Clustered Data

Michael Clark

2016-12-15

Contents

Chapter 1

Chapter 2

Introduction

2.1 What approach to use?

When dealing with clustered data we have observations that are non-independent and reside within some grouping structure. This could be replicate samples for a specimen in an experiment, children nested within schools, which themselves are nested within districts, we might obtain the same measures at different time points for the same unit of observation, etc.

In such cases standard methods may be applied, just as they can be anywhere, but they would be lacking in some respect. The following demonstrates a number of methods that might be used in these situations, but while some will definitely be better than others, in other cases, it might be a practical or theoretical choice that drives the decision on how to deal with the data. In what follows, I will create a data set for us to use so that we don't have any questions as to what might be going on, or what nuances we are not privy to that might produce differences in the results.

2.2 Preliminaries

2.2.1 Things to note

No attempt is made at providing too much detail on model specifics. It is assumed that if you're reading this you're already at least conceptually familiar in some sense with a couple of these models. The primary purpose here is to compare, contrast, and note issues. Sometimes a little more detail is required, but again, the goal is not to teach the methods themselves. See the references for more detail.

In many cases I hide the code but it is still available for viewing, especially in cases where it involves mostly data setup or wrangling, or involves more detailed technique than I am assuming some readers of this doc will understand. I prefer this document to be a more conceptual in nature and tool agnostic. Though it relies on R, the tool used doesn't matter, and all the models conducted here could be run using other packages.

In describing some of these approaches I suggest that some are more discipline specific or more common in certain areas. This assessment is based on years of consulting across many, many academic disciplines, as well as reading many discipline-specific texts and articles, but this admittedly is not a random sample, so take that info as you wish. I just provide it as a means for the reader to understand why they may not be familiar with such a technique, and where they might find more information or examples.

2.2.1.1 Terminology

Note that in the following, terms such as 'clustering' and 'cluster' have nothing to do with unsupervised learning methods such as mixture models. I use the terms 'cluster', 'grouping' and similar terms as synonyms. As 'mixed' models incorporate random effects, I don't normally distinguish between a *random effects model* (which could be seen as the general case) and a *mixed model*, except perhaps when comparing to 'fixed effects models'. I'm not going to get into it here, but 'fixed effects' and 'fixed-effect models' are not the same thing[1]. I will try to be clear.

Color coding:

- emphasis
- package
- function
- object/class
- link

This doc also available in .epub and .pdf (See the icon above), though the doc is automatically converted to those formats, and to be perfectly honest I don't care what may or may not look good or transfer appropriately.

[1] Blame the economists.

Chapter 3

Data setup

First we need to set up the data, and to do so we'll consider the longitudinal setting in which individuals are observed over time[1]. The goal is to create a data set with several (four) observations per individual, i.e. the 'repeated measures' scenario. Aside from variables that represent individual and time point, I will also create an individual level variable, which means that it is constant for each individual. In this case it will be a simple binary variable indicating 'treatment' or 'control' as in an experiment.

The data will also have a certain correlation structure, an autoregressive structure. This means that the residuals of each time point will be notably correlated with the previous, but less so as the time between observations increases. If this is unfamiliar territory to you, just see it as an extra parameter (called rho in the following code) we'll have to estimate and by which you can check the model results against the 'true' value.

Here are the main parameters of interest.

```
library(Matrix)
set.seed(8675309)
tp = 4              # number of timepoints
n = 2500            # number of individuals
sigmasq = 1         # residual variance
rho = .7            # rho of AR1 model

intercept = .2      # intercept
time_beta = .5      # time effect
treat_beta = -.5    # treatment effect

intsd = .5          # intercept standard devation
timesd = .25        # slope of time variable standard deviation
```

Now we can create the data. Here are the main components- time, individual id, and treatment. We will also create an individual specific effect for intercepts and the slope for time. These are our 'random' effects[2].

```
time = rep(0:(tp-1), n)                                    # time
id = rep(1:n, e=tp)                                        # group id
treatment = gl(2, n/2, labels=c('control', 'treatment'))[id]     # cluster level variable
re_int = rnorm(n, sd=intsd)                                # random intercepts
re_time = rnorm(n, sd=timesd)                              # random slopes
```

[1]This will allow us to examine another technique later, growth curve models, that would not apply to the non-longitudinal case.

[2]Note there is nothing *random* about them, they merely represent all causes not identified by the model that vary amongst individuals. They are an additional source of uncertainty.

Next we create the residual structure, made somewhat easier since the residual variance is set to 1. For those interested, this is an autoregressive structure of lag 1. This means that when time points are 1 measure apart, the correlation is ρ, at two measures apart, ρ^2, at three measures ρ^3. You get the gist. This structure is the same within each individual. Once that is set we draw the residuals based on a multivariate normal with mean zero and covariance based on the structure we have provided.

```
# create residual
ar1 = bandSparse(tp, tp, 0:(tp-1), list(rep(1    , tp),
                                        rep(rho  , tp-1),
                                        rep(rho^2, tp-2),
                                        rep(rho^3, tp-3)), symmetric=T)
Sig = kronecker(diag(1, n), ar1)
Sig[1:10, 1:10]                 # inspect, note that dots are 0s
```

```
10 x 10 sparse Matrix of class "dgTMatrix"

 [1,] 1.000 0.70 0.49 0.343 .     .    .    .     .   .
 [2,] 0.700 1.00 0.70 0.490 .     .    .    .     .   .
 [3,] 0.490 0.70 1.00 0.700 .     .    .    .     .   .
 [4,] 0.343 0.49 0.70 1.000 .     .    .    .     .   .
 [5,] .     .    .    .     1.000 0.70 0.49 0.343 .   .
 [6,] .     .    .    .     0.700 1.00 0.70 0.490 .   .
 [7,] .     .    .    .     0.490 0.70 1.00 0.700 .   .
 [8,] .     .    .    .     0.343 0.49 0.70 1.000 .   .
 [9,] .     .    .    .     .     .    .    .     1.0 0.7
[10,] .     .    .    .     .     .    .    .     0.7 1.0
```

```
# e = MASS::mvrnorm(1, mu=rep(0, n*tp), Sigma=sigmasq*Sig)              # residual error
e = c(replicate(n, MASS::mvrnorm(1, mu=rep(0, tp), Sigma=sigmasq*ar1))) # much faster
```

