**Resumen metodológico del estudio del Genotipado de la colección de guisante (IAS)**

1. **Descripción de los datos recibidos de DiversityArray**

* Tras secuenciar nuestra colección de guisante por DArTSeq, DiversityArray analizo los datos brutos de secuenciación para identificar los marcadores moleculares.
* Diversity array ha analizado mis muestras junto con las de Eleonora lo que permitirá comparar sus resultados con los míos (gracias a los nombres que tienen en común).
* Diversity array ha analizado por separado lo que llama los SilicoDArT markers que son los fragmentos de ADN secuenciados (marcadores largos) y los SNP que han sido identificado en cada uno de estos SilicoDarT markers. Un mismo SilicoDarT puede contener varios SNP.
* Después compararon y alinearon nuestros datos con 3 bases de datos diferentes:

1. El genoma de referencia del guisante
2. Genome de referencia de *Medicago truncatula,* del garbanzo y genoma provisional del guisante (incompleto)
3. La base de datos nr y de los procariotas del NCBI para descartar contaminantes

* Al final nos mandaron los siguientes ficheros:
  + Report\_DPea19-4080\_1\_moreOrders\_SilicoDArT\_1.csv del 11/04/2019: SilicoDArT alineados con *Medicago truncatula* y *Cicer arietinum*.
  + Report\_DPea19-4080\_1\_moreOrders\_SilicoDArT\_1.csv del 27/02/2020: SilicoDArT alineados con el genoma de referencia del guisante.
  + DPea19\_4080\_1moreOrder\_SilicoDArTsBLAST.csv del 12/04/2019: resultado del BLAST de nuestros datos con las bases de datos del NCBI para detectar posibles contaminaciones.
  + Report\_DPea19-4080\_1\_moreOrders\_SNP\_2.csv del 27/2/2020: SNP markers alineados con *Medicago truncatula* y *Cicer arietinum* en “2-row format” es decir que cada SNP esta descrito en 2 líneas, 1 con la información del alelo de referencia (el principal) con presencia /ausencia score y otra con la información del alelo alternativo.
  + Report\_DPea19-4080\_1\_moreOrders\_SNP\_3.csv del 27/2/2020: SNP markers alineados con el genoma de referencia del guisante en “2-row format” es decir que cada SNP esta descrito en 2 líneas, 1 con la información del alelo de referencia (el principal) con presencia /ausencia score y otra con la información del alelo alternativo.
  + Report\_DPea19-4080\_1\_moreOrders\_SNP\_mapping\_2: SNP markers alineados con *Pisum sativum* descritas en una sola línea.
  + Report\_DPea19-4080\_1\_moreOrders\_SNP\_mapping\_2\_old: SNP markers alineados con *Medicago truncatula* y *Cicer arietinum* descritas en una sola línea.

1. **Limpieza de las bases de datos SilicoDArT y SNP**

* Como Diversity Array analizó mis secuencias junto con los datos de Eleonora, tengo que eliminar la información de sus genotipos antes de empezar el análisis.
* Además, hay que recalcular las estimaciones de calidad, call, PIC, etc. ya que han sido determinado con los datos conjuntos (Eleonora y mías).
* También sería bueno juntar las diferentes anotaciones en el mismo fichero para tener toda la información a mano.
* La limpieza de la base de datos se realiza con el programa R utilizando el script llamado Matrixpreparation.R.
* En primer lugar, he creado una carpeta para poner todos los ficheros intermedios generado durante la limpieza de la base de datos que he llamado “Database genotipado”.

**B.1. Limpieza de la base de Datos de marcadores Silico DArT**

* Para empezar limpio la base de datos de SilicoDArT. Para ello, he pegado los 3 ficheros .csv generado por Diversity Array sobre estos marcadores.
* Para diferenciar los ficheros les he renombrado así:
  + DPea19-4080\_SilicoDArT\_Pisum.csv: base de datos alineada con el genoma de referencia del guisante = Report\_DPea19-4080\_1\_moreOrders\_SilicoDArT\_1.csv del 27/02/2020
  + DPea19-4080\_1\_SilicoDArT.csv: SilicoDArT alineados con *Medicago truncatula* y *Cicer arietinum* = Report\_DPea19-4080\_1\_moreOrders\_SilicoDArT\_1.csv del 11/04/2019
  + DPea19\_4080\_1\_SilicoDArt\_BLAST.csv: comparación de nuestros datos con las bases de datos del NCBI = DPea19\_4080\_1moreOrder\_SilicoDArTsBLAST.csv

**B.1.1 Eliminar las secuencias de Eleonora**

* Todos los genotipos de las muestras de Eleonora empiezan con “Gen” seguido de un número de identificación mientras que los míos son identificados por su Numero de referencia en la colección IAS.
* En consecuencias, para eliminar de la base de datos todo lo relativo a las secuencias de Eleonora:
  + Abrimos el fichero DPea19-4080\_SilicoDArT\_Pisum.csv
  + filtramos la base de datos todas las columnas que contienen “Gen” con la función “select” del paquete de R “dplyr”.
  + Usando los comandos:

# abrir la base de datos y elimina los genotipos de Eleonora (todos empiezan por "Gen")

Gen<-read.csv ("DPea19-4080\_SilicoDArT\_Pisum.csv", skip=6, header=TRUE) %>%

select (-c(starts\_with("gen"))) %>% arrange (CloneID) %>% select (2:3,1, 4:7,8:339)

**B.1.2. Juntar las tres anotaciones en un solo fichero**

* Para ello abrimos los otros dos ficheros (DPea19-4080\_1\_SilicoDArT.csv y DPea19\_4080\_1\_SilicoDArt\_BLAST.csv) en R.
* Seleccionamos las columnas que contienen la información de los marcadores y las anotaciones.
* Se junta las 3 bases de datos con la función “full-join” del paquete de R “dplyr”.
* Como algunos marcadores no tienen el mismo nombre en los diferentes ficheros por una razón desconocida utilizamos la secuencia de cada marcador contenido en la columna “Allele\_Sequence” para fusionar las bases de datos.
* Tras estos pasos se guarda la base de datos en formato .csv (genotipos\_SilDArT\_coleccionGWAS.csv).

# abre y selecciona los datos utiles del archivo con los alineamientos contra el genoma de Medicago y garbanzo

P1<-read.csv ("DPea19-4080\_1\_SilicoDArT.csv", skip=6, header=TRUE) %>%

select (2,1,4,5,7,12,13,15) %>%

rename (Chr\_Medicago=Chrom\_Medicago\_v4.0.2, Pos\_Medicago=ChromPos\_Medicago\_v4.0.2, Eval\_Medicago=AlnEvalue\_Medicago\_v4.0.2, Chr\_Chickpea=Chrom\_Chickpea\_cdcfrontier\_v03, Pos\_Chickpea=ChromPos\_Chickpea\_cdcfrontier\_v03, Eval\_Chickpea=AlnEvalue\_Chickpea\_cdcfrontier\_v03) %>%

arrange (CloneID)

# abre y selecciona los datos utiles del archivo con los BLASTs contra la base de datos nr y de procriota

Blast<- read.csv ("DPea19\_4080\_1\_SilicoDArt\_BLAST.csv", skip=3, header=TRUE) %>%

select (2,1,4,7,8,11) %>%

arrange (CloneID) %>%

rename (Blast\_nr=Chrom\_nt, Eval\_nr=AlnEvalue\_nt,Blast\_proc=Chrom\_ref\_prok\_rep\_genomes, Eval\_proc=AlnEvalue\_ref\_prok\_rep\_genomes)

# Union de les tres bases de datos...

# Las tres tienen el mismo numero de secuencias pero no siempre con el mismo nombre

# asi que se hace la union según la secuencia (AlleleSequence) y no el nombre (CLoneID)

P<- P1 %>% full\_join (Blast, by="AlleleSequence") %>% full\_join (Gen, by="AlleleSequence") # Une las tres bases de datos

write.csv (P,"genotipos\_SilDArT\_coleccionGWAS.csv", row.names=FALSE) # guarda la base de datos un archivo .csv

**B.1.3. Calcular parámetros de polimorfismos (PIC, Call rate, % heterocidad,etc.)**

* La evaluación de los marcadores Silico\_DArTs en cada genotipo se estimó con:
  + “1”: presencia del marcador.
  + “0”: ausencia del marcador.
  + “-“: secuencia encontrada en el genotipo pero con un numero de copia demasiado bajo para asignarle un “1” con seguridad – suele representar heterocigoto.
* Para determinar los diferentes parámetros básicos hay que contar el numero de “1” (presencia), “0” (ausencia) y “-“ (heterocigoto).
* No he conseguido hacer esto en R así que lo hago en Excel con la función “CONTAR.SI”.
* Tras contar el numero de veces que aparece “1”, “0” y “-“ en los genotipos por cada marcador en Excel, he salvado el fichero en formato .xlsx y .csv con el nombre “gen\_SilDArT\_coleccionGWAS.csv”.
* Para calcular los diferentes parámetros importantes se utiliza las siguientes formulas:
  + CallRate =
  + One Ratio = MAF =
  + % Heterocigocidad = Het=
  + PIC =
* Los cálculos de estos parámetro se hizo en R con las fórmulas anteriores con la función “mutate” del paquete de R “dplyr”.
* Después se salvó el fichero en formato .csv con el nombre “gen\_SilDArT\_coleccionGWAS.csv”
* Se utilizó los siguientes comandos en R:

G<-read.csv2 ("gen\_SilDArT\_coleccionGWAS.csv", header =TRUE, row.names=NULL) %>%

mutate (CallRate= ((Presencia + Ausencia)/Total),OneRatio= Presencia/Total, Missing=Missing/Total,PIC=(2\*OneRatio\*(1-OneRatio))) %>%

rename (Het=Missing,MAF=OneRatio)

write.csv (G, "gen\_SilDArT\_coleccionGWAS.csv", row.names=FALSE)

**B.1.4. Filtrar marcadores de mala calidad y alelos raros**

* El filtrado se hace en R con la función “filter” del paquete de R “dplyr”
* Los parámetros de filtrados fueron los siguientes:
  + Presencia ≥ 1 (marcadores detectado al menos en uno de mis genotipos)
  + MAF ≥ 0.05 (se conserva solo los marcadores que están presentes en al menos 5% de la colección)
  + %Het ≤ 0.1 (se conserva solo los marcadores con menos de 10% de heterocigosis)
* Con este filtrado se elimina por completo (o casi) todas las secuencias contaminantes de humano, y otros animales encontrado según Diversity Array ya que la contaminación viene de una o dos muestras de Eleonora.
* Tras filtrar se guarda el fichero en formato .csv con el nombre de “gen\_SilDArT\_colGWAS\_Filtered.csv”.
* El filtrado se hizo con estos comandos:

Db<- G %>% filter (Presencia >=1, MAF>=0.05, Het<0.1)

write.csv (Db,"gen\_SilDArT\_colGWAS\_Filtered.csv", row.names=FALSE)

* En los diferentes pasos de la limpieza obtuvimos:
  + Antes de limpiar cada base de datos contenían: **66643** marcadores
  + Tras fusionar las bases de datos obtuvimos: **66685** marcadores (no se porque hay más marcadores…)
  + Tras el filtrado tenemos: **27616** marcadores de buena calidad
* De estos marcadores **6259** (22% de los marcadores) no se encontraron en el genoma del guisante (criterio de BLAST de Diversity array E-value= 5e-05 y min %identify=80%)
* Antes de seguir con el análisis de los datos voy a intentar mapear estos marcadores

**B.1.5. Mapeo de los marcadores sin localización sobre el genoma de referencia**

* Para mapear estos marcadores voy a realizar un Blast frente al genoma de referencia del guisante
* Para ello primero hay que crear un fichero con el nombre y la secuencia de todos los marcadores no mapeado. Esto lo hizo en R con los siguientes comandos:

Db<- read.csv ("gen\_SilDArT\_colGWAS\_Filtered.csv", header=TRUE)

Nomap<- Db %>% filter (Chrom\_Pisum\_sativum\_v1a =="")

#crear un fichero con las secuencias de los marcadores sin información de mapa

FASTA <- Nomap %>% select (1:2)

write.table (FASTA,"nomapseq.txt", row.names=FALSE)

#crear un archivo con las secuencias trimmed de los marcadores sin información de mapa de guisante

Ftrim<- Nomap %>% select (1,3)

write.table (Ftrim, "nomapseqtrimmed.txt", row.names=FALSE)

* Con estos comandos creamos 2 ficheros con la información de los marcadores no mapeado con su secuencia completa o trimmed (elimina cebadores, y parte de baja calidad)
* Después abrimos estos ficheros .txt en Excel y Edit+ para transformarlos en formato fasta para que pueda servir para el BLAST.
* El análisis BLAST se hará en local así que me descargue la base de datos genómica del guisante de la página del INRA así como el programa MAGICBLAST y BLAST del NCBI. El MAGICBLAST es un nuevo programa del NCBI especialmente diseñado para mapear Secuencia de RNAseq y GBS sobre genomas de referencias
* Tras descargarme la base de datos y los programas se trabaja en el “símbolo de sistema” de Windows (antiguo MS-DOS)
* Para utilizar el magicblast se hace
  + Formatear la base de Datos con el comando

>makeblastdb -in <nombre base de datos.fa> -out <nombre de la base de datos> -parse\_seqids -dbtype nucl

* + Hacer el BLAST

>magicblast -query <nombre fichero con las secuencias a “blastear”> -db <nombre de la base de datos de referencia> -out<nombre del fichero output aquí:MagicBlast\_nomap.txt>

* También hice un local blast con la función blastn-short specifico por secuencias cortas de ADN.
* Para ello use como valor límites: E-value 1e-4 y min%Identity 80
* La línea de comando fue la siguiente:

>blastn -task blastn-short -query <nombre fichero query> -out <nombre fichero output aqui: Blastn\_short\_nomap.csv> -perc\_identity 80 -evalue 0.0001 -outfmt 10 (=formato .csv) -max\_target\_seqs 1 -max\_hsps 1

* Una vez realizado los blast con las secuencias completas y Trimmed con los dos métodos, he comparado los resultados juntándolos todos en un mismo fichero en R
* Tras compararlo se ve que los resultados de los diferentes BLAST son idénticos o casi.
* Tras eliminar las secuencias repetidas conservo los datos del Blastn\_short\_trimed que encuentran más hits.
* Finalmente hay que añadir los datos de mapa nuevo a la base de datos.
* Esto lo hago en R y tengo que seguir estos pasos:
  + Seleccionar los marcadores que no estaban mapeado
  + A estos marcadores se añade la información obtenido por los BLAST que he hecho
  + Añadir esta información a la base de datos completos.
  + Finalmente guardo la base de datos en formato .csv con el nombre “gen\_SilDArT\_colGWAS\_Filteredmapped.csv”
  + Estos pasos se hicieron con los siguientes comandos:

#abre base de datos completa filtrada y crear 2 base de datos uno con los datos mapeados y otros con los datos sin info de mapa

Db<- read.csv ("gen\_SilDArT\_colGWAS\_Filtered.csv", header=TRUE) %>%

rename (Chrom\_Pisum=Chrom\_Pisum\_sativum\_v1a,Pos\_Pisum=ChromPos\_Pisum\_sativum\_v1a,Eval\_Pisum=AlnEvalue\_Pisum\_sativum\_v1a)

DbMap<- Db %>% filter (Chrom\_Pisum !="")

Dbnomap<- Db %>% filter (Chrom\_Pisum =="")%>% arrange (CloneID)

#Abre el archivo con el resultado BLASTn-short-trimmed

Bt<-read.csv("Blast-short\_nomaptrimmed.csv",header=TRUE)%>%

rename (CloneID=queryacc,Identity=X.identity,Bs\_Pisumchr=reference.acc, BS\_Position=reference.start)%>%

select (1:2,9,11) %>% arrange (CloneID)

# junta los dos archivos

DbnomapBt<- Dbnomap %>% select (-c(4:6)) %>%

left\_join (Bt, by="CloneID") %>%

select (1:3,350:352,4:349) %>%

rename (Chrom\_Pisum=Bs\_Pisumchr,Pos\_Pisum=BS\_Position,Eval\_Pisum=e\_value)

#añadir la base de datos mapeado nuevamente a la base de datos mapeado por DiversityArray

Dbnew<- rbind(DbMap,DbnomapBt) %>% arrange(CloneID)

#guardar la base de datos completa

write.csv (Dbnew,"gen\_SilDArT\_colGWAS\_Filteredmapped.csv", row.names=FALSE)

* Tras este intento de mapearlo, queda **2104** marcadores sin localizar en el genoma del guisante lo que representa unos 7.6% de los marcadores. Por lo que hemos conseguido mapear a unos 4000 marcadores.
* Con esto tendríamos ya la base de datos limpiada definitiva (a falta de quitar los marcadores con Pic>0.01 es decir los no polimórficos).
* Para poder usarlo en los diferentes programas TASSEL, Plink, etc… hay que cambiar el formato del fichero.
* En total y despues de filtrar en TASSEL hay un total de **24279** markers de los cuales **19514** SilicoDArT maped to pea chromosome (80.4%), **2703** mapped to unanchored contigs (11.1%) and **2062** unmapped markers (8.5%).

**B.2. Limpieza de la base de datos de marcadores SNPs**

* Tras limpiar la base de datos de los marcadores tipo SilicoDArT, hago lo propio con las bases de datos SNP
* Para ello he creado una nueva carpeta llamada “SNPs” dentro de la carpeta “Database Genotipado”.
* Dentro de esta carpeta he pegado los 2 archivos .csv generado por Diversity Array con los datos SNP en 1 linea.
* Para diferenciar y simplificar estos dos archivos les he renombrado así:
  + DPea19-4080\_SNP\_Pisum.csv: base de datos alineada con el genoma de referencia del guisante = Report\_DPea19-4080\_1\_moreOrders\_SNP\_mapping\_2 del 27/02/2020
  + DPea19-4080\_SNP.csv: SilicoDArT alineados con *Medicago truncatula* y *Cicer arietinum* = Report\_DPea19-4080\_1\_moreOrders\_SNP\_mapping\_2\_old del 11/04/2019

**B.2.1 Eliminar las secuencias de Eleonora de la base de datos SNP**

* Todos los genotipos de las muestras de Eleonora empiezan con “Gen” seguido de un número de identificación mientras que los míos son identificados por su Número de referencia en la colección IAS.
* Para eliminar de la base de datos SNP todo lo relativo a las secuencias de Eleonora editamos el script creado para los marcadores silicoDArT para cambiar el nombre del archivo a abrir y modificar.
* Como los dos archivos tienen en realidad la misma información y la única información importante es la localización de los marcadores sobre el genoma de guisante, solo uso el archivo DPea19-4080\_SNP\_Pisum.csv y me salto la etapa de juntar las anotaciones de los dos ficheros.
* Guardo el archivo filtrado con el nombre “DbSNP\_coleccionGWAS.csv”. Este archivo contiene **54269** SNP y 325 genotipo.
* Nota: por una razón que desconozco hay menos marcadores de tipo SNP que de tipo SilicoDArT.

**B.2.2. Calcular parámetros de polimorfismos (Call rate, AlleleFrequency, % heterocidad,etc.) y filtrado de la base de datos**

* La evaluación de los marcadores SNP en cada genotipo se estimó con:
  + “0”: alelo de referencia homocigoto.
  + “1”: alelo alternativo homocigoto.
  + “2”: alelo heterocigoto
  + “-“: ausencia del marcador SilicoDArT que contiene este SNP
* Para determinar los diferentes parámetros básicos hay que contar el número de “0” (presencia), “1” (ausencia) y “2“ (heterocigoto).
* No he conseguido hacer esto en R así que lo hago en Excel con la función “CONTAR.SI”.
* Tras contar el número de veces que aparece “1”, “0”, “2” y “-“ en los genotipos por cada marcador en Excel, he salvado el fichero en formato.csv con el mismo nombre (“DbSNP\_coleccionGWAS.csv”).
* Para calcular los diferentes parámetros importantes se utiliza las siguientes formulas:
  + CallRate =
  + One RatioRef = AFRef =
  + One RatioSNP = AFSNP =
  + % Heterocigocidad = Het=
* Los cálculos de estos parámetros se hicieron en R con las fórmulas anteriores con la función “mutate” del paquete de R “dplyr” modificando el script preparado para los marcadores SilicoDArT.
* En R he podido calcular las frecuencias alélicas para el alelo de referencia y el alelo alternativo, pero no siempre el de referencia es el principal por lo que no sé cómo calcular el MAF (frecuencia alélica del alelo minoritario) para filtrar.
* Por lo tanto, voy a filtrar la base de datos en R para quitar los marcadores con >20% missing data (call rate>0.8) y con una tasa de heterocigozidad ≤ 0.1.
* Después quitaré los marcadores con MAF<0.05 en TASSEL.
* El filtrado por missing data y heterocigozidad deja **28911** SNP.
* Después se guardó el fichero en formato .csv con el nombre “DbPea\_SNP\_coleccionGWAS\_filtered.csv”
* El ultimo filtro (MAF<0.05) se hará en TASSEL tras transformar el archivo en formato hapmap. Esto deja **11511** SNP (salvado como “PisumSNP\_filtered.hmp.txt”)

**B.2.3. Mapear los marcadores sin alinear sobre el genoma de guisante**

* De los **11511** SNP marcadores obtenidos tras filtrar, del ellos, **9363** SNP mapearon sobre cromosoma del guisante (81.3% marcadores), **1054** SNP (9,2% de los marcadores) mapearon sobre unaligned scaffolds and **1094** (9,5% de los marcadores) no se encontraron en el genoma del guisante (criterio de BLAST de Diversity array E-value= 5e-05 y min %identify=80%)
* Antes de seguir con el análisis de los datos voy a intentar mapear estos marcadores
* Como los SNP son contenido dentro de los Silico-DArT, para mapear los marcadores no-mapeado voy a comparar la matrix de marcadores SNP con la info que tenemos de los Silico-DArT para completar con los datos obtenidos para ello.
* De esta forma he añadido 206 marcadores mapeado sobre los cromosomas de guisante y 78 marcadores mapeados sobre super contigs. El resto sigue sin información probablemente porque estos marcadores no eran polimórficos a nivel Silico-DArT.
* Para el resto voy a hacer un blast sobre el genoma del guisante del mismo modo que hice con los Silico-DArT.
* El primer paso es obtener las secuencias de los marcadores sin info de mapa y transformarlos en formato fasta. Para ello he comparado la lista con los datos incluido en el archivo “DbPea\_SNP\_coleccionGWAS\_filtered.csv” y filtrado para tener solo una copia de cada clone junto con su secuencia.
* Los codigos utilizados fueron:

list2<- read.delim ("SNPlist\_unmapped2.txt", header=TRUE)

ref2<- read.csv("DbPea\_SNP\_coleccionGWAS\_filtered.csv", header=TRUE) %>% select (1:4)

listmap2 <- left\_join (list2,ref2,by="CloneID") %>% group\_by (CloneID) %>% slice\_sample(n=1) # con estos comendos he juntado la lista sin mapa con las secuencias de los clones y conservado solo uns instancia en caso de clones con varios marcadores

Symbol= ">"

Nomap<- listmap2 %>% select (1,3) %>% mutate (S1=Symbol) %>% select (3,1,2)

write.table (Nomap,"SNP\_nomapseq.txt", row.names=FALSE)

