while designing a study to evaluate an intervention’s effect on meat consumption. After a fairly extensive pilot study, we had determined that participants can have quite a difficult time recalling precise quantities of meat consumption, so we were forced to move to a categorical response. (This was somewhat unfortunate, because we would not have continuous or even count outcomes, and as a result, might not be able to pick up small changes in behavior.) We opted for a question that was based on 30-day meat consumption: none, 1-3 times per month, 1 time per week, etc. - six groups in total. The question was how best to evaluate effectiveness of the intervention?

Since the outcome was categorical and ordinal - that is category 1 implied less meat consumption that category 2, category 2 implied less consumption that category 3, and so on - a model that estimates the cumulative probability of ordinal outcomes seemed like a possible way to proceed. Cumulative logit models estimate a number of parameters that represent the cumulative log-odds of an outcome; the parameters are the log-odds of categories 2 through 6 versus category 1, categories 3 through 6 versus 1 & 2, etc. Maybe not the most intuitive way to interpret the data, but seems to plausibly fit the data generating process.

I was concerned about the proportionality assumption of the cumulative logit model, particularly when we started to consider adjusting for baseline characteristics (more on that in the next post). I looked more closely at the data generating assumptions of the cumulative logit model, which are quite frequently framed in the context of a continuous latent measure that follows a logistic distribution. I thought I’d describe that data generating process here to give an alternative view of discrete data models.

I know I have been describing a context that includes an outcome with multiple categories, but in this post I will focus on regular logistic regression with a binary outcome. This will hopefully allow me to establish the idea of a latent threshold. I think it will be useful to explain this simpler case first before moving on to the more involved case of an ordinal response variable, which I plan to tackle in the near future.

### A latent continuous process underlies the observed binary process

For an event with a binary outcome (true or false, A or B, 0 or 1), the observed outcome may, at least in some cases, be conceived as the manifestation of an unseen, latent continuous outcome. In this conception, the observed (binary) outcome merely reflects whether or not the unseen continuous outcome has exceeded a specified threshold. Think of this threshold as a tipping point, above which the observable characteristic takes on one value (say false), below which it takes on a second value (say true).

### The logistic distribution

Logistic regression models are used to estimate relationships of individual characteristics with categorical outcomes. The name of this regression model arises from the logistic distribution, which is a symmetrical continuous distribution. In a latent (or hidden) variable framework, the underlying, unobserved continuous measure is drawn from this logistic distribution. More specifically, the standard logistic distribution is typically assumed, with a location parameter of 0, and a scale parameter of 1. (The mean of this distribution is 0 and variance is approximately 3.29.)

Here is a plot of a logistic pdf, shown in relation to a standard normal pdf (with mean 0 and variance 1):

**library**(ggplot2)

**library**(data.table)

my\_theme <- **function**() {

theme(panel.background = element\_rect(fill = "grey90"),

panel.grid = element\_blank(),

axis.ticks = element\_line(colour = "black"),

panel.spacing = unit(0.25, "lines"),

plot.title = element\_text(size = 12, vjust = 0.5, hjust = 0),

panel.border = element\_rect(fill = NA, colour = "gray90"))

}

x <- seq(-6, 6, length = 1000)

yNorm <- dnorm(x, 0, 1)

yLogis <- dlogis(x, location = 0, scale = 1)

dt <- data.table(x, yNorm, yLogis)

dtm <- melt(dt, id.vars = "x", value.name = "Density")

