

Classes *gTrio* and *iTrio*: Derivatives of *TrioSet* for use with genotype and intensity data in package *trioClasses*

Samuel G. Younkin

December 9, 2012

I found it difficult to construct an extension of *gSet* that was flexible with the *AssayData*, e.g., *geno*, *lrr* and *baf*. I think it makes sense to begin with the individual class scheme, and to strip the *TrioSet* class down to bare-bones so we can build it up into something that suits our present needs. Ultimately, I think a one class scheme is best, but I think it will be challenging to implement.

```
> rm(list = ls())
> library("trioClasses")
> library("trio")
```

First we load the sample pedigree data frame included in local versions of the *trioClasses* package.

```
> data(ped)
> head(ped.df)
```

| | id | mid | fid | Population | PI | Ethnicity |
|---------|---------|---------|---------|-------------|--------|-----------|
| 578_01 | 578_01 | 578_03 | 578_02 | PHILIPPINES | Murray | filipino |
| 578_02 | 578_02 | <NA> | <NA> | PHILIPPINES | Murray | filipino |
| 578_03 | 578_03 | <NA> | <NA> | PHILIPPINES | Murray | filipino |
| 1539_01 | 1539_01 | 1539_03 | 1539_02 | IOWA | Murray | european |
| 1539_02 | 1539_02 | <NA> | <NA> | IOWA | Murray | european |
| 1539_03 | 1539_03 | <NA> | <NA> | IOWA | Murray | european |

```
> pedigreeInfo <- within(ped.df, {
  F <- as.character(fid)
  M <- as.character(mid)
  O <- as.character(id)
})
> tg.ped <- Pedigree(pedigreeInfo = pedigreeInfo)
> tg.ped
```

This pedigree object contains 1812 complete trios.

For access to the data frame use the *trios()* accessor function.

After we ensure that F, M and O exist in the data frame we create a Pedigree object. Note the terse show method for the Pedigree object.

1 *gTrio* class

Next we load the genotype matrix with well-named rows and columns, with rows for subjects and columns for SNPs.

```
> data(geno)
> head(geno.mat[, 1:6])
```

| | snp1 | snp2 | snp3 | snp4 | snp5 | snp6 |
|---------|------|------|------|------|------|------|
| 578_01 | 2 | 2 | 0 | 2 | 2 | 1 |
| 578_02 | 1 | 0 | 1 | 0 | 2 | 2 |
| 578_03 | 0 | 2 | 2 | 0 | 2 | 2 |
| 1539_01 | 2 | 0 | 2 | 2 | 2 | 0 |
| 1539_02 | 1 | 1 | 2 | 0 | 1 | 0 |
| 1539_03 | 1 | 0 | 2 | 2 | 0 | 2 |

Now we format the genotype matrix for input into `gTrio()` and use the `completeTrios` method to remove trios that do not have genotype information for all members.

```
> geno.trio <- genoMat(tg.ped, geno.mat)
> (tg.ped.comp <- completeTrios(tg.ped, colnames(geno.trio)))
```

This pedigree object contains 33 complete trios.
For access to the data frame use the `trios()` accessor function.

Now we create the `gTrio` object from a complete pedigree and properly formatted, well-named, genotype matrix.

```
> (gTrio.obj <- gTrio(tg.ped.comp, geno = geno.trio))
```

```
gTrio (storageMode: lockedEnvironment)
assayData: 10 features, 33 samples
  element names: geno
protocolData: none
phenoData: none
featureData
  featureNames: snp1 snp2 ... snp10 (10 total)
  fvarLabels: position chromosome isSnp
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
genome: hg19
```

```
> class(gTrio.obj)
```

```
[1] "gTrio"
attr(,"package")
[1] "trioClasses"
```

Now we use the `getGeno` method to retrieve a genotype matrix formatted, in this case, for use in Holger's trio package.

```
> geno <- getGeno(gTrio.obj, type = "holger")
> dim(geno)
```

```
[1] 99 10
```

```
> (aTDT <- allelicTDT(mat.snp = geno, size = 10000))
```

Allelic TDT

Top 5 SNPs:

| | Statistic | p-value |
|-------|-----------|---------|
| snp10 | 3.8571 | 0.04953 |
| snp4 | 2.5714 | 0.10881 |
| snp5 | 1.1429 | 0.28505 |
| snp8 | 1.0000 | 0.31731 |
| snp6 | 0.6923 | 0.40538 |

2 *iTrio* class

```
> (iTrio.obj <- iTrio(tg.ped.comp, lrr = geno.trio, baf = geno.trio))
```

```
iTrio (storageMode: lockedEnvironment)
assayData: 10 features, 33 samples
  element names: baf, lrr
protocolData: none
phenoData: none
featureData
  featureNames: snp1 snp2 ... snp10 (10 total)
  fvarLabels: position chromosome isSnp
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
genome: hg19
```