Now we put it all together and create a data.frame object. A few entries are shown.

```
# target variable
y = (intercept + re_int[id]) +
    (time_beta + re_time[id])*time +
    treat_beta*(treatment=='treatment') +
    e

d = data.frame(y, time, treatment, id)
```

Now we are ready to proceed.

Chapter 4

Ignore data dependency

The first thing we can do is ignore the situation and just run a standard regression. This is actually okay if you have very few clusters, and put the cluster id in the model as a fixed effect. Otherwise, this is not acceptable with regard to the standard errors (SE), as cluster level covariates will be treated as if there are N*timepoint observations (typically underestimating the SE as a result), while the standard error for the time-varying covariates will not account for the clustering (typically overestimating).

```
lm_mod = lm(y ~ time + treatment, data=d, x=T)
```

	Estimate	Std. Error	t value	Pr(>|t|)
time	0.51	0.01	46.74	0
treatmenttreatment	-0.42	0.02	-17.33	0
(Intercept)	0.13	0.02	5.52	0

Table 4.2: Fitting linear model: y ~ time + treatment

Observations	Residual Std. Error	R^2	Adjusted R^2
10000	1.216	0.1991	0.1989

First be aware that the 'treatmenttreatment' label just tells us that the coefficient refers to moving from the reference group (i.e. 'control') to the treatment group, i.e. considers treatment a binary variable where 1 equals treatment and 0 control. Note that the coefficients are in the ballpark of where the true values are, save for the estimate of the residual variance, which packs in all sources of variance into one estimate. As mentioned though, the standard errors for the effects would be problematic.

4.1 Pros

- Easy
- Provides estimation of the effects most are primarily interested in

4.2 Cons

- Standard errors are off

- Ignores the cluster-specific effects, which may be highly interesting

Gist: Probably not viable for most situations.

Chapter 5

Model every group

We could also run a regression for every group. This is problematic since we would typically have so little data per group, and too many groups to make sense of. We'll go ahead and do it anyway. What if we take the average of all the estimates?

```
library(nlme)
lottalm = lmList(y ~ time | id, d)
colMeans(coef(lottalm))
```

```
(Intercept)        time
-0.07981721  0.50855877
```

```
coef(lm(y ~ time, data=d, x=T))
```

```
(Intercept)        time
-0.07981721  0.50855877
```

Surprise! We get the same result as if we had simply run the standard linear model (I abbreviate this as SLiM sometimes). However, we would not get identical coefficients if the data are unbalanced, where individuals may have a different numbers of observations.

In general, this is not the way we want to do things, and one of the biggest drawbacks is that we can never examine cluster level covariates, which may be of key interest, nor is there a straightforward way to do inference for this scenario. As we will see later, techniques are available that serve as a compromise between these first two alternatives of ignoring the clustering and overfitting to each cluster.

5.1 Pros

- None really.

5.2 Cons

- Inefficient
- Overly complex
- Overfit at each cluster
- High variance in estimates at each cluster
- Ignores cluster level effects

Gist: While it might be a fun exercise, there is little to be gained by this approach.

Chapter 6

Cluster robust variances

As mentioned, if we ignore the clustering, the so-called fixed effects estimates (i.e. coefficients) are correct, but their variances are not. We can get proper estimates of the standard errors via cluster robust standard errors, which are very popular in econometrics and fields trained in that fashion, but not widely used elsewhere in my experience. Essentially, these allow one to fire-and-forget, and treat the clustering as more of a nuisance.

In programs like Stata, obtaining these are basically an option for most modeling procedures. In R, it's not quite as straightforward, but not difficult. There are packages such as sandwich that can provide heteroscedastic robust standard errors, but won't necessarily take into account clustering. I provide a custom function that will work in this example so that the curtain can be pulled back a little, but the plm package would be the way to go for cluster robust standard errors. The plm package can take into account the serial autocorrelation via the 'arellano' input to the `type` argument.

```
library(plm)
# create numeric time so plm won't treat as a factor, time is a reserved name in plm
t2 = d$time
clusterrob_mod <- plm(y ~ t2 + treatment, data=d, model='pooling', index='id')
summary(clusterrob_mod)
```

```
Oneway (individual) effect Pooling Model

Call:
plm(formula = y ~ t2 + treatment, data = d, model = "pooling",
    index = "id")

Balanced Panel: n=2500, T=4, N=10000

Residuals :
   Min. 1st Qu.  Median 3rd Qu.    Max.
-4.7200 -0.8110  0.0302  0.8160  4.2300

Coefficients :
                   Estimate Std. Error  t-value  Pr(>|t|)
(Intercept)         0.130929   0.023712   5.5217 3.442e-08 ***
t2                  0.508559   0.010880  46.7436 < 2.2e-16 ***
treatmenttreatment -0.421493   0.024328 -17.3255 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    18469
```

```
Residual Sum of Squares: 14792
R-Squared:      0.1991
Adj. R-Squared: 0.19904
F-statistic: 1242.57 on 2 and 9997 DF, p-value: < 2.22e-16
```