ggplot(data = dtm) +

geom\_line(aes(x = x, y = Density, color = variable)) +

geom\_hline(yintercept = 0, color = "grey50") +

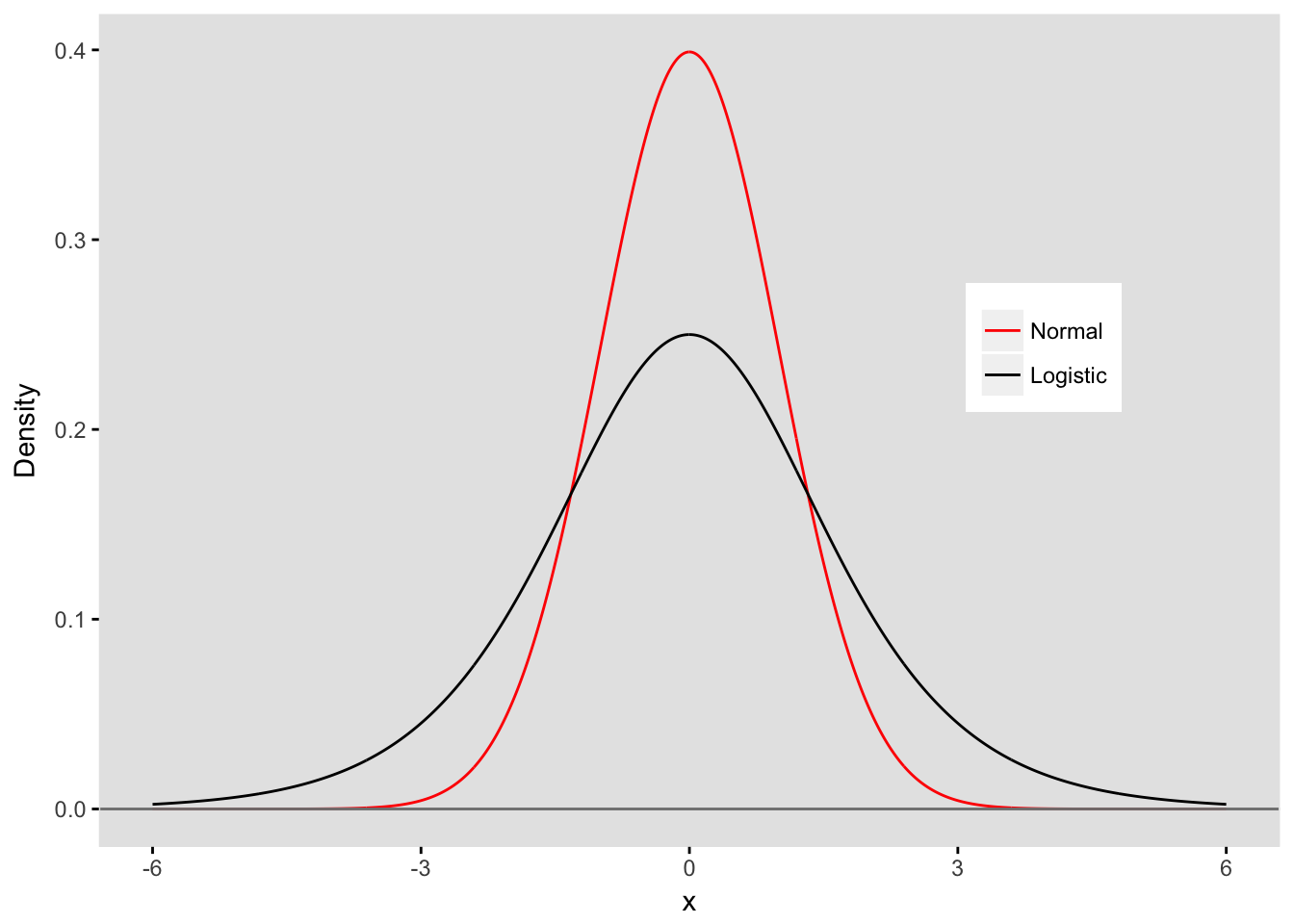
my\_theme() +

scale\_color\_manual(values = c("red", "black"),

labels=c("Normal", "Logistic")) +

theme(legend.position = c(0.8, 0.6),

legend.title = element\_blank())



### The threshold defines the probability

Below, I have plotted the standardized logistic pdf with a threshold that defines a tipping point for a particular Group A. In this case the threshold is 1.5, so for everyone with a unseen value of X &lt; 1.5*X*<1.5, the observed binary outcome Y*Y* will be 1. For those where X \geq 1.5*X*≥1.5, the observed binary outcome Y*Y* will be 0:

xGrpA <- 1.5

ggplot(data = dtm[variable == "yLogis"], aes(x = x, y = Density)) +

geom\_line() +

geom\_segment(x = xGrpA, y = 0, xend = xGrpA, yend = dlogis(xGrpA), lty = 2) +

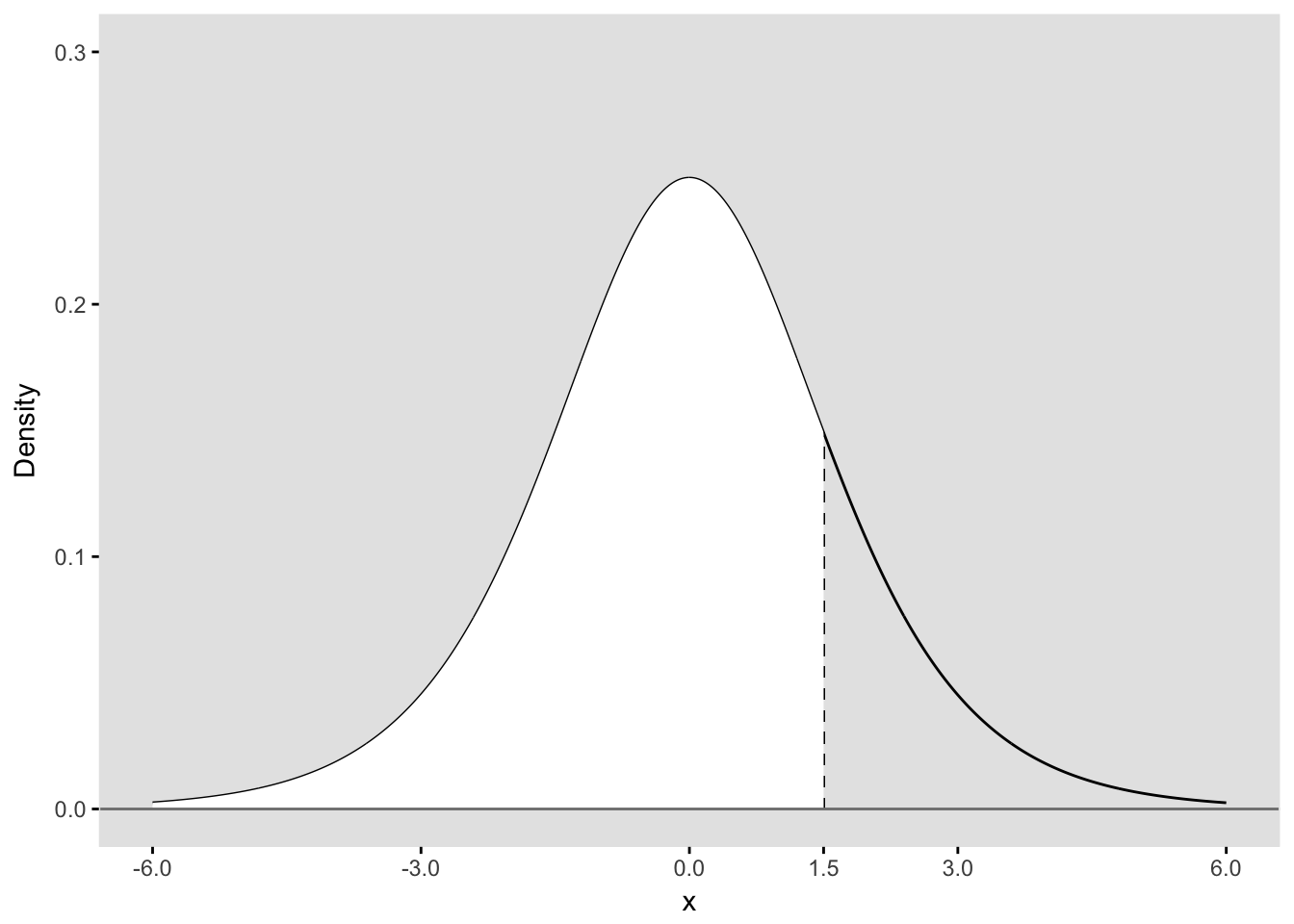
geom\_area(mapping = aes(ifelse(x < xGrpA, x, xGrpA)), fill = "white") +

geom\_hline(yintercept = 0, color = "grey50") +

ylim(0, 0.3) +

my\_theme() +

scale\_x\_continuous(breaks = c(-6, -3, 0, xGrpA, 3, 6))

