Script to convert from Onemap (map file and marker genotype file) to R/qtl .csv file

Before you can run this script, you need to upload your files in the correct format. A) You need a map file from Onemap. After determining the appropriate LOD score and ordering markers within each linkage group, you will create a map file including all linkage groups and marker order: ex. map11 <- list(lg1, lg2, lg3, lg4) You can now export the map file, including the parental linkage phase information, using: parents_haplotypes(list(lg1_final, lg2_final, lg3), out_file = "Fam11_haplotypes_final.txt", group_names=c("chr1", "chr2","chr3")) Read this file into R

```
map <- read.table("Fam11_haplotypes_final.txt", header=TRUE)
names(map) <- c("Chr", "MarkerNumb", "MarkerNames","Position","Parent1_1","Parent1_2","Parent2_1","Pare
head(map)
```

```
##    Chr MarkerNumb          MarkerNames Position Parent1_1 Parent1_2 Parent2_1
## 1 chr1         33 NC_035780.1_4528456      0.00         a         a         a
## 2 chr1          8 NC_035780.1_2493214      2.21         a         a         a
## 3 chr1         20 NC_035780.1_1818622      3.15         a         b         b
## 4 chr1         24 NC_035780.1_2768349      7.50         a         b         a
## 5 chr1         29 NC_035780.1_3964420      9.30         a         b         a
## 6 chr1         46 NC_035780.1_7203389     12.54         b         a         b
##   Parent2_2
## 1         b
## 2         b
## 3         a
## 4         a
## 5         b
## 6         a
```

Wrangle the data so you have Chr, Marker, Position, Parent1, Parent2

```
map$MarkerNumb = NULL                # Remove this column
colnames(map)[2] <- "Marker"         # Rename Marker column
map$Chr <- substr(map$Chr, 4,7)      # Remove 'chr' from Chr column so values are just numbers

# Combine parental genotypes into two columns, one for Parent 1 and one for Parent 2
library(tidyr)
map <- transform(map, Parent1=paste(Parent1_1, Parent1_2, sep=""))
map <- transform(map, Parent2=paste(Parent2_1, Parent2_2, sep=""))
map[,4:7] = NULL
head(map)
```

```
##   Chr               Marker Position Parent1 Parent2
## 1   1 NC_035780.1_4528456      0.00      aa      ab
## 2   1 NC_035780.1_2493214      2.21      aa      ab
## 3   1 NC_035780.1_1818622      3.15      ab      ba
## 4   1 NC_035780.1_2768349      7.50      ab      aa
## 5   1 NC_035780.1_3964420      9.30      ab      ab
## 6   1 NC_035780.1_7203389     12.54      ba      ba
```

B) The genotype file: includes the parental cross type for each marker and the genotype of the marker for every individual. This file was written out (i.e. write_onemap_raw(Fam11_bins, file.name = "Fam11.raw", cross="outcross")) after the vcf file was loaded into Onemap and redundant markers were removed by grouping markers into bins (see Outcrossing Populations Onemap tutorial on cran)

- Need to change file format to .txt in order to read in.

```
geno <- read.table("Fam11.txt", header=T)
geno[1:5,1:7]
```

```
##                Marker Cross Fam11_01 Fam11_02 Fam11_03D Fam11_03 Fam11_04
## 1  *NC_035780.1_82578 D2.15        a        a        ab       ab        a
## 2  *NC_035780.1_94761 D2.15        a        a        ab       ab        a
## 3 *NC_035780.1_147311  B3.7        a       ab        ab       ab        a
## 4 *NC_035780.1_595461 D2.15       ab       ab         a        a       ab
## 5 *NC_035780.1_291218 D1.10        a       ab         a        a        a
```

Onemap inserts a '*' next to every marker in the genotype file, remove this.

```
geno$Marker = as.character(factor(gsub("\\*", "", geno$Marker)))
```

Join the map and genotype file together based on markers present in the map file.

```
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
```

```
comb <- merge(map, geno, by='Marker')
comb[1:5,1:8]
```

```
##                 Marker Chr Position Parent1 Parent2 Cross Fam11_01 Fam11_02
## 1 NC_035780.1_11034723   9    17.05      aa      ba D2.15        a       ab
## 2 NC_035780.1_11034723  10    17.05      aa      ba D2.15        a       ab
## 3 NC_035780.1_11034723   4    17.05      aa      ba D2.15        a       ab
## 4 NC_035780.1_11034723   6    17.05      aa      ba D2.15        a       ab
## 5 NC_035780.1_11034723   7    17.05      aa      ba D2.15        a       ab
```

Now, you can edit the script below to change your marker genotype information into correct codings for Rqtl. (See read.cross() to determine correct codes.)