The following compares the different robust SEs we could produce at this point.

```
resids = resid(lm_mod)                    # residuals
xvar = solve(crossprod(lm_mod$x))         # (X'X)^-1
gd0 = data.frame(id=d$id, r=resids, lm_mod$x)

# custom function to be applied within cluster
cluscom = function(mat){
  X = as.matrix(mat)[, 3:5]               # magic numbers represent int, time and treatment columns
  crossprod(X, tcrossprod(mat$r)) %*% X
}

# calculate within cluster variance
gd = gd0 %>%
  group_by(id) %>%
  do(xvar = cluscom(.))

# plm
plm_se = vcovHC(clusterrob_mod, method='arellano', cluster='group') %>% diag %>% sqrt %>% round(3)

# custom output
custom_se = (xvar %*% Reduce(`+`, gd$xvar) %*% xvar) %>% diag %>% sqrt %>% round(3)

# original lm
lm_se = vcov(lm_mod) %>% diag %>% sqrt  %>% round(3)

# non-clustered 'robust' se
lm_robse = vcovHC(lm_mod, type='HC0') %>% diag %>% sqrt %>% round(3)
```

```
# plm
plm_se
```

```
      (Intercept)                t2 treatmenttreatment
            0.030             0.009                0.042
```

```
# custom output
custom_se
```

```
      (Intercept)              time treatmenttreatment
            0.030             0.009                0.042
```

```
# original lm
lm_se
```

```
      (Intercept)              time treatmenttreatment
            0.024             0.011                0.024
```

```
# non-clustered 'robust' se
lm_robse
```

```
      (Intercept)              time treatmenttreatment
            0.023             0.011                0.024
```

6.1 Pros

- Don't have to think too much about the issues at play
- Fewer assumptions about the model
- A general heteroscedastic 'robust' approach

6.2 Cons

- Ignores cluster-specific effects
- By default, assumes no intracluster correlation
- Few tools go beyond simple clustering, requiring manual approach
- Not as general as GEE approach
- Suggests that there are problems in model specification that the method itself does not correct

Gist: if your goal is to simply do a standard GLM approach and essentially ignore the clustering, treating it more or less as a nuisance to overcome, and your grouping structure is simple, then this may be for you. However, note that the GEE approach as a possibly more flexible alternative that can still incorporate cluster robust standard errors[1]. Furthermore, differences between the robust and regular SE may suggest a model misspecification issue. 'Fixing' standard errors won't solve that problem.

[1]The cluster robust standard errors as depicted above assume independent error structure in the GEE model, and thus are a special case of GEE. When you peruse the GEE model, rerun it with the argument `corst='ind'` to duplicate the plm and custom results above.

Chapter 7

A Survey Approach

I will only briefly mention an approach using survey design to show the similarity of results in that scenario to using cluster robust standard errors. We'll use the survey package and subsequent svyglm function.

For comparison, we'll use a cluster-based sampling design and nothing more. This assumes we are sampling clusters from the population of interest for which we want to make inferences to. To use most survey versions of models, the design must be specified a priori.

```
library(survey)

design = svydesign(ids=~id, data=d)
svy_mod = svyglm(y ~ time + treatment, data=d, design=design)
summary(svy_mod)
```

```
Call:
svyglm(formula = y ~ time + treatment, data = d, design = design)

Survey design:
svydesign(ids = ~id, data = d)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         0.130929   0.030500   4.293 1.83e-05 ***
time                0.508559   0.009143  55.624  < 2e-16 ***
treatmenttreatment -0.421493   0.041647 -10.121  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 1.479315)

Number of Fisher Scoring iterations: 2
```

These are quite similar to the cluster robust standard errors we got earlier.

```
      (Intercept)              time treatmenttreatment
         0.030494          0.009141           0.041638
```

In fact, they'd be identical by using a finite population correction on the latter.

```
      (Intercept)              time treatmenttreatment
         0.030500          0.009143           0.041647
```

I only note pros and cons that are relevant for our purposes. The pros and cons of dealing with survey design in general are quite complex and better hashed out elsewhere.

7.1 Pros

- Can incorporate different and quite complicated sampling designs
- More confidence in inference to the populations of interest

7.2 Cons

- The complexity of incorporating complex design and associated weights
- Beyond simpler settings it can be difficult to tell how best to utilize survey design within the modeling context

Gist: Our goal here was merely to provide a connection to survey design, but that's a whole other situation that will not be considered further.

Chapter 8

Fixed Effects Models

Fixed Effects (FE) models are a terribly named approach to dealing with clustered data, but in the simplest case, serve as a contrast to the random effects (RE) approach in which there are only random intercepts[1]. Despite the nomenclature, there is mainly one key difference between these models and the 'mixed' models we discuss. Both allow a (random) cluster-specific effect to be added to model, but the FE approach allows that effect to correlate with the predictors, while the RE does not (by default). In practice however, this means that they may end up being quite different conceptual models as well. As with cluster robust standard errors, economists, and again those trained in that fashion, have historically preferred these models. In my experience they are rarely used in other disciplines.

First, let us understand this cluster-specific effect. In the standard regression setting we have a basic intercept, while here, each cluster will provide a nudge above or below that overall effect. Consider the following model (ignoring treatment for now):

$$y = \text{Int} + \text{ClusterEffect} + b * \text{time} + \epsilon$$

The cluster effect is different from one cluster to the next, but constant for a given cluster. One way we could perform such a model is just to include `id` as a predictor, thereby getting a unique estimate for each cluster added to the model. In other words, we can also see the situation as if one had simply created a dummy variable for `id` and conducted a standard linear model. This is in fact how one can think of the FE model, though where the cluster-specific effects are assumed constants to be estimated, and in the past these models were sometimes referred to as least squares dummy variable (LSDV) regression models[2]. If you actually run the LSDV model, the statistical results for time will be identical.