Nomaptrim<- listmap2 %>% select (1,4) %>% mutate (S1=Symbol) %>% select (3,1,2)

write.table (Nomaptrim,"SNP\_nomapseq\_trim.txt", row.names=FALSE)

* Después he hecho el local BLAST frente a las secuencias de referencias del guisante igual que para los SilicoDArT.
* Al final tras mapear tengo
  + 10125 SNP mapeado sobre cromosomas del guisante (88%de los marcadores)
  + 1155 SNP mapeado sobre contigs (10% de los marcadores)
  + 231 no mapeados (2% de los marcadores)

1. **Cambio de Formato de la base de datos**

**C.1. Formato Hapmap**

* TASSEL reconoce varios formatos, pero uno de lo más común es el formato Hapmap que tiene la información de cada marcador en líneas y de los genotipos en columnas más información de calidad etc.
* Así para poder realizar el análisis en TASSEL 5.0 vamos a pasar la base de datos en el formato Hapmap.
* Este formato requiere incluir una serie de columna obligatorias.
* La base de datos en formato Hapmap se llamó “GenPea\_SilDArT.hmp.txt”
* El cambio de formato se hace en R con los siguientes comandos:

panelLSID="NA"

assayLSID="NA"

protLSID="NA"

strand="NA"

assembly="NA"

center="NA"

QCcode="NA"

alleles="A/C/N"

Hmp<- Dbnew %>% select (1,4:5,28:352) %>%

mutate(rs=CloneID,alleles,chrom=Chrom\_Pisum,pos=Pos\_Pisum, strand,assembly,center,protLSID,assayLSID,panelLSID, QCcode) %>%

select(rs,alleles,chrom,pos,strand,assembly,center,protLSID,assayLSID, panelLSID,QCcode,4:328)

write.table (Hmp," GenPea.hmp.txt", sep="\t", row.names=FALSE)

Hmp2<- Hmp %>% replace (Hmp=="1","AA") %>% replace(Hmp=="0","CC") %>% replace (Hmp=="-","NN")

write.table (Hmp2," GenPea\_SilDArT.hmp.txt", sep="\t", row.names=FALSE)

* Antes de abrir el fichero hapmap en TASSEL hay que eliminar todas las comillas que R introduce para diferenciar caracteres y se añade un “#” después de “rs” y de “assembly”
* Nota: para que TASSEL reconozca el fichero como secuencias y no datos numéricos hay que ordenar los genotipos (función “Sort genotype File” en el menú data) y salvar el fichero output. Aquí lo salvamos como “GenPea\_SilDArT\_sort.hmp.txt”.

**C.2. Formato Plink**

* + - * Plink es un programa que funciona a partir del “símbolo del sistema” (antiguo MS-Dos - En windows10 se encuentra en la pestaña de programa llamado “sistema de windows”) y permite hacer muchos análisis básicos como estimar LD, estudio de asociación, etc…. Sobre todo, permite filtrar los marcadores en LD para eliminar los datos redundantes facilitando los análisis posteriores y disminuyendo el ruido. Habrá seguramente otros métodos para hacer eso, pero plink es de lejos el programa más comúnmente empleado.
      * Consideración a tener en cuenta al usar plink:
  + este programa se ha diseñado para el genoma Humano (por defecto acepta 22 chr + chr X y Y, diploide), etc…así que necesitara algunas adaptaciones.
  + Codifica los genotipos de manera diploide. Es decir que hay 2 alelos por cada marcadores.
  + Es exclusivamente command-line con lenguaje propio.
  + Fichero con formato especifico (.bed + .fam +.bim or .ped + .map)
    - * Explicación paso a paso para cambiar el formato de hapmap a plink:

1. Pasar el fichero hapmap en formato de plink (ficheros .map y .ped) en TASSEL 5.
   * Abrir el fichero hmp (aquí GenPea\_SilDArT\_sort.hmp.txt) en TASSEL 5
   * Ticar en la pestaña “File” y en “Save As”
   * En la ventana que se abre se cambia el “Format” de “Hapmap” a “plink” en el menú desplegable.
   * Se cliquea en “Browse” para selectionar la carpeta de destino y el nombre del fichero a guardar (aquí GenPea\_SilDArT\_sort.plink)
   * Esto va a generar dos ficheros con extensiones .ped y .map respectivamente (aquí GenPea\_SilDArT\_sort.plink.plk.ped y GenPea\_SilDArT\_sort.plink.plk.map)
   * Nota: Los ficheros .ped y .map generado por TASSEL no pueden ser utilizados directamente por plink y tenemos que modificar tanto el fichero .ped que el .map
2. Modificaciones del fichero .ped

* En el formato hapmap generado por TASSEL para los marcadores Silico-DArT, cada marcador tiene 3 alelos codificado con las siguientes letras (A, C, N) siendo A presencia marcador, C ausencia marcador y N missing data o heterocigoto.
* Al pasar al formato plink, TASSEL transformo los “N” en “0”
* Plink está diseñado para marcadores de tipo SNP por lo que reconoce 4 tipos de alelos (A, C, G y T o 1, 2, 3 y 4) por lo que antes de usar plink hay que transformar los alelos “0” en “G” o “T” . Para ello usaremos el programa R o EDIT+
* En R, el script seria lo siguiente:

install.packages(readr)

library (readr) #usamos la libreria "readr" que son funciones mejorado para importar/exportar y modificar datos

ped<-read\_table2("GenPea\_SilDArT\_sort.plink.plk.ped", col\_names = FALSE) #abrimos el fichero .ped a modificar

sum (ped=="N") # con la función “sum” contamos el número de veces que aparece los símbolos "N", "O" y "T" en el fichero a modificar

sum (ped=="0")

sum (ped=="T")

Ped1<-replace(ped, ped=="0", "T") # con la función “replace” cambiamos los "0" en "T"

sum(Ped1=="0") # confirmamos que los "+" se han cambiado adecuadamente

sum (Ped1=="T")

write.table(Ped1," GenPea\_SilDArT\_sort.ped", col.names = FALSE, row.names = FALSE)

* A usar la función “write.table”, R introduce comillas para todos los caracteres (A, G, T y, C) y cadena de caracteres (nombre de los genotipos) de la tabla por si estos símbolos adicionales influencian a plink los borramos todos. Para ellos
* Abrimos el fichero en un editor de texto (EditPlus, Notepad++ o EmEditor)
* En la pestaña “buscar” seleccionamos “reemplazar”
* En “buscar” ponemos “ y en “reemplazar” lo dejamos en blanco y presionamos en reemplazar todo.
* Salvamos el fichero (aquí GenPea\_SilDArT\_sort.ped)
  + - * + Tras pensarlo un rato creo que esta transformación no esta correcta ya que en realidad los marcadores SilicoDArT han sido codificado según tres estado: “1” presencia, “0” ausencia y “-“ missing data y sobre todo heterocigoto. Para TASSEL lo hemos transformado en “AA”, “CC” y “NN” respectivamente pero según la transformación que realizamos a pasar los “NN” a “TT” durante la transformación del fichero en formato .ped, le decimos a plink que hay 3 alelos diferentes aunque en realidad solo hay 2 + heterocigoto. Lo más correcto seria transformar estos “TT” en “AC”. Para ello, he creado otro fichero hapmap (GenPea\_SilDArT\_het.hmp.txt) transformando los “-“ en “AC”. Tras abrirlo en TASSEL 5, lo he transformado en formato .ped y salvado como GenPea\_SilDArT\_het.ped. En este caso no hace falta transformar nada ya que directamente codifica los heterocigoto en “AC”.

1. Modificaciones del fichero .map

* Para que el programa reconozca los cromosomas hay que ponerlos un número en vez de los códigos “XLGX” por lo que hay que cambiar los nombres de los cromosomas. Por ejemplo, reemplazar “chr1LG6” por “1” (no “01”). Lo haremos en un editor de texto de la misma manera que para el fichero .ped.
* El guisante tiene 7 cromosomas y es diploides asi que simplemente tenemos que reemplazar los codigos chrXLGX por su número correspondiente de 1 al 7.
* Para los marcadores con homología a los scaffold sin asignación en ningún chromosoma y los que no pudieron ser mapeado sobre el genoma de referencia, le pondremos un “8” y un “9” respectivamente.
* Tras reemplazar el nombre de los cromosomas salvamos el fichero con otro nombre (aquí GenPea\_SilDArT\_sort.map y GenPea\_SilDArT\_het.map).

**C.3. Formato STRUCTURE**

* STRUCTURE necesita la información de cada genotipo en cada línea y cada marcador en una columna. Es decir que en comparación con el formato hapmap está invertido.
* Es un formato parecido al formato del fichero .ped para plink.
* STRUCTURE codifica cada marcador de manera diploide por lo que el valor de cada marcador puede estar en 2 líneas consecutivas o dos columnas consecutivas (habrá que definirlo en STRUCTURE).
* STRUCTURE coge los alelos como “1” y “0” por lo que hay que transformar de nuevo los “A” en “1”, los “C” en “0” y los heterocigotos los codificaremos como “1” y “0” como hemos hecho para el formato .ped de plink.
* Visto la similitud con el formato .ped se puede usar el fichero .ped como base para crear el formato de STRUCTURE quitando las columna innecesarias.
* Structure necesita 1 columna con los nombres de los genotipos seguido con las anotaciones de cada marcador en 1 o dos columnas consecutivas. Opcionalmente puede incluir una columna con info i.e. subespecies, localización geográfica, etc. Codificado con números. Esta columna opcional ayuda el programa a asignar los grupos. Pero en general prefiero dejar el programa libre de esta columna para que no tenga “a priori”.
* Si usamos el fichero .ped generado para plink, lo abrimos en Excel, eliminamos las columnas 1,3,4,5 y 6, se transforma las “A” en “1” y los “C” en “0” y se salva el fichero.
* Si usamos el fichero hapmap hay que seguir estos pasos:
  + Transformarlos datos en datos numéricos seleccionando “numeric genotype” en el menú “Data” esto transforma el alelo principal en “1” y el minoritario en “0”.
  + Salvamos este archivo dándole a “sabe as” en el menú “File” y seleccionamos el formato “numeric genotype” en la ventanilla que se abre.
  + Abrimos el archivo en Excel o un editor de texto.
  + Copiamos todo en un nuevo archivo o página de Excel para tener dos versiones del archivo.
  + En una copia transformamos los “0.5” o espacios (“ “) en “1”.
  + En la otra copia transformamos los “0.5” o espacios (“ “) en “0”
  + Copiamos la segundo copia y la pegamos al final de la primera copia
  + Ordenamos todo por genotipo de manera a tener los genotipos en formato diploide con los datos de cada genotipo codificado en dos líneas.
  + Salvamos el fichero y ya lo tenemos apto para STRUCTURE.

**C.4. Formato para MEGA X**

* MEGA X es un programa especializado en el calculo y construcción de árboles filogenéticos.
* Esta especializado por el análisis de secuencias de ADN o proteínas por lo tanto necesita un archivo conteniendo un alineamiento de secuencias.
* El programa reconoce todo tipos de formato relacionado con alineamiento de secuencias como el formato phylip, fasta y .msf así como en formato de arbol filogenético como newick y nexus pero es incapaz de entender los formatos hapmap y .vcf.
* Para transformar los archivos que queremos analizar con MEGA utilizaremos TASSEL y EditPlus siguiendo estos pasos:
  + Abrimos el archivo con la base de datos en formato .hmp.txt o .vcf en TASSEL 5 (no olvidar seleccionar la opción “sort genotype” al abrir el archivo)
  + Seleccionamos la base de datos del apartado “sequence”
  + La salvamos en formato “phylip sequential” seleccionando “save as” en el menú “File” y el formato correcto en la ventanilla “Format”.
  + Abrir el archivo .phy con EditPLus
  + Reemplazar la primera línea por <Número de genotipo> <Número de marcadores<> separado por un espacio. Hay que borrar del todo la primera línea y después escribir los dos números.
  + Salvar el archivo modificado en formato.phy
  + Abrir MEGA X y seleccionar “Convert File Format to MEGA”.
  + En la ventanilla emergente seleccionar el fichero y seguir los pasos que MEGA nos da.
  + Finalmente se salva el archivo en format .meg que es el formato que MEGA reconoce.

