What's In My Raw Data?

James Sims

February 2017

Summary

This document provides a high level overview of the raw data files customers can download from certain vendors that sell direct-to-consumer DNA tests for ancestry in the United States. These files contain the results of single nucleotide polymorphism (SNP) assays obtained and reported using technology described on the vendor web sites. The three direct-to-consumer DNA testing vendors covered in this report are, in alphabetical order, 23AndMe[1], AncestryDNA[2] and Family Tree DNA[3]. Downloaded information was imported into the statistical programming environment known as R, and exploratory data analysis was performed. The target audience for this document is the curious amateur genetic genealogist.

Introduction

This document is part of the supplemental information provided online for a series of articles on genetic genealogy written by the author for *Acorns to Oaks*, the quarterly publication of the Oakland County Genealogical Society (Michigan, USA ocgsmi.org). At the time this document was prepared, AncestryDNA and 23AndMe offered only one ancestry DNA test, while Family Tree DNA offered many kinds of DNA testing for ancestry. In the case of Family Tree DNA, the raw data for the Family Finder test is described here. In all cases, the raw data was downloaded from the author's account at each vendor, and the results are for samples of the author's own DNA.

This document can be read in several different ways. First, it can be read for the high level summary information it contains. This should provide genetic genealogists with a better understanding of their raw data. The document can also be read as an example of applying a few 3rd party software tools, most notably the open-source R statistical programming environment and the free version of RStudio, to understand DNA testing data in more detail. As a result of reviewing this document, genetic genealogists should have a better appreciation for how raw data from different vendors must be pre-processed before meaningful comparisons can be made between vendors. This certainly applies to 3rd party tools such as GEDmatch[4], which can be used to find DNA matches of genealogical interest when the raw data comes from different vendors.

This document was authored entirely in the RStudio development environment using Rmarkdown and R. In this way, authors can combine prose, programming code and calculation results in a single document. This technique goes by the name of literate programming, and is a part of the reproducible research approach to reporting the results of data science projects. This study, not including the author's raw data, is available on the author's GitHub account[5] and is provided to others under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) license.

Raw Data Sources

The author's raw SNP data for this report was manually downloaded following the instructions on each vendor's web site on 20 Dec 2016. A guide to download raw data is available[6]. Family Tree DNA offered several options for downloading the raw data for the Family Finder test. The option chosen for this report

[1] 23AndMe web site link: https://www.23andme.com/
[2] AncestryDNA web site link: https://www.ancestry.com/dna/
[3] Family Tree DNA web site link: https://www.familytreedna.com/
[4] GEDmatch web site https://www.gedmatch.com
[5] GitHub repository for this study.
[6] Family Tree genotype codes and reading instructions link

was Build 37 concatenated. AncestryDNA's process involved sending an email to the customer to complete the download process. The other vendors allowed the author to download the raw data without leaving the vendor's web site after providing the correct password for the account.

Vendor	Compression	File format	Decompressed file size (MB)
23AndMe	.zip	.txt	25.6
Ancestry	.zip	.txt	18.8
Family Tree DNA	.gz	.csv	24.5

Table 1. Brief description of files downloaded from vendor web sites. A .txt file is a plain text file. A .csv is a comma separated value file commonly used to transfer text data to databases, spreadsheets and programming environments. The abbreviation MB stands for megabytes.

Raw Data File Processing and Import in R

23AndMe—The file downloaded from 23AndMe begins with several lines of comments describing the file and the format of the information. As described in the comments at the beginning of the file, the actual data consists of four pieces of information separated by tabs with a return at the end of each line. The comments at the top of the file are reproduced below in a code chunk.

```
# This data file generated by 23andMe at: Tue Dec 20 05:51:03 2016
#
# This file contains raw genotype data, including data that is not used in 23andMe reports.
# This data has undergone a general quality review however only a subset of markers have been
# individually validated for accuracy. As such, this data is suitable only for research,
# educational, and informational use and not for medical or other use.
#
# Below is a text version of your data.  Fields are TAB-separated
# Each line corresponds to a single SNP.  For each SNP, we provide its identifier
# (an rsid or an internal id), its location on the reference human genome, and the
# genotype call oriented with respect to the plus strand on the human reference sequence.
# We are using reference human assembly build 37 (also known as Annotation Release 104).
# Note that it is possible that data downloaded at different times may be different due to ongoing
# improvements in our ability to call genotypes. More information about these changes can be found at:
# https://www.23andme.com/you/download/revisions/
#
# More information on reference human assembly build 37 (aka Annotation Release 104):
# http://www.ncbi.nlm.nih.gov/mapview/map_search.cgi?taxid=9606
#
# rsid  chromosome  position    genotype
```

The data variables are named *rsid*, which contains the scientific name of the SNPs; *chromosome*, the name of the chromosome; the base pair *position* on the chromosome where the SNP occurs; and *genotype*, which identifies which alleles (DNA bases) were detected.