Why would we be worried about the potential correlation between the cluster-specific effects and the model covariates? In typical social science and economic data it's probably likely that unspecified cluster level effects might have some correlation with the individual level covariates. This leads to inconsistent estimates in the RE approach, and as such the FE might be used instead.

In the following we use the plm package to estimate the FE model. I highly recommend reading the excellent vignette for this package if you are one of those econometrically trained folk new to R or the mixed model approach, or conversely, other folk wishing to understand the econometric perspective.

```
FE_mod = plm(y ~ as.numeric(time) + treatment, data=d, index='id', model='within')
summary(FE_mod)
```

```
Oneway (individual) effect Within Model
```

[1] Actually, FE models extend beyond this but I've never seen the treatment in textbook presentations, nor am familiar with tools that do so aside from the latent variable approach.

[2] This Stata note highlights the distinction.

```
Call:
plm(formula = y ~ as.numeric(time) + treatment, data = d, model = "within",
    index = "id")

Balanced Panel: n=2500, T=4, N=10000

Residuals :
    Min.  1st Qu.   Median  3rd Qu.     Max.
-2.66000 -0.40500  0.00265  0.40000  2.56000

Coefficients :
                  Estimate Std. Error t-value  Pr(>|t|)
as.numeric(time) 0.5085588  0.0064962  78.285 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    7188.7
Residual Sum of Squares: 3955.8
R-Squared:      0.44972
Adj. R-Squared: 0.33725
F-statistic: 6128.62 on 1 and 7499 DF, p-value: < 2.22e-16
```

Note how there is no intercept or treatment effect. In this circumstance of a random intercept model, the FE model can also be seen as a 'demeaning' approach, were the model within a cluster is:

$$y_i - \bar{y_i} = (X_i - \bar{X_i})\beta + (\epsilon - \bar{\epsilon_i})$$

In other words, we subtract the mean from each covariate and response and run the model that way (this is also known as the within transformation). Note the following produces the same result, although the standard error for time is off[3].

```
d %>%
  group_by(id) %>%
  mutate(ybar = y-mean(y),
         timebar = time-mean(time)) %$%
  lm(ybar ~ timebar) %>%
  summary
```

```
Call:
lm(formula = ybar ~ timebar)

Residuals:
     Min       1Q   Median       3Q      Max
-2.66127 -0.40475  0.00265  0.40017  2.56068

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 5.741e-19  6.290e-03    0.00        1
timebar     5.086e-01  5.626e-03   90.39   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.629 on 9998 degrees of freedom
```

[3]This is due to the fact that estimation of the group means was not taken into account.

```
Multiple R-squared:  0.4497,    Adjusted R-squared:  0.4497
F-statistic:  8171 on 1 and 9998 DF,  p-value: < 2.2e-16
```

Because of this, if something is constant within a cluster, it drops out of the model, and this includes anything that has only one observation even if the covariate is normally time-varying. So, not only do you lose the ability to model cluster level effects, though these are 'controlled for', you also lose data. In this we lose the treatment effect entirely, which would be completely unacceptable in most circumstances[4].

There seem to be philosophical reasons why some prefer FE models that go beyond the practical, because I don't understand the often rigid preference by some adherents over RE models given the drawbacks. I personally have never come across a valid justification for not investigating cluster level covariates if they are available (i.e. almost always in social science, educational, economic, epidemiological and other data, and often would simply include cluster-level averages of available variables). In addition, few of the applications of FE models actually seem interested in the cluster-specific effects, in short treating the clustering as a nuisance, much like the cluster-robust standard error approach[5].

8.1 Pros

- Does not assume X and random effects are uncorrelated.

8.2 Cons

- Ignores cluster level covariates or anything cluster constant (i.e. will almost always lose data).
- Doesn't easily extend to more complex clustering structures.
- Less efficient that RE if RE assumption holds
- Technically one can do random slopes also (mentioned in passing in Greene), but nothing out there does.
- Awkward (my opinion) extension to GLM setting for binary and counts
- More will be pointed out with the mixed models

Gist: If your goal is statistical consistency above all other considerations, this approach is for you. However, given that with mixed models we can potentially overcome the primary issue that the FE model addresses (RE correlated with covariates[6]), this seems a difficult modeling approach to justify.

[4]Note that you could still get the interaction of time x treatment, which you'd definitely want to examine in the experimental longitudinal setting. In other circumstances and with numerous covariates, this may become unwieldy, and then there are the issues of when the interaction is not significant, you have no main effect to fall back on, and you're also testing an interaction without all the component main effects.

[5]Still applies here, i.e. we can still use cluster robust standard errors.

[6]One can use aggregated values of the potentially offending covariates as cluster level covariates. For example, if we had people clustered within political district, we could use average income as a district-level covariate. Such models are sometimes referred to as hybrids, incorporating both the FE and RE approaches, but this is unwarranted. All three are simply random effects models of different kinds.

Chapter 9

Mixed Models

Mixed models, also known by other names, explicitly model the random effects due to the clustering in the data. They are extremely flexible approaches that can handle crossed and nested clustering structures, as well as different residual dependency structures. In addition, they can fit within other modeling approaches that can be expressed essentially identically[1], allowing for even more modeling considerations. They also have a very natural extension to Bayesian estimation, and, as we will see later, an alternative 'latent variable' interpretation. In short, there is a lot going on here.

