acorn Vignette

Tychele N. Turner, Ph.D., Washington University School of Medicine

2023-01-18

Contents

call up acorn

```
library('acorn')
```

check out acorn functions

```
lsf.str("package:acorn")
```

```
## annotateCpG : function (DNVobject = NULL, CpGannot = NULL)
## calculateDeletionInsertionRatio : function (DNVobject = NULL)
## calculateDeletionLengths : function (DNVobject = NULL)
## calculateInsertionLengths : function (DNVobject = NULL)
## calculateMNVLengths : function (DNVobject = NULL)
## calculateTiTvRatio : function (DNVobject = NULL)
## countsPerIndividual : function (DNVobject = NULL)
## extractAutosomes : function (DNVobject = NULL)
## extractINDELs : function (DNVobject = NULL)
## extractIndividual : function (DNVobject = NULL, individual = NULL)
```

```
## extractMNVs : function (DNVobject = NULL)
## extractSNVs : function (DNVobject = NULL)
## extractX : function (DNVobject = NULL)
## extractY : function (DNVobject = NULL)
## fatherAge : function (parentalAgeObject = NULL)
## motherAge : function (parentalAgeObject = NULL)
## parentalAge : function (parentalAgeObject = NULL)
## parentalAgeObject : function (counts = NULL, parentalData = NULL)
## readDNV : function (DNVfile = NULL)
```

load in test data

readDNV = Reads in a de novo variant (DNV) file in the format of sample, chromosome, genomic position, reference allele, alternate allele, and then any optional columns. File must be tab-delimited and the file must have the data in the order listed above (i.e., sample is field 1, chromosome is field 2, genomic position is field 3, reference allele is field 4, and alternate allele is field 5. The file can either be a uncompressed file or can be a gz compressed file. Please note that the chromosome data should take the form with a "chr" at the beginning (e.g., chr1).

Returns back a loaded in version of the DNV file that can be assigned to an object.

```
input <- readDNV(paste(path.package("acorn"),"/extdata/dnms_from_Ng_et_al_2022_Human_Mutation_paper.txt
head(input)
```

```
##    SAMPLE CHROM POS_B38 REFERENCE                        ALTERNATE
## 1 HG01928  chr1  913941         G                                T
## 2 HG03915  chr1  917676         C                                A
## 3 HG03915  chr1  918783         G                                C
## 4 HG00450  chr1 1216505         A                                G
## 5 HG02257  chr1 1217502         G                                A
## 6 HG00465  chr1 1366883         G GGTGTGAATTGGTGTAGTGTGAATGAGT
##                                       ID
## 1                          chr1_913941_G_T
## 2                          chr1_917676_C_A
## 3                          chr1_918783_G_C
## 4                         chr1_1216505_A_G
## 5                         chr1_1217502_G_A
## 6 chr1_1366883_G_GGTGTGAATTGGTGTAGTGTGAATGAGT
```

```
str(input)
```

```
## 'data.frame':    9741 obs. of  6 variables:
##  $ SAMPLE   : chr  "HG01928" "HG03915" "HG03915" "HG00450" ...
##  $ CHROM    : chr  "chr1" "chr1" "chr1" "chr1" ...
##  $ POS_B38  : int  913941 917676 918783 1216505 1217502 1366883 1765426 2332062 2645102 3355666 ...
##  $ REFERENCE: chr  "G" "C" "G" "A" ...
##  $ ALTERNATE: chr  "T" "A" "C" "G" ...
##  $ ID       : chr  "chr1_913941_G_T" "chr1_917676_C_A" "chr1_918783_G_C" "chr1_1216505_A_G" ...
```