Before the SNP data can be imported into the R statistical programming environment for further description and analysis, the uncompressed file was opened with TextEdit, a text editor for Apple, Inc. Macintosh computers, and the comments were deleted such that the variable names separated by tabs formed the first line (row) of the file. The file was saved to disk with the name my_23andme.txt.

Before importing the raw data from this vendor, several packages of functions were loaded into the R environment to make reading files and manipulating data easier and more readable.

```
# set working directory to a GitHub linked local repository
setwd("~/git/myrawdata")
# load library packages to read and manipulate data
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gtools)
```

The following code chunk reads the file containing the 23AndMe raw data into R, and places the data in a data object R calls a dataframe. The last line of the code chunk tells R to print out the first two rows of the dataframe.

```
# import 23AndMe raw SNP alleles as a dataframe, specifying all variables as
# character variables except position, which is an integer
snps_23AndMe <- data.frame(read_tsv("my_23andme.txt", col_names = TRUE,
                                    col_types = "ccic"))
# print first two rows of dataframe, show variable names as column headings
snps_23AndMe[1:2,]
```

```
##        rsid chromosome position genotype
## 1 rs4477212          1    82154       AA
## 2 rs3094315          1   752566       AG
```

As can be seen in the code chunk above, each row of the snps_23AndMe dataframe contains all the values for a single SNP, including its name, which chromosome the SNP occurs on, the position on the chromosome where the SNP occurs and finally, the genotype information. In the case of row 1, the author's two chromosomes named 1 had the same DNA base, A, at that SNP. In contrast, on row 2 of the dataframe, the author had different bases (an A or a G) on each chromosome number 1 at that SNP.

The naming conventions for chromosomes was reviewed. The following code chunk tells R to print the names of chromosomes in the 23AndMe raw data file.

```
# print one instance of each name of each chromosome in the dataframe
# R encloses printed values of the type character with quotes,
# which can be ignored
unique(snps_23AndMe$chromosome)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "X"  "Y"  "MT"
```

As can be seen from the output of the code chunk above, the names of the chromosomes in the snps_23AndMe dataframe are the numbers 1 through 22, plus X, Y and MT. MT, means the mitochondrial chromosome.

AncestryDNA—The raw data file downloaded from AncestryDNA begins with several lines of comments describing the file and the format of the information. As described in the comments at the beginning of the file, the actual data consists of five pieces of information separated by tabs with a return at the end of each line. The comments at the top of the file are reproduced below in a code chunk.

```
#AncestryDNA raw data download
#This file was generated by AncestryDNA at: 12/20/2016 14:00:21 UTC
#Data was collected using AncestryDNA array version: V1.0
#Data is formatted using AncestryDNA converter version: V1.0
#Below is a text version of your DNA file from Ancestry.com DNA, LLC.  THIS
#INFORMATION IS FOR YOUR PERSONAL USE AND IS INTENDED FOR GENEALOGICAL RESEARCH
#ONLY.  IT IS NOT INTENDED FOR MEDICAL OR HEALTH PURPOSES.  THE EXPORTED DATA IS
#SUBJECT TO THE AncestryDNA TERMS AND CONDITIONS, BUT PLEASE BE AWARE THAT THE
#DOWNLOADED DATA WILL NO LONGER BE PROTECTED BY OUR SECURITY MEASURES.
#WHEN YOU DOWNLOAD YOUR RAW DNA DATA, YOU ASSUME ALL RISK OF STORING,
#SECURING AND PROTECTING YOUR DATA.  FOR MORE INFORMATION, SEE ANCESTRYDNA FAQS.
#
#Genetic data is provided below as five TAB delimited columns.  Each line
#corresponds to a SNP.  Column one provides the SNP identifier (rsID where
#possible).  Columns two and three contain the chromosome and basepair position
#of the SNP using human reference build 37.1 coordinates.  Columns four and five
#contain the two alleles observed at this SNP (genotype).  The genotype is reported
#on the forward (+) strand with respect to the human reference.
# rsid  chromosome  position    allele1 allele2
```

The data variables are *rsid*, which is the scientific name of the SNP; *chromosome*, the name of the chromosome; the base pair *position* on the chromosome where the SNP occurs; *allele1*, which identifies which allele (DNA base) was detected on one chromosome, and *allele2*, which identifies the DNA base on the other chromosome.

Before the SNP data can be imported into the R statistical programming environment for further description and analysis, the file was opened with TextEdit, a text editor for Apple, Inc. Macintosh computers, and the comments were deleted such that the variable names separated by tabs formed the first line (row) of the file. The file was saved to disk with the name my_ancestryDNA.txt.