For observations i nested within cluster c, we can depict the model in a couple of different ways, but the following is fairly straightforward and in keeping with how the data was created. The main idea is that we have cluster specific coefficients for the intercept and time.

$$
\begin{aligned}
y_{ic} &= \beta_{0c} + \beta_{1c} * \text{time}_{ic} + \epsilon_{ic} \\
\beta_{0c} &= \beta_0 + \beta_2 * \text{Treat}_\text{c} + \gamma_c \\
\beta_{1c} &= \beta_1 + \nu_c \\
\gamma_c &\sim \mathcal{N}(0, \tau^2) \\
\nu_c &\sim \mathcal{N}(0, \eta^2) \\
\epsilon_i c &\sim N(0, \sigma^2)
\end{aligned}
$$

Putting the model together we get:

$$y_{ic} = (\beta_0 + \gamma_c) + (\beta_1 + \nu_c) * \text{time}_{ic} + \beta_2 * \text{Treat}_\text{c} + \epsilon_{ic}$$

So what we end up with conceptually is the standard linear model, but with cluster specific deviations/effects. These effects are *random* in that they are drawn from a specific distribution, in this case normal, with mean 0 and some standard deviation, which is estimated as part of the model. Recall that we specified these values at the beginning though in terms of standard deviation. In addition, the cluster level effect of treatment enters the model as would any other covariate.

To estimate this model we'll use the nlme package, as it will also allow us to easily estimate the residual correlation structure. The key part is the `random =` argument, where we note that we want random effects for the intercept and time.

```
library(nlme)
mixed_mod = lme(y ~ time + treatment, random = ~1+time|id, correlation=corAR1(form=~time|id))
```

[1]Such as spatial random effects, additive models, network effects etc. See the Fahrmeier et al. reference.

Table 9.1: Fixed effects: y ~ time + treatment

	Value	Std.Error	DF	t-value	p-value
(Intercept)	0.133	0.03	7499	4.448	0
time	0.507	0.009	7499	55.85	0
treatmenttreatment	-0.425	0.04	2498	-10.57	0

Table 9.2: Standardized Within-Group Residuals

Min	Q1	Med	Q3	Max
-3.197	-0.5489	0.0173	0.5486	3.282

Table 9.3: Linear mixed-effects model fit by REML : y ~ time + treatment

	Observations	Groups	Log-restricted-likelihood
id	10000	2500	-12630

Variance	StdDev
0.332	0.577
0.067	0.258
0.908	0.953

Phi
0.6748

The results show that the primary regression coefficients, known in the mixed model literature as 'fixed effects', are reasonably recovered, as are the standard deviations for the random effects (recall `intsd`, `timesd`, and `sigmasq`). In addition, `Phi` estimates what we were calling `rho`. Furthermore, there is a low estimated correlation between intercepts and slopes (not shown), in keeping with how the data was generated.

A common method for determining whether to use the FE vs. RE is the Hausman test, but it comes with its own assumptions and the standard version is problematic except in the simplest of settings. The plm package provides this if you want to try it, but will only be applicable to the random intercepts scenario. It tests whether the assumption underlying the random effects approach is viable, and if rejected, one would go with the FE model. As it is an 'accept the null' test, it is fundamentally illogical, to go along with the other issues.

It may be instructive to compare the random coefficients to the approach where we conducted a model for each individual.

What the visualization makes clear is that the mixed model has a regularizing effect on the coefficients relative to the by-group model, shrinking the coefficients toward the population average. The by-group approach overfits, treating each individual as a unique snowflake, while the mixed model recognizes that they might have things in common. Practically and philosophically this is quite appealing.

9.1 Pros

- Examine cluster-specific effects
- Can retain cluster level variable effects
- More efficient than FE (and cluster robust approach) if assumptions hold
- Easily incorporate additional sources of variance
- Additional sources of variance may be due to nested or crossed grouping structure and even interactions of random effects
- Can model residual dependency
- Can allow any coefficient to be 'random'
- Close ties to other modeling approaches

9.2 Cons

- Inconsistent estimates if predictors are correlated with random effect(s) and other steps not taken
- May have difficult estimation in the generalized case, e.g. logistic model (my experience)

Gist: Basically the mixed model can provide everything you need, and not necessarily at the cost of consistency.

Chapter 10

Generalized Estimating Equations

A generalized estimating equation is an *estimation procedure*[1] for dealing with clustered data, and is seemingly very popular in disciplines trained with a biostatistics perspective, but perhaps not too commonly used elsewhere. Models using this approach are sometimes called marginal models, and can be seen as follows, where the target y is multivariate normal with mean vector $X\beta$ and covariance matrix Σ, which is typically block diagonal as we created at the beginning[2].

$$y \sim \mathcal{N}(X\beta, \Sigma)$$

Here the focus is on the 'population average' effects, akin to 'fixed' effects in the RE model, and in some circumstances they are identical. In addition, one specifies the type of covariance structure thought to underlie the data. Among the more common are:

- *independence*: no correlation
- *autoregressive*: as described previously
- *exchangeable*: same correlation everywhere (aka 'compound symmetry' or 'spherical')
- *unstructured*: all possible correlations are estimated as parameters

And there are many others. The GEE approach is identical to RE intercept-only model approach if one conducts a linear Gaussian model, as in this case. In addition, these correlation structures are often available to mixed models tools, so this extension alone should not be a reason[3] to use the GEE approach.

We'll use the geepack package in order to conduct the gee approach to the model. We'll specify an autoregressive correlation structure as we did with the mixed model, and as the data was in fact designed with.

```
library(geepack)

gee_mod = geeglm(y ~ time + treatment, data=d, corstr='ar1', id=id, waves=time)
```

	Estimate	Std.err	Wald	Pr(>|W|)
(Intercept)	0.127	0.03	17.7	0
time	0.507	0.009	3118	0
treatmenttreatment	-0.413	0.041	101	0

[1] Not a model! We're just doing a GLM here.