This script uses the case_when() function in tidyverse. Ex. If the cross is 'B3.7' and the parents are 'ab' and 'ab' at that marker, and the sample marker genotype is 'a', change to 1.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------- tid
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------- tidyverse
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
comb2 <- comb %>% mutate_at(
  vars(starts_with('Fam11_')), # all the columns you want to change start in a similar manner, in my ca
  funs(case_when(
       Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ab' & . == 'a' ~ 1,        #B3.7abab
       Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ab' & . == 'ab' | . == 'ba' ~ 10,
       Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ab' & . == 'b' ~ 4,
       Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ba' & . == 'a' ~ 3,        #B3.7abba
```

```
      Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ba' & . == 'ab' | . == 'ba' ~ 9,
      Cross == 'B3.7' & Parent1 == 'ab' & Parent2 == 'ba' & . == 'b' ~ 6,
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ab' & . == 'a' ~ 6,        #B3.7baab
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ab' & . == 'ab' | . == 'ba' ~ 9,
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ab' & . == 'b' ~ 3,
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ba' & . == 'a' ~ 3,        #B3.7baba
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ba' & . == 'ab' | . == 'ba' ~ 9,
      Cross == 'B3.7' & Parent1 == 'ba' & Parent2 == 'ba' & . == 'b' ~ 6,
      Cross == 'D1.10' & Parent1 == 'ab' & Parent2 == 'aa' & . == 'a' ~ 5,        #D1.10abaa
      Cross == 'D1.10' & Parent1 == 'ab' & Parent2 == 'aa' & . == 'ab' | . == 'ba' ~ 6,
      Cross == 'D1.10' & Parent1 == 'ba' & Parent2 == 'aa' & . == 'a' ~ 6,        #D1.10baaa
      Cross == 'D1.10' & Parent1 == 'ba' & Parent2 == 'aa' & . == 'ab' | . == 'ba' ~ 5,
      Cross == 'D1.10' & Parent1 == 'aa' & Parent2 == 'ab' & . == 'a' ~ 7,        #D1.10aaab
      Cross == 'D1.10' & Parent1 == 'aa' & Parent2 == 'ab' & . == 'ab' | . == 'ba' ~ 8,
      Cross == 'D1.10' & Parent1 == 'aa' & Parent2 == 'ba' & . == 'a' ~ 8,        #D1.10aaab
      Cross == 'D1.10' & Parent1 == 'aa' & Parent2 == 'ba' & . == 'ab' | . == 'ba' ~ 7,
      Cross == 'D2.15' & Parent1 == 'aa' & Parent2 == 'ab' & . == 'a' ~ 7,        #D2.15aaab
      Cross == 'D2.15' & Parent1 == 'aa' & Parent2 == 'ab' & . == 'ab' | . == 'ba' ~ 8,
      Cross == 'D2.15' & Parent1 == 'aa' & Parent2 == 'ba' & . == 'a' ~ 8,        #D2.15aaba
      Cross == 'D2.15' & Parent1 == 'aa' & Parent2 == 'ba' & . == 'ab' | . == 'ba' ~ 7,
      Cross == 'D2.15' & Parent1 == 'ab' & Parent2 == 'aa' & . == 'a' ~ 5,        #D2.15abaa
      Cross == 'D2.15' & Parent1 == 'ab' & Parent2 == 'aa' & . == 'ab' | . == 'ba' ~ 6,
      Cross == 'D2.15' & Parent1 == 'ba' & Parent2 == 'aa' & . == 'a' ~ 6,        #D2.15baaa
      Cross == 'D2.15' & Parent1 == 'ba' & Parent2 == 'aa' & . == 'ab' | . == 'ba' ~ 5)))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```
# - = missing data => coded as NA
comb2[1:5,1:7]
```

```
##                 Marker Chr Position Parent1 Parent2 Cross Fam11_01
## 1 NC_035780.1_11034723   9    17.05      aa      ba D2.15        8
## 2 NC_035780.1_11034723  10    17.05      aa      ba D2.15        8
## 3 NC_035780.1_11034723   4    17.05      aa      ba D2.15        8
## 4 NC_035780.1_11034723   6    17.05      aa      ba D2.15        8
## 5 NC_035780.1_11034723   7    17.05      aa      ba D2.15        8
```

Need to order each linkage group by position. Use tidyverse filter() to separate each linkage group into its own data frame and then order the markers in ascending order, starting at 0 cM.