The following code chunk imports the AncestryDNA raw data into an R dataframe, and the last line of code tells R to print out the first two rows of the dataframe.

```
# import AncestryDNA raw SNP alleles as a dataframe, specifying all variables as
# character variables except position, which is an integer
snps_ancestryDNA <- data.frame(read_tsv("my_ancestryDNA.txt", col_names = TRUE,
                                        col_types = "ccicc"))
#print first two rows of dataframe, show variable names as column headings
snps_ancestryDNA[1:2,]
```

```
##        rsid chromosome position allele1 allele2
## 1 rs4477212          1    82154       T       T
## 2 rs3131972          1   752721       A       G
```

It was desirable to combine the values for *allele1* and *allele2* into a variable called *genotype*. It was also desirable to delete the variables *allele1* and *allele2* so that the dataframe for AncestryDNA matched that for 23AndMe in terms of the number of variables and the names of those variables.

```
# combine variables
snps_ancestryDNA$genotype <- paste(snps_ancestryDNA$allele1,snps_ancestryDNA$allele2,sep="")
# delete unneeded variables
snps_ancestryDNA <- snps_ancestryDNA[,c(1:3,6)]
# print first two rows of dataframe, show variable names as column headings
snps_ancestryDNA[1:2,]
```

```
##        rsid chromosome position genotype
## 1 rs4477212          1    82154       TT
## 2 rs3131972          1   752721       AG
```

The naming conventions used for chromosomes was reviewed for the AncestryDNA raw data file. The following code chunk shows the names of chromosomes reported on by AncestryDNA.

```
# print one instance of each name of each chromosome in the dataframe
unique(snps_ancestryDNA$chromosome)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "23" "24" "25"
```

In addition to the conventional numbers 1 through 22 for autosomal chromosomes, AncestryDNA reports data for chromosomes numbered 23, 24 and 25. These last three numbers are often used to refer to X-chromosome SNPs, the pseudo-autosomal SNPS on X & Y chromosomes, and SNPs on the Y-chromosome. The names of SNP associated with chromosomes 23, 24 and 25 were inspected for the corresponding chromosome name in the 23AndMe dataframe, and were re-coded as X, Y and X, respectively by the code chunk below.

```
# rename chromosomes 23, 24 and 25
snps_ancestryDNA$chromosome[snps_ancestryDNA$chromosome == 23] <- "X"
snps_ancestryDNA$chromosome[snps_ancestryDNA$chromosome == 24] <- "Y"
snps_ancestryDNA$chromosome[snps_ancestryDNA$chromosome == 25] <- "X"
```

The results of renaming chromosomes 23, 24 and 25 was reviewed as shown in the code chunk below.

```
# print one instance of each name of each chromosome in the dataframe
unique(snps_ancestryDNA$chromosome)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "X"  "Y"
```

Family Tree DNA—The file downloaded from Family Tree DNA contained no comments section, unlike the files downloaded from 23AndMe and AncestryDNA. The file was opened with a modern version of Microsoft Excel, but other software could have been used. The first line of the .csv file contained the the names of four variables: *RSID*, which is the scientific name of the SNP, *CHROMOSOME*, which is the name of the chromosome, *POSITION*, the base pair on the chromosome where the SNP is located, and *RESULT*, which contains the alleles (DNA bases) detected for the SNP. The file was closed without changes. The file was renamed my_ftdna.csv and closed without modification. The following code chunk reads the raw data into R.

```
# import Family Tree DNA raw SNP alleles as a dataframe, specifying all variables as
# character variables except position, which is an integer
snps_ftdna <- data.frame(read_csv("my_ftdna.csv", col_names = TRUE,
                                  col_types = "ccic"))
```

```
## Warning: 1 parsing failure.
##    row      col   expected   actual
## 707270 POSITION an integer POSITION
```

The parsing failure on row 707270 while importing the data from Family Tree DNA is due to a row of variable names that precedes the X-chromosome data in the file. It was appropriate therefore to delete that one row before further description and analysis. It was also useful to rename the variables (variable names are case sensitive in R) so they have the same names as the variables associated with the 23AndMe data.

```
# delete extra row containing variable names
snps_ftdna <- snps_ftdna[-707270,]
# rename variables
colnames(snps_ftdna) <- c("rsid","chromosome","position","genotype")
# print first two rows of dataframe, show variable names as column headings
snps_ftdna[1:2,]
```

```
##        rsid chromosome position genotype
## 1 rs3094315          1   752566       AG
```

```
## 2 rs3131972          1   752721       AG
```

The naming conventions for chromosomes was reviewed. The following code chunk tells R to print a list of chromosome names for Family Tree DNA.

```
# print one instance of each name of each chromosome in the dataframe
unique(snps_ftdna$chromosome)
```

```
##  [1] "1"  "2"  "0"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13"
## [15] "14" "15" "16" "17" "18" "19" "20" "21" "22" "X"
```