[2] The corresponding matrix formulation for the *conditional* model of the random effects approach is:

$$y \sim \mathcal{N}(X\beta + Z\Gamma, \Sigma)$$

where Z is some subset of X, and Γ contains the random effects.

[3] lme4 is a very widely used mixed model package that does not allow the specification of a correlation structure for the residuals.

Residual Variance
1.479

	Estimate	Std.err
alpha	0.757	0.007

We should be getting used to the coefficient estimates for the population-average, a.k.a. fixed effects, by now. We also obtain an estimate for the residual correlation, here noted as *alpha*.

Note the similarities here compared with the mixed model where there is only a random intercept. A couple of changes are made to keep things as similar as possible.

```
mixed_mod_ri_only = lme(y ~ time + treatment, data=d, random = ~1|id, cor=corAR1(form=~time),
                        control=lmeControl(opt='optim'), method='ML')  # changed opt bc nlminb had issu
```

Table 10.4: Fixed effects: y ~ time + treatment

	Value	Std.Error	DF	t-value	p-value
(Intercept)	0.126	0.032	7499	3.92	0
time	0.507	0.008	7499	60.6	0
treatmenttreatment	-0.412	0.042	2498	-9.82	0

Table 10.5: Standardized Within-Group Residuals

Min	Q1	Med	Q3	Max
-3.847	-0.6583	0.02588	0.6645	3.448

Table 10.6: Linear mixed-effects model fit by maximum likelihood
: y ~ time + treatment

	Observations	Groups	Log-restricted-likelihood
id	10000	2500	-12707

Variance	StdDev
0.003	0.051
1.504	1.227

Phi
0.7808

The population average effects are identical (though the geeglm function automatically does cluster robust standard errors). The estimated correlations for both are similar, and a bit high. There is essentially no

cluster variance in the mixed model, and both estimated residual variances are similar, and similar to the standard linear model we started with. This makes sense as the variance is equal to the residual variance + the intercept variance + slope variance.

```
summary(lm_mod)$sigma^2                    # similar to gee and random intercept only
```

```
[1] 1.479611
```

```
sum(as.numeric(VarCorr(mixed_mod)[,1]))    # model that incorporates all sources of variance
```

```
[1] 1.306739
```

```
intsd^2 + timesd^2 + sigmasq               # 'truth'
```

```
[1] 1.3125
```

GEE models are generally robust to misspecification of the covariance structure. They are rarely implemented for more than simple clustering but some tools allow for it[4].

10.1 Pros

- Easy modeling of different correlation structures
- Focus on population level effects
- Extends the cluster robust approach to glm setting and other correlation structures[5]
- Robust to misspecification of correlation structure
- May be more feasible with larger data situations than mixed models

10.2 Cons

- Cluster-specific effects are not estimated
- Not easily extendable to other clustering situations, e.g. nested
- Missing data is assumed Missing Completely at Random (MCAR) (RE and FE assume MAR)

Gist: If your goal is to focus on population average effects and ignore subject specific effects, without the drawbacks of the FE model, this might be considered

[4]CSCAR director Kerby Shedden has worked on a Python implementation for nested structures as part of the statsmodels module.

[5]Recall the note at the end of the cluster robust SE chapter.

Chapter 11

Latent Growth Curve

An alternative approach to mixed models considers the random effects as latent variables with the outcome at each time point an indicator for the latent variable. I have details elsewhere, but I want to explore this as it is a commonly used technique in the social sciences, especially psychology. Latent Growth Curve Models are a special case of structural equation modeling, a highly flexible tool that can incorporate latent variables, indirect effects, multiple outcomes etc. Growth curve models are actually somewhat irregular SEM in the way that they are specified, but for our purposes, we only want to see how the approach works and compare it to previous methods.

The first thing is that the data has to be in *wide* format, such that we have one column per time point, and thus only one row per individual. Once the data is ready we specify the model syntax. By default, the SEM approach also assumes unequal variances across time, so to make it more comparable, we fix that value to be constant. We'll use lavaan to estimate the model.

```
dwide = spread(d, key=time, value=y, sep='_') %>%
  mutate(treatment = treatment=='treatment')  # otherwise converted to numeric directly as 1-2 instead
head(dwide)
```

```
  treatment id     time_0    time_1    time_2     time_3
1     FALSE  1  0.1760974 0.6928733 0.1920017 -0.2205356
2     FALSE  2  1.4265781 1.6607311 1.2515623  2.9685755
3     FALSE  3 -0.1383776 0.8689109 1.2446484  2.9954976
4     FALSE  4  2.0575643 1.7831405 1.6413706  1.7078853
5     FALSE  5  0.9899045 1.7892770 1.7883325  2.4579697
6     FALSE  6  0.6079163 0.1161267 0.6956824  0.8326176
```

```
growthmod_syntax = "
# model for the intercept and slope latent variables
  int   =~ 1*time_0 + 1*time_1 + 1*time_2 + 1*time_3
  slope =~ 0*time_0 + 1*time_1 + 2*time_2 + 3*time_3

# cluster-level effect
  int ~ treatment

# intercept-slope correlation
  int ~~ slope

# fix to equal variances (parameter 'res')
  time_0 ~~ res*time_0
  time_1 ~~ res*time_1
  time_2 ~~ res*time_2
```

```
  time_3 ~~ res*time_3
"

library(lavaan)
growth_mod = growth(growthmod_syntax, data=dwide)
summary(growth_mod, standardized=T)
```

```
lavaan (0.5-22) converged normally after  27 iterations

  Number of observations                          2500

  Estimator                                         ML
  Minimum Function Test Statistic              234.733
  Degrees of freedom                                11
  P-value (Chi-square)                           0.000

Parameter Estimates:

  Information                                 Expected
  Standard Errors                             Standard

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  int =~
    time_0            1.000                               1.048    0.896
    time_1            1.000                               1.048    0.931
    time_2            1.000                               1.048    0.863
    time_3            1.000                               1.048    0.742
  slope =~
    time_0            0.000                               0.000    0.000
    time_1            1.000                               0.394    0.350
    time_2            2.000                               0.788    0.648
    time_3            3.000                               1.182    0.836

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  int ~
    treatment        -0.434    0.041  -10.686    0.000   -0.414   -0.207

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
 .int ~~
    slope            -0.128    0.011  -11.522    0.000   -0.317   -0.317

Intercepts:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
   .time_0            0.000                               0.000    0.000
   .time_1            0.000                               0.000    0.000
   .time_2            0.000                               0.000    0.000
   .time_3            0.000                               0.000    0.000
   .int               0.137    0.030    4.553    0.000    0.131    0.131
    slope             0.509    0.009   55.635    0.000    1.291    1.291

Variances:
```

```
                Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
.time_0   (res)    0.269    0.005   50.000    0.000    0.269    0.197
.time_1   (res)    0.269    0.005   50.000    0.000    0.269    0.212
.time_2   (res)    0.269    0.005   50.000    0.000    0.269    0.182
.time_3   (res)    0.269    0.005   50.000    0.000    0.269    0.135
.int               1.052    0.035   29.817    0.000    0.957    0.957
 slope             0.155    0.006   25.828    0.000    1.000    1.000
```