```
chr1 <- comb2 %>% filter(Chr == 1) %>% arrange(Position)
chr2 <- comb2 %>% filter(Chr == 2) %>% arrange(Position)
chr3 <- comb2 %>% filter(Chr == 3) %>% arrange(Position)
chr4 <- comb2 %>% filter(Chr == 4) %>% arrange(Position)
chr5A <- comb2 %>% filter(Chr == '5A') %>% arrange(Position)
```

```
chr5B <- comb2 %>% filter(Chr == '5B') %>% arrange(Position)
chr6 <- comb2 %>% filter(Chr == 6) %>% arrange(Position)
chr7 <- comb2 %>% filter(Chr == 7) %>% arrange(Position)
chr8 <- comb2 %>% filter(Chr == 8) %>% arrange(Position)
chr9 <- comb2 %>% filter(Chr == 9) %>% arrange(Position)
chr10 <- comb2 %>% filter(Chr == 10) %>% arrange(Position)
```

Combine all linkage groups in preferred order.

```
map11 <- rbind(chr1, chr2, chr3, chr4, chr5A, chr5B, chr6, chr7, chr8, chr9, chr10)
map11[1:5,1:8]
```

```
##                 Marker Chr Position Parent1 Parent2 Cross Fam11_01 Fam11_02
## 1 NC_035780.1_11034723   1    17.05      aa      ba D2.15        8        7
## 2 NC_035780.1_61054953   1    20.93      aa      ab D2.15        8        7
## 3 NC_035780.1_61021676   1    21.91      aa      ba D2.15        8        7
## 4 NC_035780.1_12067265   1    22.71      ba      ab  B3.7        3        6
## 5 NC_035780.1_20730105   1    24.18      ab      aa D1.10        5        6
```

Now, clean up file for R/qtl format. Remove Parent1, Parent2, and Cross.

```
map11[,4:6] = NULL
map11[1:5,1:7]
```

```
##                 Marker Chr Position Fam11_01 Fam11_02 Fam11_03D Fam11_03
## 1 NC_035780.1_11034723   1    17.05        8        7         7        7
## 2 NC_035780.1_61054953   1    20.93        8        7         7        7
## 3 NC_035780.1_61021676   1    21.91        8        7         7        7
## 4 NC_035780.1_12067265   1    22.71        3        6         9        9
## 5 NC_035780.1_20730105   1    24.18        5        6         5        5
```

Transpose the dataframe so Marker, Chr, and Position are columns and each individual is a row.

```
Tfam11 <- t(map11)
Fam11_T <- as.data.frame(Tfam11)
Fam11_T[1:5,1:3]
```

```
##                             V1                  V2                  V3
## Marker   NC_035780.1_11034723 NC_035780.1_61054953 NC_035780.1_61021676
## Chr                         1                   1                   1
## Position                17.05               20.93               21.91
## Fam11_01                    8                   8                   8
## Fam11_02                    7                   7                   7
```

Add in phenotype(s) as the first columns in the dataframe. Make sure they are in the same order as the individuals. Load in phenotype file. NOTE: phenotype file is blank for first 3 rows since it needs to match the transposed dataframe.

```
phen <- read.table("Fam11_phen.txt", header=T)
head(phen)
```

```
##   daydead surv DAY
## 1      NA   NA  NA
## 2      NA   NA  NA
## 3      NA   NA  NA
## 4       9    1   9
## 5      11    1  11
## 6      11    1  11
```

Insert phenotype(s) as first columns, and remove column and row headers.

```
Fam11_fin <- cbind(phen, Fam11_T)
colnames(Fam11_fin) = NULL
rownames(Fam11_fin) = NULL
Fam11_fin[1:5,1:6]
```

```
##
## 1 NA NA NA NC_035780.1_11034723 NC_035780.1_61054953 NC_035780.1_61021676
## 2 NA NA NA                    1                    1                    1
## 3 NA NA NA                17.05                20.93                21.91
## 4  9  1  9                    8                    8                    8
## 5 11  1 11                    7                    7                    7
```

Write file out in csv format.

```
write.csv(Fam11_fin, "Fam11_rqtl.csv")
```

(Might need to clean-up in a text editor or Excel. i.e. remove NAs from first 3 phenotype rows, remove blank column headers, remove row header column)

Ready to upload into Rqtl for mapping!