1. **FILTRAR MARCADORES EN LD**

- Para filtrar los marcadores en LD vamos a usar el programa Plink

* 1. Poner los ficheros .ped y .map en la carpeta donde se encuentra el programa plink. Idealmente se ha descardado en el disco C:
  2. Abrir el símbolo de sistema y ir a la carpeta donde se encuentra el programa plink y los ficheros .ped y .map a analizar. Se usa el comando “cd “ (por ejemplo: cd Documents\Bioinfo…
  3. Uso de plink
     1. Cargar el fichero y crear los fichero binario (ficheros con extensiones .bed + .bim + .fam) que plink usa preferentemente.

Se hace con los comandos siguientes:

plink --file {*nombre fichero sin su extensión*} --make-bed

Nota: en plink cada orden se inicia con un dos signos “-“ seguidos

Nota 2: Tal como indique antes plink se diseñó para tratar marcadores del genoma Humano y Animal así que hay que decirle que hay más cromosoma para ello usaremos el comando “--autosome-num” seguido del número del último cromosoma (En el fichero GenPea\_SilDArT tenemos los 7 cromosomas del guisante plus el cromosoma 8 y 9 que contienen las secuencias con homología con los contigs no mapeado y las secuencia sin homologia respectivamente). Del mismo modo considera que cada individuo pertenece a una familia y desciende de 1 padre y 1 madre. Como en nuestro caso no es posible hay que decirle que todos nuestros genotipos son como “entradas fundadoras” sino plink no hará ningún análisis posterior. Se hace con el comando “--make-founders”

Ejemplo:

plink -file Genpea\_SilDArT\_het --autosome-num 9 --make-founders --make-bed

Nota3: en plink se puede concatener tanto ordenes que se quiere

ii. Para empezar y comprobar que todo va bien vamos cargar los ficheros y estimar la frecuencia alélica de todos los marcadores y los % de missing data

plink --file {*nombre fichero sin su extensión*}--autosome-num <número del ultimo cromosoma*>* --make-founders --make-bed --freq --out {*nombre del fichero output sin extensión*} --missing

Ejemplo:

plink -file Genpea\_SilDArT\_het --autosome-num 9 --make-founders --make-bed --freq --out Peahet --missing

- He hecho esto con la base de datos codificado como “AA, CC y TT (heterocigotos)” (fichero sort) y el fichero codificando los heterocigotos como “AC” (fichero “het). Se guardan como “Peasort” y “Peahet” respectivamente.

- Tal como se esperaba el resultado cambia un poco según la manera de codificar los heterocigotos. Parece que lo más lógico es usar el fichero het.

1. Reducir el número de marcadores basado en la LD

Según los tutoriales hay dos maneras de realizar esta reducción, el método de “pruning” y el método de “clumping”. Algunos dicen que clumping es mejor que pruning pero como no lo hemos hecho nunca voy a probar los dos para empezar.

“**pruning**”: este método se diseñó para filtrar los marcadores en LD para eliminar ruido y reducir el tiempo de cálculo de los análisis posteriores como STRUCTURE. Para ello plink coge el primer marcador y estima la correlación con los siguientes marcadores. En caso de estar correlacionado elimina el marcador con menor MAF después pasa al siguiente marcador (no eliminado todavía) y repite el procedimiento. En caso extremo este procedimiento puede remover muchos marcadores y dejar grande área del genoma sin marcadores. Aun así, es el procedimiento por defecto de muchos análisis de población genómico (STRUCTURE, etc…)

Comando para el pruning:

plink --bfile *{nombre del fichero binario sin extensión*} –autosome-num <número del ultimo cromosoma> --indep-pairwise <tamaño de la ventana de marcador> <tamaño de escalón =número de marcadores a saltar> <valor límite de r2>

Ejemplo

1. plink --bfile Peahet --autosome-num 9 --indep-pairwise 50 5 0.1 --out Peahet

el resultado del pruning por LD (r2 >0.1) del fichero “het” da lo siguiente:

Pruned 2218 variants from chromosome 1, leaving 514.

Pruned 1921 variants from chromosome 2, leaving 573.

Pruned 2142 variants from chromosome 3, leaving 536.

Pruned 2338 variants from chromosome 4, leaving 753.

Pruned 3418 variants from chromosome 5, leaving 810.

Pruned 2741 variants from chromosome 6, leaving 713.

Pruned 2828 variants from chromosome 7, leaving 941.

Pruned 1930 variants from chromosome 8, leaving 1138.

Pruned 1099 variants from chromosome 9, leaving 1005.

Pruning complete. 20635 of 27618 variants removed, remaining 6983 markers

1. plink --bfile Peasort --autosome-num 9 --indep-pairwise 50 5 0.1 --out Peasort

Pruned 2048 variants from chromosome 1, leaving 684.

Pruned 1759 variants from chromosome 2, leaving 735.

Pruned 1988 variants from chromosome 3, leaving 690.

Pruned 2197 variants from chromosome 4, leaving 894.

Pruned 3151 variants from chromosome 5, leaving 1077.

Pruned 2525 variants from chromosome 6, leaving 929.

Pruned 2666 variants from chromosome 7, leaving 1103.

Pruned 1777 variants from chromosome 8, leaving 1291.

Pruned 1139 variants from chromosome 9, leaving 965.

Pruning complete. 19250 of 27618 variants removed, remaining 8368 markers

* Al final queda 6983 y 8368 markers lo que es mucho para STRUCTURE. Lo ideal sería reducir a un número máximo de 2000 a 4000 marcadores. Para lograrlo voy a aumentar la ventana de marcadores a 200 (estima la LD entre los 200 primeros marcadores filtra los marcadores en LD y después salta 5 marcadores (del inicio del fichero) y repite el proceso por cada cromosoma.
* para el fichero “het” lo hago asi

plink --bfile Peahet --autosome-num 9 --indep-pairwise 200 5 0.1 --out Peahet2

Pruned 2425 variants from chromosome 1, leaving 307.

Pruned 2148 variants from chromosome 2, leaving 346.

Pruned 2353 variants from chromosome 3, leaving 325.

Pruned 2651 variants from chromosome 4, leaving 440.

Pruned 3757 variants from chromosome 5, leaving 471.

Pruned 3015 variants from chromosome 6, leaving 439.

Pruned 3218 variants from chromosome 7, leaving 551.

Pruned 2485 variants from chromosome 8, leaving 583.

Pruned 1566 variants from chromosome 9, leaving 538.

Pruning complete. 23618 of 27618 variants removed, remaining 4000 markers

* para el fichero “sort”:

plink --bfile Peasort --autosome-num 9 --indep-pairwise 200 5 0.1 --out Peasort

Pruned 2246 variants from chromosome 1, leaving 486.

Pruned 1952 variants from chromosome 2, leaving 542.

Pruned 2165 variants from chromosome 3, leaving 513.

Pruned 2497 variants from chromosome 4, leaving 594.

Pruned 3455 variants from chromosome 5, leaving 773.

Pruned 2756 variants from chromosome 6, leaving 698.

Pruned 2998 variants from chromosome 7, leaving 771.

Pruned 2299 variants from chromosome 8, leaving 769.

Pruned 1639 variants from chromosome 9, leaving 465.

Pruning complete. 22007 of 27618 variants removed, remaining 5611 marcadores

* Lo bueno de hacerlo de este método es que el número que se queda es más o menos homogéneo entre cromosomas.
* Al final queda 4000 y 5611 marcadores tras el pruning de las bases de datos por LD.
* El comando “--indep-pairwise” crea dos fichero prune.in y prune.out que contiene los nombres de los marcadores que se guarda y los que se eliminan respectivamente. Para crear la base de datos con los marcadores filtrados en el formato binario de plink (ficheros .bed+.bim+.fam) se usa estos comandos:

plink --bfile *{nombre del fichero binario sin extensión*} --autosome-num <número del ultimo cromosoma> --extract {*nombre del fichero prune.in*} --make-bed --out {*nombre fichero filtrado*}

Ejemplo

plink --bfilePeahet --autosome-num 9 --extract Peahet.prune.in --make-bed --out Peahetpruned

* Nota: Tras filtrar la base de datos, el programa estima la frecuencia alélica. Filtrar permite subir el ratio genotípico total de 0.9513 a 0.962676 para el fichero “sort”.

Para exportar estos ficheros en formato plink (.ped + .fam) y en formato .vcf (fichero compatible con Tassel, etc…) usamos estos comandos:

plink --bfile *{nombre del fichero binario sin extensión*} --autosome-num <número del ultimo cromosoma> --recode vcf –out {*nombre del fichero creado en formato .vcf}*

plink --bfile *{nombre del fichero binario sin extensión*} --autosome-num <número del ultimo cromosoma> --recode --out {*nombre del fichero creado en formato .bed +.map*}

Ejemplo:

plink --bfile Peahetpruned --autosome-num 9 --recode vcf --out Peahetpruned

plink --bfile Peahetpruned --autosum-num 9 --recode --out Peahetpruned

**Clumping:** Método de filtrado diseñado para quitar marcadores en LD pero guardando los SNPs esenciales. Se creó originalmente para asegurar que los SNP asociados con un carácter según el GWAS se mantienen en la base de datos tras simplificarla. Para ello plink clasifica los marcadores por importancia (normalmente por valor de p) y cuando un par de marcadores son en LD guarda el de mayor importancia (normalmente el que tiene un p inferior al límite que le hemos especificado). Como el clumping/pruning se hace por ventana de tamaño es decir por área del genoma este método asegura que 1) todos los SNPs que tienen una p < al límite definido (i.e. 0.001) y 2) que se quede al menos un marcador en cada ventana de tamaño mientras que el pruning a solaparse las ventana de tamaño puede que se elimine todos los marcadores en los casos más extremos. Algunos autores indican que se podría usar el clumping basándose en el MAF (en vez de los valores de p). De esta forma los dos métodos son muy similares pero el clumping podría ser un poco más conservador.

Para el clumping necesitamos el fichero .bed y el fichero .frq que contiene los valores de MAF. Como clumping fue diseñado para usar valores de *p*, eliminará los marcadores que tienen un valor superior a *p* limite (i.e. 0.001). Así si queremos que conserve los marcadores que tengan un MAF superior a 0.2 por ejemplo, tendremos que calcular los valores de “1-MAF” y usar como límite de 0.8. Para transformar la columna MAF en 1-MAF usamos Excel. Para ello, abrimos el fichero en Excel y reemplazamos los datos MAF por 1-MAF. Después copiamos la nueva tabla en un editor de texto y reemplazamos las tabulaciones por espacios y lo salvamos como .frq. Después volvemos al “símbolo de sistema” y usamos los siguientes comandos:

plink --file *{nombre del fichero sin extensión*} --clump {*nombre del fichero .frq*} --clump-field MAF --clump-p1 0.8 --clump-p2 0.2--clump-r2 0.1 –out {*nombre del fichero output*}

Ejemplo:

plink --bfile Peahet --autosome-num 9 --clump Peahet.frq --clump-field MAF --clump-p1 0.8 --clump-p2 1 --clump-r2 0.1 –out PeahetClumped

Nota: Algunos marcadores estan dos veces y no me deja hacer el análisis. Para solucionar el problema cambio el nombre de los marcadores repetido en los ficheros .frq y .bim.