extract individual

extractIndividual = Extracts the DNVs out of a dnvObject from a particular individual. Returns a DNV object containing only DNVs in the specified individual.

```
ind <- extractIndividual(input, "HG01928")
head(ind)
```

```
##        SAMPLE CHROM    POS_B38 REFERENCE ALTERNATE                ID
## 1    HG01928  chr1     913941         G         T    chr1_913941_G_T
## 12   HG01928  chr1    3393842         G         A   chr1_3393842_G_A
## 166  HG01928  chr1   44230922         C         T  chr1_44230922_C_T
## 304  HG01928  chr1   94001171         C         T  chr1_94001171_C_T
## 405  HG01928  chr1  151473815         T         C chr1_151473815_T_C
## 422  HG01928  chr1  156638884         G         A chr1_156638884_G_A
```

```
nrow(ind)
```

```
## [1] 85
```

```
table(ind[,1])
```

```
##
## HG01928
##      85
```

extract individuals if you want

```
ind <- extractIndividual(input, c("HG01928", "HG03915"))
head(ind)
```

```
##        SAMPLE CHROM  POS_B38 REFERENCE ALTERNATE               ID
## 1    HG01928  chr1   913941         G         T   chr1_913941_G_T
## 2    HG03915  chr1   917676         C         A   chr1_917676_C_A
## 3    HG03915  chr1   918783         G         C   chr1_918783_G_C
## 12   HG01928  chr1  3393842         G         A  chr1_3393842_G_A
## 74   HG03915  chr1 18766956         T         A chr1_18766956_T_A
## 166  HG01928  chr1 44230922         C         T chr1_44230922_C_T
```

```
nrow(ind)
```

```
## [1] 158
```

```
table(ind[,1])
```

```
##
## HG01928 HG03915
##      85      73
```

extract SNVs

extractSNVs = Extracts single-nucleotide variants (SNVs) out from a DNV object generated using the readDNV function. Returns a DNV object containing only SNVs.

```
snvs <- extractSNVs(input)
nrow(snvs)
```

```
## [1] 8558
```

extract indels

extractINDELs = Extracts small insertions/deletions (INDELs) out from a DNV object generated using the readDNV function. Returns a DNV object containing only INDELs.

```
indels <- extractINDELs(input)
nrow(indels)
```

```
## [1] 1183
```

3

extract MNVs (there are none in the test set)

extractMNVs = Extracts multi-nucleotide variants (MNVs) out from a DNV object generated using the readDNV function. Returns a DNV object containing only MNVs.

```
mnvs <- extractMNVs(input)
nrow(mnvs)
```

```
## [1] 0
```

calculate the Transition/Transversion ratio

calculateTiTvRatio = This function will automatically grab only the SNVs from the DNV object for the calculation of the transition/transversion (Ti/Tv) ratio. Returns the counts of transitions, the counts of transversions, the Ti/Tv ratio, and a barplot of the different types of SNV changes observed in the DNV object.
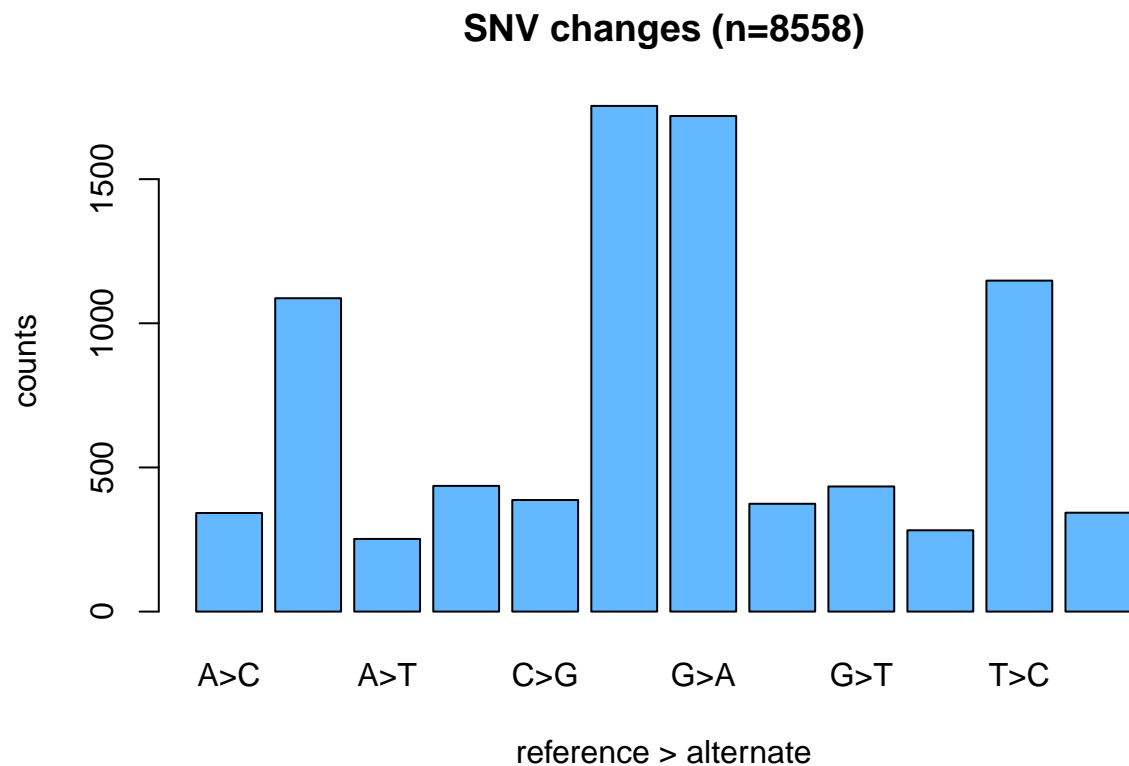
```
calculateTiTvRatio(input)
```

```
## number of transitions (A>G, C>T, G>A, T>C): 5708
```

```
## number of transversions (A>C, A>T, C>A, C>G, G>C, G>T, T>A, T>G): 2850
```

```
## Ti/Tv ratio: 2.00280701754386
```

```
## Plot of different nucleotide changes:
```