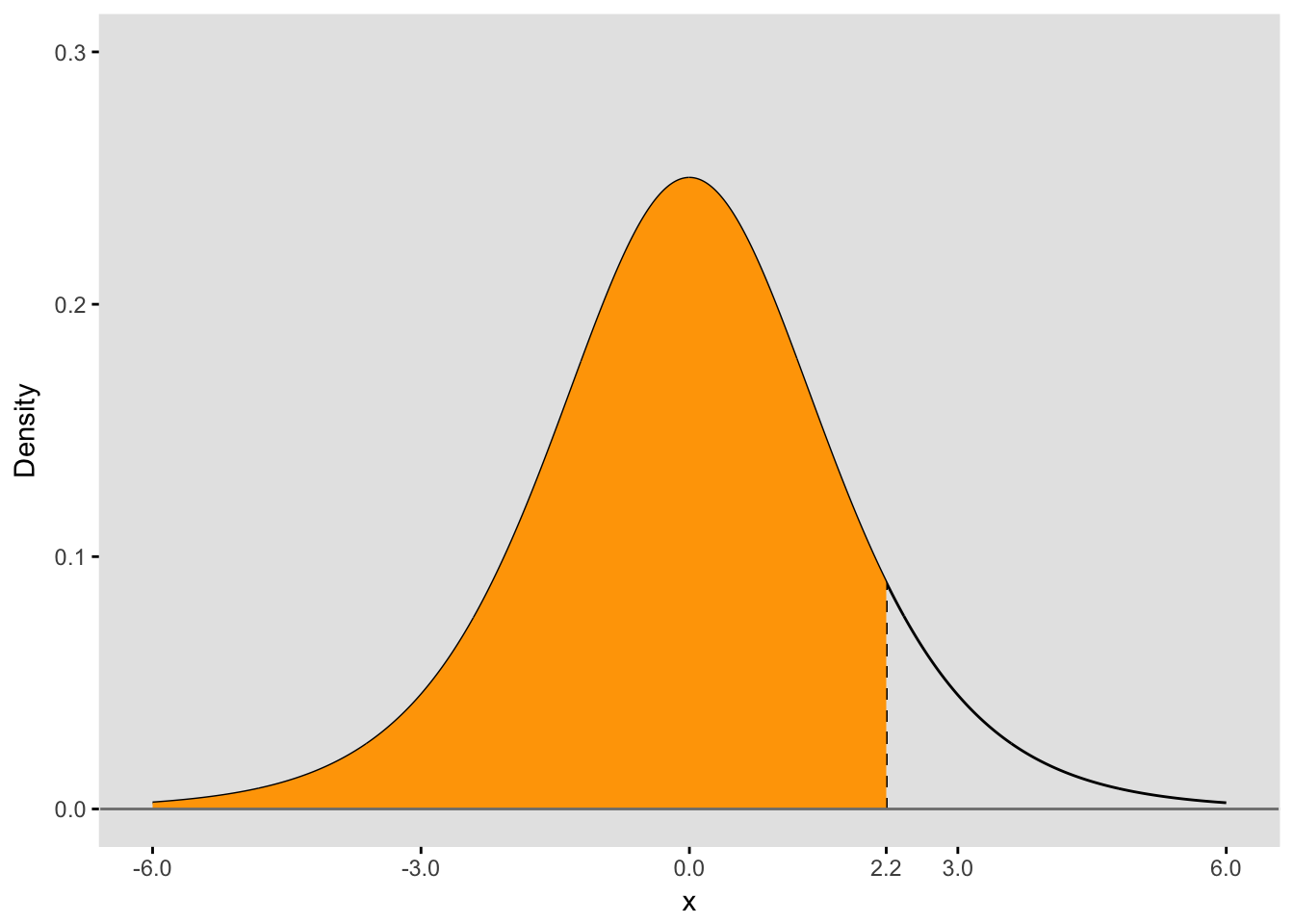


Since we have plot a probability density (pdf), the area under the entire curve is equal to 1. We are interested in the binary outcome Y*Y* defined by the threshold, so we can say that the area below the curve to the left of threshold (filled in white) represents P(Y = 1|Group=A)*P*(*Y*=1∣*Group*=*A*). The remaining area represents P(Y = 0|Group=A)*P*(*Y*=0∣*Group*=*A*). The area to the left of the threshold can be calculated in R using the plogis function:

(p\_A <- plogis(xGrpA))

## [1] 0.8175745

Here is the plot for a second group that has a threshold of 2.2:



The area under the curve to the left of the threshold is P(X &lt; 2.2)*P*(*X*<2.2), which is also P(Y = 1 | Group=B)*P*(*Y*=1∣*Group*=*B*):

(p\_B <- plogis(xGrpB))

## [1] 0.9002495

### Log-odds and probability

In logistic regression, we are actually estimating the log-odds of an outcome, which can be written as

log \left[ \frac{P(Y=1)}{P(Y=0)} \right]*log*[*P*(*Y*=0)*P*(*Y*=1)​].