Si usamos como limite MAF= 0.2 y r2<0.1 deja 5748 marcadores.

Si usamos limite MAF=0.3 y r2=0.1 deja 3720 marcadores

Tras hacer el clumping hay que extraer los SNP eliminado del fichero usando la función extract y crear los nuevos ficheros .bed y .vcf para poder usarlos en TASSEl y otro programas.

Extraer los SNP eliminado

plink --bfilePeahet --autosome-num 9 --extract PeahetClumped2.clumped --make-bed --out PeahetClumped

Exportar los datos en .ped y format vcf

plink --bfile PeahetClumped --autosome-num 9 --recode vcf --out Peahetclumped

plink --bfile PeahetClumped --autosome-num 9 --recode --out Peahetclumped

1. **ESTUDIO DE LA DIVERSIDAD GENETICA**

**E.1. PCA**

- He realizado una PCA con los datos filtrados con el programa TASSEL e independiente con R con el mismo dataset (GenPea\_SilDArT\_sort.hmp.txt).

- Para hacer el PCA en R, hay que transponer la base de datos.

- Esto se hace con los siguientes comandos:

#

## 1. transponer la base de datos

#

Dbgen<- Dbnew %>% select (1,28:352)

row.names (Dbgen) <- make.names (Dbgen$CloneID, unique=TRUE)

Dbgen <- Dbgen %>% select (2:326)

Dbgen2<- as.data.frame(t(as.matrix(Dbgen)))

#

## 2. Añadir una columna con el nombre de los genotipos

#

Genot<-colnames(Dbgen) # crea un lista con los nombres de los genotipos

Genot<-sub("X","",Genot) #elimina los X añadido por R a los nombres que empiezen con un numero

DbGeno <- Dbgen2 %>% mutate (Genot) %>% select (Genot,1:27619) %>% rename (Genotype=Genot)

#

## 3. transformar los datos de presencia/ausencia en valores numericas

#

DbGeno <-DbGeno %>% replace (DbGeno=="0","2") %>% replace (DbGeno=="-","3")

write.table (DbGeno,"peaGWAS\_transpose.txt", sep="\t",row.names=FALSE)

# en un editor de texto abrir el archivo y eliminar las comillas

* Después se añade la información de pasaporte de los diferentes genotipos

DbGen3<-read.delim("peaGWAS\_transpose.txt",header=TRUE) %>% select (1:27619)

Info<- read.delim ("Coleccion\_info.txt", header=TRUE)

Db.info <- Info %>% full\_join(DbGen3, by="Genotype")

* Finalmente se hace el PCA y se representa según diferentes parámetros del pasaporte (color flor, subespecie, origen geográfica, etc.) y salvamos los valores de PC en un fichero para servir en el GWAS con estos comandos:

install.packages("ggfortify")

library (ggplot2)

library(ggfortify)

library (dplyr)

PCs<- select(Db.info,contains ("X"))

PCA<-prcomp (PCs, scale.=FALSE, rank.=10) # estima los PCA

autoplot(PCA, data = Db.info)

autoplot(PCA, data = Db.info, colour = 'Growth\_Type')

autoplot(PCA, data = Db.info, colour = 'Material')

autoplot(PCA, data = Db.info, colour = 'Continent')

autoplot(PCA, data = Db.info, colour = 'Color\_Flor')

autoplot(PCA, data = Db.info, colour = 'Species')

autoplot(PCA, data = Db.info, colour = 'Flower\_Type')

summary(PCA)

ggplot (PCA,aes(x=PC1, y=PC3))+ geom\_point(aes(colour= Db.info$Species))

PCAs<-as.data.frame(PCA$x)# crea un data frame con los valores de PCs

PCAs<- mutate (PCAs, Entrada)

plot(PCAs)

write.table (PCAs,"PCA\_peaGWAS\_R.txt", row.names=FALSE)

* Para estimar PCA en TASSEL:
  + Abrimos le fichero hapmap en TASSEL 5 (no olvidar de ticar en la opción de “sort genotype” )
  + Seleccionamos la base de datos y ticamos en “PCA” dentro del submenu “Relatedness” del menú “Analysis”.
  + Se puede representar las PCA en TASSEL pero los graphs no son muy bonitos así que la representación grafica se hace en R
  + Para ello salvamos el fichero PC\_GEnPea\_SilDArT\_sort con el formato “phenotype” y lo abrimos en R. Se salvo con el nombre (ColGWAS\_PCA.txt)
  + Después en R añadimos la información de pasaporte del mismo modo que para el método con R y representamos las PCs gráficamente con estos comandos.

info<- read.delim ("Coleccion\_info.txt", header=TRUE)

TASSEL <- read.delim ("ColGWAS\_PCA.txt",header=TRUE)

TAS.info<- full\_join (info,TASSEL)

PCATASSEL <- select (TAS.info,contains ("PC"))

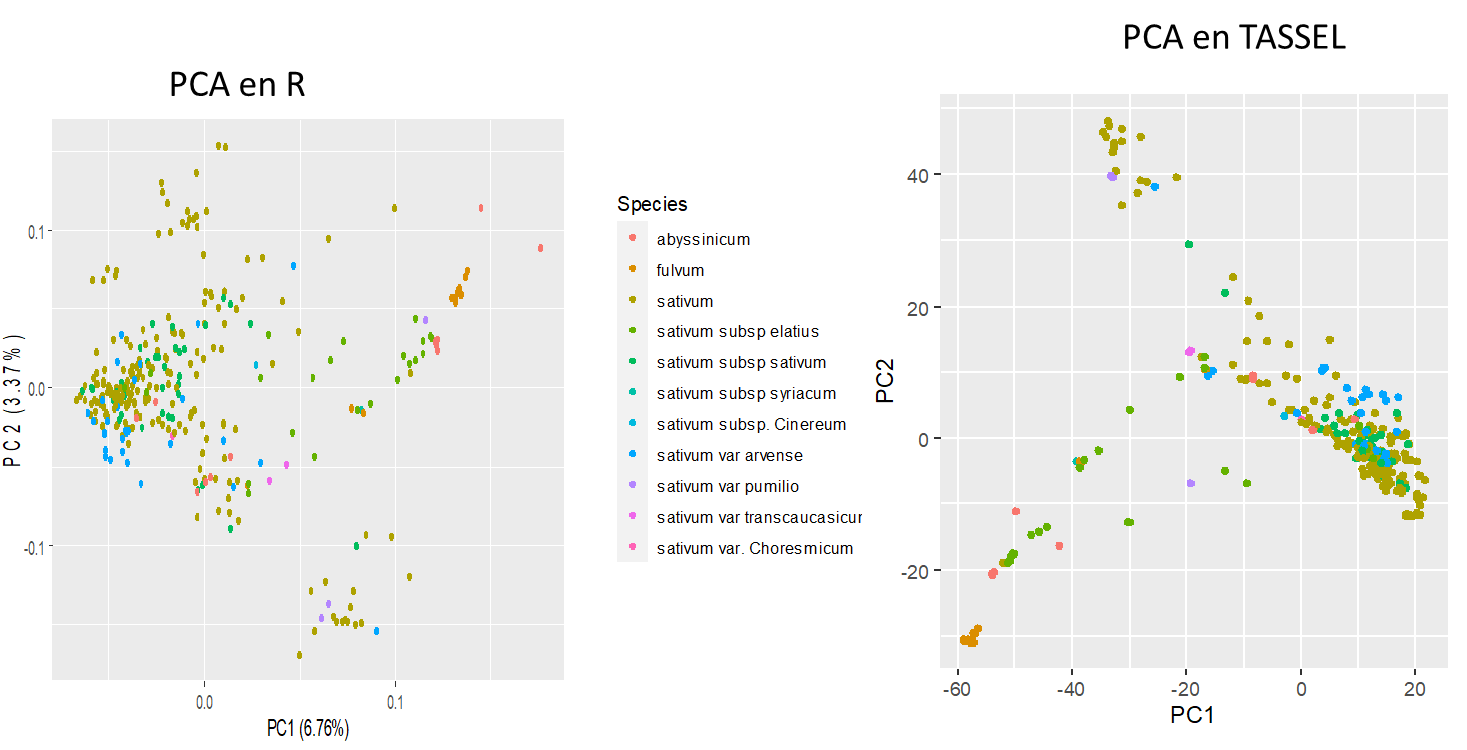
plot(PCATASSEL, data = TAS.info,shape =TRUE,color = 'species', label.size=3) # este comando dibuja una matrice de graficos con todos los PCn vs PCn

summary(PCATASSEL)

ggplot (TAS.info,aes(x=PC1, y=PC2))+ geom\_point(aes(colour= Continent))

ggplot (TAS.info,aes(x=PC1, y=PC2))+ geom\_point()

* Ambos programas dan resultados similares con varios grupos visible. En concreto en los dos los genotipos de P. fulvum se junta en un pequeño subgrupo. La falta de definición de las subespecies por cada genotipo que tenemos en los datos de pasaporte impide identificar los otros subgrupos. El PCA estimado en R es más disperso lo que ayuda a observar entre 4 y 6 grupos distintos.



- Para ver el efecto del filtro drástico (para usar STRUCTURE (ver parametros en la subsección E.2.) y del LD Pruning repito las PCA con estos dos datasets.

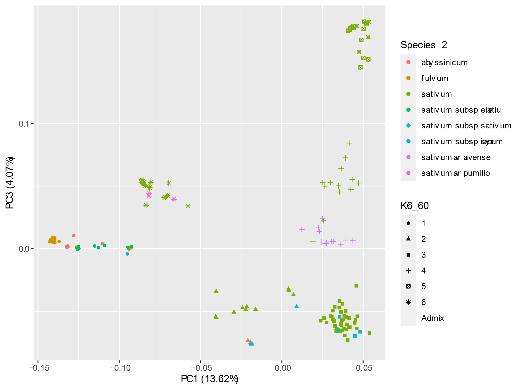
- Para ello utilizo R para estimar el PCA.

- Antes de usar R hay que abrir los ficheros filtrados en TASSEL.

* + Los ficheros creados por plink son en formato .vcf que es compatible con TASSEL.
  + Todos los archivos se abren en TASSEL
  + Se transforman en datos numéricos (seleccionando la opción “Numeric genotype” en el menú “Data”) lo que transforma el alelo principal en 1, el alelo alternativo en 0 y los heterocigotos en 0,5).
  + Y los salvamos como Numeric Genotype:
    - Genpea\_het\_Num.txt (4001 marcadores)
    - Genpea\_sort\_Num.txt (5611 marcadores)
    - Genpea\_TAS\_Num.txt (3575 marcadores)
  + Abrimos estos ficheros en Edit+, quitamos si hace falta los -9 puesto en la columna de los genotipos y transformamos las comas en puntos y salvamos de nuevo el fichero.
  + Nota: solamente el fichero “het” contiene datos heterocigotos los demás tienen missing values en vez de esto. Como para calcular la PCA R no admite missing values, se transforman los “NA” en “0.5” después de salvarlos
  + Después abrimos R y modificamos el script anterior con el nombre del fichero filtrado correcto.