The `Intercepts:` section of the output shows what would be the fixed effects in the mixed model, and in this case, they are in fact 'intercepts' in this latent variable approach, so that is why they are named as such. The Regression of `int` on treatment depicts the treatment effect, and will make more sense to those who come to mixed models from the multilevel modeling literature. If you go back to the model depiction for the mixed model, this model more explicitly denotes $\beta_{0c} = \beta_0 + \beta_2 * \text{Treatment} + \gamma_c$. The `res` parameter is the arbitray name I've given for the residual variance, and is roughly equivalent to the square of the residual standard deviation in the mixed model output. The above model does not allow for correlated residuals, though this is possible[1].

The primary point here is not to precisely reproduce the correct model but to show the identity between the mixed model and the latent growth curve approach. Proper specification will lead to identical results between latent growth curve and mixed models. The following creates a mixed model that is the equivalent.

```
mixed_mod_nocorr = lme(y ~ time + treatment, data=d, random=~1+time|id, method="ML")
```

Table 11.1: Fixed effects: y ~ time + treatment

	Value	Std.Error	DF	t-value	p-value
(Intercept)	0.137	0.03	7499	4.552	0
time	0.509	0.009	7499	55.63	0
treatmenttreatment	-0.434	0.041	2498	-10.69	0

Table 11.2: Standardized Within-Group Residuals

Min	Q1	Med	Q3	Max
-3.578	-0.5069	0.005247	0.5049	3.276

Table 11.3: Linear mixed-effects model fit by maximum likelihood
: y ~ time + treatment

	Observations	Groups	Log-restricted-likelihood
id	10000	2500	-12732

Variance	StdDev
1.052	1.026
0.155	0.394
0.269	0.519

[1]See my LGC chapter in this SEM document. Once you see it there you'll know why I did not do so here.

11.1 Pros

- Can be utilized on less data than typical SEM
- Very efficient estimation
- Can deal with very complex models, including mediation, parallel processes etc.

11.2 Cons

- Tedious to specify even the simplest of models
- Very tedious to specify even common extensions (e.g. time-varying covariates)
- Even worse to get into correlated residuals
- More complex cluster structure is not dealt with well (if at all)[2]
- Assumes balanced time points
- Doesn't deal with many time points well (if also time-varying covariates especially)

Gist: Growth curve models are very flexible, but they are also problematic simply because they are from the SEM world, which is one where models are notoriously misapplied. Furthermore, there are no common uses of growth curve models that would not be more easily implemented in one of several R packages[3] and various other languages and statistical programs. While I find the latent variable interpretation very much intriguing, the latent variable approach is not something I'd normally consider for this setting.

[2]MPlus has recently incorporated the ability to handle crossed random effects (and see example 9.24 in the version 7 manual), but I have no idea how they work in realistic situations with potentially many, possibly time-varying, covariates, and it's actually done with their multilevel approach rather than the LGC we've been discussing. Furthermore, it requires the Bayesian estimator, which, if you're going that route you might as well use rstan, rjags or similar and have a lot more utility (and clarity) at your disposal. For tools like lme4 and similar, incorporating crossed random effects are no more difficult than other situations, i.e. are 1 line of code, while you'd be debugging the MPlus output for days.

[3]See the mediation package for mediation with mixed models, flexMix for growth mixture models, Bayesian approaches for parallel processes etc.

Chapter 12

Summary

Let's revisit the results all at once. Seeing the output side-by-side may prove helpful in comparing the different approaches.

12.1 Fixed Effects

We'll start with the 'fixed effects' or 'population average' estimates. It may be worth sorting by `Effect`[1]. I also include an random intercept only model for a more direct comparison between the RE and FE models.

The main thing to note is that we would come to no grand differences in substantive conclusions, except for the Grouped and FE approach, where we can come to no conclusion about the treatment effect. Even statistically, the conclusions would be the same, so what can one conclude about these main effects?

