Noname manuscript No.
(will be inserted by the editor)

NlsyLinks

An R package for research with the NLSY (National Longitudinal Survey of Youth)

William H Beasley ·

Received: date / Accepted: date

Abstract The text of your abstract. 150 – 250 words.

Keywords key · dictionary · word ·

1 Introduction

1.1 Abstract

1.2 Introduction

- NLSY Structure
- Benefits of Accounting for Kinships
  - BG
  - D'Onofrio-type research
- Terminology

1.2.1 Structural and Topical information

The NlsyLinks package offers two types of datasets: topical and structural. *Topical datasets* contain predictor and outcome variables typically used to test a focused hypotheses. For instance, the NLSY79 Gen2 variables Rqqq.qq and Rqqq.qq are critical when studying the relationship between conduct disorder and menarche (*e.g.*, Rodgers et al, 2015), but are not relevant to many hypotheses outside these fields.

William H Beasley
Howard Live Oak, LLC
E-mail: `wibeasley@hotmail.com`

In contrast, variables in *structural datasets* are not typically directly stated in the hypotheses, yet are essential to many NLSY-related investigations including:

- familial relationships (*e.g.*, Subjects 301 and 302 are half-brothers; Subjects 301 and 403 are first-cousins),
- subject characteristics (*e.g.*, Subject 301 is a Native American female; Subject 607 died from heart disease in 2005; Subject 802 is part of the military over-sample), and
- subject-survey characteristics (*e.g.*, Subject 301 was 15 years old for the 1981 Survey; Subject 301 did not respond to the 1990 survey; Subject 702 completed the NLSY-C survey in 1996 and the NLSY-YA survey in 1998).

The NlsyLinks includes small topical datasets which allows the vignettes and examples to be reproducible and more realistic. The structural datasets are intended to be the authoritative representations, and are the product of two NIH grants (for a complete history of the familial relationships, see Rodgers et al., 2016).

1.2.2 Terminology

The package pertains to multiple generations of the 'Nlsy79' and multiple generations of the 'Nlsy97'. Because the NlsyLinks package structures information within and between generations of the NLSY simultaneously, it requires slightly unconventional NLSY terminology to reduce ambiguity.

The 'Nlsy79 sample' refers to both the original 12,686 subjects interviewed in 1979, and their 11,500+ children (termed 'Nlsy79 Gen1' and 'Nlsy79 Gen2', respectively). Data for the 'Nlsy79 Gen1' comes from the original NLSY79 study, while data for the 'Nlsy Gen2' comes from both the NLSY-C study and the NLSY-YA study ('C' stands for children, and 'YA' stands for young adult). More specifically, the Gen2 subjects are the biological offspring of the Gen1 mothers; they initially completed the NLSY-C survey until roughly age 14, and then completed the NLSY-YA survey (the oldest 'Young Adult' respondent was QQ in the 2016 survey). Although the NLSY does not interview 'Nlsy79 Gen0' (the parents of Gen1) or 'Nlsy79 Gen3' (the children of Gen2), it does contain direct and indirect information about them.

The terminology for the 'Nlsy97' sample is similar yet simpler than the 'Nlsy79', because the explicit respondents come from a single generation (*i.e.*, the 'Nlsy97 Gen1'). A few variables reflect Gen0 and Gen2. In contrast to the Nlsy79, the Nlsy97 contains more information about the housemates.

They both are nationally-representative samples.

Common NLSY79 terminology refers second generation subjects as 'children' when they are younger than age 15 (NLSYC), and as 'young adults' when they are 15 and older (NLSY79-YA); though they are the same respondents, different funding mechanisms and different survey items necessitate the distinction. This cohort is sometimes abbreviated as 'NLSY79-C', 'NLSY79C', 'NLSY-C' or 'NLSYC'. This packages uses 'Gen2' to refer to subjects of this generation, regardless of their age at the time of the survey.

Ambiguous Categories: In some cases, the kinship of relatives can be safely narrowed to two categories, but not one. The most common scenario involves the *ambiguous siblings* (among Nlsy Gen2 siblings); due to the sampling design, we know that the two participants share the same mother, so they are either full siblings (R=.50) or half siblings (R=.25). For the qqq% that we could not satisfactorily classify, we categorize them as "ambiguous siblings" and assign them R=.375. Another common scenario involves the *ambiguous twins* (among Nlsy97 and both generations of Nlsy79). For qqq% of the twins we comfortably classify their relationship as either MZ or DZ; the remaining qqq% percent are classified as "ambiguous twins" and assigned R=0.75. Empirically, this approach has been successful in the previous 20 years of our previous research, and is discussed further in (Rodgers, qqq).