The value zero is included in the list of chromosome names. It is often the case that SNPs which do not have known precise locations are assigned to a chromosome 0. There were 15 SNPs attributed to chromosome zero by Family Tree DNA. For the purposes of this document, the chromosome zero SNPs were removed from the dataframe for Family Tree DNA.

```
# delete the SNPs attributed to chromosome zero for Family Tree DNA
snps_ftdna <- filter(snps_ftdna, chromosome != "0")
# print revised list of chromosomes
unique(snps_ftdna$chromosome)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "X"
```

Exploratory Data Analysis

Number of SNPs Reported—Having imported and processed the raw data downloaded from vendor web sites so that there was one R dataframe object for each vendor the three dataframes created as described above were then compared.

Vendor	Number of SNPs
23AndMe	991791
AncestryDNA	701478
Family Tree DNA	725276

Table 2. A comparison of the number of SNPs reported by each vendor in the downloaded raw data. It should be noted in the case of 23AndMe, the author had ordered autosomal DNA tests twice in different years when different gene chips were in use. The data downloaded from 23AndMe reflects this history.

As can be seen in Table 2 above, 23AndMe reported on many more SNPs that the other vendors did. This is primarily due to the health research emphasis at 23AndMe. For example, the dataframe for 23AndMe contained 991791 rows of data, one row per SNP, compared to AncestryDNA's 701478 rows of SNPs. Vendors do change their technology over time and it may be the case that other customers have different numbers of SNPs in their raw data files than are reported here for the author's data.

Next, the SNP lists for vendors were compared with regard to reporting X-chromosome SNPs.

Vendor	Number of X-chromosome SNPs
23AndMe	26770
AncestryDNA	18044
Family Tree DNA	18022

Table 3. A listing of the number of SNPs on the X-chromosome reported by vendors in downloads of autosomal DNA test results.

Next, the SNP lists for vendors were compared with regard to reporting Y-chromosome SNPs. Family Tree DNA was the only vendor that does not include Y-chromosome SNPs in the raw data. Family Tree DNA offers an extensive line of Y-chromosome SNP tests as separate products.

Vendor	Number of Y-chromosome SNPs
23AndMe	3092
AncestryDNA	885
Family Tree DNA	0

Table 4. A listing of the number of SNPs on the Y-chromosome reported by vendors in downloads of autosomal DNA test results.

Next, the SNP lists for vendors were compared with regard to reporting mitochondrial SNPs. AncestryDNA reported no mitochondrial SNPs. Family Tree DNA also reports no mitochondrial SNPs, however it should be noted that Family Tree offers additional products to analyze mitochondrial DNA including full sequencing of mitochondrial DNA.

Vendor	Number of mitochondrial SNPs
23AndMe	2678
AncestryDNA	0
Family Tree DNA	0

Table 5. A listing of the number of mitochondrial SNPs reported by vendors in downloads of autosomal DNA test results.

Based on the exploratory analysis performed above, it was desirable to exclude all of the Y-chromosome and mitochondrial SNPs for each vendor before making additional comparisons between vendors. This process will result in a dataframe for each vendor containing SNPs for all the autosomes plus the X-chromosome SNPs. Autosomal SNPs and X-chromosome SNPs can be used to find autosomal and X-chromosome matches between customers of vendors, if the vendor provides tools to support such matching, or 3rd party tools such as GEDmatch are used to find matches when vendor-supplied tools are lacking.

```
# remove Y and MT SNPs and save rsid values
# use this information in the following tables
# 23AndMe:
rsid_23andme <- filter(snps_23AndMe, chromosome != "Y",
                       chromosome != "MT") %>%
        select(rsid)

# AncestryDNA:
rsid_ancestryDNA  <- filter(snps_ancestryDNA, chromosome != "Y",
                            chromosome != "MT")  %>%
        select(rsid)

# FamilyTree DNA
rsid_ftdna <- filter(snps_ftdna, chromosome != "Y",
                     chromosome != "MT") %>%
        select(rsid)

# use this information later in the article:
# 23AndMe:
small_23Andme <- filter(snps_23AndMe, chromosome != "Y",
                        chromosome != "MT") %>%
```

```
        select(rsid,genotype)
colnames(small_23Andme) <- c("rsid","geno23")

# AncestryDNA:
small_ancestryDNA <- filter(snps_ancestryDNA, chromosome != "Y",
                            chromosome != "MT")  %>%
        select(rsid,genotype)
colnames(small_ancestryDNA) <- c("rsid","genoAn")

# Family Tree DNA:
small_ftdna <- filter(snps_ftdna, chromosome != "Y",
                      chromosome != "MT") %>%
        select(rsid,genotype)
colnames(small_ftdna) <- c("rsid","genoFt")

# create dataframes for pairs of vendors with SNPs in common
common_snps_23AndMe_ftdna_geno <- Reduce(function(x,y) merge(x,y,all=FALSE),
                                  list(small_23Andme, small_ftdna))
common_23AndMe_ancestryDNA_geno <- Reduce(function(x,y) merge(x,y,all=FALSE),
                                  list(small_23Andme, small_ancestryDNA))

common_ancestryDNA_ftdna <- Reduce(function(x,y) merge(x,y,all=FALSE),
                                  list(small_ancestryDNA,small_ftdna))

# create one dataframe with SNPs in common for all three vendors
# use this information in text immediately below
common_snps_all_genotypes <- Reduce(function(x,y) merge(x,y,all=FALSE),
                                  list(small_23Andme,small_ancestryDNA,
                                       small_ftdna))
```

