Fitting a Multilevel Model

This analysis will be focusing on a longitudinal study that was conducted on children with autism[1]. We will be looking at several variables and exploring how different factors interact with the socialization of a child with autism as they progress throughout the beginning stages of their life.

The variables we have from the study are:

- AGE is the age of a child which, for this dataset, is between two and thirteen years
- VSAE measures a child's socialization
- SICDEGP is the expressive language group at age two and can take on values ranging from one to three. Higher values indicate more expressive language.
- CHILDID is the unique ID that is given to each child and acts as their identifier within the dataset

We will first be fitting a multilevel model with explicit random effects of the children to account for the fact that we have repeated measurements on each child, which introduces correlation in our observations.

[1] Anderson, D., Oti, R., Lord, C., and Welch, K. (2009). Patterns of growth in adaptive social abilities among children with autism spectrum disorders. Journal of Abnormal Child Psychology, 37(7), 1019-1034.

###Importing Data and Packages Before we begin, we need to include a few packages that will make working with the data a little easier.

In [1]:
```
# Upgrade to statsmodels 0.9.0
#!pip install --upgrade --user statsmodels

# Import the libraries that we will need for the analysis
import csv
import numpy as np
import pandas as pd
import statsmodels.api as sm
from sklearn import linear_model
import matplotlib.pyplot as plt
import patsy
from scipy.stats import chi2 # for sig testing
from IPython.display import display, HTML # for pretty printing

# Read in the Autism Data
dat = pd.read_csv("autism.csv")

# Drop NA's from the data
dat = dat.dropna()
```

In [2]:

```
# Print out the first few rows of the data
dat.head()
```

Out[2]:

	age	vsae	sicdegp	childid
0	2	6.0	3	1
1	3	7.0	3	1
2	5	18.0	3	1
3	9	25.0	3	1
4	13	27.0	3	1

###Fit the Model without Centering We will first begin by fitting the model without centering the age component first. This model has both random intercepts and random slopes on age.

In [*]:

```
# Build the model
mlm_mod = sm.MixedLM.from_formula(
    formula = 'vsae ~ age * C(sicdegp)',
    groups = 'childid',
    re_formula="1 + age",
    data=dat
)

# Run the fit
mlm_result = mlm_mod.fit()

# Print out the summary of the fit
mlm_result.summary()
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/base/model.py:508: Con
vergenceWarning: Maximum Likelihood optimization failed to converge. Chec
k mle_retvals
  "Check mle_retvals", ConvergenceWarning)
```

We can see that the model fails to converge. Taking a step back, and thinking about the data, how should we expect children's socialization to vary at age zero? Would we expect the children to exhibit different socialization when they are first born? Or is the difference in socialization something that we would expect to manifest over time?

We would expect the socialization differences should be negligible at age zero or, at the very least, difficult to discern. This homogeneity of newborns implies the variance of the random intercept would be close to zero, and, as a result, the model is having difficulty estimating the variance parameter of the random intercept. It may not make sense to include a random intercept in this model. We will drop the random intercept and attempt to refit the model to see if the convergence warnings still manifest themselves in the fit.

In [*]:

```
# Build the model - note the re_formula definition now
# has a 0 instead of a 1. This removes the intercept from
# the model
mlm_mod = sm.MixedLM.from_formula(
    formula = 'vsae ~ age * C(sicdegp)',
    groups = 'childid',
    re_formula="0 + age",
    data=dat
)

# Run the fit
mlm_result = mlm_mod.fit()

# Print out the summary of the fit
mlm_result.summary()
```

The model now converges, which is an indication that removing the random intercepts from the model was beneficial computationally.

First, we notice that the interaction term between the expressive language group and the age of children is positive and significant for the third expressive language group. This is an indication that the increase in socialization as a function of age for this group is significantly larger relative to the first expressive language group (i.e., the age slope is significantly larger for this group relative to the first expressive language group).

When we think about the interpretation of the parameters, however, we need to be cautious. The intercept can be interpreted as the mean socialization when a child in the first expressive language group is zero years old. This may not be sensible to estimate. To improve this interpretation, we should center the age variable and, again, fit the model.

In [*]:

```
# Center the age variable
dat["age"] = dat.groupby("childid")["age"].transform(lambda x: x - x.mean())

# Print out the head of the dataset to see the centered measure
dat.head()
```

In [*]:

```
# Refit the model, again, without the random intercepts
mlm_mod = sm.MixedLM.from_formula(
    formula = 'vsae ~ age * C(sicdegp)',
    groups = 'childid',
    re_formula="0 + age",
    data=dat
)

# Run the fit
mlm_result = mlm_mod.fit()

# Print out the summary of the fit
mlm_result.summary()
```

Now, our intercept of represents the mean socialization of the children at the mean age for their measurements. For most children, this measures the socialization around around 6.5 years of age.