****

* + - * He probado también hacer la PCR tras filtrar la base de datos con plink con el método de Clumping (“Genpea\_het\_Clump\_Num.txt”). Pero da exactamente lo mismo que la PCA obtenida tras filtrar en TASSEL, probablemente porque usé el mismo parámetro de filtrado para ambos.
      * El PCA bueno es el estimado con R o TASSEL tras transformar los datos genómicos en datos numéricos que transforma los Datos por cada sitio con un “1” para el alelo principal y “0” para el menor independientemente si es una A o una C. De este modo el PCA estimado en R con la base de dato completo es igual que el PCA estimado en R. De este modo codificar los heterocigotos (evaluado como “-“ en el fichero inicial) con “TT” o con “AC” en TASSEL
      * Tampoco hay mucha diferencia entre filtrar por LD con el archivo “het” y con el archivo “sort” aunque parece más lógico utilizar el archivo “het”.
      * La forma general de las PCA sin filtrar y tras filtrar son parecidos, aunque filtrar los datos dispersa más los datos complicando la separación de los grupos.
      * Tras hacer el análisis con STRUCTURE, hemos añadido al fichero la información de los grupos estimado (K3, K5 y K6) para poder diferenciar los puntos de la PCA con los agrupamientos de STRUCTURE. Lo que nos ha dados los resultados siguientes:





* + - * Resultados destacables:
        + Juntado los datos de la PCA con STRUCTURE es evidente que hay 6 grupos
        + La separación se ve más claro cuando usamos la PC1 vs PC3 o la PC2 vs PC3 que con la separación PC1 vs PC2.
        + Es muy variable más de un tercio de la colección no se puede asignar a ningún grupo a tener menos del 60% de pertenencia a un grupo.
        + Esta colección esta muy diversa y con mucho admixture debido a la hibridación interespecífica
        + Los grupos más o menos respetan las subespecies y serian asi:

Grupo 1: *P. fulvum, P. sativum* subsp *elatius* y la mitad de los *P. abyssinicum*

Grupo 2: los *P. abyssinicum* restante y *P. sativum subsp sativum* con flor de color

Grupo 3: cultivares de *P. sativum subsp sativum* todos con flor blancas

Grupo 4: Landraces de *P. sativum* var arvense, la mitad con flor blanca y la otra con flor de color

Grupo 5: grupo de *P. sativum* de la india con flor blanca

Grupo 6: *P. sativum* subsp *elatius* var pumelio y también P. sativum var tibetanicum

* + - * + Con esta separación he corregido la identificación de unas pocas entradas por las que había error claro de identificación.
      * Una vez hemos probado las diferentes representaciones (PCA organizada por subespecies, grupo structure, color de flor, etc…) y encontrado la más adecuada para representar los datos la salvamos en formato .eps con los siguientes comandos:

graph<- autoplot(PCA, data = Db.info2, x=1, y=2,colour="K3\_60") #creamos un objeto con el grafico a guardar

ggsave(

"GraphPCA\_FiltPlink2.eps", plot = graph, device = "eps", path = NULL,

scale = 1, width = 20, height = 15, units = "cm", dpi = 600,

limitsize = TRUE)

**E.2. STRUCTURE**

- No se puede analizar la estructura de la población con la base de datos completo de marcadores ya que tardaría demasiado.

- Hay dos maneras de reducir el número de marcadores:

* en TASSEL seleccionando solamente los marcadores con alta variación alélica y presente en la mayoría de genotipos. Por ejemplo, Fran uso como parámetro para la Avena (Site Min Count: 645 [presente en 95% de la población]; Site Min Allele Freq: 0.3 [min frecuencia alélica 30%; Site Max Allele Freq: 1.0; Min Heterozygous Proportion: 0.0; Max Heterozygous Proportion: 0.01 [máxima proporción de heterocigoto 1%).
* Filtrando los marcadores que están en LD. Lo que se puede hacer con el programa Plink.
* Para determinar si el método de filtrado y el modo de codificar los alelos tiene una consecuencia en la estructura final de la colección voy a probar los dos métodos.

**E.2.1. Instrucciones de uso de STRUCTURE**

- Para hacer los análisis es mejor que los ficheros de la base de datos estén en el disco C: para evitar problema de conexión ya que el programa va a durar varios días.

- Una vez hemos puesto la base de datos de plink (archivo .ped) o hapmap (archivo .hmp.txt) en un formato apto para STRUCTURE les hemos guardado en una carpeta del disco duro local, abrimos el programa STRUCTURE y haremos lo siguiente:

* Creamos un nuevo proyecto seleccionando “New Project” en el menú “File”. Lo que abrirá una ventana emergente
  + - 1. En la 1ª página de la ventana ponemos:

Name Project: <nombre de la carpeta que se va a crear> (por ejemplo “Structure\_Peahet”)

Seleccionamos el directorio donde se creará este proyecto. En concreto selecciono la carpeta que he creado y donde puse la base de dato (aquí c:/… documentos/bioinformatics/Structure Pea)

Seleccionamos el archivo con los datos (por ejemplo “Pea\_SilDArt\_het\_LDpruned.txt”).

* 1. En la 2ª página ponemos:

Number Individual: 325

Ploidy Data 2

Number Loci: <Número total de marcadores> (para este archivo es 4001)

Missing Data -9 (le indicamos como tiene que reconocer el missing data de nuestra matriz)

* 1. En la 3ª página:
     1. Si la base de datos se formateo a partir del archivo .ped de plink ticamos en “Data file stores data for individuals in a single line”
     2. Si la base de datos se formateo a partir del fichero hapmap o tras trasnformación númerica en TASSEL no ticamos nada en esta pagína.
  2. En la página 4 seleccionamos únicamente “Individual ID for each individual”
  3. Si no hay mensaje de error habremos cargado los datos correctamente en STRUCTURE
* Después de crear el proyecto vamos a definir los parámetros a utilizar. Para elloseleccionamos “New” en el menú “Parameter Set”:
  1. En la ventana “Run length ponemos:

Length of Burnin Period: 10000

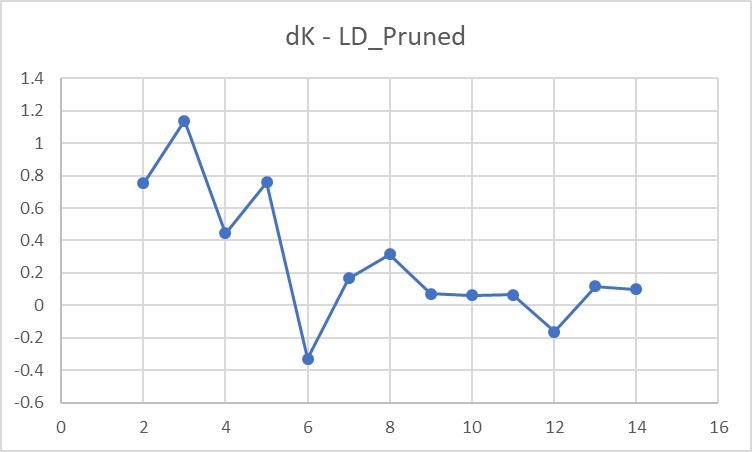
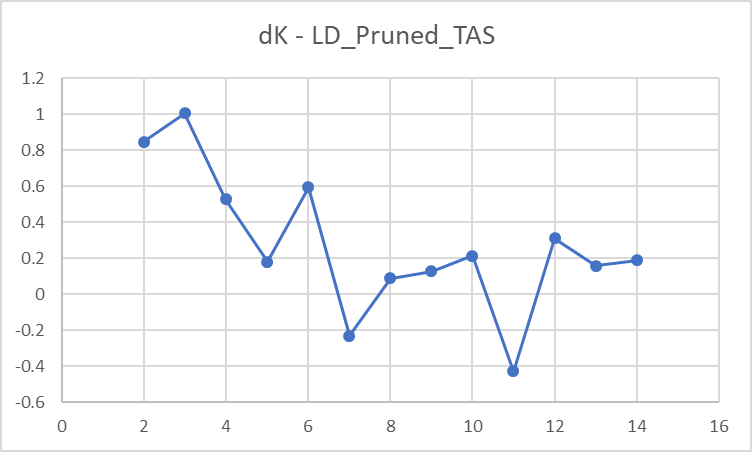
Number of MCMC Reps after Burnin: 20000

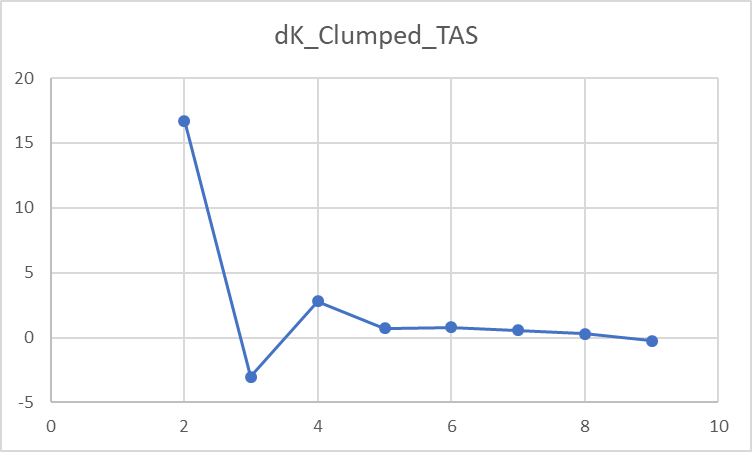
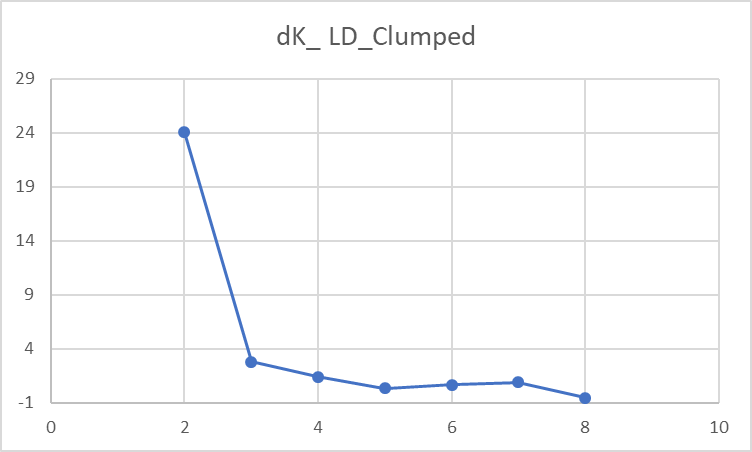
* 1. En la ventana “Ancestry Model Info” seleccionamos “Admixture Model”
  2. En la ventana “Frequency Model Info” seleccionamos “Allele Frequency Correlated”
  3. En la ventana “Advance” seleccionamos “Compute probability of the data (for estimating K). todo el resto lo dejamos tal como está.
  4. Tras rellenar/revisar todas las ventanas le damos a OK y le ponemos nombre al método (aquí “PeaStructure”)
* Finalmente lanzamos el programa. Para eso le damos a “Start a job” en el menú “Project”
* Se abre una ventana en la cual seleccionamos el método que acabamos de crear, el número de K (aquí de 1 a 10), El número de seed (opcional aquí he puesto 1) y el número de iteración (para la prueba pongo 1. Para una análisis completa poner al menos 10). Después le damos a “Start”.
* Como el analisis completo puede tardar más de una semana se puede hacer por tanda por ejemplo tanda 1) de 1 a 5 K, 2) de 6 a 8 K y 3) de 9 a 10 K cada vez con 5 o 10 run cada uno.

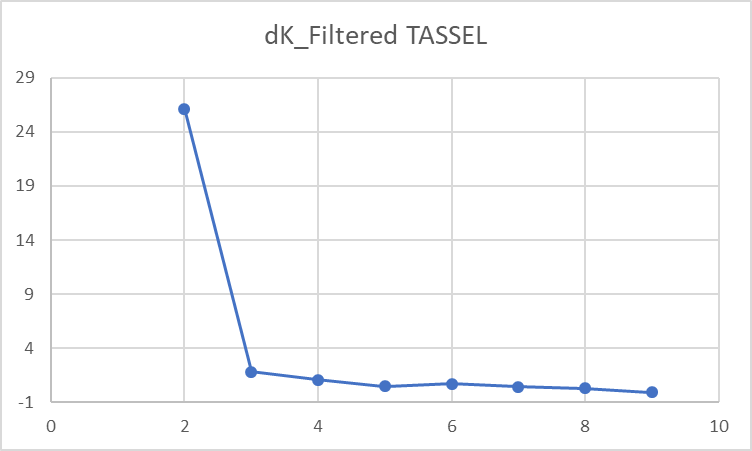
**E.2.2 Comparación del modo de filtración y modo de codificación de los alelos**

- Para saber si la filtración y el modo de codificar los alelos repercuta sobre la estructura de la población he lanzado el programa STRUCTURE con las diferentes bases de datos:

* Base de datos filtrado por LD (método pruning): “Pea\_SilDArT\_het\_LDpruned.txt”
* Base de datos filtrado por LD (método pruning) y alelos principales (transformado en formato numérico con TASSEL): “Pea\_SilDArT\_het\_LDprunedTAS.txt”
* Base de datos filtrado por LD/MAF (método clumping): ”Pea\_SilDArT\_het\_LDclump.txt”
* Base de datos filtrado por LD/MAF (método clumping) y alelos principales (transformado en formato numérico con TASSEL): ”Pea\_SilDArT\_het\_LDclump\_TAS.txt”
* Base de datos filtrado en TASSEL (transformado en formato numérico con TASSEL): “Pea\_SilDArT\_FilTAS.txt”
  + - * En todos los casos he usado los mismos parámetros (10000 burnin, 20000runs, admixture y frecuencia alélica correlacionadas)
      * He corrido 1 run para cada K de 1 a 10 K. Además, las dos bases de datos “LDpruned” fueron probado por K de 11 a 15 con 1 run cada uno también.







* + - * Como solo se ha hecho un solo run, hay que tomar los datos con cautela. Sin embargo, se puede observar que hay grandes diferencias en la estimación de la estructura según los diferentes métodos y/o modo de codificación de los alelos.
      * Del mismo modo que para la PCA, la filtración con TASSEL y con el método de Clumping da un resultado similar. En ambos casos, el análisis del deltaK indica 2 grupos principales y cuando los datos están codificados según alelo principal/minoritario vemos también un segundo pico a K=4 haciendo suponer que podría tener 4 subgrupos.
      * El análisis con la base de datos filtrado por LD con el método de pruning da resultados muy diferentes a los otros dos métodos.
      * Estos resultados se asemejan más a lo que se espera tras observar las PCAs y los arboles filogeneticos.
      * Con la base de datos “Pea\_SilDArt\_het\_LDpruned.txt” observamos un pico principal a K=3 y otros dos más pequeños a K=5 y K=8.
      * Con la base de datos “Pea\_SilDArT\_het\_LDprunedTAS.txt”, se observa también el pico principal a K=3 pero se ve tres picos más pequeños a K=6, 10 y 12.
      * Las diferencias entre los dos métodos de codificación de los alelos puede ser un artefacto debido a que se hizo solo el análisis una vez.
      * Los resultados que más me convencen son los obtenidos con la base de datos filtrado por LD con el método pruning. En consecuencia, voy a repetir el análisis de STRUCTURE con esta base de datos con 10 runs per categoría de K (hasta 15 K).
      * Para asegurar que el método de codificación de los alelos no afecta el resultado voy a hacer en paralelo el análisis con la base de dato LD prunedTAS.

**E.2.1 Estimación de la estructura de la población tras filtrar por LD (Plink)**

* + - * Como indicado en el apartado anterior, los resultados preliminares de STRUCTURE indican que los datos filtrados por LD reflejan mejor la realidad que el filtrado por polimorfismo.
      * En consecuencia, he repetido el análisis con la base de datos “Pea\_SilDArt\_het\_LDpruned.txt” para un número de K entre 1 y 15 con 10 run independiente cada K.
      * En paralelo, he analizado la base de datos “Pea\_SilDArt\_het\_LDprunedTAS.txt” para un número de K entre 1 y 15 y con 5 run independientes por cada K.
      * Para determinar el número de K optimo he utilizado el servidor web STRUCTURE HARVESTER que implementa método de EVANNO (ΔK).
      * Como algunos runs para algunas K a veces se desvía mucho de los demás, he calculado de nuevo el ΔK eliminando los runs outliers.





* + - * Según la evolución del ΔK con la base de datos “LDPruned” habría 3 grupos principales y 6 subgrupos si eliminamos los outliers (5 subgrupos con los outliers).
      * Con los datos “LDPRunedTAS” observamos 3 grupos principales y 5 subgrupos independientemente si consideramos los outliers o no.



* + - * Observando los valores de -Ln(K) vemos que son muy similares entre el análisis de la base de datos “LDPruned” y “LDPRunedTAS”) por lo que podriamos juntar todos los resultados (en total 15 runs).
      * En este caso observamos que habría 3 grupos y 5 o 6 subgrupos dependiendo si eliminamos los runs outliers o no.



* + - * Como los resultados obtenidos con “LDPRuned” son iguales que si utilizamos todos los runs (“LDPRUned + LDPrunedTAS”) creo que lo correcto es usar los datos de “LDPruned” que tienen más sentido.

**E.3. Filogenia**

* + - * Para los estudios de filogenia usaremos el programa gratuito MEGA X.
      * Vamos a construir dos tipos de árboles basado en distancia: UPGMA y Neigbor-Joining.
      * Previamente al análisis hay que transformar la base de datos en formato .hmp o .vcf en un formato aceptable para MEGA X siguiendo las instrucciones indicadas en la sección C.4
      * Vamos a comparar los arboles obtenidos con:
        + La base de datos completa (“GenPea\_SilDArT\_sort.meg”
        + Base de datos filtrado por LD con el método pruning (“Peahetpruned.meg”)
        + Base de datos filtrado por LD y MAF con el método clumping (“Peahetclumped.meg”)
        + Base de datos filtrado en TASSEL (GenPea\_SilDArT\_sort\_FilTAS.meg”)
      * Una vez abierto el archivo estimamos el mejor modelo de sustitución que servirá para calcular la distancia genética entre cada genotipo.
        + Lo hacemos con el programa MEGA X:

Hacemos clic en el icono “Models” y seleccionamos “Find Best DNA/protein models (ML)

En la ventana emergente seleccionamos:

Tree to use: Neighbour-Joining

Gaps: Use all site

Branch swap: none

Nbr. Thread: 8

Finalmente hacemos clic en OK

Hacemos lo mismo con la opción de gap “partial deletion con 95% cutoff”.

* + - * + Los modelos óptimos para cada base de datos son:

Db completa: HKY + G (BIC); HKY +G +I (AICc)

Db filtrado con TASSEL: HKY + G (BIC y AICc)

Db filtrado por LD pruning: HKY +G + I (BIC y AICc)

Db filtrado por LD clumping: HKY + G (BIC y AICc)

* + - * Con esta información hacemos los arboles filogenéticos. Usaremos los métodos UPGMA y Neigbor-Joining basado en matrices de distancia entre genotipos.
        + Para el método UPGMA hacemos

Ticamos en “filogenia” y seleccionamos “construct/test UPGMA…”

En la ventanilla que se abre seleccionamos:

Phylogeny ⇒Test of Phylogeny: Bootstrap

Phylogeny ⇒No of Bootstrap…: 1000

Substitution Model ⇒ Model/Method: p-distance

Substitution Model ⇒Substitution to include: Transitions + Transversions

Rates and Pattern ⇒ Rates among sites: seleccionar entre “Uniform Rates”, “Gamma distributed”, “Invariant site” o “Gamma distributed with Unvariant sites” según el resultado del mejor modelo de substitución estimado anteriormente.

Rates and Pattern ⇒Gamma parameter: rellenar con el valor del modelo estimado si la distribución es +G

Data Subset to Use ⇒ Gaps/Missing data: elegir “pairwise deletion “ o “partial deletion – 95% cutoff”

* + - * + Para el método Neigbor-Joining (NJ)

Ticamos en “filogenia” y seleccionamos “construct/test Neighbor-Joining…”

En la ventanilla que se abre Seleccionamos/rellenamos todo igual que para el UPGMA

* + Una vez hecho los cálculos, MEGA dibuja el árbol filogenetico en una nueva ventana ( ventana “Tree Editor”) donde podemos modificar el aspecto del árbol cambiando el tipo de árbol (cladogram, unrooted, circular, etc…), rotar y colapsar ramas etc…
  + Para poder volver al árbol posteriormente guardamos la sesión lo que guardara el archivo en formato .mtsx que es especifico de MEGA
  + Para poder dibujar el árbol en R y modificar más cosas lo vamos a exportar en formato formato “newick” que es un formato universal para codificar los arboles filogenéticos.
    - Para ello seleccionamos “Export Current Tree (Newick)” en el menú “File”
    - En al ventana emergente ticamos “Branch lengths” y “Bootstrap values” y ticamos en “OK”
    - Esto abre otra ventana “Text Editor”
    - Para guardar el archivo, ticamos en el símbolo del disquete y la salvamos con formato “.tre” que es reconocido por R (por ejemplo: “GenPea\_Full\_UPGMA.tre”)
      * Al final he construido los arboles filogenéticos tratando los huecos por deleción parcial y por pares y por todos los métodos de filtrado lo que nos da lo siguientes archivo de árboles:
        + UPGMA con la base de datos completa:

“GenPea\_Full\_UPGMA\_MEGA\_partialdel.tre” (11635 sitios)

“GenPea\_Full\_UPGMA\_MEGA\_pairwisedel.tre” (27618 sitios)

* + - * + UPGMA con la base de datos filtrado por LD (plink):

“GenPea\_LDPruned\_UPGMA\_MEGA\_partialdel.tre” (1165 sitios)

“GenPea\_LDPruned\_UPGMA\_MEGA\_pairwisedel.tre” (4001 sitios)

* + - * + UPGMA con la base de datos filtrado por polimorfismo (TASSEL):

“GenPea\_FilTAS\_UPGMA\_MEGA\_partialdel.tre” (575 sitios)

“GenPea\_FilTAS\_UPGMA\_MEGA\_pairwisedel.tre” (3575 sitios)

* + - * + NJ con la base de datos completa:

“GenPea\_Full\_NJ\_MEGA\_partialdel.tre” (11635 sitio)

“GenPea\_Full\_NJ\_MEGA\_pairwisedel.tre” (27618 sitios)

* + - * + NJ con la base de datos filtrado por LD (plink):

“GenPea\_LDPruned\_NJ\_MEGA\_partialdel.tre” (1165 sitios)

“GenPea\_LDPruned\_NJ\_MEGA\_pairwisedel.tre” (4001 sitios)

* + - * + NJ con la base de datos filtrado por polimorfismo (TASSEL):

“GenPea\_FilTAS\_NJ\_MEGA\_partialdel.tre” (575 sitios)

“GenPea\_FilTAS\_NJ\_MEGA\_pairwisedel.tre” (3575 sitios)

* + - * Para modificar los árboles y por ejemplo separar por colores las entradas según el grupo de Structure a la que pertenecen abriremos el árbol en Mega y lo editaremos con las herramientas de edición de árbol que contiene.
      * Para poner una forma de colores al final de cada rama:
        + Ticamos en el icono con forma de llave inglesa
        + Seleccionamos “labels”
        + En “Shape” seleccionamos el circulo y modificamos el color
        + Finalmente seleccionamos los genotipos al cual queremos que se añada este símbolo.
        + Si queremos podemos también cambiar el color y el tipo de letra usado para los nombres de los genotipos o incluso quitarlos en la sección “taxon name”
        + Después aplicaremos los cambios dándole a “OK”
        + Utilizaremos un color distinto por cada grupo de Q y seleccionaremos manualmente cada genotipo de cada grupo para incluir el símbolo.
      * Para cambiar el color de la rama:
        + Seleccionamos el nodo a editar ticando encima con el ratón
        + Después ticamos en el martillo en la barra de herramienta vertical
        + En la ventana emergente cambiaremos el color y tamaño de línea en la parte “Branch line” de la sección “Property”
        + Allí podemos también ponerle un nombre al subárbol rellenando y modificando el color o el tipo de fuente de letra en “name/Caption”
        + Después aplicamos los cambios presionando “OK”
      * La estructura general de los arbóles es muy similar independientemente de la base de datos utilizado. Asi que para el artículo usare el árbol obtenido con la base de datos completo con el método de Neigbor-Joining
      * Tras editar los árboles para añadir la información de los grupos de Structure se ve claramente los 6 grupos separados por un montón de genotipos mixtos.
      * Aunque se realizó los árboles con 1000 boostraps, para simplificar las figuras he representado el árbol de forma circular y eliminado los valores de bootstrap y los nombres de los genotipos para tener solamente los clústers visibles.