The names of SNPs reported by each vendor were compared. When the SNPs of all three vendors were examined, 684859 were common to all three vendors. In Table 6 below, the number of SNPs in common between pairs of vendors is shown.

In Common with:	AncestryDNA	Family Tree DNA
SNPs at 23AndMe	687299	709916

In Common with:	23AndMe	Family Tree DNA
SNPs at AncestryDNA	687299	698118

In Common with:	23AndMe	AncestryDNA
SNPs at Family Tree DNA	709916	698118

Table 6. The number of SNPs in common between direct to consumer DNA testing for ancestry vendors offering autosomal DNA testing. These comparisons include autosomal chromosomes and X-chromosome SNPs.

Next, the following table shows the number of differences in SNPs between pairs of vendors.

Different from:	AncestryDNA	Family Tree DNA
SNPs at 23AndMe	298722	276105

Different from:	23AndMe	Family Tree DNA
SNPs at AncestryDNA	13294	2475

Different from:	23AndMe	AncestryDNA
SNPs at Family Tree DNA	15360	27158

Table 7. The number of SNPs that are different at pairs of vendors. These comparisons include autosomal chromosomes and X-chromosome SNPs. Read these entries as, for example (middle), AncestryDNA reported 13294 SNPs that 23AndMe did not report, and AncestryDNA reported 2475 SNPs that were not reported at Family Tree DNA.

The following code chunk prints out a table showing the number of SNPs reported by each vendor for each chromosome excluding the Y-chromosome and mitochondrial DNA.

```
# for each vendor:
# filter out Y and mitochondrial SNPs
# summarize the number of SNPs by chromosomes
# 23AndMe:
autoXcount23AndMe <- filter(snps_23AndMe, chromosome != "Y",
                        chromosome != "MT") %>%
        group_by(chromosome) %>%
        summarise(snpc_23AndMe = length(rsid))
# AncestryDNA:
autoXcountAncestryDNA <- filter(snps_ancestryDNA, chromosome != "Y",
                        chromosome != "MT") %>%
        group_by(chromosome) %>%
        summarise(snpc_anDNA = length(rsid))
# Family Tree DNA:
autoXcountFtdna <- filter(snps_ftdna, chromosome != "Y",
                        chromosome != "MT") %>%
        group_by(chromosome) %>%
        summarise(snpc_ftdna = length(rsid))
# combine count lists, clean up table headings
autoXThreeVendors <- bind_cols(autoXcount23AndMe, autoXcountAncestryDNA,autoXcountFtdna )
autoXThreeVendors <- autoXThreeVendors[,c(1,2,4,6)]
# sort a mixed variable, print summary showing the number of SNPs per chromosome
options(tibble.print_max = Inf)
autoXThreeVendors[mixedorder(autoXThreeVendors$chromosome),]
```

```
## # A tibble: 23 × 4
##    chromosome snpc_23AndMe snpc_anDNA snpc_ftdna
##         <chr>        <int>      <int>      <int>
## 1           1        79071      57267      59310
## 2           2        79682      55972      57780
## 3           3        65161      45769      47299
## 4           4        56736      39105      40526
## 5           5        57802      40899      42169
```

```
## 6           6        65100      46134      48272
## 7           7        52561      36681      38215
## 8           8        50816      35718      37086
## 9           9        44335      31838      32878
## 10         10        51869      37867      39160
## 11         11        49508      35412      36704
## 12         12        48622      34348      35618
## 13         13        37209      26965      27912
## 14         14        31757      22615      23377
## 15         15        29258      21010      21702
## 16         16        31112      22013      22822
## 17         17        27586      19633      20296
## 18         18        28866      21072      21765
## 19         19        19205      14406      15138
## 20         20        24560      17876      18461
## 21         21        13810       9940      10257
## 22         22        14625      10009      10507
## 23          X        26770      18044      18022
```

