Universitat Autònoma de Barcelona

TOWARDS INCREASING GENETIC VARIABILITY AND IMPROVING FRUIT QUALITY IN PEACH USING GENOMIC AND BIOINFORMATIC TOOLS

Octávio Manuel Ribeiro Serra

TOWARDS INCREASING GENETIC VARIABILITY AND IMPROVING FRUIT QUALITY IN PEACH USING GENOMIC AND BIOINFORMATIC TOOLS

OCTÁVIO MANUEL RIBEIRO SERRA

TESIS DOCTORAL 2017

Tesi Doctoral

Universitat Autónoma de Barcelona

Facultat de Biociències

Departament de Biologia Animal, de Biologia Vegetal i d'Ecologia

Doctorat en Biologia i Biotecnologia Vegetal

TOWARDS INCREASING GENETIC VARIABILITY AND IMPROVING FRUIT QUALITY IN PEACH USING GENOMIC AND BIOINFORMATIC TOOLS

Memòria d'investigació presentada per Octávio Manuel Ribeiro Serra per optar al títol de doctor per la Universitat Autónoma de Barcelona

Octávio Manuel Ribeiro Serra

Dr. Pere Arús	Dr. Werner Howad	Dra. Roser Tolrà
Director de Tesi	Director de Tesi	Tutora de Tesi

Bellaterra, Febrer 2017

Na vida mais vale levarmos connosco a certeza de onde partimos, a nossa identidade, as nossas raízes, porque só assim saberemos onde podemos chegar. São essas raízes que nos ajudam a crescer, são esses valores que nos impulsam. Podemos estar longe, mas estando seguros das nossas raízes sentímo-nos como em casa. É nelas que pensamos durante a nossa viagem. E quando chegarmos ao destino, saberemos olhar para trás e reconhecer, que sem elas não teríamos conseguido.

Esta tese é dedicada aos meus pais, porque lhes devo tudo. Obrigado pelas raízes.

Acknowledgments

This thesis would not be possible without the collaboration and friendship of several people. The feeling of gratitude is enormous by the time I write these lines and I hope I do not forget to mention anyone.

I would like to thank to my supervisors. First to Dr. Pere Arús, for the possibility to do my PhD in his group and for the role he played during these years. For the brainstorming meetings and the new ways of thinking that he imparted, which I´m sure will be extremely useful for me in the future. Also for the knowledge transmitted during this period and for the criticism that always leads to perfection. I also thank to Dr. Werner Howad for his help and support in this work. Thank you both.

To Dra. Amparo Monfort, for accepting me as a master student back in 2012, which allowed me to make contact with crop genomics for the first time and ultimately opened the doors for my PhD. Also for her support and help during the period of this thesis. Thank you very much.

To Roser Tolrà and Charlotte Poschenrieder for their help with all academic procedures.

A special thanks to Dr. Ibán Eduardo, not only for the healthy discussions on most important topics of this thesis, but mostly for the friendship. For his support when I started this PhD and also for the good moments shared during fieldwork and outside of work. You are an older brother to me, and I will be forever grateful.

To Lara Pereira, a great partner and a great friend. Thank you for all the conversations and good moments shared both in and out of work. Our friendship is one of the best things I will take with me after this thesis.

To Dr. Jose Manuel Donoso, for his friendship and for his knowledge and his help at the beginning of my doctorate. Also for the great work developed before this thesis that allowed me to reach my objectives.

To Dr. Konstantinos Alexiou and Marc Tormo, for their patience, for their knowledge and for the great work developed with the resequence project. To Dr. Jordi Giné and all postharvest lab from Fruitcentre for their collaboration in the SMF phenotyping.

To all pollination teams of seasons 2014, 2015 and 2016. Your help was crucial to achieve the results presented here in the MAI project, and to finish the collections of ILs in next years.

To the Rosaceae group and all researchers and post-docs from the department that, in one way or another, helped me during this thesis. To all partners from 3.01 and all PhD students, former or new, that may have helped me in this work.

To the in vitro department for the help with embryo rescue, mainly to Elena Del Blanco and Ramón Dolcet.

To all the technician staff from lab 3.03, Esteve Collell, Àngel Montejo, Fuensi, Vanessa, Dani and Joana. Thanks for your help since my master and for all the good moments shared ever since.

To Pilar Fontanet and the greenhouse staff at CRAG for the maintenance of plants.

To all IRTA field staff in Gimenells, Mollerusa, Cabrils and Torre Marimon. Your work was crucial to the development of this project and your help with all field tasks are really appreciated.

To my professors from UTAD for the good teaching skills and for all the knowledge transmitted during the Bachelor and Master. All the good background knowledge was very relevant to me.

To all my college friends, Ivo Pavia, Andreia Mendes, Ana Costa, Jorge Ferreira, Joana Fernandes, Ágata Carvalho. Thank you so much for the biannual meetings and the good moments shared since almost 10 years ago. Thanks to Joana Bastos for the cover image.

To all my friends back home, for the incredible moments in summer and Christmas.

A special thanks to Nídia, for the adventure started in 2012 when we decided to move to Barcelona, for her support, for her love, and for her commitment. It was key to have someone side by side these times. I am forever thankful to have met you <3.

To my father, my mother and my sister, for all the support during this thesis. Thanks to my parents for the education and for the possibility to reach this objective. Special thanks to my sister for being able, somehow, of living without her older brother close by since 11. You were an essential source of motivation to accomplish this project. This work belongs to you.

Table of contents

Abbreviations

a – Additive effect

AFLP – Amplified fragment length polymorphism

Ak – 'Armking'

avg. – Average

BAC – Bacterial artificial chromosome

BAM – Compressed SAM

BC – Backcross population

bp – Base pair

Bt – 'Bigh Top'

Bt×Ak – 'Big Top' × 'Armking' population

Bt×Nr – 'Big Top' × 'Nectaross' population

Chi – Chinese cultivars and landraces

cm – Centimeters

CO – Crossover

COR – Crossover region

CP – Cross pollination

CRAG – Center for research in agricultural genomics

CTAB – Cetrimonium bromide

DH – Double haploid

DNA – Deoxyribonucleic acid

DSB – Double strand break

ERS – Economic research service

EST – Expressed sequence tag

FL – Firmness loss

FL3d – Firmness loss three days after harvest

FL5d – Firmness loss five days after harvest

FRP – Fill recpoints

g – Grams

G – Linkage group

GBA – Genotype by absence

Gbp – Giga base pairs

GC – Gene conversion

IAA – Indol acetic acid

IBD – Identical by descent

IL – Introgression line

IL HET – Introgression line in heterozygosis

IL HOM – Introgression line in homozygosis

IM – Interval mapping

IRTA – Institut de reserca I tecnologia agroalimentaries

Kbp – Kilo base pairs

Kg – Kilograms

Land – Oriental and Occidental landraces

LOD – Logarithm of odds

M – Million (tonnes)

MAI – Marker assisted introgression

MAS – Marker assisted selection

Mbp – Mega base pairs

MD – Maturity date

MF – Melting flesh

mm – Milimeters

N – Newtons

n.s. – Not significant

NCO – Noncrossover

NIL – Near isogenic line

NMF – Non melting flesh

Nr – 'Nectaross'

ºC – Celsius degrees

Occ – Occidental cultivars

Or – Oriental cultivars

PCR – Polymerase chaine reaction

Pp – Peach chromosome

PPV – Plum pox virus

prIL – Pre introgression line

QTL – Quantitative trait *locus*

QTLs – Quantitative trait l*oci*

R^2 – Percentage of phenotypic variance

RAPD – Random amplified polymorphic DNA

RefE1 – Reference genome E1

RefE2 – Reference genome E2

RFLP – Restriction fragment length polymorphism

RIL – Recombinant Inbred Lines

RNA – Ribonucleic acid

SA – SNP_IGA_

SAM – Sequence alignment/map

SC – Synaptonemal complex

sd – Standard deviation

SDSA – Synthesis-dependent strand annealing

SH – Stony hard

SMF – Slow melting flesh

SNP – Single nucleotide polymorphism

SR – Slow ripening

SSR – Simple sequence repeat or microsatellite

SV – Structural variation

SW – Shapiro-wilk

T×E – 'Texas' × 'Earlygold' population

T1E – ('Texas' × 'Earlygold') × 'Earlygold'population

Ta – Annealing temperature

TPA – Textural profile analysis

TSS – Transcription strat site

TTS – Transcription termination site

USDA – United states department of agriculture

VCF – Variant call format

List of Figures

List of Tables

Resumo

O pessegueiro é uma das espécies frutais de maior relevo a nível mundial, com uma adaptação a uma grande diversidade de climas, e cuja produção global duplicou nas últimas décadas. Aumentar o consumo de pêssegos implica aumentar a sua qualidade, um desafio tendo em conta a sua vida pós-colheita. Para além disso, o pessegueiro é uma espécie com baixa variabilidade genética, o que representa outro obstáculo para o seu melhoramento genético. Outros factores que podem condicionar a produção deste fruto são as alterações climatéricas, a globalização do mercado e as mudanças nos hábitos alimentários dos consumidores. Para superar estes desafios será necessário implementar novas estratégias para explorar a pouca variabilidade existente dentro da espécie e introduzir novos genes de outras espécies silvestres ou cultivadas próximas, que são os objectivos desta tese. Em primeiro lugar estudámos a base genética da maduração lenta do fruto (SMF), uma característica em que a polpa se mantém firme durante mais tempo que a polpa macia (MF) que actualmente domina o mercado, resultando numa vida pós-colheita mais extensa. Em Espanha, a variedade SMF de referência é a nectarina 'Big Top', que se usou como progenitor feminino para gerar duas descendências F_1 com progenitores MF. Construiu-se um mapa genético de alta densidade com SNPs e mediu-se a SMF em ambas descendências. Encontrámos dois *loci* de caracteres quantitativos (QTLs) que co-localizam com dois QTLs de época de maduração nos grupos de ligação 4 (G4) e G5, explicando cada um deles >20% da variabilidade fenotípica observada. O QTL no G5 foi encontrado apenas em 'Big Top' e é provavelmente o agente causador do seu comportamento SMF. Em segundo lugar, ensaiámos uma nova estratégia que chamámos introgressão assistida por marcadores (MAI) para introduzir nova variabilidade procedente de outras fontes exóticas em espécies lenhosas em pouco tempo, usando marcadores moleculares para acelerar o processo. Como resultado adicional desenvolvemos um conjunto de linhas de introgressão (ILs) de amendoeira no fundo genético de pessegueiro, muito útil para o estudo genético de caracteres de herança complexa. Por fim, estudámos o processo de recombinação usando dados de resequência de ADN dos descendentes de um cruzamento entre um híbrido de amendoeira × pessegueiro e o seu progenitor masculino (o pessegueiro 'Earlygold'), o que permitiu comparar a recombinação inter e intraespecífica. Entender os factores que controlam a formação de sobrecruzamentos (COs) é essencial para o controlo da introgressão de um dador exótico num germoplasma cultivado. Desenvolvemos um programa bioinformático