**E.4. Linkage Desequilibrium**

* + - * El linkage Disequilibrium es una manera de determinar el nivel si un marcador esta o no ligado a otro marcador localizado cerca o en otro lugar. Es decir estimar el LD permite sabe si los marcadores son independiente o no.
      * Es importante saber esta información porque es la base del GWAS. Para que el GWAS sea eficiente tenemos que disponer de un gran número de marcadores que sean independiente y repartido por todo el genoma.
      * También es importante saber la distancia de decaimiento de LD ya que permite determinar la ventana de tamaño en pares de bases o centiMorgan donde se encuentra un QTL potencial y por tanto los genes candidatos.
      * Cuando no disponemos de un genoma de referencia o de un mapa genético no podemos saber el orden de los marcadores por lo tanto no podemos saber el orden y localización de los marcadores. Así tenemos que estimar el LD por cada par de marcadores. Esto genera millones de comparación y genera ficheros grandes y puede colapsar el ordenador
      * Si nuestros marcadores han sido alineados sobre un genoma de referencia o un mapa genético podemos estar el LD de solo una parte de los marcadores utilizando una ventana de tamaño para reducir el número de comparación y de recurso de CPU.
      * Se puede utilizar varios programas para estimar el LD (TASSEL, PLink, etc…).
      * Para estimar el decaimiento de LD, se necesita saber la distancia entre cada marcador por lo que se debe de comparar solo marcadores de un mismo chromosoma por lo que voy a eliminar todos los marcadores que no han sido mapeado o que mapean sobre supercontigs ya que realmente no sabemos donde están.
      * Guardo esta nueva base de datos como GenPea\_SilDArT\_sortCHR.hmp.txt.
      * Esta base de datos contiene 22446 marcadores
      * Para estimar el LD en una ventana de tamaño utilizaré el programa TASSEL. Para ello
        + Lanzo el programa TASSEL y abro el archivo con mi base de datos genotípica (GenPea\_SilDArT\_sortCHR.hmp.txt)
        + Antes de lanzar el análisis filtro los marcadores con MAF <0.05 para asegurar que todos los marcadores sean polimórficos. Esto deja 19514
        + Selecciono “Analysis” > “Diversity” > “Linkage Disequilibrium”
        + En la ventana emergente pongo lo siguiente:

Select Type: “Sliding Window”

LD Window Size: 100 (cada marcador se compara con los 100 marcadores que tiene alrededor)

How to treat Heterozygous: “set as missing”

Finalmente se presiona”run”

* + - * + Guardamos el archivo output como “GenPea\_SilDArT\_chr\_LD\_sliding100.txt”.
        + Este archivo contiene 1,946,350 líneas o sea 1,946,350 comparaciones.
        + Repetimos lo mismo usando una ventana de 50 (por si simplifica los gráficos a tener menos puntos).
        + Esta segunda análisis la guardamos como “GenPea\_SilDArT\_chr\_LD\_sliding50.txt”.
        + Una vez hecho el análisis, pasamos a R para hacer los gráficos y estimar el decaimiento de LD.
        + En R primero eliminaremos todas las comparaciones realizadas entre marcadores de cromosoma diferente ya que no sabemos la distancia que existe entre ellos (con el archivo de 100 queda 1,916,050 comparaciones)
        + Eliminamos también todos las comparaciones que no son estadísticamente significativa (pDiseq<0.05). Esto deja 1,107,940 comparaciones.
        + Para ello usamos el siguiente script:

LDout <- read.delim("GenPea\_SilDArT\_chr\_LD\_sliding100.txt")

LDist <- LDout %>% na.omit %>%

filter (pDiseq <0.05) %>%

rename (sitei=Position1, sitej=Position2,rsq=R.2) %>%

replace (LDist=="1LG6","chr1") %>%

replace(LDist=="2LG1","chr2") %>%

replace(LDist=="3LG5","Chr3") %>%

replace(LDist=="4LG4","chr4") %>%

replace(LDist=="5LG3","chr5") %>%

replace(LDist=="6LG2","chr6") %>%

replace(LDist=="7LG7","chr7")

* + - * + Después usaremos una función denominada LDit y sus modificaciones para dibujar los graficos de LD
        + La función de LDit fue creado por Jeoffrey en la Universidad de Davis y se puede encontrar el script original https://github.com/rossibarra/r\_buffet/blob/master/LDit.r
        + Para dibujar tanto los valores de R2 por cada unidad de distancia que la curva de decaimiento usaremos el siguiente script

LDit<-function(x,n){

d=(x$sitei-x$sitej); # si el LD ha sido estimado por TASSEL se puede sustituir por d=(X$Dist\_bp); #

y<-x[order(d),]

ld=y$rsq

row\_bp=unique(x$sitei)

col\_bp=unique(x$sitej)

segsites=length(row\_bp)+1

vlen=length(x$rsq)

Er<-function(C\_,d){

length(d)

res<-((10+C\_\*d)/((2+C\_\*d)\*(11+C\_\*d)))\*(1+((3+C\_\*d)\*(12+12\*C\_\*d+(C\_\*d)^2)/(n\*(2+C\_\*d)\*(11+C\_\*d))))

return(res)

}

nlm<-nls(ld~Er(C\_,d[order(d)]),start=list(C\_=0.01))

plot(d[order(d)],ld,cex=.3,pch=20,col="grey",xlab="distance",ylab=expression(r^2))

C\_<-summary(nlm)$coefficients[1]

lines(d[order(d)],Er(C\_,d[order(d)]),col="red",lwd=3)

}

LDit(LDist,1107940)

* + - * + Si solo queremos dibujar la curva de decaimiento de LD usamos esta variante de la function LDit:

LDit2<-function(x,n){

d=(x$Dist\_bp);

y<-x[order(d),]

ld=y$rsq

row\_bp=unique(x$sitei)

col\_bp=unique(x$sitej)

segsites=length(row\_bp)+1

vlen=length(x$rsq)

Er<-function(C\_,d){

length(d)

res<-((10+C\_\*d)/((2+C\_\*d)\*(11+C\_\*d)))\*(1+((3+C\_\*d)\*(12+12\*C\_\*d+(C\_\*d)^2)/(n\*(2+C\_\*d)\*(11+C\_\*d))))

return(res)

}

nlm<-nls(ld~Er(C\_,d[order(d)]),start=list(C\_=0.01))

plot(d[order(d)],ld,pch=NA,xaxt="none",xlab="",ylab="",cex.axis=1.25)

axis(1,seq(0,8\*10^7,2\*10^7),labels=c("0","20","40","60","80"),cex.axis=1.25)

C\_<-summary(nlm)$coefficients[1]

lines(d[order(d)],Er(C\_,d[order(d)]),col="red",lwd=3)

mtext(side=1,line=2.5, "Distance (Mbp)", font=2,cex=1.8)

mtext(side=2, line=2, expression(LD~(r^{2})), font=2,cex=1.8)

}

LDit2(LDist,1107940)

{setEPS()

postscript(file="GenPea\_LDdecayGraph2.eps",onefile=TRUE,width=18/2.54, height=12/2.54,title="LD Decay")

LDit2(LDist,1107940)

dev.off()

}

* + - * + Finalmente para estimar la distancia que corresponde a la mitad de decaimiento de LD (LD50) usamos el siguiente script creado por Fabio Marrioni y descrito en la pagina <https://www.r-bloggers.com/2011/08/estimate-decay-of-linkage-disequilibrium-with-distance>

distance <- LDist$Dist\_bp

LD.data<- LDist$rsq

n<-1107940

HW.st<-c(C=0.1)

HW.nonlinear<-nls(LD.data~((10+C\*distance)/((2+C\*distance)\*(11+C\*distance)))\*(1+((3+C\*distance)\*(12+12\*C\*distance+(C\*distance)^2))/(n\*(2+C\*distance)\*(11+C\*distance))),start=HW.st,control=nls.control(maxiter=100))

tt<-summary(HW.nonlinear)

new.rho<-tt$parameters[1]

fpoints<-((10+new.rho\*distance)/((2+new.rho\*distance)\*(11+new.rho\*distance)))\*(1+((3+new.rho\*distance)\*(12+12\*new.rho\*distance+(new.rho\*distance)^2))/(n\*(2+new.rho\*distance)\*(11+new.rho\*distance)))

maxld<-max(fpoints)

h.decay<-maxld\_chr01/2

half.decay.distance<-distance[which.min(abs(fpoints-h.decay))]

* + - * + Según esta fórmula para los marcadores SillicoDArT, la distancia necesaria para llegar al LD50 es de 477644bp es decir 0,48Mbp.
        + Además de estimar el LD del genoma entero, he estimado también el decaimiento de ligamiento por cada cromosoma por separado. Según este análisis, obtenemos un LD50 de:

Chr1: 0,60 Mbp

Chr2: 0,32 Mbp

Chr3: 0,60 Mbp

Chr4: 0,25 Mbp

Chr5: 0,54 Mbp

Chr6: 1,05 Mbp

Chr7: 0,25 Mbp

* + - * Estimación LD para la base de datos SNP

**E.4. Estimación de la estructura de la población alternativa (uso de Chromopainter y GLobeTrotter)**

* + - * Para consolidar la estructura de la población determinada por STRUCTURE y las PCA, vamos a usar otro programa como Chromopainter y globetrotter.
      * Chromopainter “pinta” los sitios de recombinación sobre los cromosomas para reconstruir los patrones de recombinación y asi determina la estructura de una población y como ha evolucionado conjuntamente con el programe GLobetrotter y FineStructure.
      * Para usar este programa, hay que filtrar los marcadores en LD y tener en fase los haplotipos.
      * Para filtrar los marcadores en LD usamos PLink y para reconstruir la fase de los haplotipos usaremos PHASE 2.1 que es el único programa de este tipo que funciona en “Windows”.
      * El formato que necesita PHASE para funcionar es muy especifico asi que primero tenemos que poner en formato el archivo LD pruned que obtuvimos con Plink para estimar la estructura de la población con STRUCTURE.
      * Una opción para formatear los ficheros en formato de PHASE es utilizar el comando “--recode fastphase” en plink. El comando sería:

plink --bfile *{nombre del fichero binario sin extensión*} --autosome-num <número del ultimo cromosoma> --recode fastphase --out {*nombre del fichero creado en formato .phase.in*}

Ejemplo:

plink --bfile Peahetpruned --autosome-num 9 --recode fastphase --out Peahetphased

* + - * Esta función crea un fichero por cada cromosoma. Una vez transformado hay que abrir en un editor de texto cada uno de los ficheros obtenidos y añadir una línea antes de los genotipos indicando que tipo de marcadores (LocusType) usamos (S por SNP o M por Microsatelites). También hay que eliminar los espacios adicionales en la fila de los nombres de los genotipos. Este proceso codifica cada genotipo en dos líneas y locus con
        + “A”: alelo principal
        + “C”: alelo minoritario
        + “?”: Missing