Description of Reported Genotypes—The following code chunk prints out one instance of each genotype reported in the dataframe for each vendor. The numbers preceding the genotypes are row numbers of the first instance of a particular genotype, and can be ignored.

```
# show the unique geneotypes reported by each vendor
# 23AndMe:
geno23AndMe <- unique(filter(snps_23AndMe, chromosome != "Y",
                             chromosome != "MT") %>%
                              select(genotype))
geno23AndMe
```

```
##        genotype
## 1            AA
## 2            AG
## 4            GG
## 7            CC
## 9            CT
## 12           GT
## 24           TT
## 176          AC
## 233          --
## 350          CG
## 3921         AT
## 9961         II
## 24140        DD
## 51706        DI
## 959648        G
## 959650        C
## 959651        T
## 959652        A
## 961186        I
## 967446        D
```

```
# AncestryDNA:
genoAncestryDNA <- unique(filter(snps_ancestryDNA, chromosome != "Y",
                                 chromosome != "MT") %>%
                                  select(genotype))
```

```
genoAncestryDNA
```

```
##        genotype
## 1            TT
## 2            AG
## 3            GG
## 5            CC
## 6            AA
## 7            TC
## 10           TG
## 45           OO
## 210          AC
## 700159       CG
## 700195       GC
## 700361       TA
```

```
# Family Tree DNA:
genoftdna <- unique(filter(snps_ftdna, chromosome != "Y",
                           chromosome != "MT") %>%
                            select(genotype))
genoftdna
```

```
##       genotype
## 1           AG
## 3           GG
## 6           CC
## 7           AA
## 8           TC
## 11          TG
## 13          TT
## 229         AC
## 306         GC
## 4863        --
## 6931        CG
## 7197        AT
## 11285       TA
```

Looking at the different reported SNP genotypes, 23AndMe reports the most different SNP genotypes, and AncestryDNA reports the fewest different genotypes. Family Tree DNA provides an explanation of Illumina OmniExpress gene chip allele results and a guide to reading their raw data files[7]. When a vendor reports a genotype as "- -", this means the DNA bases for that SNP could not be called with the required level of confidence. When 23AndMe reports a D, this means a deletion was detected for that particular SNP. When 23AndMe reports an I, this indicates an insertion of one or more DNA bases were present for that SNP. 23AndMe reports the geneotype for X-chromosome SNPs from an male as a single base. In contrast, Family Tree DNA and Ancestry DNA reported the X-chromosome SNPs as having two identical bases, when in reality, there is only one X-chromosome for the male sample the author provided. AncestryDNA also reports a genotype of OO, the meaning of which was not apparent from Google searches.

Vendor	Number of No-call SNP genotypes
23AndMe	2467
AncestryDNA	0
Family Tree DNA	687

[7]Family Tree genotype codes and reading instructions link

Table 8. A listing of the number of no-call genotypes by vendor, where "–" was used as a no-call indicator.

Comparison of SNP Genotypes Between Vendors—Ideally, all vendors should report the same results, with perhaps the exception of no-calls, which may vary depending on sample quality, gene chip performance and software tuning factors. When looking at genotypes, two factors must be considered: (1) the DNA strand used to report the DNA base for the SNP, and (2) the order of bases reported.

With regard to strand, vendors can choose to provide a genotype based on the DNA bases present on either of the two DNA strands. Each is correct. For example, if one vendor reports a genotype of AA for a particular SNP, another vendor may report the bases on the opposite complimentary strand, which in this case would be TT. In this context, the AA and TT genotypes are the same, they are simply reported using bases on the opposite strand.

With regard to base order, the raw genotype data can be reported with the alleles in either order. For example, a SNP genotype of GT is the same as TG. The vendor simply reported the alleles in a different order. The following short listing of SNPs and the genotypes each vendor reported shows no instances of strand-switched reporting, but does contain two examples in rows 6 and 8 showing 23AndMe reported the alleles in a different order compared to the other vendors.