Significance Testing

The next question that we need to ask is if the addition of the random age effects is actually significant; should we retain these random effects in the model? First, we will fit the multilevel model including centered age again. This time, however, we will compare it to the model that does not have random effects:

In [*]:

```python
# Random Effects Mixed Model
mlm_mod = sm.MixedLM.from_formula(
    formula = 'vsae ~ age * C(sicdegp)',
    groups = 'childid',
    re_formula="0 + age",
    data=dat
)

# OLS model - no mixed effects
ols_mod = sm.OLS.from_formula(
    formula = "vsae ~ age * C(sicdegp)",
    data = dat
)

# Run each of the fits
mlm_result = mlm_mod.fit()
ols_result = ols_mod.fit()

# Print out the summary of the fit
print(mlm_result.summary())
print(ols_result.summary())
```

Now, we perform the significance test with a mixture of chi-squared distributions. We repeat the information from the Likelihood Ratio Tests writeup for this week here:

- Null hypothesis: The variance of the random child effects on the slope of interest is zero (in other words, these random effects on the slope are not needed in the model)
- Alternative hypothesis: The variance of the random child effects on the slope of interest is greater than zero

- First, fit the model WITH random child effects on the slope of interest, using restricted maximum likelihood estimation
  - -2 REML log-likelihood = 4854.18
- Next, fit the nested model WITHOUT the random child effects on the slope:
  - -2 REML log-likelihood = 5524.20 (higher value = worse fit!)
- Compute the positive difference in the -2 REML log-likelihood values (“REML criterion”) for the models:
  - Test Statistic (TS) = 5524.20 – 4854.18 = 670.02
- Refer the TS to a mixture of chi-square distributions with 1 and 2 DF, and equal weight 0.5:

In [*]:

```
# Compute the p-value using a mixture of chi-squared distributions
# Because the chi-squared distribution with zero degrees of freedom has no
# mass, we multiply the chi-squared distribution with one degree of freedom
# 0.5
pval = 0.5 * (1 - chi2.cdf(670.02, 1))
print("The p-value of our significance test is: {0}".format(pval))
```

The p-value is so small that we cannot distiguish it from zero. With a p-value this small, we can safely reject the null hypothesis. We have sufficient evidence to conclude that the variance of the random effects on the slope of interest is greater than zero.

Marginal Models

While we have accounted for correlation among observations from the same children using random age effects in the multilevel model, marginal models attempt to manage the correlation in a slightly different manner. This process of fitting a marginal model, utilizing a method known as Generalized Estimating Equations (GEEs), aims to explicitly model the within-child correlations of the observations.

We will specify two types of covariance structures for this analysis. The first will be an exchangeable model. In the exchangeable model, the observations within a child have a constant correlation, and constant variance.

The other covariance structure that we will assume is independence. An independent covariance matrix implies that observations within the same child have zero correlation.

We will see how each of these covariance structures affect the fit of the model.

In [*]:

```
# Fit the exchangable covariance GEE
model_exch = sm.GEE.from_formula(
    formula = "vsae ~ age * C(sicdegp)",
    groups="childid",
    cov_struct=sm.cov_struct.Exchangeable(),
    data=dat
    ).fit()

# Fit the independent covariance GEE
model_indep = sm.GEE.from_formula(
    "vsae ~ age * C(sicdegp)",
    groups="childid",
    cov_struct = sm.cov_struct.Independence(),
    data=dat
    ).fit()

# We cannot fit an autoregressive model, but this is how
# we would fit it if we had equally spaced ages
# model_indep = sm.GEE.from_formula(
#     "vsae ~ age * C(sicdegp)",
#     groups="age",
#     cov_struct = sm.cov_struct.Autoregressive(),
#     data=dat
#     ).fit()
```

The autoregressive model cannot be fit because the age variable is not spaced uniformly for each child's measurements (every year or every two years for each measurement). If it was, we can fit it with the commented code above. We will now see how each of the model fits compare to one another:

In [*]:

```
# Construct a datafame of the parameter estimates and their standard errors
x = pd.DataFrame(
    {
        "OLS_Params": ols_result.params,
        "OLS_SE": ols_result.bse,
        "MLM_Params": mlm_result.params,
        "MLM_SE": mlm_result.bse,
        "GEE_Exch_Params": model_exch.params,
        "GEE_Exch_SE": model_exch.bse,
        "GEE_Indep_Params": model_indep.params,
        "GEE_Indep_SE": model_indep.bse
    }
)

# Ensure the ordering is logical
x = x[["OLS_Params", "OLS_SE","MLM_Params", "MLM_SE","GEE_Exch_Params",
       "GEE_Exch_SE", "GEE_Indep_Params", "GEE_Indep_SE"]]

# Round the results of the estimates to two decimal places
x = np.round(x, 2)
# Print out the results in a pretty way
display(HTML(x.to_html()))
```

We can see that the estimates for the parameters are relatively consistent among each of the modeling methodologies, but the standard errors differ from model to model. Overall, the two GEE models are mostly similar and both exhibit standard errors for parameters that are slightly larger than each of their corresponding values in the OLS model. The multilevel model has the largest standard error for the age coefficient, but the smallest standard error for the intercept. Overall, we see that we would make similar inferences regarding the importance of these fixed effects, but remember that we need to interpret the multilevel models estimates conditioning on a given child. For example, considering the age coefficient in the multilevel model, we would say that as age increases by one year *for a given child* in the first expressive language group, VSAE is expected to increase by 2.73. In the GEE and OLS models, we would say that as age increases by one year *in general*, the average VSAE is expected to increase by 2.60.