In the case of Group A, log-odds of Y being equal to 1 is

(logodds\_A <- log(p\_A / (1 - p\_A) ))

## [1] 1.5

And for Group B,

(logodds\_B <- log(p\_B / (1 - p\_B) ))

## [1] 2.2

As you may have noticed, we’ve recovered the thresholds that we used to define the probabilities for the two groups. The threshold is actually the log-odds for a particular group.

### Logistic regression

The logistic regression model that estimates the log-odds for each group can be written as

log \left[ \frac{P(Y=1)}{P(Y=0)} \right] = B\_0 + B\_1 \* I(Grp = B) \quad ,*log*[*P*(*Y*=0)*P*(*Y*=1)​]=*B*0​+*B*1​∗*I*(*Grp*=*B*),

where B\_0*B*0​ represents the threshold for Group A and B\_1*B*1​ represents the shift in the threshold for Group B. In our example, the threshold for Group B is 0.7 units (2.2 - 1.5) to the right of the threshold for Group A. If we generate data for both groups, our estimates for B\_0*B*0​ and B\_1*B*1​ should be close to 1.5 and 0.7, respectively

### The process in action

To put this all together in a simulated data generating process, we can see the direct link with the logistic distribution, the binary outcomes, and an interpretation of estimates from a logistic model. The only stochastic part of this simulation is the generation of continuous outcomes from a logistic distribution. Everything else follows from the pre-defined group assignments and the group-specific thresholds:

n = 5000

set.seed(999)

# Stochastic step

xlatent <- rlogis(n, location = 0, scale = 1)

# Deterministic part

grp <- rep(c("A","B"), each = n / 2)

dt <- data.table(id = 1:n, grp, xlatent, y = 0)

dt[grp == "A" & xlatent <= xGrpA, y := 1]

dt[grp == "B" & xlatent <= xGrpB, y := 1]

# Look at the data

dt

## id grp xlatent y

## 1: 1 A -0.4512173 1

## 2: 2 A 0.3353507 1

## 3: 3 A -2.2579527 1

## 4: 4 A 1.7553890 0

## 5: 5 A 1.3054260 1

## ---

## 4996: 4996 B -0.2574943 1

## 4997: 4997 B -0.9928283 1

## 4998: 4998 B -0.7297179 1

## 4999: 4999 B -1.6430344 1

## 5000: 5000 B 3.1379593 0

The probability of a “successful” outcome (i.e P(Y = 1*P*(*Y*=1)) for each group based on this data generating process is pretty much equal to the areas under the respective densities to the left of threshold used to define success:

dt[, round(mean(y), 2), keyby = grp]

## grp V1

## 1: A 0.82

## 2: B 0.90

Now let’s estimate a logistic regression model:

**library**(broom)

glmfit <- glm(y ~ grp, data = dt, family = "binomial")

tidy(glmfit, quick = TRUE)

## term estimate

## 1 (Intercept) 1.5217770

## 2 grpB 0.6888526

The estimates from the model recover the logistic distribution thresholds for each group. The Group A threshold is estimated to be 1.52 (the intercept) and the Group B threshold is estimated to be 2.21 (intercept + grpB parameter). These estimates can be interpreted as the log-odds of success for each group, but also as the threshold for the underlying continuous data generating process that determines the binary outcome *YY*. And we can interpret the parameter for grpB in the traditional way as the log-odds ratio comparing the log-odds of success for Group B with the log-odds of success for Group A, or as the shift in the logistic threshold for Group A to the logistic threshold for Group B.

### binary

(setVar <- iccRE(ICC = 0.05, dist = "binary"))

## [1] 0.173151

d <- defData(varname = "a", formula = 0, variance = 0.1732, id = "grp")

d <- defData(d, varname = "size", formula = 1000, dist = "nonrandom")

a <- defDataAdd(varname = "y1", formula = "-1 + a", dist = "binary",

link = "logit")

dT <- genData(10000, d)

dCb05 <- genCluster(dtClust = dT, cLevelVar = "grp", numIndsVar = "size",

level1ID = "id")

dCb05 <- addColumns(a, dCb05)