The following code chunk executes a very inefficient loop calling the function areSame(), which contains many comparisons between vendor genotypes, once for each of the 684859 rows in a dataframe. The result of each call to areSame() is stored as a TRUE or FALSE value in the dataframe in a new variable called isSame. The number of SNPs reported with different genotypes is reported, as is a table showing all the SNPs with genotypes that were called differently by the vendors.

```
# example, first ten rows of genotypes reported by 23AndMe (geno23),
# AncestryDNA (genoAn) and Family Tree DNA (genoFt)
common_snps_all_genotypes[1:10,]
```

```
##          rsid geno23 genoAn genoFt
## 1   rs1000000     AG     AG     AG
## 2  rs10000023     TT     TT     TT
## 3   rs1000003     AA     AA     AA
## 4  rs10000030     GG     GG     GG
## 5  rs10000037     AG     AG     AG
## 6  rs10000041     GT     TG     TG
## 7  rs10000049     AA     AA     AA
## 8   rs1000007     CT     TC     TC
## 9  rs10000073     TT     TT     TT
## 10 rs10000081     TT     TT     TT
```

```
# matching:
# AA or TT = (AA AA) (TT TT) (AA TT) (TT AA)     # OK
# GG or CC = (GG GG) (CC CC) (GG CC) (CC GG)     # OK
# TA or AT = (AT AT) (TA TA) (AT TA) (TA AT)     # OK
# AG or GA = (AG AG) (TC TC) (AG TC) (TC AG)     # OK
# ----
# GT or TG = (GT GT) (TG TG) (GT TG) (GT TG)     # OK
# CT or TC = (CT CT) (TC TC) (CT TC) (TC CT)     # OK
# AC or CA = (AC AC) (CA CA) (AC CA) (CA AC)     # OK
# a function to implement match comparisons:

areSame <- function(geno1,matchAlleles){
        # a function to compare three sets of genotypes
        # takes a character string geno1, and a list of length 2
        # returns logical TRUE or FALSE; temporarily returns NA for certain unhandled cases
        if(nchar(geno1) != 2 ){
```

```
        # convert single letter genotypes to double-letter homozygous genotypes
        geno1 <- paste(geno1,geno1,sep="")
        # allow continuation of function to proceed
}
if(nchar(matchAlleles) != 4){
        # unhandled situations so far
        return(NA)
}
# AA or TT = (AA AA) (TT TT) (AA TT) (TT AA)
if(geno1 == "AA" | geno1 == "TT"){
        if(matchAlleles == "AAAA" | matchAlleles == "TTTT" |
           matchAlleles == "AATT" |  matchAlleles == "TTAA"){
                return(TRUE)
        }
        else{
                return(FALSE)
        }
}
# GG or CC = (GG GG) (CC CC) (GG CC) (CC GG)
if(geno1 == "GG" | geno1 == "CC"){
        if(matchAlleles == "GGGG" | matchAlleles == "CCCC" |
           matchAlleles == "GGCC" |  matchAlleles == "CCGG"){
                return(TRUE)
        }
        else{
                return(FALSE)
        }
}
# TA or AT = (AT AT) (TA TA) (AT TA) (TA AT)
if(geno1 == "TA" | geno1 == "AT"){
        if(matchAlleles == "ATAT" | matchAlleles == "TATA" |
           matchAlleles == "ATTA" |  matchAlleles == "TAAT"){
                return(TRUE)
        }
        else{
                return(FALSE)
        }
}
# AG or GA = (AG AG) (TC TC) (AG TC) (TC AG)
if(geno1 == "AG" | geno1 == "GA"){
        if(matchAlleles == "AGAG" | matchAlleles == "TCTC" |
           matchAlleles == "AGTC" |  matchAlleles == "TCAG"){
                return(TRUE)
        }
        else{
                return(FALSE)
        }
}
# GT or TG = (GT GT) (TG TG) (GT TG) (GT TG)
if(geno1 == "GT" | geno1 == "TG"){
        if(matchAlleles == "GTGT" | matchAlleles == "TGTG" |
           matchAlleles == "GTTG" |  matchAlleles == "GTTG"){
                return(TRUE)
```

```
                }
                else{
                        return(FALSE)
                }
        }
        # CT or TC = (CT CT) (TC TC) (CT TC) (TC CT)
        if(geno1 == "CT" | geno1 == "TC"){
                if(matchAlleles == "CTCT" | matchAlleles == "TCTC" |
                   matchAlleles == "CTTC" |  matchAlleles == "TCCT"){
                        return(TRUE)
                }
                else{
                        return(FALSE)
                }
        }
        # AC or CA = (AC AC) (CA CA) (AC CA) (CA AC)
        if(geno1 == "AC" | geno1 == "CA"){
                if(matchAlleles == "ACAC" | matchAlleles == "CACA" |
                   matchAlleles == "ACCA" |  matchAlleles == "CAAC"){
                        return(TRUE)
                }
                else{
                        return(FALSE)
                }
        }
        return(NA)

}

# run areSame on the three reports of genotyping; this loop is very slow to run
# the system.time() funciton will report the number of seconds it takes to run the loop
system.time(for(i in 1:nrow(common_snps_all_genotypes)){
        common_snps_all_genotypes$isSame[i] <- areSame(common_snps_all_genotypes$geno23[i],
                                                       paste(common_snps_all_genotypes$genoAn[i],
                                                             common_snps_all_genotypes$genoFt[i],
                                                             sep=""))
})
```

```
##     user   system  elapsed
## 1052.356  576.935 1668.360
```

```
print(paste("Excluding no-call genotypes, there were ",
            nrow(filter(common_snps_all_genotypes,
                        isSame == FALSE,
                        genoAn != "00",
                        genoFt != "--",
                        geno23 != "--")),
            " SNP genotypes that were called differently.",sep=""))
```

```
## [1] "Excluding no-call genotypes, there were 40 SNP genotypes that were called differently."
```

```
# a table of SNPs called differently by the vendors
filter(common_snps_all_genotypes,
                        isSame == FALSE,
                        genoAn != "00",
```

```
                      genoFt != "--",
                      geno23 != "--")
```

```
##          rsid geno23 genoAn genoFt isSame
## 1   rs1034009     CC     CC     AC  FALSE
## 2  rs10440882     CT     CC     CC  FALSE
## 3  rs11062619     GG     AG     AG  FALSE
## 4  rs11241755     AG     GG     GG  FALSE
## 5  rs11582478     GG     TG     TG  FALSE
## 6  rs11665831     CT     TT     TT  FALSE
## 7  rs11706310     GG     AG     GG  FALSE
## 8  rs11887432     TT     TC     TT  FALSE
## 9  rs11984341     CT     TT     TT  FALSE
## 10 rs12081621     CC     CC     TC  FALSE
## 11 rs12158884     AA     AA     AG  FALSE
## 12 rs12464824     CC     CC     TC  FALSE
## 13 rs12927146     CC     CC     TC  FALSE
## 14 rs13414624     AC     CC     CC  FALSE
## 15  rs1551078      C     CC     TC  FALSE
## 16  rs1559579     AG     GG     AG  FALSE
## 17  rs1562893     CT     CC     CC  FALSE
## 18 rs16906634     GG     GG     AG  FALSE
## 19   rs176461     GG     AG     AG  FALSE
## 20  rs1913606     AG     GG     GG  FALSE
## 21  rs2316280     CC     TC     CC  FALSE
## 22  rs2398397     GG     GG     AG  FALSE
## 23  rs2604259     AA     AA     AG  FALSE
## 24 rs28990969     GG     GG     TG  FALSE
## 25   rs299881     CT     CC     CC  FALSE
## 26  rs3130801     AA     AA     AG  FALSE
## 27   rs361359     AC     AC     TG  FALSE
## 28  rs4140483     AG     GG     GG  FALSE
## 29  rs4902843     CC     CC     AC  FALSE
## 30  rs4976858     TT     TT     TG  FALSE
## 31  rs5904558      T     TG     TG  FALSE
## 32  rs6664362     CC     CC     TC  FALSE
## 33  rs7127129     AG     AG     GG  FALSE
## 34  rs7211084     CC     CC     TC  FALSE
## 35  rs7366689     CC     CC     TC  FALSE
## 36  rs7909419     GT     TT     TT  FALSE
## 37  rs7912364     AA     AC     AA  FALSE
## 38  rs7946005     CC     CC     TC  FALSE
## 39   rs872610     GG     AG     GG  FALSE
## 40   rs881711     TT     TC     TT  FALSE
```