SNV changes (n=8558)

calculate deletion/insertion ratio

calculateDeletionInsertionRatio = This function will automatically grab only the INDELs from the DNV object for the calculation of the deletion/insertion ratio. Returns the counts of deletions, the counts of insertions, and the deletion/insertion ratio.

```
calculateDeletionInsertionRatio(input)
```

```
## number of deletions: 540
```

```
## number of insertions: 643
```

```
## deletion/insertion ratio: 0.839813374805599
```

calculate deletion lengths

calculateDeletionlengths = This function will automatically grab only the deletions from the DNV object for the calculation of the length of the deletions. Returns the length of the deletions, in the form of an object, observed in the DNV object. It also returns a barplot of the deletion lengths.

```
dellengths <- calculateDeletionLengths(input)
```

deletion lengths (n=540)
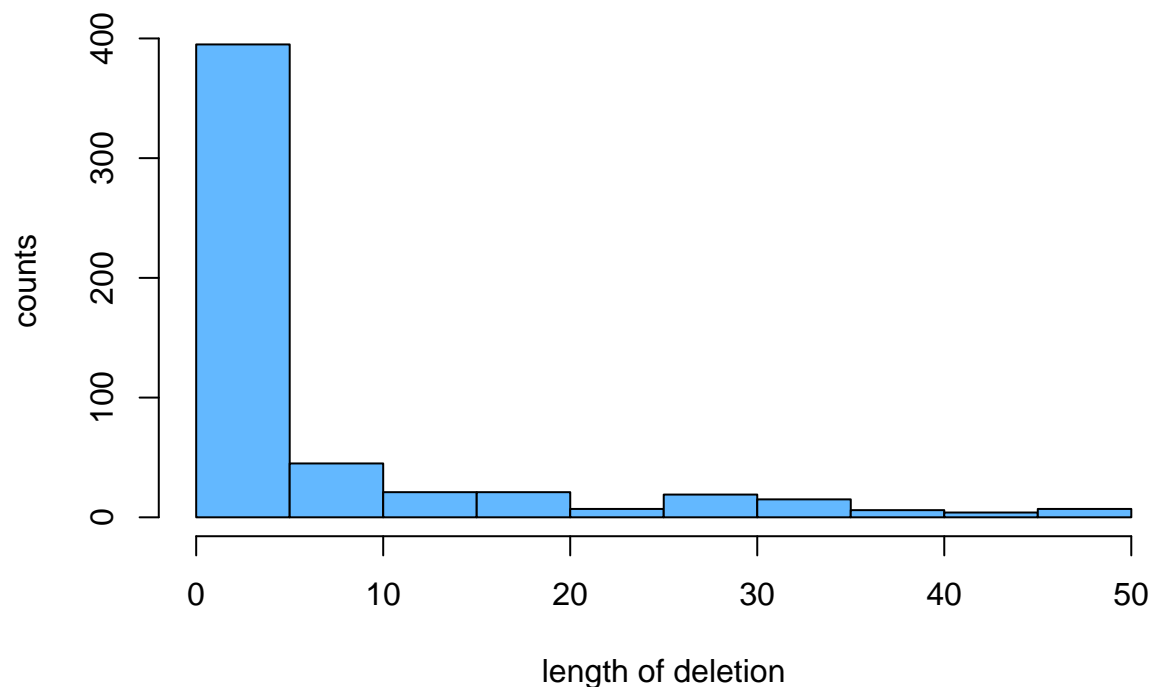

```
head(dellengths)
```

```
## [1]  2  2 27  2  2 15
```