For one, we typically don't have 10000 observations for our models, and so the differences would be more notable in that case. I invite you to rerun the simulation with a smaller sample size where n = 100 individuals instead of 2500, as well as with smaller effects (these are large), and see what you come up with. In general, with a lot of data you shouldn't come to wildly different conclusions with different techniques[2].

While I have yet to come across a client that actually cares what the precise value of the standard error is, many care about statistical significance. If that is of primary concern, and it shouldn't be, then one would prefer techniques that get better estimates of the standard error[3]. In that sense, a cluster robust approach would help, but would be better if applied in the FE/RE/GEE model settings.

12.2 Variance estimates

Here we'll compare variance estimates. I add mixed and gee models with no residual correlation estimate to make more direct comparisons to the growth curve model, as well as an random intercept only with no dependency structure assumed (other than independence) for comparison to the FE model.

Cluster robust modifications alone will change nothing compared to a SLiM except for the standard errors of the main effects. A GEE approach with an independence structure and constant residual variance is

[1] For the grouped SE, I just calculated as one would the mean `sd/sqrt(n)`. I thought about using Rubin's rules as one would for the missing value situation, but each four observation linear model has very high variance. This approach puts it in the ball park of the other estimates.

[2] "More data beats a cleverer algorithm." ~ Pedro Domingos

[3] Note that when there are multiple sources of variance, it is difficult to know what the standard error should be. Programs like Stata and SAS make the decision for you, and provide approximate p-values (via Kenward-Roger or other approximation), while the R package lme4 will not provide p-values. One can get decent interval estimates (e.g. via bootstrap or MCMC), but see the R wiki reference for some details on this issue.

equivalent to the SLiM. The FE residual variance is equivalent to the SLiM if `id` had been included in the model, and is equivalent to that estimated by an RE intercept only model with no residual dependency estimated. The estimates of RE model with random intercepts and slopes but no correlation structure assumed are identical to the growth curve estimates.

12.3 Conclusion

This is only a single and contrived data example that is based on random effects as part of the underlying data generating process. If there is no clustering effect, then the standard linear model would obviously the way to go, but that is also not the situation we're interested in. If there is no residual correlation, or only cluster-specific effects akin to an 'intercept-only' model, little changes from what has been noted above, aside from latent growth curve models being less of an option, and they wouldn't be in a non-longitudinal setting anyway. In general, the same issues noted above would still be in play for the most part for simpler settings.

Specifically, I think it's safe to say that the standard linear model approach has drawbacks and is simply not necessary given how easy it is to conduct more appropriate methods. Using cluster robust standard errors may help, but ignores what could potentially be a very rich investigation of cluster specific effects, and may be best used as a diagnostic or signal for a misspecified model. I also think it's difficult to justify the FE approach where we can't even investigate cluster level covariates, but again, that is my bias, and may not be shared by others. Latent growth curve models will probably only be useful if one is dealing with very complex SEM, and which in that case, one will have a host of other issues to contend with that aren't applicable in this setting. Otherwise, LGCM will be identical to mixed models, and even some more complex than depicted, and may only make for more tedious coding with little else gained.

That leaves the mixed model and GEE GLM approaches. Between these two, we don't even have to choose if it's a random intercept, linear model, as they would have similar 'population-average' interpretations, and similar estimates of like parameters, though the mixed model provides cluster-specific effects and predictions. Interpretations do change with nonlinear models such as with a binary outcome and logit link, but see the Gardiner reference below for how the estimated coefficients relate to one another in those cases.

Only the mixed model provides cluster-specific effects and predictions, while allowing for complex cluster structures and retaining cluster level covariates. What's more, they can be specified to overcome the primary motivation for preferring FE models. In addition, they readily extend to other types of 'random effects' models (e.g. spatial).

In conclusion then, the random effects/mixed model approach can provide most everything one could want in dealing with a variety of clustered data situations. It is highly flexible, and a very powerful tool to have at one's disposal.

These references are a mix of starting points, interesting notes, and more authoritative sources.

12.4 Robust SE

- King & Roberts. (2015). How Robust Standard Errors Expose Methodological Problems They Do Not Fix. *Political Analysis*.
- Trivedi, C. (2011). Robust Inference with Clustered Data.

12.5 Fixed effects and 'panel' data models

- Wooldridge. (2016). Introductory Econometrics: A Modern Approach (6e). link
- Wooldridge. (2010). Econometric Analysis of Cross Section and Panel Data (2e). link
- Baltalgi. (2005). Econometric Analysis of Panel Data (3e). link

12.6 Mixed models

- My overview.
- Gelman & Hill. (2006). Data Analysis Using Regression and Multilevel/Hierarchical Models. link
- Pinheiro & Bates. (2000). Mixed-Effects Models in S and S-PLUS. link
- Fahrmeier et al. (2013). Regression. link
- West et al. (2014). Linear Mixed Models: A Practical Guide Using Statistical Software (2e). link

12.7 GEE

To be honest I can't speak to this reference from experience, though I've read and would recommend Hardin and Hilbe's GLM book, and this is a similar approach (applied with Stata examples).

- Hardin & Hilbe (2013). Generalized Estimating Equations. link

12.8 Growth Curve Models

- Kline. (2015). Principles and Practice of Structural Equation Modeling. link
- My SEM notes

12.9 Comparison/Issues

- Gardiner et al. (2009). Fixed effects, random effects, and GEE: What are the differences? *Statistics in Medicine*. link
- Bell & Jones. (2015) Explaining Fixed Effects: Random Effects Modeling of Time-Series Cross-Sectional and Panel Data. link
- Bell & Jones. (2016) Fixed and Random effects: making an informed choice.link
- R mixed list FAQ. Old but still has useful information. link

Bibliography