Discussion

The results reported in this document are specific for the author's experience with autosomal DNA testing at the vendors included in this report. Vendors do change technology from time to time, and the number of SNPs tested and the identity of SNPs tested can vary for a single vendor and among vendors over time. Use these results as a guide.

As shown in Tables 2, 3, 4 and 5, vendors analyzed different numbers of SNPs on the autosomes and

the X-chromosome. One vendor, 23AndMe, reported SNPs for the Y-chromosome and for mitochondrial chromosome. One vendor, Family Tree DNA, reported no SNPs for the Y-chromosome or the mitochondrial chromosome. This choice by Family Tree DNA probably reflects the fact that the company sells an extensive line of Y-chromosome SNP tests as separate products and it also offers mitochondrial DNA analysis testing separately.

In this study, there were 684859 SNPs that each of the three vendors analyzed and reported on. Excluding the no-call genotypes, of these SNPs, only 40 SNPs were called differently. This amounts to agreement on individual SNP calls of approximately 99.994 percent, which is pretty impressive!

Computing Environment

Hardware & software	Version	Use
Apple, Inc. iMac computer	Mid 2015	hardware
OSX	10.11.6	operating system
RStudio[8]	1.0.44	authoring Rmarkdown & R programming
R[9]	3.3.2	R programming environment
dplyr package[10]	0.5.0	dataframe manipulation in R
readr package[11]	1.0.0	read data into R
knitr package[12]	1.15.1	authoring Rmarkdown
gtools package[13]	3.5.0	sorting on mixed data
MacTEX[14]	20161009	render Rmarkdown to pdf
TextEdit	1.11 (325)	examine and remove comments in files
Microsoft Excel for Mac	15.29.1 (161215)	preview .csv file

[8] RStudio
[9] R-project.org
[10] dplyr package on CRAN
[11] readr package on CRAN
[12] knitr package on CRAN
[13] gtools package on CRAN
[14] MacTEX