calculate insertion lengths

calculateInsertionLengths = This function will automatically grab only the insertions from the DNV object for the calculation of the length of the insertions. Returns the length of the insertions, in the form of an object, observed in the DNV object. It also returns a barplot of the insertion lengths.

```
inslengths <- calculateInsertionLengths(input)
```

insertion lengths (n=643)

length of insertion
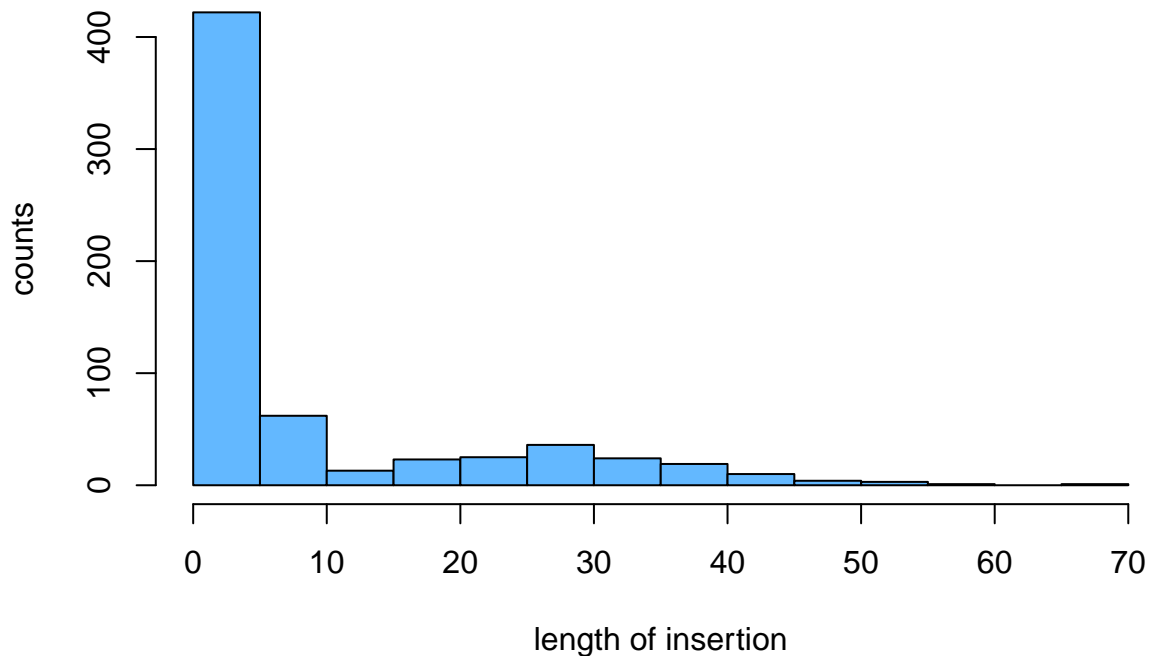
```
head(inslengths)
```

```
## [1] 28  3 37 37 41  5
```