para detectar SNPs e indels, genotipámos *in silico* cada individuo e determinámos a posição dos COs. Um primeiro resultado foi a distribuição dos Cos, que foi heterogénea ao longo do genoma, mas semelhante em ambas meioses intra e inter específicas, apesar de que a recombinação interespecífica foi muito menos frequente. Localizámos as regiões dos COs e encontrámos algumas com mais COs que o esperado (hotspots), detectando também motivos de ADN associados a ambas regiões. Outros eventos de recombinação chamados "noncrossovers" detectaram-se na meiose do híbrido com uma frequência aproximada cinco vezes superior aos COs. Finalmente associámos a baixa recombinação do progenitor híbrido com a baixa fertilidade do pólen, indicando que a diversidade da sequência de ADN é uma causa possível para o isolamento reprodutivo em plantas. Em conjunto, os nossos resultados proporcionam informação nova sobre a herança de caracteres chave para o melhoramento do pessegueiro, ferramentas para a análise detalhada de caracteres de herança complexa, uma estratégia de melhoramento para a introgressão de genes de interesse procedentes de outras espécies y dados sobre o funcionamento da recombinação interespecífica. Ao mesmo tempo produzimos informação sobre ferramentas moleculares que permitem aplicar estes conhecimentos para a obtenção de variedades melhoradas

.

Resum

El préssec és un dels fruiters més conreats al món, la producció del qual s'ha duplicat en els darrers vint anys. Per augmentar el seu consum cal millorar-ne la qualitat, un repte difícil ja que es tracta d'una fruita de curta vida postcollita. Addicionalment, el préssec te poca variabilitat genètica el que implica que les seves possibilitats de millora genètica són limitades. Altres elements que condicionen el futur del conreu del préssec son el canvi climàtic, la globalització del mercat i els canvis d'hàbits alimentaris dels consumidors. Per a encarar aquests reptes caldrà implementar noves estratègies per a explotar millor la variabilitat existent dins l'espècie i per a introduir nous gens d'altres espècies silvestres o conreades properes, que son els objectius d'aquesta tesi. Primer, vam estudiar la base genètica del caràcter de maduració lenta del fruit (SMF), la carn del qual es manté ferma més temps que la carn tova (MF) que actualment domina el mercat, el que resulta en una vida postcollita més llarga. La referència pel caràcter SMF a Espanya és la varietat de nectarina 'Big Top', que es va usar per a generar dues descendències F1 amb pares MF. Vam construir mapes genètics d'alta densitat amb SNPs i vam mesurar el caràcter SMF en ambdues descendències. Els dos loci de caràcters quantitatius (QTLs) trobats colocalitzaren amb dos QTLs d'època de maduració en els grups de lligament 4 (G4) i G5, explicant >20% de la variabilitat fenotípica observada cadascun. El QTL del G5 només es trobà a 'Big Top' i és possiblement el causant principal del seu comportament SMF. En segon lloc, assajarem una nova estratègia, que anomenàrem introgressió assistida amb marcadors (MAI), per a introduir nova variabilitat d'origen exòtic en espècies llenyoses en un curt període de temps - dues generacions després de l'híbrid - usant marcadors moleculars per a accelerar el procés. Como resultat lateral vam desenvolupar un joc de línies d'introgressió (ILs) d'ametller en el fons genètic del presseguer, una eina útil per a l'anàlisi genètica de caràcters d'herència complexa. Finalment, estudiàrem el procés de recombinació usant dades de reseqüència d'ADN dels descendents d'un encreuament entre un híbrid ametller × presseguer i el seu genitor masculí (el presseguer 'Earlygold'), el que permeté comparar la recombinació inter i intraespecífica. Entendre quins factors controlen la formació dels entrecreuaments (COs) és essencial pel control de la introgressió d'un donant exòtic al germoplasma cultivat. Desenvolupàrem un programa bioinformàtic per a detectar SNPs i indels, genotipàrem in silico cada individu i determinàrem la posició dels COs. Un primer resultat fou una distribució heterogènia de COs al genoma, però semblant en meiosis intra i

interespecífiques, encara que la recombinació interespecífica va ser molt menys freqüent que la intraespecífica. Localitzàrem les regions dels COs i en vam trobar algunes amb més COs dels esperats (hotspots), detectant motius d'ADN associats a aquestes regions. Altres esdeveniments de recombinació anomenats "noncrossovers" es van detectar a la meiosi de l'híbrid a una freqüència aproximadament cinc vegades més alta que la dels COs. Finalment, associàrem la baixa recombinació en l'híbrid amb la baixa fertilitat del seu pol·len, indicant que la diversitat de seqüència del ADN es una possible causa d'aïllament reproductiu en plantes. En conjunt, els nostres resultats proporcionen informació nova sobre l'herència de caràcters clau per a la millora del presseguer, eines per a l'anàlisi fi de caràcters d'herència complexa, una estratègia de millora per a la introgressió de gens d'interès procedents d'altres espècies i dades sobre el funcionament de la recombinació interespecífica. Al mateix temps, hem produïm informació sobre eines moleculars que permeten aplicar aquests coneixements a l'obtenció de varietats millorades.

Resumen

El melocotón es uno de los más importantes cultivos frutales del mundo, adaptado a una gran diversidad de climas, cuya producción se ha duplicado en las últimas dos décadas. Aumentar su consumo implica mejorar su calidad, un difícil reto para un fruto que tiene un corto período poscosecha. Además, el melocotonero tiene poca variabilidad genética lo que supone una dificultad añadida para su mejora genética. Otros elementos que condicionan el futuro del cultivo del melocotón son el cambio climático, la globalización del mercado y los cambios de hábitos alimentarios de los consumidores. Para resolver estos retos será necesario implementar nuevas estrategias para explotar mejor la variabilidad existente dentro de la especie e introducir nuevos genes de otras especies silvestres o cultivadas próximas, que son los objetivos de esta tesis. Primero, estudiamos la base genética del carácter de maduración lenta del fruto (SMF), cuya carne se mantiene firme más tiempo que la carne blanda (MF) que actualmente domina el mercado, resultando en una vida poscosecha más larga. La referencia para los melocotoneros SMF en España es la nectarina 'Big Top', que se usó como parental femenino para generar dos descendencias F1 con parentales MF. Se construyó un mapa genético de alta densidad con SNPs y se midió el carácter SMF en ambas descendencias. Encontramos dos loci de caracteres cuantitativos (QTLs) que colocalizaron con sendos QTLs de época de maduración en los grupos de ligamiento 4 (G4) y G5, explicando >20% de la variabilidad fenotípica observada cada uno. El QTL del G5 solo se halló en 'Big Top' y es posiblemente el causante principal de su comportamiento SMF. En segundo lugar, ensayamos una nueva estrategia que llamamos introgresión asistida por marcadores (MAI) para introducir nueva variabilidad procedente de fuentes exóticas en especies leñosas en poco tiempo, usando marcadores moleculares para acelerar el proceso. Como resultado lateral desarrollamos un juego de líneas de introgresión (ILs) de almendro en el fondo genético del melocotonero, muy útil para el análisis genético de caracteres de herencia compleja. Finalmente, se estudió el proceso de recombinación usando datos de resecuencia de ADN de los descendientes de un cruzamiento entre un híbrido almendro × melocotonero y su parental masculino (el melocotonero 'Earlygold'), lo que permitió comparar la recombinación inter e intraespecífica. Entender qué factores controlan la formación de los sobrecruzamientos (COs) es esencial para el control de la introgresión de un donante exótico al germoplasma cultivado. Desarrollamos un programa bioinformático para detectar SNPs e indels, genotipamos in silico cada individuo y

determinamos la posición de los COs. Un primer resultado fue que la distribución de COs fue heterogénea en el genoma, pero parecida en meiosis intra e interespecíficas, aunque la recombinación interespecífica fue mucho menos frecuente que la intraespecífica. Localizamos las regiones de los COs y encontramos algunas con más COs de los esperados (hotspots), detectando motivos de ADN asociados a ambas regiones. Otros eventos de recombinación llamados "noncrossovers" se detectaron en la meiosis del híbrido a una frecuencia aproximadamente cinco veces mayor que los COs. Finalmente, asociamos baja recombinación en el híbrido con baja fertilidad del polen, indicando que la diversidad de secuencia del ADN es una posible causa de aislamiento reproductivo en plantas. En conjunto, nuestros resultados proporcionan información nueva sobre la herencia de caracteres clave para la mejora del melocotonero, herramientas para el análisis fino de caracteres de herencia compleja, una estrategia de mejora para la introgresión de genes de interés procedentes de otras especies y datos sobre el funcionamiento de la recombinación interespecífica. Al mismo tiempo, producimos información sobre herramientas moleculares que permiten aplicar estos conocimientos a la obtención de variedades mejoradas.

Summary

Peach is a major fruit species, cultivated worldwide, with an outstanding adaptation to contrasting climate conditions, which world production has doubled in the last two decades. Increasing peach consumption requires enhancing fruit quality, a challenging objective for a fruit that has a short postharvest life. An important shortcoming for peach breeding is its low level of variability, narrowing the possibilities for its improvement. Other elements that may further condition peach production and breeding are climate change, the globalization of peach market and the changing eating habits of the population. Facing these challenges requires implementation of new strategies allowing a better exploitation of the variability that still exists inside the species and the introduction of new variability using other cultivated or wild relatives. In this work we aim to contribute in the development of such novel approaches. First we studied the genetic basis of the slow melting flesh (SMF) trait, characterized by the longer postharvest life of fruits, with higher firmness values after harvest than regular melting flesh (MF) peaches. SMF is present in some North American peach and nectarine cultivars, one of which ('Big Top'), has become a reference for nectarine production in Spain. We studied two F_1 populations using 'Big Top' as female parent, built linkage maps using the 9k peach SNP chip, and measured SMF. Quantitative trait loci (QTL) analysis allowed us to find two consistent QTLs for SMF co-localizing with maturity date QTLs in linkage group four (G4) and G5 that explained each >20% of phenotypic variability. The QTL on G5 was exclusive to 'Big Top', which can be the cause of its specific SMF behavior. In a second topic, we tested a new strategy, marker assisted introgression (MAI), to introduce new variability from exotic sources into cultivated perennial species in a short timeframe, using molecular markers to accelerate the process. As a side result we developed a set of introgression lines (ILs) of almond in the genetic background of peach, an optimal tool for genetic analysis of complex traits. In the final topic of this thesis we aimed to study the recombination process in wide crosses (almond × peach) in comparison with that of intraspecific crosses (peach), using resequence data of a cross between an almond × peach hybrid and its peach parent ('Earlygold'). Understanding which factors control the occurrence of crossovers (COs) is critical to control the introgression process from an exotic donor to elite cultivated materials. We developed a bioinformatics pipeline to detect SNP and indel variants, *in silico* genotyped each individual, and determined the CO positions using the variants called. We found that

the distribution of COs was heterogeneous in the genome, but similar in intra and interspecific meioses, and that a strong reduction of recombination occurred at the interspecific level, which we associated with DNA sequence divergence. We studied the CO regions, found some with high CO frequency (hotspots) and identified DNA motifs associated with these regions. Other recombination events such as noncrossovers (NCOs) were also detected for the hybrid meiosis about five times more frequently than COs. Finally, we associated low recombination in the hybrid with low pollen fertility, suggesting DNA sequence divergence as a possible cause for a gradual process of reproductive isolation in plants. Overall, our results supply new information on the inheritance of key commercial peach traits, tools for the fine analysis of complex characters, a breeding strategy for the enrichment of peach genome with valuable genes from other species, and data on how interspecific recombination proceeds. At the same time, we provide molecular tools to facilitate the translation of this knowledge into new and improved cultivars.

1. Main introduction

1.1 - Peach species and the *Prunus* genus

1.1.1 - Taxonomy and phylogeny in Prunus

The Rosaceae is an important plant family with approximately 90 genera and 3,000 species which includes a large number of economically important crop and ornamental species. The most recent classification for the family (Potter et al. 2007) divides it into three sub families: *Rosoideae*, *Dryadoideae* and *Spiraeoideae*, although the name *Spiraeoideae* was corrected in 2011 to *Amygdaloideae* based on changes in the International Code of Nomenclature for Algae, Fungi and Plants (McNeill et al., 2012). The *Prunus* genus (*Prunus* L.) belongs to the *Amygdaloideae* subfamily and is composed by 200 species of trees and shrubs, some of them economically relevant fruit and nut crops. The infrageneric classification of *Prunus* by Rehder (1940) is the most accepted nowadays and consists in five subgenera: *Amygdalus* (peaches and almonds), *Cerasus* (cherries), *Prunus* (plums), *Laurocerasus* (evergreen laurel cherries) and *Padus* (bird cherries) (Chin et al. 2014) (Figure 1.1). The cultivated peach [*Prunus persica* (L.) Batsch] shares the *Amygdalus* subgenus with almond (*P. dulcis* (Mill.) D.A.Webb) and also with several wild relatives like *P. davidiana* (Carr.) Franch, *P. mira* Koehene, *P. ferganensis* (Kost and Rjab) Kov. & Kost and *P kansuensis* Rehd (Byrne et al. 2012; Chin et al. 2014). Their systematic classification is as follows:

Kingdom: *Plantae*
Division: *Magnoliophyta*
Class: *Magnoliopsida*
Order: *Rosales*
Family: *Rosaceae*
Subfamily: *Amygdaloideae*
Tribe: *Amygdaleae*
Genus: *Prunus*
Subgenus: *Amygdalus*

All the previous peach relatives are originated from China, considering also some regions of Nepal, India and other peripheral countries. The genetic proximity among all of these species makes it possible to produce fertile hybrids between peach and them. Other *Prunus* distantly related may also be hybridized with peach, but produce generally sterile hybrids (Scorza and Okie, 1991).

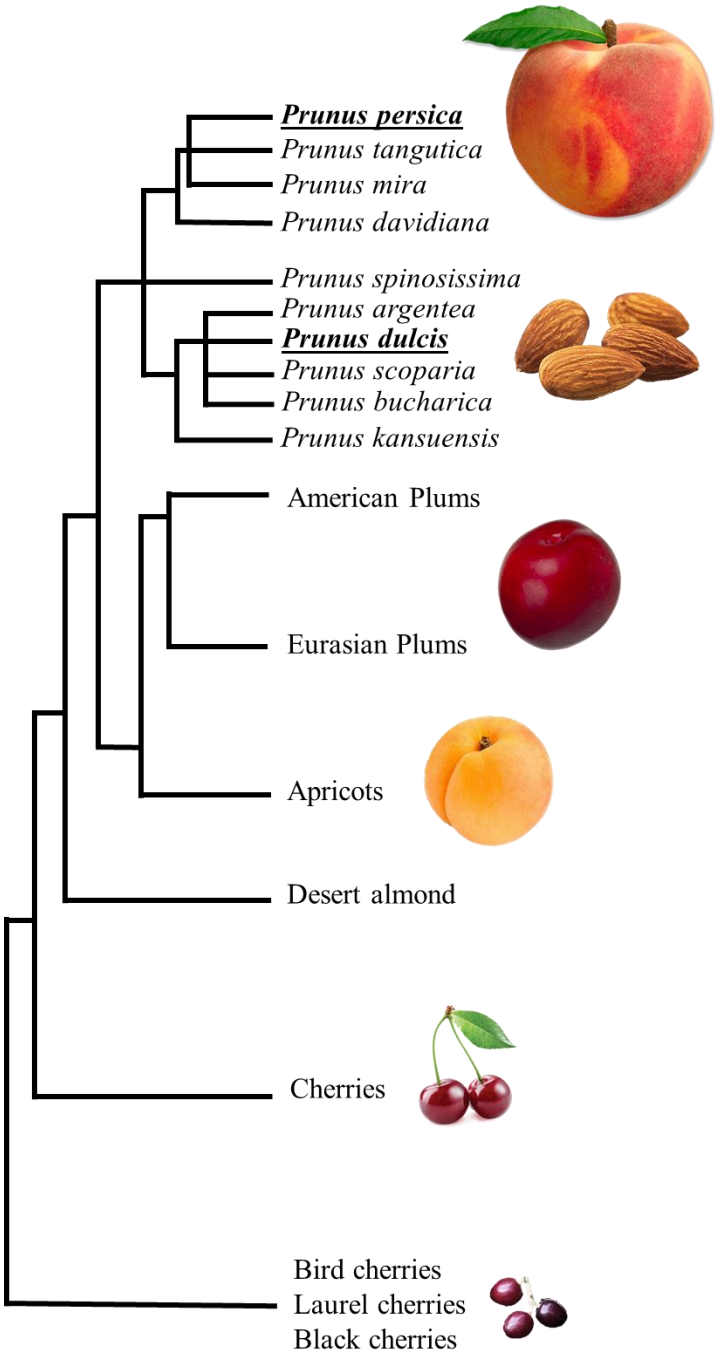


Figure 1.1 – The phylogeny of *Prunus* based on the plastid DNA sequence of four genes. Adapted from Chin et al. (2014).

The majority of these interspecific crosses were performed in the past mainly for the development of rootstocks for peach. Interspecific crosses between almond and peach were historically done also for rootstock development, for use primarily in calcareous soils, since these hybrids often tolerate iron chlorosis. In the last years a growing interest has emerged in the use of these related species for peach breeding, mainly as a source of resistances to plum pox virus (PPV) and powdery mildew (Gradziel 2003; Martínez-Gómez et al. 2004; Foulongne et al. 2003). Almond has become an interesting choice for introgressing new genes into peach, mainly due to the high variability of the species. Recently, Donoso et al. (2016) have evidenced the potential of almond species to enrich the peach gene pool, not only for disease resistance traits but also for fruit quality traits.

1.1.2 - Peach origin and dissemination

The botanical name of peach (*P. persica*) was attributed in 1801 by August Johann Georg Karl Batsch (1761-1802), a German naturalist, and refers to the geographic location of Persia (today's Iran) mainly because peach was brought into Europe from there. By the end of the nineteen century, some authors suggested that peach was most likely native from China due to the higher variability of the species in that country. Nowadays there is enough evidence to establish the origin of peach in China, specifically in the Northwest part of the country, from Lanzhou (Gansu Province) to the west until the Tarim basin, a region flanked by the mountains of Tian Shan (North), Pamir (West) and Karakoram in the South (Faust and Simon, 1995). Recently, the discovery of fossils of peach endocarps (*P. kunmingensis*) with approximately 2.5 million years old in the region of Kunming (Yunnan Province) by Su and colleagues (2015) proves the existence of fruits similar to modern cultivated peach in China much before the presence of humans in the country, which unequivocally establishes the origin of peach in that Asian country.

Peach domestication and cultivation started at least 4,000 years ago, when Chinese started to recognize the value of this fruit and began to select and cultivate it (Wang et al. 1985; Huang et al.2008). Since then, peach has become a part of Chinese culture. The different climatic conditions of the country, as well as the different traditions and cultures of each

region, led to the early selection of a wide variety of plants for their adaptability to climate, fruit traits or ornamental characteristics (Wang, 1985). The dissemination of peach from China to the rest of the world was started 3,000 years ago, when peach was moved to all temperate and subtropical climates within the Asian continent (Byrne et al. 2012), mainly using the ancient communication routes. About 2,000 years ago, the first peaches arrived to Japan (Yamamoto et al. 2003) and probably by the same time, to Persia. During the Han dynasty (207 BC – AD 220), the network of trade routes throughout Asia was a key factor to connect China to the Mediterranean Sea. The silk route allowed the trade of goods between China and Persia starting from 105 BC, and peach probably arrived to Persia by that time along with other goods coming from the Chinese empire. The Romans occupied Syria in the 70 BC and established the Roman-Persian connection, which allowed the appearance of silk in Rome – and probably peaches – by that time. Alternative theories to the entrance of peaches into Europe do exist (Faust and Timon 1995), but despite the route and the date of arrival, the majority of authors do agree that the Romans were the main responsible for the dissemination of peaches throughout all Europe (Scorza and Okie, 1991; Faust and Timon, 1995).

In the 16th century peaches arrived to the American continent for the first time, taken by the Portuguese and Spanish explorers on their maritime explorations. The plant was rapidly adopted and disseminated by the natives and by the 19th century it was already cultivated throughout the entire American country. In 1850 a direct shipment of seedlings originated from the southern group of Chinese cultivars was sent to America, which represented a second entry of peach germplasm into the continent and contributed to the extensive cultivation of the very well-known varieties descendent from the ‘Chinese cling’ (Scorza and Okie, 1991).

1.1.3 - Peach botany, production and consumption

P. persica is a tree species that can live up to 30 years, although for commercial uses trees with half that age are typically discarded due to lower production. (Bassi and Monet, 2008). Peach buds present a common layout of three buds per node, where the middle bud is vegetative and the lateral ones produce flowers. The internode length between varieties is diverse, and it can be as small as one cm in dwarf genotypes (Bassi and Monet,

2008). There are six growth habits (tree architectures) described for peach: standard, columnar or pillar, upright, open or spur-type, semi-dwarf or compact and weeping (Scorza et al. 2006). Different growth habits are mainly determined by the branch angle of the trees, and the overall architecture may influence fruit production as well as the pruning labor. Peach leaves emerge each year after anthesis and develop into adult lanceolate form. Several leaf traits were described in the past, mainly for leaf blade (Scott and Cullinan, 1942), narrow-width (Chaparro et al. 1994), serrated forms (Okie and Scorza, 2002), presence of glands at the margin of the blade (Connors, 1922) and leaf color (Blake, 1937). Peach flowers are hermaphroditic, with one pistil and 20 to 30 stamens. The flower typically has five petals, except in the case of double flower phenotypes where the petal number can vary from 12 to 24. Petals are normally separated and two types of corolla exist: showy (rose-shaped flowers with large petals) and non-showy (bell-shaped flowers with small petals). Petal color ranges from white to dark pink with a color pallet between these two extremes. Anthers are pink, unless there is male sterility, where yellowish or white anthers are observed (Bassi and Monet, 2008). Peach fruits are known to have three phases of growing during fruit development period, following a double sigmoidal curve: a first phase of cell division, followed by a stage of pit hardening and seed and embryo development, and a third phase of cell expansion (Bassi and Monet, 2008). When mature, peach fruit is fleshy and juicy, commonly with a globose form.

Peach is a species well adapted to contrasting climate conditions. Peach production is possible from humid climates in south China to dry conditions in Iran and California, and from cold climates in Canada to subtropical regions in Florida (Faust and Timon, 1995). Despite this high adaptability to different climates, the majority of varieties require at least 100 hours of chilling below 7º C and are sensitive to spring frosts (Hancock et al. 2008) whereas the incidence of high humidity can lead to an increase of diseases (Byrne et al. 2012).

Peach is one of the most important fruits worldwide. The production of peaches have doubled in the last two decades, mainly due to more efficient agricultural practices and the development of new varieties and rootstocks well adapted to specific climate conditions. The overall world production of peach reached almost 23 million tonnes in

2014 (FAOSTAT, 2017). In Spain, peach is the main deciduous fruit species, with a clear increase on production for the last decades (Iglesias, 2013). China is the top producer (Figure 1.2), with more than half of world total production (~12M tonnes), followed by Spain (1.5M tonnes) and Italy (1.3M tonnes). Asian countries taken together represented in 2014 more than 66% of the worldwide production, followed by Europe (20%) and America (10%) (FAOSTAT, 2017).

Despite the increase on overall peach production, the consumption of this fruit has been decreasing. According to the Economic research service (ERS) from the United States Department of Agriculture (USDA), the annual per person consumption of peaches in the United States peaked at 5.95 kg in the year 1980. In 2014 the consumption was 2.91 kg, 13% less than the previous year and 52% less than year 1980. The comparison of peach vs apple consumption in European countries has also shown some interesting consumer behaviors. In countries such as Spain and France, consumers eat more peaches (when season) than apples, which is in contrast with northern European countries such as Germany or Poland, and may be mainly explained by the higher availability of the fruit and the lower prices in southern countries (Konopacka et al. 2010). The decrease on peach consumption over the last decades has been associated mainly with two characteristics: (1) the lack of consumer understanding between mature and ripe peaches and (2) the presence of flesh browning and flesh mealiness (chilling injury and internal breakdown) in the fruits.

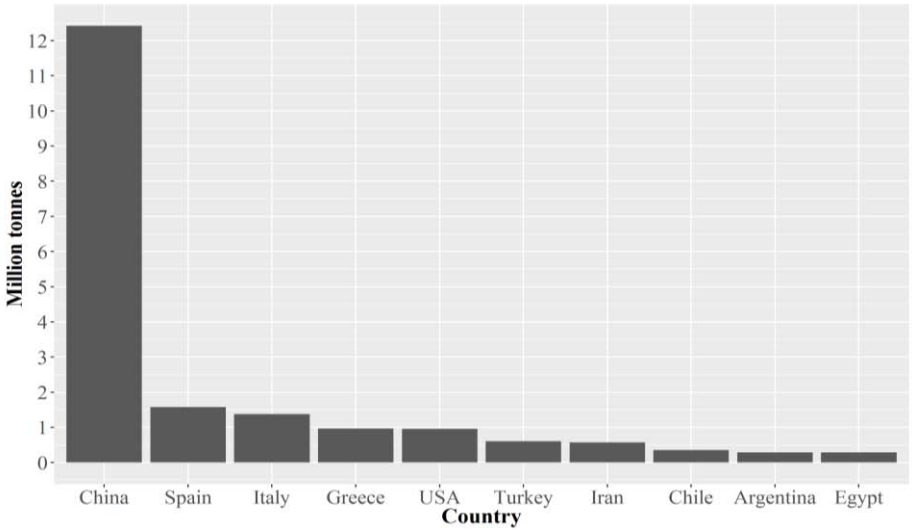


Figure 1.2 - The 10 most important producers of peaches and nectarines in the year 2014 (FAOSTAT, 2017).

1.1.4 - Genetic characterization of the species

P. persica is a diploid species with eight chromosomes (2n=*2x=16*) and has a small genomic complexity when compared with other closely related species such as European plum or sour cherry and also with more distant species as apple, pear or the cultivated strawberry, since all of them are polyploid. Peach has been one of the best genetically characterized species in the Rosaceae family since it has extensive genome regions conserved with apple and strawberry (Illa et al. 2011) and is often pointed out as a model species for the family (Shulaev et al. 2008), and also for tree species in general (Arús et al. 2012; The International Peach Genome Initiative 2013). In addition, the genome size of peach is relatively small (~230 Mbp) when compared with the majority of crop species, and is just 1.7× larger than Arabidopsis genome (The Arabidopsis genome initiative, 2000). Further genetic advantages of peach are the absence of a self-incompability system for the species, which permits the creation of F2 populations, and the short intergeneration period of 2-4 years that is much lower than the majority of fruit tree species (5-10 years). These characteristics facilitate and accelerate the generation of segregating populations for genetic mapping studies.

1.2 - Breeding in peach

1.2.1 - The history of peach breeding

Peach was domesticated in China about 4,000 years ago from wild populations of *P. persica*. It has been extensively cultivated and selected by the Chinese since then, mainly for fruit traits and ornamental characteristics. In Europe, during the industrial revolution of the 16th century, more people acquired substantial wealth and began to garden, which resulted in the release of some peach cultivars during this period (Hancock et al. 2008). In North America several cultivars of unknown parentage were released after the American Revolution, from 1770's to 1860's, including 'Early Crawford', 'Late Crawford' and 'Oldmixon Cling'. With the direct import of peaches from China to North America in 1850, several varieties based on the cultivar 'Chinese Cling' were released, and they became the most important germplasm to the development of fresh market cultivars in North America (Byrne et al. 2012). After the civil war 'Chinese Cling' was planted in Marshville, Georgia, by Samuel Rumph who later released two important varieties: 'Belle of Georgia' (or 'Belle') and 'Elberta' that probably were progeny of 'Chinese Cling'. Other cultivars released later, such as 'Hiley' and 'J.H. Hale' were in fact seedlings of 'Belle' and 'Elberta', respectively. This reduced group of cultivars formed the basis of almost all subsequent breeding activities in North America and also Europe (Hancock et al. 2008). The real peach breeding activities, taken by institutional breeding programs, were first established about 120 years ago in North America (Geneva, New York, 1895). Other breeding programs were started in Iowa (1905), Illinois (1907), California (1907), Ontario (1911), New Jersey (1914), Virginia (1914), Massachusetts (1918) and New Hampshire (1918). Other states also followed the tendency and started their own breeding programs in the following decades (Byrne et al. 2012). Private breeding programs started in California in 1930's (Faust and Timon 1995). The main objective of these breeding programs at that time was to develop peaches and nectarines with melting flesh for fresh market. In South America, breeding programs were initiated in Brazil (1950) and Mexico in the 1980's (Byrne et al. 2000), followed by Chile, Uruguay and Argentina. In Europe the first peach breeding program started in Italy in the 1920's and later in France, in the 1960's. These countries were followed by Spain, Romania, Serbia, Greece, Bulgaria, Ukraine and Poland. In South Africa and Australia breeding

programs also emerged in last decades with focus mainly to fresh market peaches (Byrne et al. 2012). In Asia, the first breeding program was started in Japan, about 60 years ago, followed by China in the 1970's and recently in Korea, India and Thailand (Byrne et al. 2000). Despite the great number of breeding programs, where each one has specific breeding objectives related with adaptation to local climate or selection of traits appreciated by the local consumers, there are some common traits that are present in every program goal´s. The selection for an increased fruit quality, best postharvest behavior, resistance to pest and disease, greater variety of fruit types and the adaptation to low-chill zones are some of the traits being selected in almost all breeding programs worldwide (Byrne et al. 2005). In last decades, the majority of the public breeding programs have been step by step replaced by private sector breeding programs, which already release the majority of new cultivars in the United States, Spain and France (Byrne et al. 2012).

1.2.2 - Peach breeding nowadays

The breeding of peach species have started long ago, at the beginning of the 20th century. Even before that period, peaches were extensively selected for desired traits around the world during its expansion. Some of the most important achievements of peach breeding have been the expansion of the areas of cultivation, the increase of the harvest period and the diversification of its market (Byrne et al. 2012). Nowadays peach breeders worldwide select for a great amount of traits, but some of these traits are being specifically targeted for their higher economic importance. Expanding the environmental ranges of peach cultivation is typically a goal for breeders, either to reduce the chilling requirements and allow cultivation in subtropical climates or to increase the frost damage tolerance by selecting for a bloom delay and expand the cultivation into colder climates (Hancock et al. 2008). Also important is the development of new varieties to increase the harvesting window, either for earlier varieties or later varieties, as well as the increase of fruit quality.

Improve the shelf life of peach fruits is probably one of the most important goals for the majority of breeding programs. Fruit texture is a trait highly correlated with storability. The melting flesh peaches (MF) are used for fresh market since they have the shortest shelf life. In contrast, non-melting flesh peaches (NMF) that present a harder flesh texture for a long time, are commonly used for canning (Ghiani et al. 2011). Other fruit textures like stony hard (SH) are characterized by higher firmness values during postharvest storage, probably due the absence of ethylene production (Haji et al. 2005). Another peach fruit texture, slow ripening (SR), is characterized by a non-softening of the fruit at all, even when external ethylene is applied. Thus in order to increase the postharvest life of peach fruits, breeders are focusing on producing varieties with higher firmness values, and those phenotypes presenting a delay on fruit softening may be extremely promising to extend fruit durability over MF standard range (Sansavini et al. 2006). In addition, the reduction of postharvest disorders related with long distance shipping is extremely important for southern hemisphere countries like Chile, South Africa and New Zealand. Fruit sensory traits are also being selected as an intent to captivate consumers. In European programs an effort is being made to recover the flavor and aroma of old local cultivars, whereas in China the main effort is to introduce European and American cultivars types but with the low-acid trait (Sansavini et al. 2006). Further interests of breeders worldwide are new fruit types, like blood-fleshed peaches (or anthocyaninless), as well as the Chinese flat peach phenotype (Sansavini et al. 2006).

The control of tree architecture and growth has also been the aim of several breeding programs, mainly because a growth habit compatible with high density orchards would optimize the yield and reduce management costs. Controlling the tree vigor would also allow to reduce pruning costs and facilitate harvesting (Sansavini et al. 2006). The consumers of 21th century, especially in developed countries, have special concerns about food that previous generations did not have, which need to be met with new breeding goals such as: (1) increasing the health, nutrition and organoleptic properties of fruit, and (2) incorporating resistances to a wide variety of pests and diseases, in order to reduce or eliminate the usage of pesticides, since there is a growing concern by the consumers about the presence of chemicals in fruits and vegetables (Llácer, 2009).

The ultimate goal for breeders is the development of cultivars with superior and consistent fruit production, quality and market appeal (Byrne et al. 2012). This would be a challenging task in the future, mainly due to ungovernable factors like climatic change, global marketing of fruits and changing eating habits of populations. To meet these challenges, peach breeders will need to explore new germplasm as well as integrate new genomic breeding technologies like marker assisted breeding and genetic engineering (Hancock et al. 2008).

1.2.3 - The low variability of peach and new sources of variability

Despite the great variety of fruits that is seen in the markets, peach is a species with a very low level of variability genome-wide. Indeed, previous studies demonstrated that peach is the most homozygous of the *Prunus* crops (Byrne et al. 1990; Mnejja et al. 2010), a fact that is mainly explained by the absence of a functional gametophytic self-incompatibility system for the species. The self-pollinating behavior of peach, together with the domestication process of the species and the initial breeding activities at the beginning of the 20th century, have contributed to the scarce gene pool available for the nowadays occidental breeding programs (Scorza et al. 1985). It is known that European and North American cultivars share the same parents, a small group of cultivars used in North America more than 100 years ago that probably descend, in the majority of cases, from 'Chinese Cling' and a few more cultivars. The low variability of peach was confirmed in several studies with isoenzymes (Byrne et al. 1990), AFLPs (Aranzana et al. 2003a), microsatellites (Aranzana et al. 2003b; Aranzana et al. 2010; Mnejja et al. 2010; Li et al. 2013) and SNPs (Aranzana et al. 2012; Micheletti et al. 2015), typically with lower levels of observed heterozygosity in occidental cultivars than in oriental cultivars, as represented in Figure 1.3 (Li et al. 2013). The overall nucleotide diversity, π, of peach genome was also investigated in the past and consists of 1.6×10^{-3} for eastern varieties and 1.1×10^{-3} for western varieties, which is much lower than the closely related species *P. davidiana* (4.8×10^{-3}) (The International Peach Genome Initiative 2013) and almond (18.4×10^{-3}) (Velasco et al. 2016).

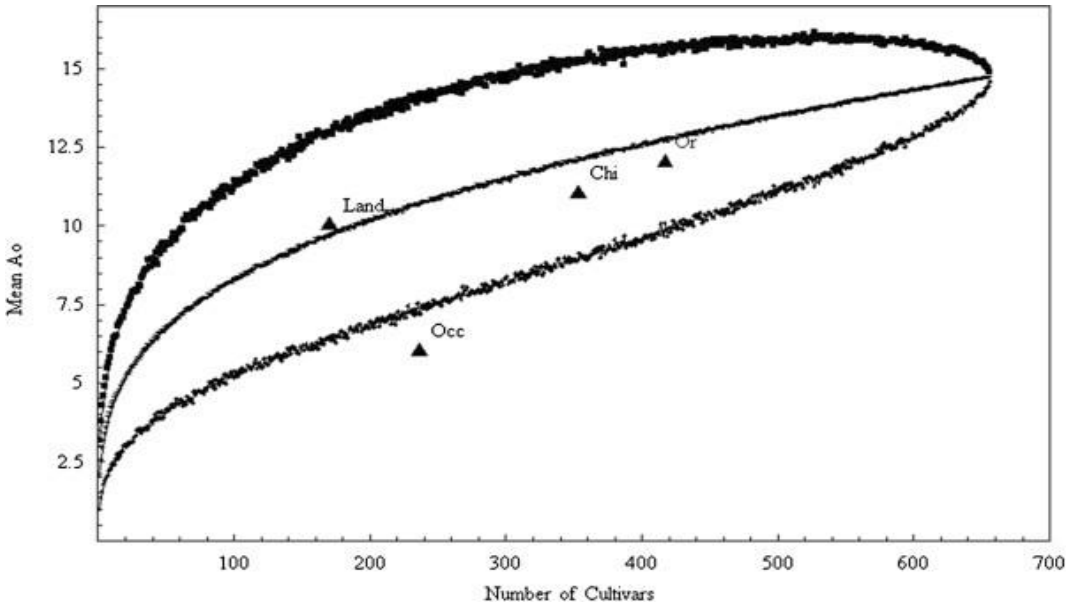


Figure 1.3 – Comparison of the mean number of observed alleles in four different groups of *Prunus persica*: Oriental cultivars (Or), Occidental cultivars (Occ), Chinese cultivars and landraces (Chi) and both occidental and Chinese landraces (Land). Groups containing oriental cultivars or landraces are close to the mean of the rarefaction curve whereas the occidental group is far from the mean and outside the 95% confidence interval, revealing that the mean number of observed alleles for Occ is low and contributes the least for the variability of the whole collection (Li et al. 2013).

To accomplish the breeding goals that are required for the species, breeders must look for other sources of genetic variability besides western cultivars. Aranzana et al. (2003b; 2010) have shown that the majority of varieties coming from occidental breeding programs share the same founders, which explains the low genetic variability seen in western materials. The incorporation of new genes into western peach gene pool can be done mainly by three routes: (1) using local European varieties (Aranzana et al. 2003b), (2) using new Chinese accessions (Li et al. 2013) or (3) using other cultivated species of the genus that are compatible to peach (Donoso et al. 2016). The first route seems plausible since peaches were moved from China long ago, mainly by seed propagation, throughout Europe, and were extensively cultivated and selected for approximately 2,000 years. As a consequence, several landraces well adapted to local climates and selected for regional environments and quality preferences exist. The second route considers the exploration of new accessions that are not available in western countries. The most extensive collections of germplasm have been established in China 50 years ago and they include many of the local cultivars and landraces (Byrne et al. 2012). The use of these ancient oriental cultivars could be a good source of new genes for peach breeding (Li et

al. 2013; Cao et al. 2014). While these two first routes consider primary germplasm (cultivars, varieties or landraces of *P. persica*), the third route explores sources of secondary germplasm. Peach is compatible with several other *Prunus* species, such as almond (*P. dulcis*), myrobolan plum (*P. cerasifera*) and Japanese plum (*P. salicina*), as well as wild *Prunus* close to peach like *P. mira*, *P. kansuensis* and *P. davidiana*. Interspecific hybrids between peach and these species have been done in the past, mainly for rootstock development (Byrne et al. 2012) as well as for genetic analysis with the aim to find resistances to pest and diseases (Gradziel, 2003; Foulongne et al. 2003; Rubio et al. 2010; Sauge et al. 2012). These hybrids have also been used to study fruit quality traits (Gradziel, 2003; Quilot et al. 2004; Donoso et al. 2016) and tree architecture (Carrillo-Mendoza et al. 2013). However there is no evidence of any peach cultivar currently under commercial cultivation that possess genes from other *Prunus* species.

1.2.4 - Interspecific breeding

The integration of novel genes coming from exotic species into elite materials is a great challenge. First of all, several generations are needed after the original interspecific cross in order to recover the elite background for the majority of the genome. For a species like peach, this process can take decades due to the intergeneration time of 2-4 years. Second, the introgressed fragment with a desired exotic gene may carry other genes, which may produce undesirable phenotypes, a phenomenon known as linkage drag. Third, interspecific hybrids are known to have reduced recombination frequencies, probably due to genomic instabilities. This would emphasize the linkage drag problem since less recombination means less probability of reducing the introgressed fragment from one generation to another. And forth, there are several issues related with low fertility that may occur when generating of interspecific progenies, thus reducing the production of offspring.

On the other hand, the most relevant advantage for interspecific breeding is the access to a large reservoir of variability that does not exist in the intraspecific gene pool (Tanksley

and McCouch 1997). Compatible exotic species do not only represent a source of resistance genes for pests or diseases, but they could also contribute with genes for novel (and unseen) fruit quality traits, which would bring, after all, novelty to consumers. Other advantage when developing interspecific populations is the usefulness of these populations for genetic studies, generating high density genetic linkage maps due to the higher polymorphisms that exist between the two parents and allowing the genetic characterization of all traits that may segregate.

One way to accurately study the inheritance of complex characters, facilitating at the same time the introgression of novel genes into an elite genotype is the development of near isogenic line (NIL) collections. Each NIL has a single introgressed fragment from a donor (exotic) parent in the background genome from an elite variety, and all fragments are different between lines. Taken together, the collection of NILs covers the entire exotic genome with fixed fragments of the donor parent (Eshed and Zamir 1995). When certain phenotype is observed in one line, the gene or genes causing such phenotype must be, theoretically, in the donor fragment that only that line possesses (Eshed and Zamir 1994). This facilitates the dissection of quantitative traits into component Mendelian traits much more efficiently than with other population types. Furthermore NIL populations are a great tool to introduce variability into breeding programs, mainly because the gene or genes of interest are already isolated in small introgressed (fixed) fragments, thus requiring one single generation to introduce novel genes into elite varieties (Grandillo et al. 2007). Additionally, as these lines only have a small fragment from the donor parent, fertility issues associated with interspecific crosses are more likely to be limited. NIL populations were first developed in tomato (Eshed and Zamir 1994) and since then have been replicated for other annual species like maize (Szalma et al. 2007), lettuce (Jeuken and Lindhout 2004), melon (Eduardo et al. 2005), strawberry (Urrutia et al. 2015), among others. The cultivated tomato is probably the most successful example of interspecific breeding, resulting in the introgression of disease resistance genes (Zamir et al. 1994; Griffiths and Scott 2001; Bai et al. 2003; Verlaan et al. 2013) as well as other important traits like soluble solids content, yield, early fruit ripening, color and viscosity (Eshed and Zamir 1995; Tanksley and Nelson 1996; Fulton et al. 2000; Frary et al. 2003).

1.2.5 - Almond as a donor of genetic variability for peach

Almond [*P. dulcis* (Mill.) D.A.Webb] is a nut crop native from the central Asia (Zohary et al. 2012) which cultivation was expanded to all Mediterranean regions, and then from there to South Africa, Australia and America. *P. dulcis* is the most important nut worldwide (Hummer & Janick, 2009) with an annual production of 2.7 M tonnes in 2014 (FAOSTAT, 2017), being the United States (Mainly in California state) the world leader in almond production, followed by Spain, Iran and Italy (FAOSTAT, 2017). Peach and almond are closely related (Chin et al. 2014) and some authors consider that the differences seen nowadays between both species are a consequence of the adaptation, from a common progenitor, to different climates (Yazbek and Al-Zein, 2014). This theory states that, from the same progenitor, peach developed in high humidity climates, typically in low elevations (China) whereas almonds adapted to drier and semi desert conditions of central Asia (Figure 1.4) (Bielenberg et al. 2009; Yazbek and Al-Zein, 2014).

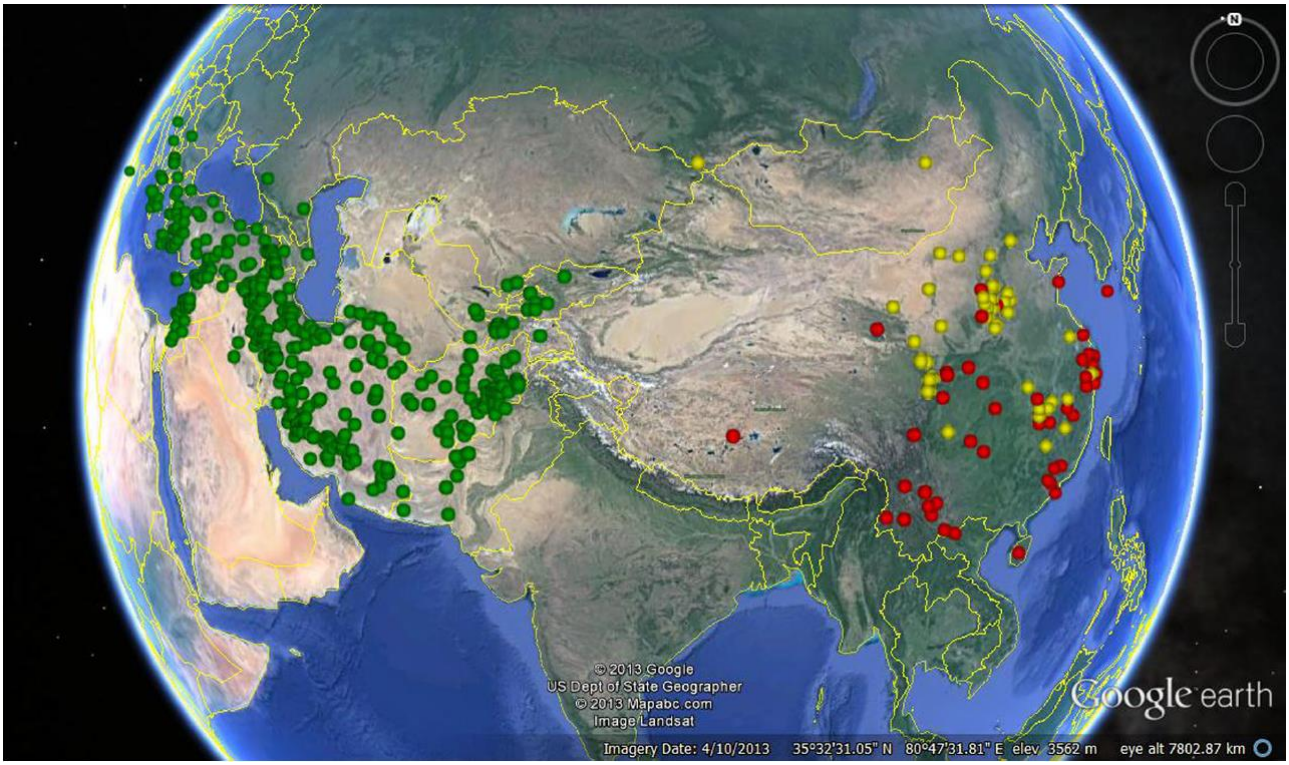


Figure 1.4 – Distribution of almond (*Prunus dulcis*) and its close relatives in arid and semiarid regions of West Asia (green dots) and peach (*Prunus persica*) and its relatives in East Asia, were more humid climates exist (red and yellow dots). Image from Yazbek and Al-Zein (2014).

Despite their close phylogeny, and their sex compatibility, peach and almond have two completely different levels of variability. Almond is highly polymorphic, probably due to its self-incompability and to the old cultivation practices of using open-pollinated seedlings (Fernández i Martí, 2009). Almond variability was estimated to be seven times higher than peach (Velasco et al. 2016) and is probably the most heterozygous species of the cultivated *Prunus* (Mnejja et al. 2010), while peach is the most homozygous one. Therefore, the use of almond as a donor of novel genes to the peach gene pool seems a logical choice. Almond × peach hybrids have been developed in the past, mainly for rootstocks, since almond is a good source of resistances for root-related pathogens (Gómez-Aparisi, et al. 2001; Esmenjaud et al. 2009). Martínez-García et al. (2013) performed a QTL mapping for brown rot resistance in one interspecific population derived from an almond × peach cross, and found a total of three QTLs from almond conferring resistance to peach.

Other studies were developed in the past with interspecific almond × peach hybrids to study fruit quality traits (Gradziel, 2003). However, the most complete study on useful almond variability for peach species was done by Donoso et al (2015, 2016), which identified and mapped 10 major genes coming from almond: anther color (*Ag/ag* and *Ag2/ag2*), flower color (*Fc2/fc2*), maturity date (*MD/md*), almond fruit type (almond *vs.* peach; *Alf/alf*), juiciness (*Jui/jui*), blood flesh (*DBF2/dbf2*), powdery mildew resistance (*Vr3*) and two fertility restorer genes (*Rf1* and *Rf2*). Two of these genes are of particular interest since they represent the main differences between peach and almond fruits: the thickness of the mesocarp (*Alf/alf*) and the presence or absence of juice (*Jui/jui*). Furthermore, the study of quantitative traits (Donoso et al. 2016) allowed the mapping of 32 consistent QTLs over the years, which reflects the usefulness and richness of almond genome for peach species (Figure 1.5).

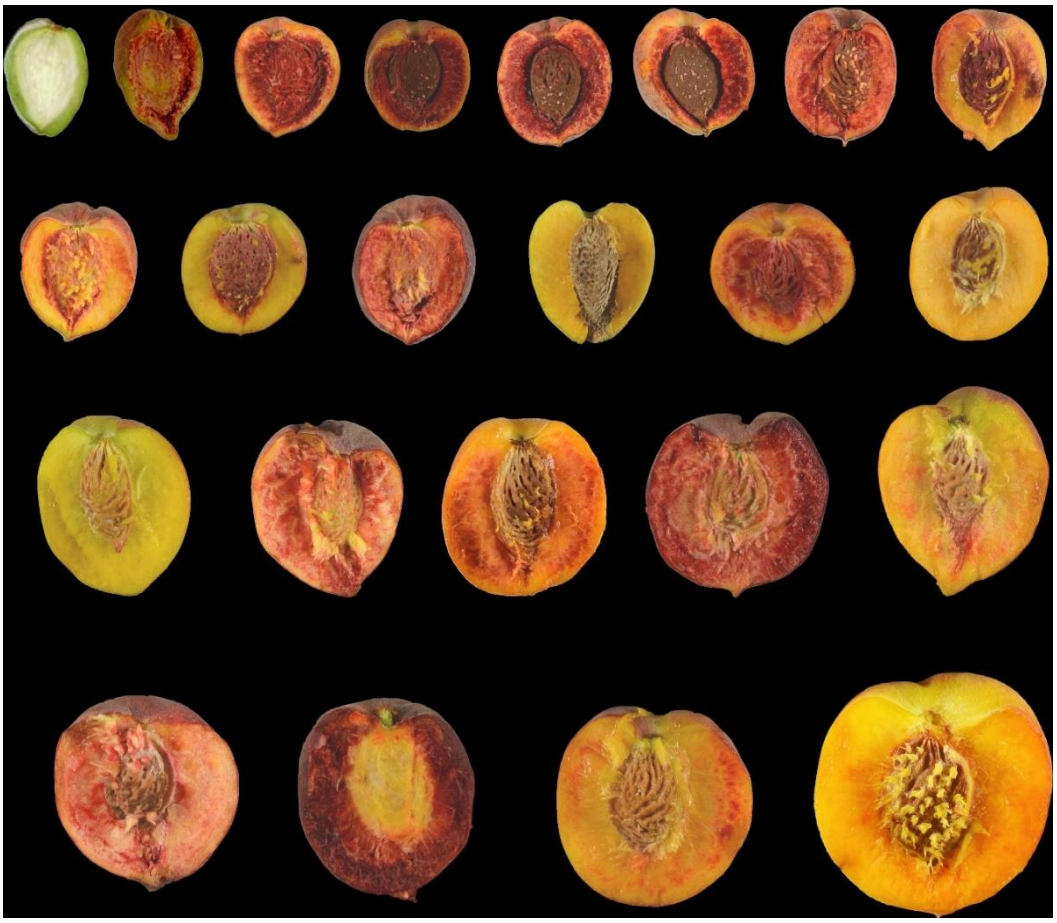


Figure 1.5 - Fruits from ‘Texas’ almond (top left), ‘Earlygold’ peach (bottom right) and some of the progeny between both.

1.3 - Genetic and genomic tools available for peach

1.3.1 - Segregating populations and genetic maps

Peach is one of the best genetically characterized species of the Rosaceae family, mainly because of the vast repertoire of genetic and genomic tools that have been developed during the last decades for the species. The first genetic map for peach was produced more than 20 years ago by Chaparro et al. (1994) using an F_2 population derived from the cross NC174RL x 'Pillar'. This first map was built with two morphological markers, one isoenzyme and 83 RAPDs (random amplified polymorphic DNAs) and presented a total of 15 linkage groups. One year later, Rajapakse et al. (1995) developed a genetic map with more reliable and informative markers (RFLPs), also for an F_2 progeny, and detected the eight expected linkage groups. Other linkage maps based on RAPDs, AFLPs and/or RFLPs markers arose for peach, but they were still incomplete maps, with large gaps and few mapped markers. One major issue when constructing linkage maps in peach is the low variability of the species, which translates into a high proportion of monomorphic markers and large uncovered chromosome regions. In order to overcome this problem, some maps were built using interspecific progenies, since they allow high and constant coverage of polymorphic markers along the genome.

One of these interspecific progenies is TxE, an F2 population derived from the cross between almond 'Texas' and Peach 'Earlygold'. TxE linkage map was initially built using 11 isoenzymes and 235 RFLPs (Joobeur et al. 1998) and was adopted by the scientific community as the reference map for peach and other *Prunus* species, providing the terminology and orientation of linkage groups and contributing with a large dataset of transferable markers for other *Prunus*. TxE was continuously improved over the years with the addition of sequence-based markers (Dirlewanger et al. 2004) and ultimately was rebuilt using only SSR and SNP markers (Donoso et al. 2015). More than 30 genetic maps were developed for *Prunus* in the past (Arús et al. 2005; Pozzi and Vecchietti, 2009) and all of them share a framework of markers with TxE (Arús et al. 2012). This marker transferability, together with the availability of a SNP array (Verde et al. 2012) allow the fine mapping of major genes for a great amount of traits with an increased resolution. To

date, over 50 major genes for different *Prunus* species were mapped in the reference map TxE, the majority of them with high economic impact (Table 1.1).

1.3.2 - The use of molecular markers linked to traits of interest

The selection of plants with desired traits by breeders is a process commonly based on phenotyping or visual evaluation of those traits. For tree species like peach, whose intergeneration period can last up to four years, this selection step represent a severe delay on the creation of new varieties for both public and private breeding programs. Other limitations of traditional selection are the pyramiding of several alleles (i.e. for durable resistance to specific diseases) or the identification of genetically superior lines when the phenotypic variance is influenced by the environment (Ru et al. 2015).

The development of molecular markers linked to traits of interest allows an early selection of individuals, a process known as Marker Assisted Selection (MAS), and is especially important for traits that manifest late on plant's life cycle, such as fruit traits (Byrne et al. 2012). In peach, the application of MAS is possible for a few Mendelian traits, mainly for low-acidity, skin pubescence, flesh color, stone adhesion, flesh texture and fruit shape (Aranzana et al. 2012) and also for pollen sterility and aborting fruit (Dirlewanger et al. 2006), slow ripening (Meneses et al. 2016), resistance to green peach aphid (Lambert et al. 2016) and skin color (Bretó et al. 2017). In addition to the monogenic traits already mapped for peach, several complex traits were also detected for the species in several segregating populations during last decades (Arús et al. 2012). Despite the discovery of markers associated with these quantitative trait loci, further work is still necessary before integrating these into breeding programs (Byrne et al. 2012). Moreover, solutions should be found to other important shortcomings such as the lack of trained staff, or the need of simplified and cheap strategies to screen large progenies (Byrne 2007).

Table 1.1 - Genes described and mapped in *Prunus* crosses and their linkage map positions in the peach TxE reference map

Characters	LG	Symbol	References
Sharka resistance	G1	*Sharka*	Hurtado et al. (2002)
Flesh color (white/yellow)	G1	*Y*	Warburton et al. (1996); Bliss et al. (2002)
Anther color (yellow/anthocyanic)	G1	*Ag2*	Donoso et al. (2016)
Juiciness	G1	*Jui*	Donoso et al. (2016)
Evergrowing	G1	*Evg*	Wang et al. (2002)
Flesh color (normal/anthocyanic)	G1	*DBF2*	Donoso et al. (2016)
Flower color	G1	*B*	Jauregui (1998)
Root-knot nematode resistance	G2	*Mi*	Claverie et al. (2004); Lu et al. (1998); Jáuregui (1998)
Root-knot nematode resistance	G2	*Mj*	Yamamoto et al. (2005)
Root-knot nematode resistance	G2	*RMia*	Duval et al. (2014)
Powdery mildew resistance	G2	*Vr3*	Donoso et al. (2016)
Male fertility restorer	G2	*Rf1*	Donoso et al. (2015)
Shell hardness	G2	*D*	Arús et al. (1998)
Broomy (or pillar) growth habit	G2	*Br*	Scorza et al. (2002)
Double flower	G2	*Dl*	Chaparro et al. (1994)
Anther color (yellow/anthocyanic)	G3	*Ag*	Joobeur (1998); Donoso et al. (2016)
Polycarpel	G3	*Pcp*	Bliss et al. (2002)
Flesh color around the stone	G3	*Cs*	Yamamoto et al. (2001)
Flower color	G3	*Fc*	Yamamoto et al. (2001)
Skin color (Highlighter/Anthocyanic)	G3	*H*	Bretó et al. (2017)
Flesh color (normal/anthocyanic)	G4	*Bf*	Werner et al. (1998); Bliss et al. (2002)
Flower color	G4	*Fc2*	Donoso et al. (2016)
Late blooming	G4	*Lb*	Ballester et al. (2001)
Maturity date	G4	*MD*	Eduardo et al. 2011; Donoso et al. (2016)
Slow ripening	G4	*SR*	Eduardo et al. 2015
Almond fruit type	G4	*Alf*	Donoso et al. (2016)
Flesh adhesion (clingstone/freestone) / Flesh type (Melting/nonmelting)	G4	*F-M*	Verde et al. (2002); Dettori et al. (2001); Yamamoto et al. (2001)
Hybrid incompatibility	G4	*Hls1*	Tsuruta and Mukai (2015)
Non-acid fruit	G5	*D*	Dirlewanger et al. (1998, 1999); Etienne et al. (2002)
Flesh color (normal/anthocyanic)	G5	*DBF*	Shen et al. (2013)
Skin hairiness (nectarine/peach)	G5	*G*	Dirlewanger et al. (1998, 1999); Bliss et al. (2002)
Kernel taste (bitter/sweet)	G5	*Sk*	Bliss et al. (2002)
Leaf shape (narrow/wide)	G6	*Nl*	Yamamoto et al. (2001)
Plant height (normal/dwarf)	G6	*Dw*	Yamamoto et al. (2001)
Male sterility	G6	*Ps*	Dirlewanger et al. (1998, 2006)
Male fertility restorer	G6	*Rf2*	Donoso et al. (2015)
Fruit skin color	G6-G8	*Sc*	Yamamoto et al. (2001)
Leaf color (red/yellow)	G6-G8	*Gr*	Jauregui (1998); Yamamoto et al. (2001)
Powdery mildew resistance	G6-G8	*Vr2*	Pascal et al. (2010)
Aborting fruit	G6	*Af*	Dirlewanger et al. (2006)
Fruit shape (flat/round)	G6	*S*	Dirlewanger et al. (1998, 1999, 2006)
Gametophytic incompatibility	G6	*Si*	Olmstead et al. (2008); Vilanova et al. (2003)
Root-knot nematode resistance	G7	*Ma*	Claverie et al. (2004)
Powdery mildew resistance	G7	*Sf*	Dirlewanger et al. (1996)
Leaf gland (reniform/globose/eglandular)	G7	*E*	Dettori et al. (2001)
Flower morphology	G8	*Sh*	Ogundiwin et al. (2009); Fan et al. (2010); Donoso et al. (2016)

1.3.3 - The peach genome

The development of genomic tools based on genome sequence represents a very useful source of information for comparative studies and identification of important genes (Byrne et al. 2012). Since the first announcement of the construction of a BAC framework physical map by Abbott et al. (2002), peach has become an even more appealing model species for the Rosaceae family, and several BAC libraries have been constructed for the species. Two BAC libraries, the one constructed from peach rootstock 'Nemared' and the other based on peach rootstock 'Lovell', were used for the construction of a peach physical map (Zhebentyayeva et al. 2008). The anchorage of 52 clones from peach physical map to the eight linkage groups of the *Prunus* reference map paved the way to the mapping of EST sequences already developed for the species and contributed to the fine mapping of QTLs, positional cloning of genes and development of EST-based SSRs. Efforts to generate a peach genome sequence date back to 2002, being later expanded from the US-Italian consortium to the International Peach Genome Initiative (IPGI) (Arús et al. 2012). The initiative clustered together researchers from US (JGI, Washington State University, Clemson University and NC State University), Italy (CRA, IGA, Udine University, FPTP, and Pisa S. Anna School), Spain (IRTA), France (INRA) and Chile (Andres Bello). A double haploid individual from the peach cultivar 'Lovell' was selected for genome sequencing, which allowed a deeper effective genome depth and reduced the complexity of genome assembly (Arús et al. 2012). The first version of peach genome (Peach v1.0) was published in 2013 by The International Peach Genome Initiative and presented a genome sequence of 227 Mbp in size, which was significantly lower than a previous nuclear content prediction of 300 Mbp (Baird et al. 1994). The sequencing of this first version of the genome consisted of 8.47× whole genome shotgun sequencing, employing the accurate Sanger methodology, and was assembled using Arachne. It consisted of eight pseudomolecules representing the eight chromosomes of peach, with the assembled scaffolds covering about 99% of peach genome. Altogether, peach genome is one the best sequenced genomes among plants (Table 1.2). A total of 27,852 protein-coding genes and 28,689 protein-coding transcripts were predicted in this irst version, the great majority of them (24,423) having Arabidopsis homologs (The International Peach Genome Initiative 2013). Two years later a new release of the peach genome (Peach

v2.0.a1) was available[1] to the public and included several improvements with respect to the first version. Efforts were made to map previously unmapped sequences as well as to correct misassembly issues, altogether contributing to an outstanding 99.2% of mapped sequences, and from those 97.9% correctly oriented. Annotation of repetitive elements was improved as well as the gene prediction, the latter based on the assembly of 2.2 billion of RNA-seq reads. Peach v2.1 annotation presents 26,873 protein-coding genes (991 less than Peach v1.0) and 47,089 protein-coding transcripts, about 20,000 new isoforms discovered when compared to version 1.0.