Within our own team, we've mostly stopped using terms like 'NLSY79', 'NLSY79-C' and 'NLSY79-YA', because we conceptualize it as one big sample containing two related generations. It many senses, the responses collected from the second generation can be viewed as outcomes of the first generation. Likewise, the parents in the first generation provide many responses that can be viewed as explanatory variables for the 2nd generation. Depending on your research, there can be big advantages of using one cohort to augment the other. There are also survey items that provide information about the 3rd generation and the 0th generation.

The `SubjectTag` variable uniquely identify NLSY79 subjects when a dataset contains both generations. For Gen2 subjects, the `SubjectTag` is identical to their CID (*i.e.*, C00001.00 -the ID assigned in the NLSY79-Children files). However for Gen1 subjects, the `SubjectTag` is their CaseID (*i.e.*, R00001.00), with "00" appended. This manipulation is necessary to identify subjects uniquely in inter-generational datasets. A Gen1 subject with an ID of 43 becomes 4300. The `SubjectTag`s of her four children remain 4301, 4302, 4303, and 4304.

The *expected coefficient of relatedness* of a pair of subjects is typically represented by the statistical variable `R`. Examples are: Monozygotic twins have `R`=1; dizygotic twins have `R`=0.5; full siblings (*i.e.*, those who share both biological parents) have `R`=0.5; half-siblings (*i.e.*, those who share exactly one biological parent) have `R`=0.25; adopted siblings have `R`=0.0. Other uncommon possibilities are mentioned the documentation for `Links79Pair`. The font (and hopefully their context) should distinguish the variable `R` from the software R. To make things slightly more confusing the computer variable for `R` in the `Links79Pair` dataset is written with a monospace font: `R`.

A subject's `ExtendedID` indicates their extended family. Two subjects will be in the same extended family if either: [1] they are Gen1 housemates, [2] they are Gen2 siblings, [3] they are Gen2 cousins (*i.e.*, they have mothers who are Gen1 sisters in the NLSY79), [4] they are mother and child (in Gen1 and Gen2, respectively), or [5] they are (aunt|uncle) and (niece|nephew) (in Gen1 and Gen2, respectively).

An **outcome variable** is directly relevant to the applied researcher; these might represent constructs like height, IQ, and income. A **plumbing variable** is necessary to manage BG datasets; examples are `R`, a subject's ID, and the date of a subject's last survey.

An **ACE model** is the basic biometrical model used by Behavior Genetic researchers, where the genetic and environmental effects are assumed to be additive. The three primary variance components are (1) the proportion of variability due to a shared genetic influence (typically represented as a^2, or sometimes h^2), (2) the proportion of variability due to shared common environmental influence (typically c^2), and (3) the proportion of variability due to unexplained/residual/error influence (typically e^2).

The variables are scaled so that they account for all observed variability in the outcome variable; specifically: $a^2 + c^2 + e^2 = 1$. Using appropriate designs that can logically distinguish these different components (under carefully specified assumptions), the basic biometrical modeling strategy is to estimate the magnitude of a^2, c^2, and e^2 within the context of a particular model. For gentle introductions to Behavior Genetic research, we recommend Plomin (1990) and Carey (2003). For more in-depth ACE model-fitting strategies, we recommend Neale & Maes, (1992).

1.3 Retrieving Data with the NLS Investigator

– This will use much of the existing NLS Investigator vignette.

When a researcher pursues a new idea, we suggest to start by exploring what the NLSY can offer by poking around the (a) vast online documentation and (b) NLS Investigator. The documentation online (www.qqq), and has general information (*e.g.*, how to connect the nationally representative sample was collected), topical information (*e.g.*, what medical and health information has been collected across survey waves and subject ages), and descriptive summaries (*e.g.*, attrition over time for different race and ethnic groups). This material has helpful suggestions which variables are available and appropriate.

With these hints, it's time to identify and download the specific variables from the NLS Investigator. The NLS Investigator is described briefly here (see the NLS Investigator vignette for more detailed instruction). Researchers new to the NLSY should expect at least a dozen round trips as they iteratively improve and complete their set of variables. First, select the 'Study', such as 'NLSY79 Child & Young Adult' (which corresponds to 'Nlsy79 Gen2' in our terminology. Second, select your desired variables, out of the tens of thousands available ones.

Before starting the examples, first verify that the NlsyLinks package is installed correctly. If not, please refer to Appendix.

```r
any(.packages(all.available = TRUE) == "NlsyLinks") #Should evaluate to TRUE.
```

```
## [1] TRUE
```

```
library(NlsyLinks) #Load the package into the current session.
```

1.4 ACE DF Analysis of One Generation (NLSY79 Gen1)

{This very basic analysis that should provide an initial feel for the inputs, mechanics, and goals of the analysis.}

The first example uses a simple statistical model and all available Gen2 subjects. The `CreatePairLinksDoubleEntered` function will create a data frame where each represents one pair of siblings, respective of order (*i.e.*, there is a row for Subjects 201 and 202, and a second row for Subjects 202 and 201). This function examines the subjects' IDs and determines who is related to whom (and by how much). By default, each row it produces has at least six values/columns: (i) ID for the older member of the kinship pair: `Subject1Tag`, (ii) ID for the younger member: `Subject2Tag`, (iii) ID for their extended family: `ExtendedID`, (iv) their estimated coefficient of genetic relatedness: `R`, (v *and beyond*) outcome values for the older member; (vi *and beyond*) outcome values for the younger member.

A DeFries-Fulker (**DF**) Analysis uses linear regression to estimate the a^2, c^2, and e^2 of a univariate biometric system. The interpretations of the DF analysis can be found in Rodgers & Kohler (2005) and Rodgers, Rowe, & Li (1999). This example uses the newest variation, which estimates two parameters; the corresponding function is called `DeFriesFulkerMethod3`. The steps are:

1. Use the NLS Investigator to select and download a Gen2 dataset.
2. Open R and create a new script (see Appendix: R Scripts and load the NlsyLinks package. If you haven't done so, install the NlsyLinks package). Within the R script, identify the locations of the downloaded data file, and load it into a data frame.
3. Within the R script, load the linking dataset. Then select only Gen2 subjects. The 'Pair' version of the linking dataset is essentially an upper triangle of a symmetric sparse matrix.
4. Load and assign the `ExtraOutcomes79` dataset.
5. Specify the outcome variable name and filter out all subjects who have a negative value in this variable. The NLSY typically uses negative values to indicate different types of missingness (see 'Further Information' below).
6. Create a double-entered file by calling the 'CreatePairLinksDoubleEntered' function. At minimum, pass the (i) outcome dataset, the (ii) linking dataset, and the (iii) name(s) of the outcome variable(s). *(There are occasions when a single-entered file is more appropriate for a DF analysis. See Rodgers & Kohler, 2005, for additional information.)*
7. Use 'DeFriesFulkerMethod3' function (*i.e.*, general linear model) to estimate the coefficients of the DF model.

```
### R Code for Example DF analysis with a simple outcome and Gen2 subjects
#Step 2: Load the package containing the linking routines.
library(NlsyLinks)

#Step 3: Load the LINKING dataset and filter for the Gen2 subjects
dsLinking <- subset(Links79Pair, RelationshipPath=="Gen2Siblings")
summary(dsLinking) #Notice there are 11,088 records (one for each unique pair).
```

```
##    ExtendedID    SubjectTag_S1     SubjectTag_S2           R
##  Min.   :    2   Min.   :    201   Min.   :    202   Min.   :0.250
##  1st Qu.: 3154   1st Qu.: 315401   1st Qu.: 315403   1st Qu.:0.250
##  Median : 6105   Median : 610703   Median : 610705   Median :0.500
##  Mean   : 5927   Mean   : 593061   Mean   : 593063   Mean   :0.417
##  3rd Qu.: 8507   3rd Qu.: 851001   3rd Qu.: 851003   3rd Qu.:0.500
##  Max.   :12673   Max.   :1267301   Max.   :1267302   Max.   :1.000
##         RelationshipPath
##  Gen1Housemates:    0
##  Gen2Siblings  :11114
##  Gen2Cousins   :    0
##  ParentChild   :    0
##  AuntNiece     :    0
##
```

```
#Step 4: Load the OUTCOMES dataset, and then examine the summary.
dsOutcomes <- ExtraOutcomes79 #'ds' stands for 'Data Set'
summary(dsOutcomes)
```

```
##    SubjectTag        SubjectID        Generation    HeightZGenderAge
##  Min.   :    100   Min.   :      1   Min.   :1.000   Min.   :-2.985
##  1st Qu.: 314025   1st Qu.:   5998   1st Qu.:1.000   1st Qu.:-0.724
##  Median : 620050   Median :  12000   Median :1.000   Median :-0.045
##  Mean   : 618600   Mean   : 289254   Mean   :1.476   Mean   :-0.006
##  3rd Qu.: 914501   3rd Qu.: 577403   3rd Qu.:2.000   3rd Qu.: 0.648
##  Max.   :1268600   Max.   :1267501   Max.   :2.000   Max.   : 2.996
##                                                      NA's   :4711
##  WeightZGenderAge AfqtRescaled2006Gaussified      Afi             Afm
##  Min.   :-2.985   Min.   :-2.895             Min.   : 2.00   Min.   : 0.00
##  1st Qu.:-0.677   1st Qu.:-0.692             1st Qu.:15.00   1st Qu.:12.00
##  Median :-0.149   Median :-0.024             Median :17.00   Median :13.00
##  Mean   : 0.001   Mean   :-0.011             Mean   :16.66   Mean   :12.78
##  3rd Qu.: 0.533   3rd Qu.: 0.660             3rd Qu.:18.00   3rd Qu.:14.00
##  Max.   : 4.945   Max.   : 2.994             Max.   :27.00   Max.   :19.00
##  NA's   :4719     NA's   :12510              NA's   :12740   NA's   :18165
##  MathStandardized
##  Min.   : 65.0
##  1st Qu.: 92.5
##  Median :100.0
##  Mean   :100.1
##  3rd Qu.:108.5
##  Max.   :135.0
##  NA’s   :15085
```

```
#Step 5: This step isn’t necessary for this example, because Kelly Meredith already
#   groomed the values.  If the negative values (which represent NLSY missing or
#   skip patterns) still exist, then:
dsOutcomes$MathStandardized[dsOutcomes$MathStandardized < 0] <- NA

#Step 6: Create the double entered dataset.
dsDouble <- CreatePairLinksDoubleEntered(
  outcomeDataset   = dsOutcomes,
  linksPairDataset = dsLinking,
  outcomeNames     = c(’MathStandardized’)
)
summary(dsDouble) #Notice there are 22176=(2*11088) records now (two for each unique pair).
```

```
##  SubjectTag_S1     SubjectTag_S2       ExtendedID          R
##  Min.   :     201   Min.   :     201   Min.   :    2   Min.   :0.250
##  1st Qu.: 315402   1st Qu.: 315402   1st Qu.: 3154   1st Qu.:0.250
##  Median : 610704   Median : 610704   Median : 6105   Median :0.500
##  Mean   : 593062   Mean   : 593062   Mean   : 5927   Mean   :0.417
##  3rd Qu.: 851002   3rd Qu.: 851002   3rd Qu.: 8508   3rd Qu.:0.500
##  Max.   :1267302   Max.   :1267302   Max.   :12673   Max.   :1.000
##
##        RelationshipPath MathStandardized_S1 MathStandardized_S2
##  Gen1Housemates:    0   Min.   : 65.0       Min.   : 65.0
##  Gen2Siblings  :22228   1st Qu.: 90.0       1st Qu.: 90.0
##  Gen2Cousins   :    0   Median : 98.5       Median : 98.5
##  ParentChild   :    0   Mean   : 98.4       Mean   : 98.4
##  AuntNiece     :    0   3rd Qu.:107.0       3rd Qu.:107.0
##                         Max.   :135.0       Max.   :135.0
##                         NA’s   :3926        NA’s   :3926
```

```
#Step 7: Estimate the ACE components with a DF Analysis
ace <- DeFriesFulkerMethod3(
    dataSet   = dsDouble,
    oName_S1 = "MathStandardized_S1",
    oName_S2 = "MathStandardized_S2")
ace
```

```
## [1] "Results of ACE estimation: [show]"
##     ASquared     CSquared     ESquared    CaseCount
## 7.734447e-01 1.469204e-01 7.963486e-02 1.668000e+04
```

Further Information: If the different reasons of missingness are important, further work is necessary. For instance, some analyses that use item `Y19940000` might need to distinguish a response of "Don't Know" (which is coded as -2) from "Missing" (which is coded as -7). For this example, we'll assume it's safe to clump the responses together.

1.5 ACE DF Analysis of One Generation (NLSY79 Gen2)

The second example differs from the previous example in two ways. First, the outcome variables are read from a CSV (comma separated values file) that was downloaded from the **NLS Investigator**. Second, the DF analysis is called through the function `AceUnivariate`; this function is a wrapper around some simple ACE methods, and will help us smoothly transition to more techniques later.

The steps are:

1. Use the NLS Investigator to select and download a Gen2 dataset. Select the variables 'length of gestation of child in weeks' (`C03280.00`), 'weight of child at birth in ounces' (`C03286.00`), and 'length of child at birth' (`C03288.00`), and then download the *.zip file to your local computer.
2. Open R and create a new script and load the NlsyLinks package.
3. Within the R script, load the linking dataset. Then select only Gen2 subjects.
4. Read the CSV into R as a `data.frame` using `ReadCsvNlsy79Gen2`.
5. Verify the desired outcome column exists, and rename it something meaningful to your project. It is important that the `data.frame` is reassigned (*i.e.*, `ds <- RenameNlsyColumn(...)`). In this example, we rename column `C0328800` to `BirthWeightInOunces`.
6. Filter out all subjects who have a negative `BirthWeightInOunces` value. See the 'Further Information' note in the previous example.
7. Create a double-entered file by calling the `CreatePairLinksDoubleEntered` function. At minimum, pass the (i) outcome dataset, the (ii) linking dataset, and the (iii) name(s) of the outcome variable(s).
8. Call the `AceUnivariate` function to estimate the coefficients.

```r
### R Code for Example of a DF analysis with a simple outcome and Gen2 subjects
#Step 2: Load the package containing the linking routines.
library(NlsyLinks)

#Step 3: Load the linking dataset and filter for the Gen2 subjects
dsLinking <- subset(Links79Pair, RelationshipPath=="Gen2Siblings")

#Step 4: Load the outcomes dataset from the hard drive and then examine the summary.
#   Your path might be: filePathOutcomes <- 'C:/BGResearch/NlsExtracts/gen2-birth.csv'
filePathOutcomes <- file.path(path.package("NlsyLinks"), "extdata", "gen2-birth.csv")
```

```r
dsOutcomes <- ReadCsvNlsy79Gen2(filePathOutcomes)
summary(dsOutcomes)
```

```
##    SubjectTag        SubjectID         ExtendedID      Generation
##  Min.   :    201   Min.   :    201   Min.   :    2   Min.   :2
##  1st Qu.: 310302   1st Qu.: 310302   1st Qu.: 3101   1st Qu.:2
##  Median : 604607   Median : 604607   Median : 6045   Median :2
##  Mean   : 601313   Mean   : 601313   Mean   : 6007   Mean   :2
##  3rd Qu.: 876203   3rd Qu.: 876203   3rd Qu.: 8757   3rd Qu.:2
##  Max.   :1267501   Max.   :1267501   Max.   :12675   Max.   :2
##                                      NA's   :2
##  SubjectTagOfMother    C0005300        C0005400          C0005700
##  Min.   :    200    Min.   :1.000   Min.   :-3.000   Min.   :  -3
##  1st Qu.: 310300    1st Qu.:2.000   1st Qu.: 1.000   1st Qu.:1981
##  Median : 604600    Median :3.000   Median : 1.000   Median :1985
##  Mean   : 601311    Mean   :2.338   Mean   : 1.489   Mean   :1986
##  3rd Qu.: 876200    3rd Qu.:3.000   3rd Qu.: 2.000   3rd Qu.:1990
##  Max.   :1267500    Max.   :3.000   Max.   : 2.000   Max.   :2008
##
##     C0328000        C0328600        C0328800
##  Min.   :-7.00   Min.   : -7.0   Min.   :-7.00
##  1st Qu.:37.00   1st Qu.: 99.0   1st Qu.:18.00
##  Median :39.00   Median :115.0   Median :20.00
##  Mean   :33.51   Mean   :103.9   Mean   :16.51
##  3rd Qu.:39.00   3rd Qu.:128.0   3rd Qu.:21.00
##  Max.   :51.00   Max.   :768.0   Max.   :48.00
##
```

```r
#Step 5: Verify and rename an existing column.
VerifyColumnExists(dsOutcomes, "C0328600") #Should return ’10’ in this example.
```

```
## [1] 10
```

```r
dsOutcomes <- RenameNlsyColumn(dsOutcomes, "C0328600", "BirthWeightInOunces")

#Step 6: For this item, a negative value indicates the parent refused, didn’t know,
#   invalidly skipped, or was missing for some other reason.
#   For our present purposes, we’ll treat these responses equivalently.
#   Then clip/Winsorized/truncate the weight to something reasonable.
dsOutcomes$BirthWeightInOunces[dsOutcomes$BirthWeightInOunces < 0] <- NA
dsOutcomes$BirthWeightInOunces <- pmin(dsOutcomes$BirthWeightInOunces, 200)

#Step 7: Create the double entered dataset.
dsDouble <- CreatePairLinksDoubleEntered(
  outcomeDataset    = dsOutcomes,
  linksPairDataset = dsLinking,
  outcomeNames     = c('BirthWeightInOunces')
)

#Step 8: Estimate the ACE components with a DF Analysis
ace <- AceUnivariate(
  method   = "DeFriesFulkerMethod3",
  dataSet  = dsDouble,
  oName_S1 = "BirthWeightInOunces_S1",
  oName_S2 = "BirthWeightInOunces_S2"
)
ace
```

```
## [1] "Results of ACE estimation: [show]"
##     ASquared     CSquared     ESquared    CaseCount
## 5.103108e-01 1.752415e-01 3.144477e-01 1.744000e+04
```

For another example of incorporating CSVs downloaded from the NLS Investigator, please see the “Race and Gender Variables” entry in the FAQ.

1.6 ACE SEM of One Generation (NLSY79 Gen2)

The example differs from the first one by the statistical mechanism used to estimate the components. The first example uses multiple regression to estimate the influence of the shared genetic and environmental factors, while this example uses structural equation modeling (SEM).

The `CreatePairLinksSingleEntered` function will create a `data.frame` where each row represents one unique pair of siblings, *irrespective of order*. Other than producing half the number of rows, this function is identical to `CreatePairLinksDoubleEntered`.

The steps are:

(Steps 1-5 proceed identically to the first example.)

6. Create a *single*-entered file by calling the `CreatePairLinksSingleEntered` function. At minimum, pass the (i) outcome dataset, the (ii) linking dataset, and the (iii) name(s) of the outcome variable(s).
7. Declare the names of the outcome variables corresponding to the two members in each pair. Assuming the variable is called ‘ZZZ’ and the preceding steps have been followed, the variable ‘ZZZ_S1’ corresponds to the first members and ZZZ_S2’ corresponds to the second members.
8. Create a GroupSummary `data.frame`, which identifies the `R` groups that should be considered by the model. Inspect the output to see if the groups show unexpected or fishy differences.
9. Create a `data.frame` with cleaned variables to pass to the SEM function. This `data.frame` contains only the three necessary rows and columns.

10. Estimate the SEM with the lavaan package. The function returns an `S4` object, which shows the basic ACE information.
11. Inspect details of the SEM, beyond the ACE components. In this example, we look at the fit stats and the parameter estimates. The lavaan package has additional methods that may be useful for your purposes.

```
### R Code for Example lavaan estimation analysis with a simple outcome and Gen2 subjects
#Steps 1-5 are explained in the vignette's first example:
library(NlsyLinks)
dsLinking <- subset(Links79Pair, RelationshipPath=="Gen2Siblings")
dsOutcomes <- ExtraOutcomes79
dsOutcomes$MathStandardized[dsOutcomes$MathStandardized < 0] <- NA

#Step 6: Create the single entered dataset.
dsSingle <- CreatePairLinksSingleEntered(
  outcomeDataset   = dsOutcomes,
  linksPairDataset = dsLinking,
  outcomeNames     = c('MathStandardized')
)

#Step 7: Declare the names for the two outcome variables.
oName_S1 <- "MathStandardized_S1" #Stands for Outcome1
oName_S2 <- "MathStandardized_S2" #Stands for Outcome2

#Step 8: Summarize the R groups and determine which groups can be estimated.
dsGroupSummary <- RGroupSummary(dsSingle, oName_S1, oName_S2)
dsGroupSummary
```

```
##       R Included PairCount    O1Mean    O2Mean O1Variance O2Variance
## 1 0.250     TRUE      2689  95.10450  95.97936   126.9489   150.1775
## 2 0.375     TRUE       137  93.63139  93.36861   160.0120   136.6628
## 3 0.500     TRUE      5491  99.89374 100.02868   168.7326   172.7293
## 4 0.750    FALSE         2 108.50000 106.00000   220.5000    18.0000
## 5 1.000     TRUE        21  98.21429  96.02381   289.4393   215.2369
##   O1O2Covariance Correlation Determinant PosDefinite
## 1       41.96914   0.3039577   17303.459        TRUE
## 2       50.39790   0.3408090   19327.735        TRUE
## 3       90.04116   0.5274225   21037.642        TRUE
## 4       63.00000   1.0000000       0.000       FALSE
## 5      229.10714   0.9179130    9807.933        TRUE
```

```
#Step 9: Create a cleaned dataset
dsClean <- CleanSemAceDataset(dsDirty=dsSingle, dsGroupSummary, oName_S1, oName_S2)

#Step 10: Run the model
```

```
ace <- AceLavaanGroup(dsClean)
ace
```

```
## [1] "Results of ACE estimation: [show]"
##     ASquared     CSquared     ESquared    CaseCount
##    0.6219253    0.2097338    0.1683408 8338.0000000
```

```
#Notice the ’CaseCount’ is 8,390 instead of 17,440.
#  This is because (a) one pair with R=.75 was excluded, and
#  (b) the SEM uses a single-entered dataset instead of double-entered.
#
#Step 11: Inspect the output further
library(lavaan) #Load the package to access methods of the lavaan class.
```

```
## This is lavaan 0.6-6

## lavaan is BETA software! Please report any bugs.
```

```
GetDetails(ace)
```

```
## lavaan 0.6-6 ended normally after 54 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of free parameters                         32
##   Number of equality constraints                    28
##
##   Number of observations per group:
##     1                                             2689
##     2                                              137
##     3                                             5491
##     4                                               21
##
## Model Test User Model:
##
##   Test statistic                               447.241
##   Degrees of freedom                                16
##   P-value (Chi-square)                           0.000
##   Test statistic for each group:
##     1                                          281.866
##     2                                           30.277
##     3                                          127.671
##     4                                            7.428
```

```
#Examine fit stats like Chi-Squared, RMSEA, CFI, etc.
fitMeasures(GetDetails(ace)) #’fitMeasures’ is defined in the lavaan package.
```

```
##                npar                fmin               chisq                  df
##               4.000               0.027             447.241              16.000
##              pvalue      baseline.chisq         baseline.df     baseline.pvalue
##               0.000            2106.324               4.000               0.000
##                 cfi                 tli                nnfi                 rfi
##               0.795               0.949               0.949                  NA
##                 nfi                pnfi                 ifi                 rni
##                  NA               3.151               0.794               0.795
##                logl   unrestricted.logl                 aic                 bic
##          -65103.779          -64880.158          130215.557          130243.671
##              ntotal                bic2               rmsea      rmsea.ci.lower
##            8338.000          130230.960               0.114               0.105
##      rmsea.ci.upper        rmsea.pvalue                 rmr          rmr_nomean
##               0.123               0.000               9.992              12.765
##                srmr        srmr_bentler srmr_bentler_nomean                crmr
##               0.130               0.130               0.089               0.178
##         crmr_nomean          srmr_mplus   srmr_mplus_nomean               cn_05
##               0.026               0.153               0.083             491.246
##               cn_01                 gfi                agfi                pgfi
##             597.581               0.999               0.999               0.799
##                 mfi
##               0.974
```

```
#Examine low-level details like each group’s individual parameter estimates and standard
#  errors.  Uncomment the next line to view the entire output (which is roughly 4 pages).
#summary(GetDetails(ace))
```

1.7 ACE SEM of Two Generations

{This is a more moderately difficult analysis with a more common estimation mechanism.}

{Benefits of cross-generational analysis}

The example differs from the previous example in one way –all possible pairs are considered for the analysis. Pairs are only excluded (a) if they belong to one of the small R groups that are difficult to estimate, or (b) if the value for adult height is missing. This includes all 42,836 relationships in the follow five types of NLSY79 relationships.

```
xt <- xtable::xtable(table(Links79Pair$RelationshipPath, dnn=c("Relationship Frequency")),
             caption="Number of NLSY79 relationship, by ‘RelationshipPath‘.(Recall that ’AuntNiece’
xtable::print.xtable(xt, format.args=list(big.mark=","), type = "html")
```

Number of NLSY79 relationship, by `RelationshipPath`.(Recall that ‘AuntNiece’ also contains uncles and nephews.)

Relationship Frequency

Gen1Housemates
5,302
Gen2Siblings
11,114
Gen2Cousins
5,000
ParentChild
11,521
AuntNiece
9,899

In our opinion, using the intergenerational links is one of the most exciting new opportunities for NLSY researchers to pursue. We will be happy to facilitate such research through consult or collaboration, or even by generating new data structures that may be of value. The complete kinship linking file facilitates many different kinds of cross-generational research, using both biometrical and other kinds of modeling methods.

```
### R Code for Example lavaan estimation analysis with a simple outcome and Gen1 subjects
#Steps 1-5 are explained in the vignette's first example:
library(NlsyLinks)
dsLinking <- subset(Links79Pair, RelationshipPath %in%
                      c("Gen1Housemates", "Gen2Siblings", "Gen2Cousins",
                        "ParentChild", "AuntNiece"))
#Because all five paths are specified, the line above is equivalent to:
#dsLinking <- Links79Pair

dsOutcomes <- ExtraOutcomes79
#The HeightZGenderAge variable is already groomed

#Step 6: Create the single entered dataset.
dsSingle <- CreatePairLinksSingleEntered(
  outcomeDataset   = dsOutcomes,
  linksPairDataset = dsLinking,
  outcomeNames     = c('HeightZGenderAge'))

#Step 7: Declare the names for the two outcome variables.
oName_S1 <- "HeightZGenderAge_S1"
oName_S2 <- "HeightZGenderAge_S2"

#Step 8: Summarize the R groups and determine which groups can be estimated.
dsGroupSummary <- RGroupSummary(dsSingle, oName_S1, oName_S2)
dsGroupSummary
```

```
##        R Included PairCount      O1Mean      O2Mean O1Variance O2Variance
## 1 0.0625     TRUE       202  0.22968753 -0.07575395  1.0509023  0.8271487
## 2 0.1250     TRUE      2422 -0.02213186  0.01502572  1.0334694  0.9809845
## 3 0.2500     TRUE      7136 -0.05441460 -0.03885252  1.0189426  1.0294526
## 4 0.3750     TRUE        48  0.13536238 -0.10786196  1.1497643  0.9269109
## 5 0.5000     TRUE     14862 -0.05738494 -0.01738686  0.9602559  0.9851045
## 6 0.7500    FALSE         0          NA          NA         NA         NA
## 7 1.0000     TRUE        27 -0.11092113 -0.12601206  0.6934418  0.9891146
##   O1O2Covariance Correlation Determinant PosDefinite
## 1      0.1112158   0.1192871   0.8568836        TRUE
## 2      0.1487539   0.1477368   0.9916897        TRUE
## 3      0.2721042   0.2656790   0.9749124        TRUE
## 4      0.4074779   0.3947123   0.8996908        TRUE
## 5      0.4146816   0.4263636   0.7739915        TRUE
## 6             NA          NA          NA       FALSE
## 7      0.6863894   0.8287857   0.2147630        TRUE
```

```
#Step 9: Create a cleaned dataset
dsClean <- CleanSemAceDataset(dsDirty=dsSingle, dsGroupSummary, oName_S1, oName_S2)

#Step 10: Run the model
ace <- AceLavaanGroup(dsClean)
ace
```

```
## [1] "Results of ACE estimation: [show]"
##     ASquared     CSquared     ESquared    CaseCount
## 7.366543e-01 6.580003e-02 1.975456e-01 2.469700e+04
```

```
#Step 11: Inspect the output further (see the final step two examples above).
```

Notice the ACE estimates are very similar to the previous exampl, but the number of pairs has increased by 6x –from 4,185 to 24,700. The number of *subjects* doubles when Gen2 is added, and the number of *relationship pairs* really takes off. When an extended family's entire pedigree is considered by the model, many more types of links are possible than if just nuclear families are considered. This increased statistical power is even more important when the population's a^2 is small or moderate, instead of something large like 0.7.

You may notice that the analysis has 24,697 relationships instead of the entire 42,836. This is primarily because not all subjects have a value for 'adult height' (and that's mostly because a lot of Gen2 subjects are too young). There are 42,088 pairs with a nonmissing value in `RFull`, meaning that 98.3 are classified. We feel comfortable claiming that if a researcher has a phenotype for both members of a pair, there's a 99+% chance we have an `RFull` for it. For a description of the `R` and `RFull` variables, please see the `Links79Pair` entry in the package reference manual.

1.8 More Advanced ACE Analyses

- When the analysis grows beyond a single outcome at one time point, researchers are better off using the modeling software itself (*e.g.*, OpenMx, lavaan, Mplus) than then wrappers provided by `NlsyLinks`, or any other package.

1.9 Data Manipulation and Non-Biometric Analyses

- Even if the investigation doesn't involve family structure, `NlsyLinks` functions and dataset can make the research can be more efficient.

1.10 SAS Analogues

- The `NlsyLinks` datasets can be used in any statistical package, and we demonstrate that here with SAS.
- Downloadable from qqq.-qqqq

1.11 Additional Resources

1.12 Authors & Affiliation

- William Howard Beasley (Howard Live Oak LLC, Norman)
- Michael D. Hunter (Georgia Institute of Technology, Atlanta)
- David Bard (University of Oklahoma Health Sciences Center, OKC)
- Kelly Williams (Oklahoma City University, OKC)
- S. Mason Garrison (Wake Forest University, Winston-Salem)
- Patrick O'Keefe (Vanderbilt University, Nashville)
- Joseph Lee Rodgers (Vanderbilt University, Nashville)

1.13 References