keep only the autosomes

extractAutosomes = Extracts the autosomes (chromosomes 1 to 22) out from a DNV object originally
generated using the readDNV function. You can also run this on objects generated from extractSNVs,
extractINDELs, or extractMNVs. Returns a DNV object containing only DNVs on the autosomes.

```
aut <- extractAutosomes(input)
nrow(aut)
```

```
## [1] 9262
```

```
table(aut[,2])
```

```
##
##   chr1 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 chr19   chr2 chr20
##    707   408   477   408   338   479   251   273   233   232   206   833   257
## chr21 chr22  chr3  chr4  chr5  chr6  chr7  chr8  chr9
##    143   154   696   642   614   551   482   501   377
```

keep only the X chromosome

extractX = Extracts the X chromosome out from a DNV object originally generated using the readDNV
function. You can also run this on objects generated from extractSNVs, extractINDELs, or extractMNVs.
Returns a DNV object containing only DNVs on the X chromosome.

```
X <- extractX(input)
nrow(X)
```

```
## [1] 479
```

```
table(X[,2])
```

```
##
## chrX
##  479
```

keep only the Y chromosome (there are none on the Y in the test dataset)

extractY = Extracts the Y chromosome DNVs out from a DNV object originally generated using the readDNV function. You can also run this on objects generated from extractSNVs, extractINDELs, or extractMNVs. Returns a DNV object containing only DNVs on the Y chromosome.

```
Y <- extractY(input)
nrow(Y)
```

```
## [1] 0
```

calculate counts per individual

countsPerIndividual = This function will count the DNVs from a DNV object originally generated using the readDNV function. You can also run this on objects generated from extractSNVs, extractINDELs, or extractMNVs. Returns the mean of the DNV counts per individual, the standard deviation of the DNV counts per individual, a plot of the density of the DNV counts per individual, and an object consisting of the sample name and the counts of their DNVs that can be assigned to another object.

```
counts <- countsPerIndividual(input)
```

```
## mean of the counts per individual:
```

```
## 79.1951219512195
```

```
##
```

```
## standard deviation of the counts per individual:
```

```
## 12.2043116115263
```

```
##
```

```
## Plot generating of the density of the DNV counts.
```

density of DNV counts per individual
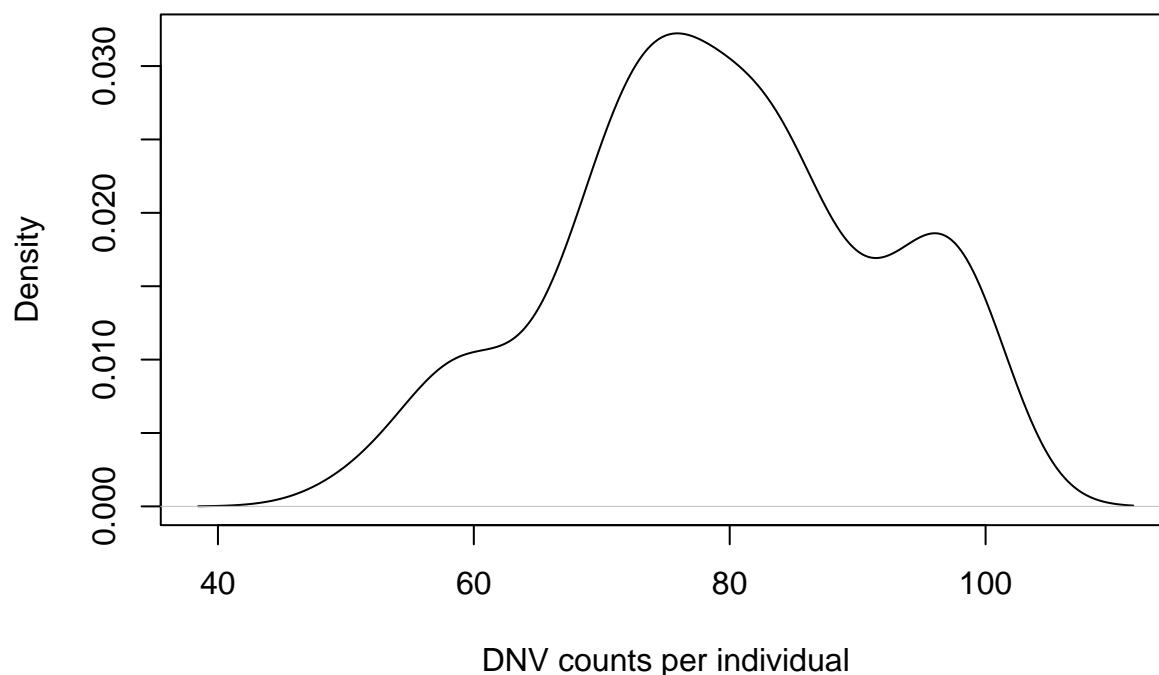

```
head(counts)
```

```
##     sample dnv_count
## 1 HG00405        70
## 2 HG00423        78
## 3 HG00429        57
## 4 HG00438        66
## 5 HG00444        74
## 6 HG00447        75
```

load in example data for parental age analyses

```
input <- readDNV(paste(path.package("acorn"),"/extdata/dnms_from_Ng_et_al_2022_Human_Mutation_paper.txt
```

```
countExample <- read.delim(paste(path.package("acorn"),"/extdata/dnm_count_example.txt",sep=""))
parentExample <- read.delim(paste(path.package("acorn"),"/extdata/parental_age_example.txt",sep=""))
```

make parental age object

parentalAgeObject = Takes in a counts object that is either the result of countsPerIndividual() or is already read into an object from a file that contains the following two fields: sample and number of DNVs. The parental age object should be read in and contain the following fields: sample, father age at child's birth, and mother age at child's birth. Returns back an object with the de novo counts and parental age data together. The fields in this file are sample, dnm_counts, fatherAge, and motherAge.

```
parents <- parentalAgeObject(countExample, parentExample)
```

run parental age analyses including both mother and father

parentalAge = This function will calculate the correlation between father's and mother's age at birth and DNV counts per individual, the results of the linear model taking the form: lm(formula = dnm_counts ~ fatherAge+motherAge, data = parentalAgeObject). Input required is output from the parentalAgeObject function in this package. Returns the results of the linear model taking the form: lm(formula = dnm_counts ~ fatherAge + motherAge, data = parentalAgeObject). It also returns a plot of father's and mother's age at birth and DNV counts.
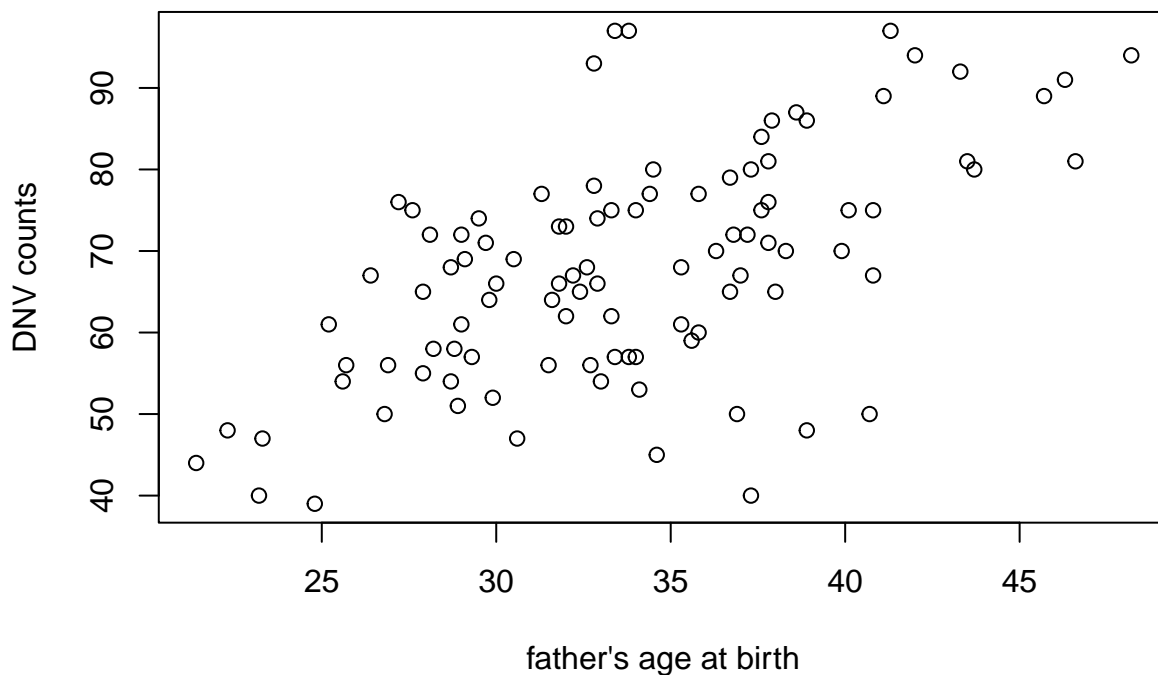
```
parentalAge(parents)
```
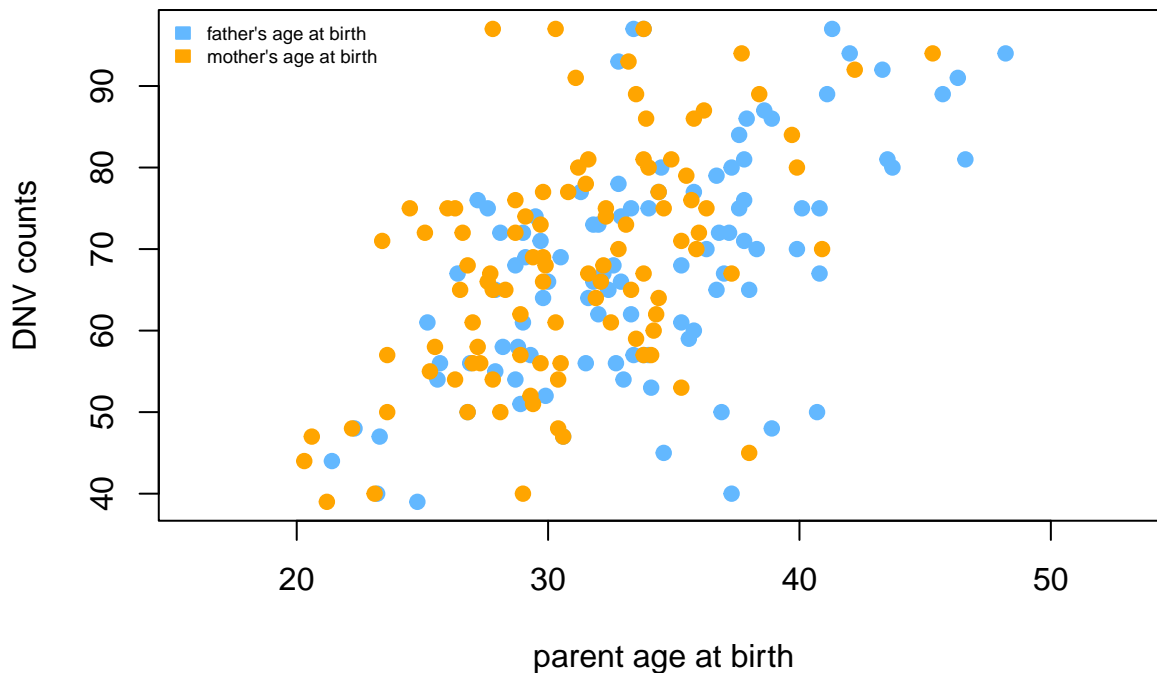


```
## $`summary of linear model for father's and mother's age at birth and DNV counts`
##
## Call:
## lm(formula = dnm_counts ~ fatherAge + motherAge, data = parentalAgeObject)
##
## Residuals:
##     Min     1Q  Median     3Q     Max
## -30.190  -6.831  -0.477   6.975  31.700
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.8374     7.7691   1.395 0.166220
## fatherAge     1.0331     0.2669   3.871 0.000196 ***
## motherAge     0.7179     0.3174   2.262 0.025955 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.14 on 97 degrees of freedom
## Multiple R-squared:  0.3732, Adjusted R-squared:  0.3603
## F-statistic: 28.87 on 2 and 97 DF,  p-value: 1.45e-10
```

```
##
##
## $`confidence interval of linear model for father's and mother's age at birth and DNV counts`
##                   2.5 %    97.5 %
## (Intercept) -4.58207432 26.256945
## fatherAge    0.50343356  1.562695
## motherAge    0.08788934  1.347946
```

run parental age analyses for father age only

fatherAge = This function will calculate the correlation between father's age at birth and DNV counts per individual, the results of the linear model taking the form: lm(formula = dnm_counts ~ fatherAge, data = parentalAgeObject). Input required is output from the parentalAgeObject function in this package. Returns the correlation between father's age at birth and DNV counts per individual and the results of the linear model taking the form: lm(formula = dnm_counts ~ fatherAge, data = parentalAgeObject). It also returns a plot of father's age at birth and DNV counts.

```
fatherAge(parents)
```



```
## $`correlation of father's age at birth and DNV counts`
##
##  Pearson's product-moment correlation
##
## data:  parentalAgeObject[, 3] and parentalAgeObject[, 2]
## t = 7.1073, df = 98, p-value = 1.927e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4368243 0.6994934
## sample estimates:
##       cor
## 0.5832034
##
##
```

```
## $`summary of linear model for father's age at birth and DNV counts`
##
## Call:
## lm(formula = dnm_counts ~ fatherAge, data = parentalAgeObject)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.027  -6.690  -0.768   7.624  29.579
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.4131     6.9218   2.805  0.00608 **
## fatherAge     1.4374     0.2022   7.107 1.93e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.37 on 98 degrees of freedom
## Multiple R-squared:  0.3401, Adjusted R-squared:  0.3334
## F-statistic: 50.51 on 1 and 98 DF,  p-value: 1.927e-10
##
##
## $`confidence interval of linear model for father's age at birth and DNV counts`
##                 2.5 %    97.5 %
## (Intercept) 5.676962 33.149157
## fatherAge   1.036030  1.838704
```

run parental age analyses for mother age only

motherAge = This function will calculate the correlation between mother's age at birth and DNV counts per individual, the results of the linear model taking the form: lm(formula = dnm_counts ~ motherAge, data = parentalAgeObject). Input required is output from the parentalAgeObject function in this package. Returns the correlation between mother's age at birth and DNV counts per individual and the results of the linear model taking the form: lm(formula = dnm_counts ~ motherAge, data = parentalAgeObject). It also returns a plot of mother's age at birth and DNV counts.

```
motherAge(parents)
```

```
## $`correlation of mother's age at birth and DNV counts`
##
##  Pearson's product-moment correlation
##
## data:  parentalAgeObject[, 4] and parentalAgeObject[, 2]
## t = 6.1173, df = 98, p-value = 1.955e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3671661 0.6545122
## sample estimates:
##       cor
## 0.5256736
##
##
## $`summary of linear model for mother's age at birth and DNV counts`
##
## Call:
## lm(formula = dnm_counts ~ motherAge, data = parentalAgeObject)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.793  -7.578  -0.467   7.825  33.927
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20.2284     7.8898   2.564   0.0119 *
## motherAge     1.5412     0.2519   6.117 1.96e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.91 on 98 degrees of freedom
## Multiple R-squared:  0.2763, Adjusted R-squared:  0.2689
```

```
## F-statistic: 37.42 on 1 and 98 DF,  p-value: 1.955e-08
##
##
## $`confidence interval of linear model for mother's age at birth and DNV counts`
##                2.5 %   97.5 %
## (Intercept) 4.571389 35.88550
## motherAge   1.041211  2.04113
```