dCb05

## grp a size id y1

## 1: 1 -0.20740274 1000 1 0

## 2: 1 -0.20740274 1000 2 0

## 3: 1 -0.20740274 1000 3 0

## 4: 1 -0.20740274 1000 4 1

## 5: 1 -0.20740274 1000 5 0

## ---

## 9999996: 10000 -0.05448775 1000 9999996 0

## 9999997: 10000 -0.05448775 1000 9999997 1

## 9999998: 10000 -0.05448775 1000 9999998 0

## 9999999: 10000 -0.05448775 1000 9999999 0

## 10000000: 10000 -0.05448775 1000 10000000 0

The ICC for the binary distribution is on the logistic scale, and the *within* variance is constant. The *between* variance is estimated on the log-odds scale:

within <- (pi ^ 2) / 3

means <- dCb05[,mean(y1), keyby = grp]

between <- means[, log(V1/(1-V1)), keyby = grp][abs(V1) != Inf, var(V1)]

round(between / (between + within), 3)

## [1] 0.051

The ICC for the **Poisson** distribution is interpreted on the scale of the count measurements, even though the random effect variance is on the log scale.

(setVar <- iccRE(ICC = 0.05, dist = "poisson", lambda = 30))

## [1] 0.0017513

d <- defData(varname = "a", formula = 0, variance = 0.0018, id = "grp")

d <- defData(d, varname = "size", formula = 1000, dist = "nonrandom")

a <- defDataAdd(varname = "y1", formula = "log(30) + a",

dist = "poisson", link = "log")

dT <- genData(10000, d)

dCp05 <- genCluster(dtClust = dT, cLevelVar = "grp",

numIndsVar = "size", level1ID = "id")

dCp05 <- addColumns(a, dCp05)

dCp05

## grp a size id y1

## 1: 1 0.035654485 1000 1 26

## 2: 1 0.035654485 1000 2 36

## 3: 1 0.035654485 1000 3 31

## 4: 1 0.035654485 1000 4 34

## 5: 1 0.035654485 1000 5 21

## ---

## 9999996: 10000 0.002725561 1000 9999996 26

## 9999997: 10000 0.002725561 1000 9999997 25

## 9999998: 10000 0.002725561 1000 9999998 27

## 9999999: 10000 0.002725561 1000 9999999 28

## 10000000: 10000 0.002725561 1000 10000000 37

The variance components and ICC for the Poisson can be estimated using the same approach as the normal distribution:

between <- dCp05[, mean(y1), keyby = grp][, var(V1)]

within <- dCp05[, var(y1), keyby = grp][, mean(V1)]

round(between / (between + within), 3)

## [1] 0.051

Finally, here are the results for the **Gamma** distribution, which I talked about in great length:

(setVar <- iccRE(ICC = 0.05, dist = "gamma", disp = 0.25 ))

## [1] 0.01493805

d <- defData(varname = "a", formula = 0, variance = 0.0149, id = "grp")

d <- defData(d, varname = "size", formula = 1000, dist = "nonrandom")

a <- defDataAdd(varname = "y1", formula = "log(30) + a", variance = 0.25,

dist = "gamma", link = "log")

dT <- genData(10000, d)

dCg05 <- genCluster(dtClust = dT, cLevelVar = "grp", numIndsVar = "size",

level1ID = "id")

dCg05 <- addColumns(a, dCg05)

dCg05

## grp a size id y1

## 1: 1 0.09466305 1000 1 14.31268

## 2: 1 0.09466305 1000 2 39.08884

## 3: 1 0.09466305 1000 3 28.08050

## 4: 1 0.09466305 1000 4 53.27853

## 5: 1 0.09466305 1000 5 37.93855

## ---

## 9999996: 10000 0.25566417 1000 9999996 14.16145

## 9999997: 10000 0.25566417 1000 9999997 42.54838

## 9999998: 10000 0.25566417 1000 9999998 76.33642

## 9999999: 10000 0.25566417 1000 9999999 34.16727

## 10000000: 10000 0.25566417 1000 10000000 21.06282

The ICC for the Gamma distribution is on the log scale:

between <- dCg05[, mean(log(y1)), keyby = grp][, var(V1)]

within <- dCg05[, var(log(y1)), keyby = grp][, mean(V1)]

round(between / (between + within), 3)

## [1] 0.05

It is possible to think about the ICC in the context of covariates, but interpretation is less straightforward. The ICC itself will likely vary across different levels of the covariates. For this reason, I like to think of the ICC in the marginal context.

I leave you with some visuals of clustered binary data with ICC’s ranging from 0 to 0.075, both on the log-odds and probability scales:

