Models for Longitudinal Data

Contents

1 Getting Set Up

1.1 Setting chunk options and generating R script

```
knitr::opts_chunk$set(warning=FALSE, message=FALSE)
knitr::purl("longitudinal-models.Rmd")
```

```
##
##
## processing file: longitudinal-models.Rmd
```

```
## output file: longitudinal-models.R
```

1.2 Installing Packages

```
install.packages('lattice')
library('lattice')
```

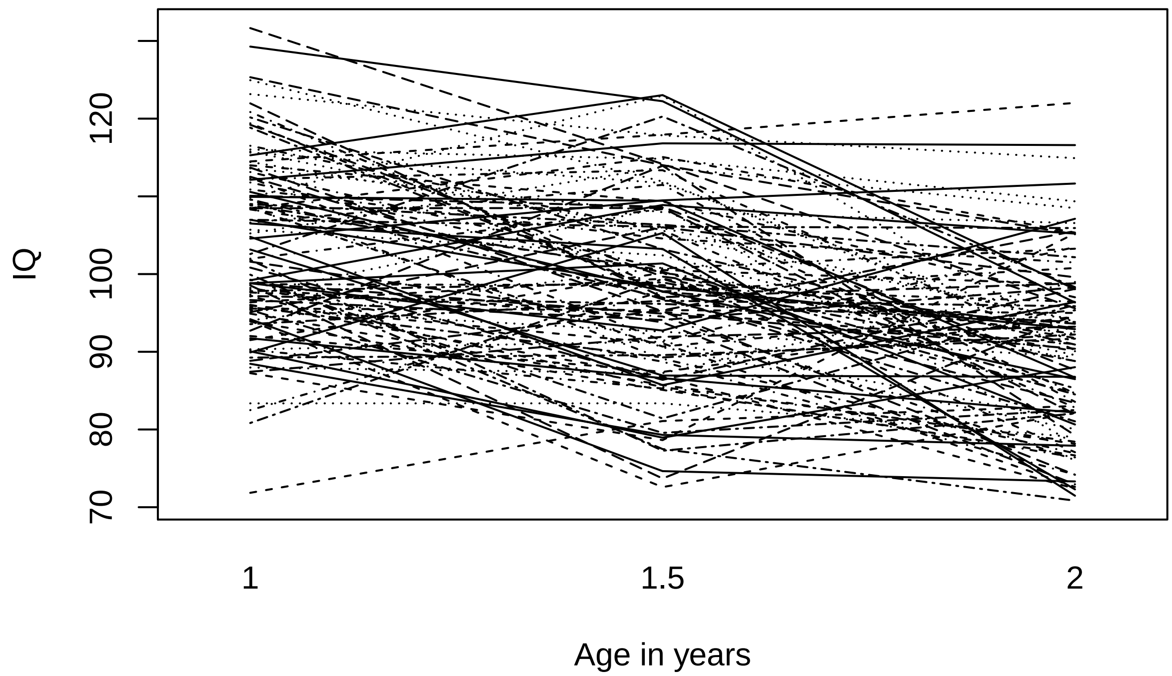
1.3 Reading in the data

```
early.int1 = read.table("earlyint.txt", header=T, sep=",")
attach(early.int1)
```

1.4 Exploring the data

Spaghetti plot.

```
n = length(unique(id))
interaction.plot(age, id, cog, xlab="Age in years", ylab="IQ", legend=F)
```

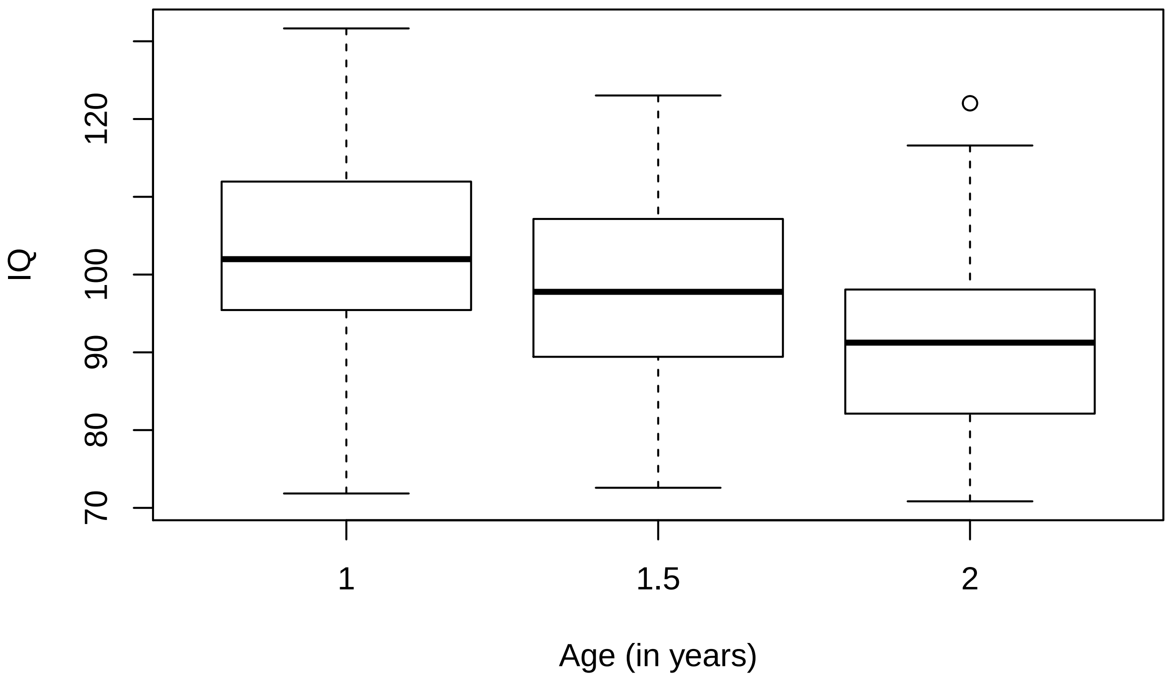


Descriptives.

```
early.mean=tapply(cog,list(age,program),mean) #mean
early.sd=tapply(cog,list(age,program),sd) #sd
early.var=tapply(cog,list(age,program),var) #variance
early.n=table(age,program) #frequency
```

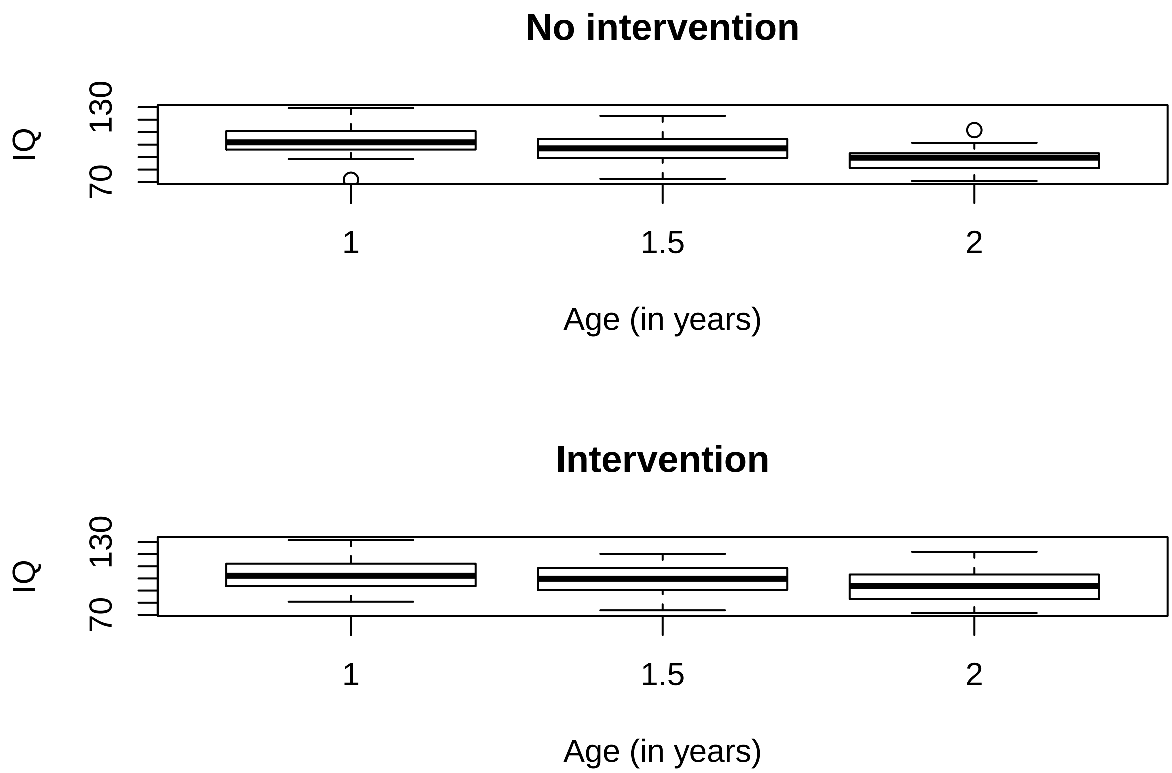
Boxplots.

```
boxplot(cog~age, xlab="Age (in years)", ylab="IQ")
```



Boxplots per program.

```
par(mfrow=c(2,1))
boxplot(cog[program==0]~age[program==0], main="No intervention", main="No intervention", xlab="Age (in y
boxplot(cog[program==1]~age[program==1], main="Intervention", main="No intervention", xlab="Age (in yea
```

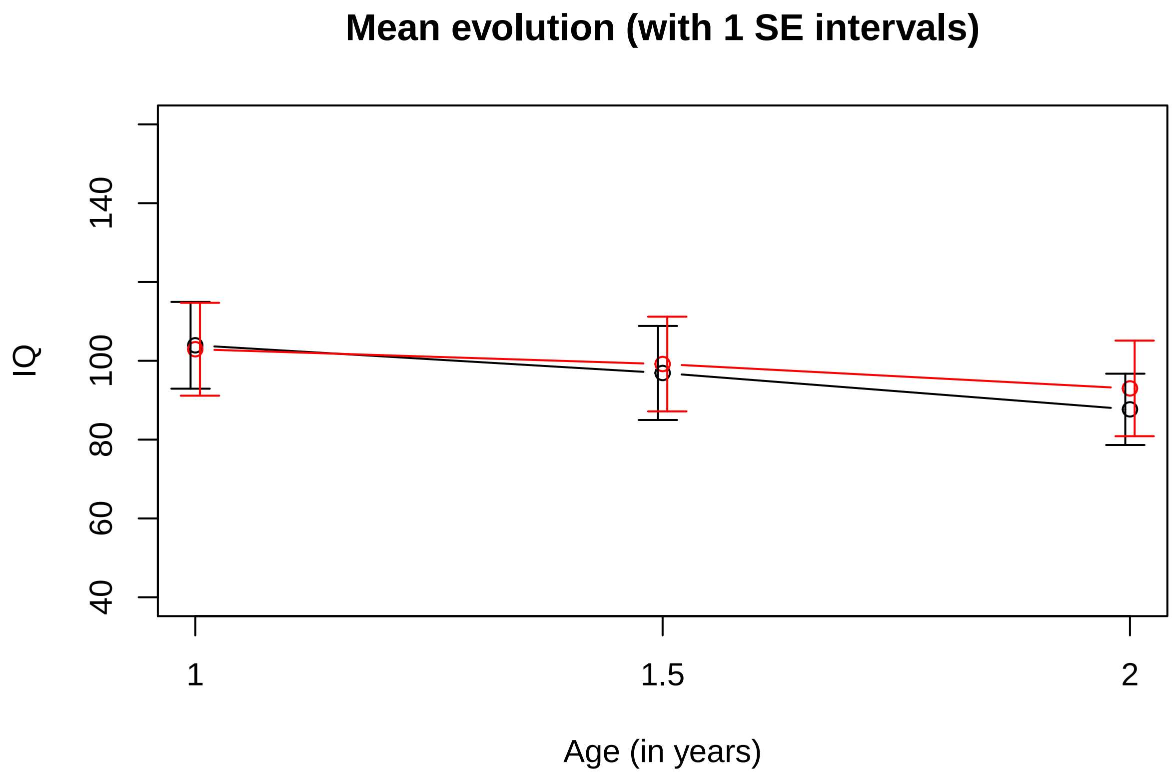


General function to plot error bars.

```
errbar=function(x,y,height,width,lty=1,col="black"){
arrows(x,y,x,y+height,angle=90,length=width,lty=lty, col=col)
arrows(x,y,x,y-height,angle=90,length=width,lty=lty, col=col)
}
```

Plotting mean evolutions.

```
plot(age[id==1],early.mean[,1],type="b",xlim=c(1,2), ylim=c(40,160),xlab="Age (in years)",ylab="IQ",axe
axis(side=1,at=c(1,1.5,2),labels=c(1,1.5,2))
axis(side=2,at=seq(40,160,20))

box()
points(age[id==1],early.mean[,2],type="b",col="red")
errbar(age[id==1]-.005,early.mean[,1],early.sd[,1],.1)
errbar(age[id==1]+.005,early.mean[,2],early.sd[,2],.1,col="red")
```



Correlation between IQ scores at different ages.

```
early.int2 <- reshape(early.int1, timevar = "age", idvar = c("id", "program"), direction = "wide") # re
early.int2
```

```
##      id program     cog.1   cog.1.5     cog.2
## 1     1       1 106.98289  98.31060  92.91342
## 2     2       1 108.86019 100.29307  85.29502
## 3     3       1 112.52438  96.76684  83.42649
## 4     4       1  90.24428  85.27380  76.41052
## 5     5       1 105.70738 102.39839  88.78872
## 6     6       1  93.88987  85.09601  76.66209
## 7     7       1 109.93899 109.43202  86.68573
## 8     8       1 106.98599 106.09735 105.92056
## 9     9       1 125.33166 114.01277 105.12163
## 10   10       1  82.49028  97.76043 100.77653
## 11   11       1 105.22740 113.77558 108.48978
## 12   12       1 114.49240 117.93460 122.02861
## 13   13       1  91.73435  86.60655  82.18117
## 14   14       1  95.37839  79.56730  82.49799
## 15   15       1 110.07345 108.65272  87.65045
## 16   16       1  98.01933  94.54154  76.07024
## 17   17       1  87.96931  87.25195  98.98177
## 18   18       1 108.31917 100.51574  98.70037
## 19   19       1  94.18916  74.64264  73.31470
## 20   20       1 120.16253 103.18718  90.92738
```

```
## 21   21        1 131.65178 114.16135  95.58954
## 22   22        1 120.85608  98.18288  96.44816
## 23   23        1 116.02689 114.82364 109.35596
## 24   24        1 108.50925  90.67410  95.33989
## 25   25        1 112.12633 116.82108 116.59443
## 26   26        1  80.83364  99.37240  84.50895
## 27   27        1  87.42831  91.92213  93.23212
## 28   28        1 112.40877  91.41400 103.47415
## 29   29        1 123.16009 117.85460 114.91998
## 30   30        1  91.50417 100.18023  77.07159
## 31   31        1  98.81854  92.73035 107.11404
## 32   32        1 102.57830 120.27905  98.31582
## 33   33        1  92.71612 113.93789  79.24296
## 34   34        1 113.13320 100.72077 103.26529
## 35   35        1  83.36287  83.36361  79.02352
## 36   36        1  94.91031  85.72218  78.16092
## 37   37        1  99.20004 108.88909 105.20391
## 38   38        1 109.00945 106.33822 102.16497
## 39   39        1  88.83725  95.32658 105.52882
## 40   40        1  97.11882  85.45274  78.42828
## 41   41        1  97.34675 108.54913  78.15538
## 42   42        1 108.46227 115.06198 105.33598
## 43   43        1 106.51196 103.23088  72.29365
## 44   44        1  93.55798  96.57147  95.85989
## 45   45        1  96.66016  94.09513  98.27780
## 46   46        1 112.22206 113.36683  90.21796
## 47   47        1 115.75380 105.04866  94.08408
## 48   48        1  92.02678  87.41049  94.53681
## 49   49        1  90.05997  78.94745  88.01800
## 50   50        1 118.84066  97.71249  93.61212
## 51   51        1 100.83136  86.38944  98.09643
## 52   52        1 101.98162 108.36219  84.64359
## 53   53        1 113.60019 107.96398 106.56405
## 54   54        1  87.24872  78.63402 105.24226
## 55   55        1  89.91285 105.28723  71.47761
## 56   56        1  97.17166  96.15962  93.83076
## 57   57        1  98.54182  73.69470  94.72776
## 58   58        1 116.51412 105.63534  82.82452
## 59   59        0 124.94527 111.91136  91.24352
## 60   60        0  71.84974  81.08563  83.02724
## 61   61        0  88.45453  79.30242  77.91012
## 62   62        0 101.81687  77.26865  82.05273
## 63   63        0  98.83434  94.90388  90.39131
## 64   64        0 106.62630 111.48140  83.56083
## 65   65        0  90.31671  91.30650  81.15678
## 66   66        0  95.55040  95.36279  94.99702
## 67   67        0 110.57535  97.77947  86.48163
## 68   68        0  95.73969  89.27407  92.22706
## 69   69        0 108.46676 108.50177  81.96456
## 70   70        0 109.57738  89.10167  76.92818
## 71   71        0 114.42217 105.91386  93.90563
## 72   72        0  96.37015  99.40169  90.07637
## 73   73        0 104.56600 109.48258 111.65059
## 74   74        0  99.27624  81.42511  96.03642
```

```
## 75   75        0 121.96798  97.03722  98.80731
## 76   76        0 110.88300 104.57459 101.48922
## 77   77        0  95.49981  92.58608  89.33631
## 78   78        0  96.03821  86.79500  72.39639
## 79   79        0 115.28714 123.02920  98.05151
## 80   80        0  89.25232  89.57222  91.68889
## 81   81        0 119.21283  99.67131  93.03702
## 82   82        0  98.68876  98.57612  92.32044
## 83   83        0 108.50237  90.58188  80.04558
## 84   84        0  97.63610  95.12270  90.99789
## 85   85        0  98.82146 101.36631  80.64234
## 86   86        0 109.60582  96.96958  81.04709
## 87   87        0  95.09582 106.02006  99.58778
## 88   88        0  98.22010  88.72318  74.19584
## 89   89        0  94.03053 100.83250  89.57231
## 90   90        0  95.94445  91.09095  72.61159
## 91   91        0 104.79640  85.71041  96.40972
## 92   92        0  97.79446  98.93178  88.80250
## 93   93        0 113.70390  95.20994  74.17274
## 94   94        0 111.78680 100.19471  91.77763
## 95   95        0 115.63780 111.71134  92.56601
## 96   96        0 113.99454 109.34016  97.01341
## 97   97        0 129.27171 122.23883  96.33804
## 98   98        0  99.92436  77.48019  70.84332
## 99   99        0 101.59745  93.78219  72.94243
## 100 100        0 119.40051  97.75317  84.70497
## 101 101        0 109.84539 122.96455  91.49998
## 102 102        0  93.98954  72.58405  82.19806
## 103 103        0 103.16078  86.94306  86.77639
```

```
cor(early.int2[,3:5])
```

```
##            cog.1   cog.1.5     cog.2
## cog.1   1.0000000 0.5816070 0.3263912
## cog.1.5 0.5816070 1.0000000 0.4371109
## cog.2   0.3263912 0.4371109 1.0000000
```

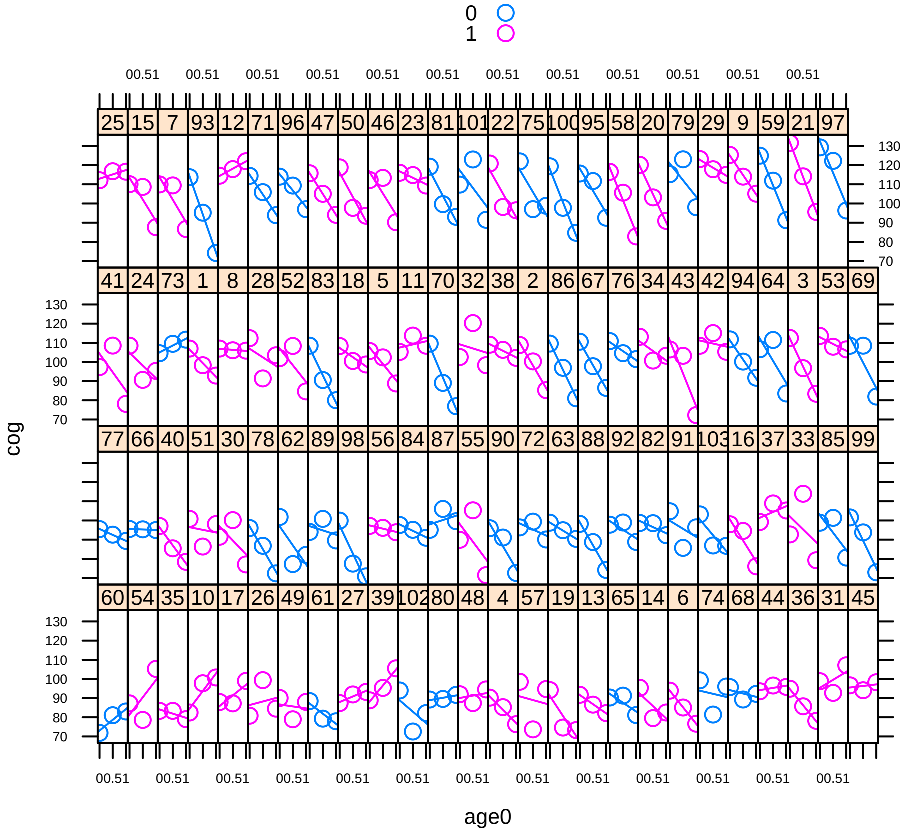
2 Linear Regression Per Person

Creating the time variable.

```
early.int1$age0<-early.int1$age-1
```

Displaying the linear regression per person.

```
cf = sapply(early.int1$id, function(x) coef(lm(cog~age0, data=subset(early.int1, id==x))))
Sx<-reorder(early.int1$id, cf[1,])

xyplot(cog ~ age0|Sx, groups=program, data=early.int1, type=c('p','r'), auto.key=T,aspect="xy", par.set
)
```



2.1 Linear regression of cog on age per participant.

Coefficients.

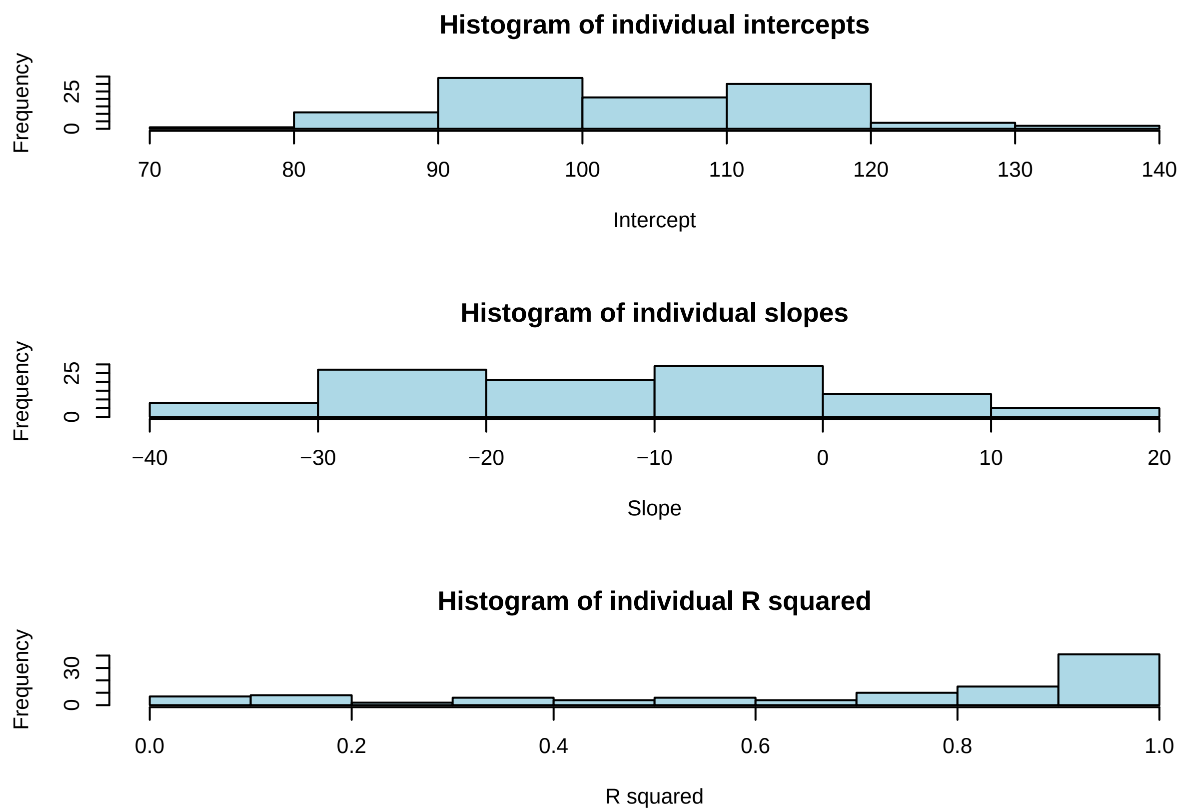
```
lin.reg.coef = by(early.int1, early.int1$id, function(data) coef(lm(cog ~ age0, data=data)))
lin.reg.coef1 = unlist(lin.reg.coef)
names(lin.reg.coef1) = NULL
lin.reg.coef2 = matrix(lin.reg.coef1,length(lin.reg.coef1)/2,2,byrow = TRUE)
```

R-Squared.

```
lin.reg.r.squared = by(early.int1, early.int1$id, function(data) summary(lm(cog ~ age, data=data))$r.squ
lin.reg.r.squared1 = as.vector(unlist(lin.reg.r.squared))
```

Histograms.

```
par(mfrow=c(3,1))
hist(lin.reg.coef2[,1],xlab="Intercept",col="lightblue",main="Histogram of individual intercepts")
hist(lin.reg.coef2[,2],xlab="Slope",col="lightblue",main="Histogram of individual slopes")
hist(lin.reg.r.squared1,xlab="R squared",col="lightblue",main="Histogram of individual R squared")
```

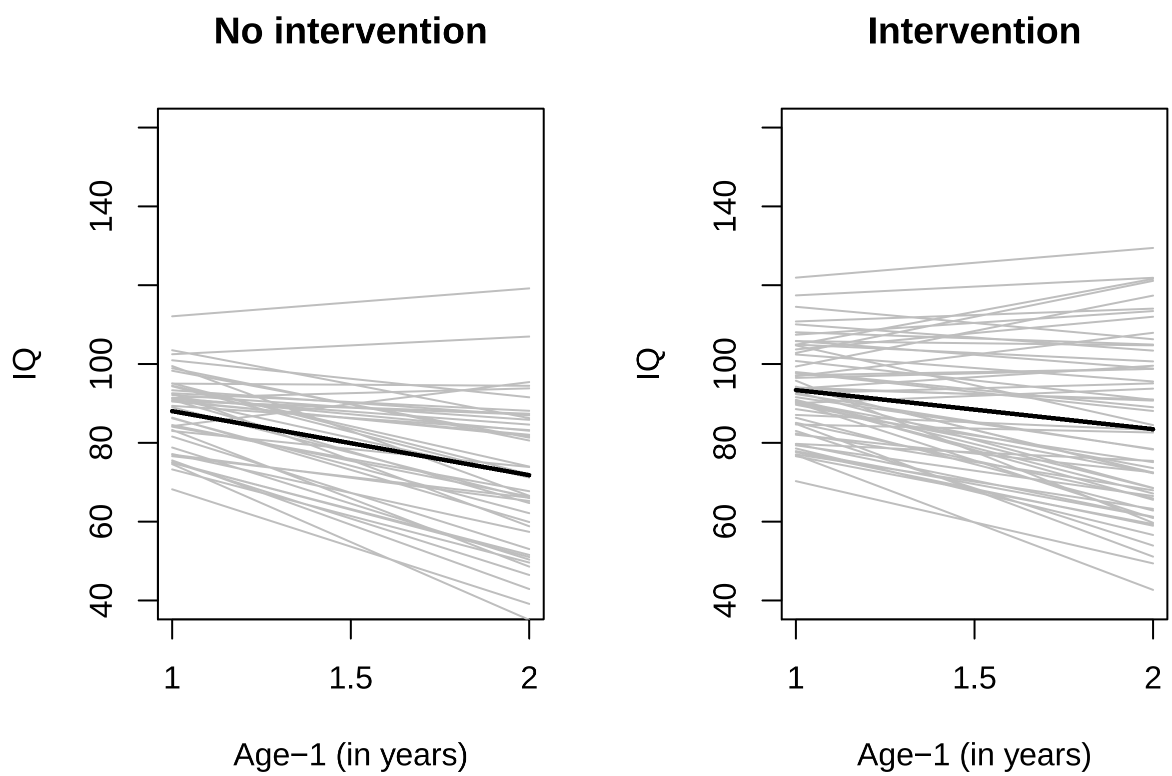


3 Linear regression per person and group

Plotting individual regression lines per group.

```
reg.coef = cbind(lin.reg.coef2, early.int1[early.int1$age==1,]$program)
mean.int = tapply(reg.coef[,1], reg.coef[,3], mean)
mean.slope = tapply(reg.coef[,2], reg.coef[,3], mean)

par(mfrow=c(1,2))
plot(age, cog, type="n", xlim=c(1,2), ylim=c(40,160), main="No intervention", xlab="Age-1 (in years)", y
axis(side=1, at=c(1,1.5,2), labels=c(1,1.5,2))
axis(side=2, at=seq(40,160,20))
box()

for (i in 1:103) {
  if (reg.coef[i,3]==0){
    curve(cbind(1,x)%*%reg.coef[i,1:2], add=T, col="gray")
    curve(cbind(1,x)%*%c(mean.int[1], mean.slope[1]), add=T, lwd=2)
  }
}

plot(age, cog, type="n", xlim=c(1,2), ylim=c(40,160),main="Intervention", xlab="Age-1 (in years)", ylab=
axis(side=1, at=c(1,1.5,2), labels=c(1,1.5,2))
axis(side=2, at=seq(40,160,20))
```

```
box()

for (i in 1:103){
  if (reg.coef[i,3]==1){
    curve(cbind(1,x)%*%reg.coef[i,1:2], add=T, col="gray")
    curve(cbind(1,x)%*%c(mean.int[2], mean.slope[2]), add=T, lwd=2)
  }
}
```



4 Fitting the Model

4.1 Installing the Packages

```
install.packages("lme4")
install.packages("arm")
install.packages("pbkrtest")

library(lme4)
library(lattice)
library(arm)
library(car)
library(pbkrtest)
```

Creating the time variable.

```
early.int1$age0<-early.int1$age-1
```

Fitting the model with maximum likelihood.

```
early.lmer1 = lmer(cog~1+age0*program+(1 + age0|id), REML = FALSE, data=early.int1)
summary(early.lmer1)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: cog ~ 1 + age0 * program + (1 + age0 | id)
##    Data: early.int1
##
##      AIC      BIC   logLik deviance df.resid
##   2332.5   2362.4  -1158.3   2316.5      301
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.25362 -0.59088  0.02131  0.56850  2.29366
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  id       (Intercept) 84.02    9.166
##           age0        39.44    6.281    -0.55
##  Residual             60.31    7.766
## Number of obs: 309, groups:  id, 103
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)  104.3007     1.7274  60.380
## age0         -16.2555     1.8860  -8.619
## program       -0.9646     2.3020  -0.419
## age0:program   6.3187     2.5133   2.514
##
## Correlation of Fixed Effects:
##             (Intr) age0   progrm
## age0        -0.629
## program     -0.750  0.472
## age0:progrm  0.472 -0.750 -0.629
```

4.2 Estimating the fixed effects via bootstrap

```
fixed.boot = bootMer(early.lmer1, fixef, use.u = TRUE, nsim = 250)
fixed.boot
```

```
##
## PARAMETRIC BOOTSTRAP
##
##
## Call:
## bootMer(x = early.lmer1, FUN = fixef, nsim = 250, use.u = TRUE)
```

```
##
##
## Bootstrap Statistics :
##        original      bias    std. error
## t1* 104.3007437  0.01424596    1.025219
## t2* -16.2554565 -0.07347919    1.606066
## t3*  -0.9646326  0.03266867    1.431340
## t4*   6.3187112  0.09511476    2.274090
```

```
summary(fixed.boot)
```

```
##
## Number of bootstrap replications R = 250
##              original  bootBias bootSE   bootMed
## (Intercept)  104.30074  0.014246 1.0252 104.28389
## age0        -16.25546 -0.073479 1.6061 -16.31605
## program      -0.96463  0.032669 1.4313  -0.99671
## age0:program   6.31871  0.095115 2.2741   6.50525
```

4.3 Calculating confidence intervals for the fixed effects via Wald, bootstrap and profile likelihood

```
confint(early.lmer1,par=5:8,method="Wald",oldNames = FALSE) # Only for fixed effects vc will return NA
```

```
##                   2.5 %     97.5 %
## (Intercept)  100.915097 107.686391
## age0        -19.951912 -12.559002
## program      -5.476396   3.547131
## age0:program   1.392761  11.244662
```

```
confint(early.lmer1,method="boot",boot.type ="perc",oldNames = FALSE,nsim=500)
```

```
##                                 2.5 %      97.5 %
## sd_(Intercept)|id          6.6248377  11.1720202
## cor_age0.(Intercept)|id  -1.0000000   0.9999961
## sd_age0|id                 0.8700550   9.9647573
## sigma                      6.6032140   8.6957184
## (Intercept)              100.7225482 107.7892045
## age0                     -19.9596571 -12.6528151
## program                   -5.4528343   3.9271159
## age0:program               0.8768822  11.1150900
```

```
confint(early.lmer1, level = 0.95,method="profile",oldNames = FALSE)
```

```
##                               2.5 %     97.5 %
## sd_(Intercept)|id          7.005366  11.406182
## cor_age0.(Intercept)|id  -1.000000   1.000000
## sd_age0|id                 0.000000   9.975354
## sigma                      6.814978   8.953288
```

```
## (Intercept)              100.883287 107.718200
## age0                     -19.986640 -12.524273
## program                   -5.518786   3.589521
## age0:program               1.346481  11.290942
```

4.4 Get the KR-approximated degrees of freedom

```
#early.lmer1.df.KR = get_ddf_Lb(early.lmer1, fixef(early.lmer1))
```

4.5 Likelihood ratio tests

```
early.lmer1.noprog<-lmer(cog~1+age0+(1 + age0|id), REML = FALSE, data=early.int1)
early.lmer1.intprog<-lmer(cog~1+age0+program+(1 + age0|id), REML = FALSE, data=early.int1)
anova(early.lmer1.noprog,early.lmer1.intprog,early.lmer1)
```

```
## Data: early.int1
## Models:
## early.lmer1.noprog: cog ~ 1 + age0 + (1 + age0 | id)
## early.lmer1.intprog: cog ~ 1 + age0 + program + (1 + age0 | id)
## early.lmer1: cog ~ 1 + age0 * program + (1 + age0 | id)
##                     npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## early.lmer1.noprog     6 2336.8 2359.2 -1162.4   2324.8
## early.lmer1.intprog    7 2336.7 2362.8 -1161.3   2322.7 2.0840  1    0.14885
## early.lmer1            8 2332.5 2362.4 -1158.3   2316.5 6.1345  1    0.01326 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.6 Random effects covariance matrix

```
D.early = unclass(VarCorr(early.lmer1))$id
D.early
```

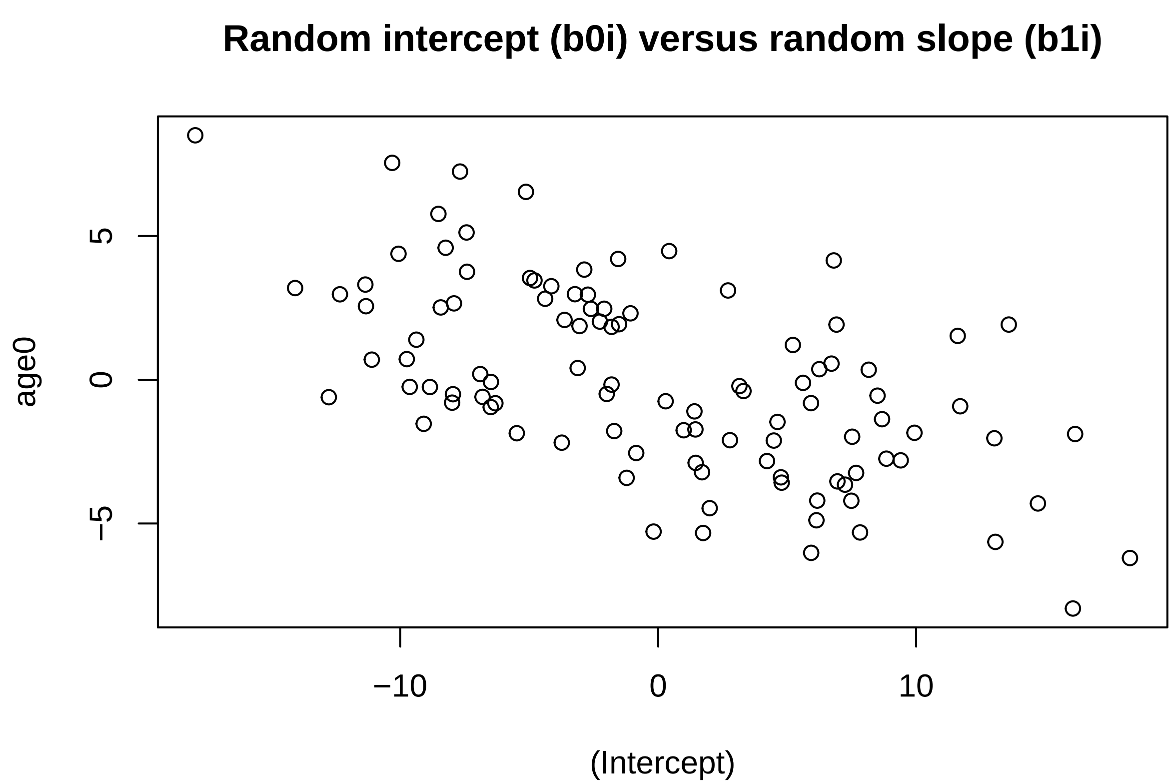
```
##             (Intercept)      age0
## (Intercept)    84.01946 -31.89378
## age0          -31.89378  39.44496
## attr(,"stddev")
## (Intercept)        age0
##    9.166213    6.280522
## attr(,"correlation")
##             (Intercept)       age0
## (Intercept)   1.0000000 -0.5540135
## age0         -0.5540135  1.0000000
```

4.7 Predicted random effects

```r
early.lmer1.re = ranef(early.lmer1)$id
head(early.lmer1.re,10)
```

```
##    (Intercept)       age0
## 1     1.406085 -1.0998501
## 2     1.700819 -3.2167576
## 3     1.996405 -4.4674896
## 4   -11.103349  0.6994181
## 5     1.444739 -1.7280748
## 6    -9.633038 -0.2479249
## 7     4.787570 -3.5798370
## 8     5.221722  1.2096924
## 9    14.723774 -4.3028253
## 10   -7.682855  7.2428573
```

```r
plot(early.lmer1.re, main="Random intercept (b0i) versus random slope (b1i)")
```



OLS vs LM Estimates ## Creating the subject specific intercepts and slopes

```r
ind.coef=coef(early.lmer1)$id
head(ind.coef)
```

```
##   (Intercept)      age0    program age0:program
## 1   105.70683 -17.35531 -0.9646326     6.318711
```

```
## 2   106.00156 -19.47221 -0.9646326     6.318711
## 3   106.29715 -20.72295 -0.9646326     6.318711
## 4    93.19739 -15.55604 -0.9646326     6.318711
## 5   105.74548 -17.98353 -0.9646326     6.318711
## 6    94.66771 -16.50338 -0.9646326     6.318711
```

```
int.subject=ind.coef[,1]+ind.coef[,3]
slope.subject=ind.coef[,2]+ind.coef[,4]
plot(int.subject,slope.subject, main="Random intercept versus random slope (Including the fixed effects)
```