Table 1.2 - Statistics and comparison of the peach assembly to other plant genomes. Adapted from The International Peach Genome Initiative (2013) – Supplemental materials.

Genome	Sequence Coverage	Assembled scaffold sequence (Mb)	Portion Mapped (Mb)	% Mapped	Scaffold N50 (Mb)	Contig N50 (Kbp)	Sequencing approach	Reference
Peach (*Prunus persica*)	8.47×	227.4	225.7	99.3	4	294	Sanger	The International Peach Genome Initiative (2013)
Chinese plum (*Prunus mume*)	8.0×	237	--	--	1.1	31.8	Illumina	Zhang et al. (2012)
Apple (*Malus × domestica*)	16.9×	598.3	528.3	88.3	80	16.171	Sanger, 454	Velasco et al. (2010)
European Pear (*Pyrus communis*)	11.4×	577.3	171.3	29.7	0.088	6.569	454	Chagné et al. (2014)
Woodland Strawberry (*Fragaria vesca*)	39×	201.9	198.1	94.0	1.3	--	454, Illumina, SOLiD	Shulaev et al. (2011)
Arabidopsis thaliana	--	119.7	119.7	100	3	--	BAC by BAC	The Arabidopsis Genome initiative (2000)
Rice (*Oryza sativa*)	--	382.2	382.2	100	6	--	BAC by BAC	The international Rice Genome Sequencing Project (2005)
Grape (*Vitis vinifera*)	8.4×	467.5	290.2	62.1	14	2.012	Sanger	Jaillon et al. (2007)
Papaya (*Carica papaya*)	<3×	271.7	235.0	86.5	1	11	Sanger	Ming et al. (2008)
Melon (*Cucumis melo*)	13.52×	375.5	316.3	87.5	4.68	18.163	454, Sanger	Garcia-Mas et al. (2012)
Soybean (*Glycine max*)	8.04×	955.1	937.3	98.1	10.0	1.492	Sanger	Schmutz et al. 2010

[1] https://www.rosaceae.org/species/*Prunus*_persica/genome_v2.0.a1

1.4 - Meiotic Recombination

Recombination results as a consequence of crossover (CO) formation during meiosis, giving rise to the reciprocal exchange of genetic material between homologous chromosomes. During the first meiotic division, COs are needed to produce the physical connections between homologous chromosomes for the generation of bivalents, which ensures the proper segregation of chromosomes in most species (De Muyt et al. 2009; Mercier et al. 2015). Besides their importance for the correct formation of gametes, COs generate new genetic combinations thus contributing to the genetic variability of sexually reproducing species. Furthermore, recombination is the basis for determining the genetic distance between genes or markers, which allows the development of genetic linkage maps in segregating populations (De Muyt et al. 2009).

1.4.1 - Molecular mechanisms of meiotic recombination

The studies conducted in the last decade on the molecular events that take place during meiotic recombination have contributed to a better understanding of the genes that are involved in this mechanism. Despite the fact that so much is still unknown about recombination, the knowledge acquired provided of models that explain this complex process.

The molecular basis of meiotic recombination was first established from studies in yeast (Osman et al. 2011). At the molecular level, recombination can be divided into three major phases: (1) DSB formation, (2) DSB processing and strand invasion and (3) DSB repair, which can produce crossovers or noncrossovers (Figure 1.6).

The first steps into meiotic recombination are triggered with the formation of programmed DNA double-strand breaks (DSBs) by the highly conserved protein SPO11 (Keeney et al. 1997). In *S. cerevisiae*, Spo11 requires a total of nine other proteins in order to form DSBs, while in other species like Arabidopsis, orthologues of some of these accessory proteins are not needed (Mercier et al. 2015). Also interesting is the fact that in most species SPO11 is encoded by a single gene, whereas in plants and other lineages

there are several homologs encoding for SPO11 proteins (Malik et al. 2007). In Arabidopsis both AtSPO11-1 and AtSPO11-2 proved to be required for meiotic recombination whereas AtSPO11-3 did not play a role in meiosis (Mercier et al. 2015). Taken together, all these data point out to the heterogeneous nature of the DSB formation machinery that exists among species. For the second step of DSB processing, studies in yeast have shown that DNA strands are truncated to generate longer 3'-OH single-strand DNA. These single strands are then bound by the Rpa proteins and loaded by recombinases Rad51 and Dmc1 creating structures capable of searching for homology and heteroduplex formation. In plants each protein of the RPA complex is encoded by a multigene family and studies in rice and Arabidopsis suggest that several RPA complexes exist, either with a role on meiosis or not (Mercier et al. 2015). In what concerns to RAD51 and DMC1, recent cytological evidence for Arabidopsis suggest that they probably localize to the two opposite sides of the DSB (Kurzbauer et al. 2012), which correlates with the different fate of each of these sides of the DSB according with the DSB repair model. The third and last major step of meiotic recombination is the repairing of the DSB, which leads a proportion of recombination intermediates into COs.

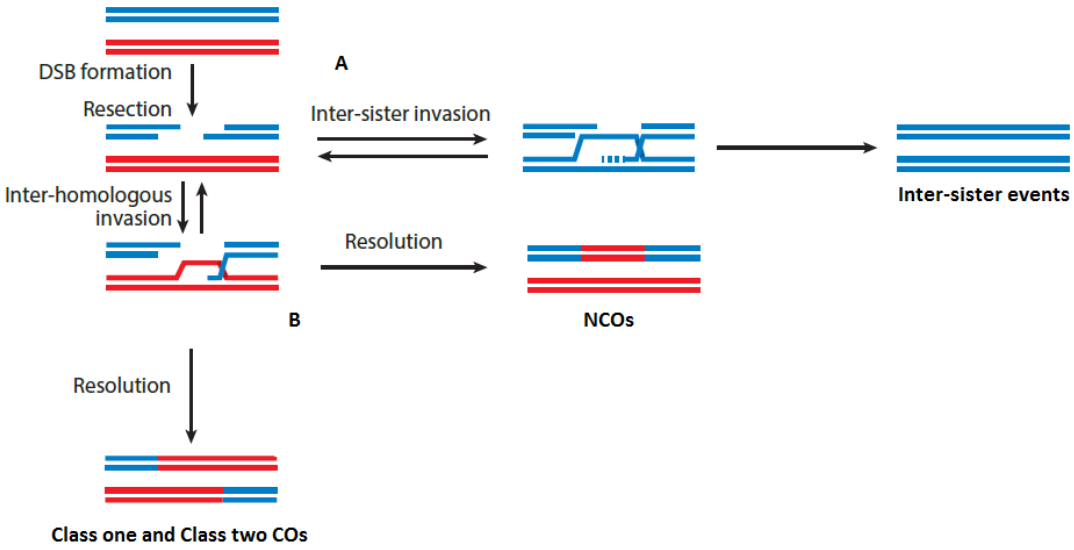


Figure 1.6 - A model for the meiotic recombination events occurring in plants. Double strand breaks (DSBs) can be repaired using the intact sister chromatid (A), leaving no genetic trace, or using the homologous chromosome (B). From the initial number of DSBs repaired using the homologous chromosome, approximately 10 are repaired as class one COs and few as class two COs, suggesting that the majority of DSBs must be repaired as noncrossovers (NCOs). Adapted from Mercier et al. (2015).

There are two pathways for the formation of COs in most eukaryotes: the class I COs and the class II COs. The class I pathway generates COs that exhibit interference, a mechanism that prevent consecutive COs to occur in adjacent regions of the same chromosome. This pathway is driven by a group of proteins (ZMM proteins) that were described in yeast as essential for the interference-sensitive COs and synaptonemal complex (SC) formation (Börner et al. 2004). Homologues of ZMM proteins have been identified in Arabidopsis and mutant lines for these proteins show a decrease of 85% on the number of overall COs per meioses, which indicates that this is the main pathway for crossover formation in plants, but is not unique (Mercier et al. 2015). The remaining CO activity belongs to the class II pathway, to which much less is known. The class II COs do not exhibit interference and MUS81 is probably one of its major players. In Arabidopsis, mutating MUS81 showed that it accounts for one part of the class II COs, reducing recombination in wild-type plants in about 10% (Berchowitz et al. 2007; Higgins et al. 2008). The involvement of further proteins in the formation of class II COs still needs to be studied (Mercier et al. 2015).

In addition to COs, DSB repair in plants originates a different genetic product known as noncrossovers (NCOs) (De Muyt et al. 2009), probably through the synthesis-dependent strand annealing (SDSA) pathway (Hunter et al. 2001). The factors controlling the decision of repairing a DSB as CO or as NCO are not yet understood, however there is evidence in budding yeast suggesting that the strand invasion is a key control step for this decision (Börner et al. 2004). In Arabidopsis it is known that, from the initial 150-250 DSBs that initiate meiotic recombination, only 10 are repaired as COs, suggesting that the remaining DSBs must be repaired as NCO (Sanchez-Moran et al. 2007). This means that the proportion of COs relative to DSBs is about 5-10% in Arabidopsis, which is much lower than yeast (50%) (Osman et al. 2011). In addition, there is also cytological evidence for several species suggesting that meiotic DSBs are mostly repaired as NCOs (De Muyt et al. 2009).

1.4.2 - Distribution and abundance of recombination events

In the majority of organisms, meiotic recombination is a highly regulated process and the distribution of COs is non-random, since multiple COs in the same chromosome pair are spaced further apart than what would be expected by chance (Mercier et al. 2015). This phenomenon is known as CO interference and is characterized by the lower probability of a second CO to take place close to the region where a first CO emerged (De Muyt et al. 2009). Several models have been proposed to explain interference since it was first observed in *Drosophila* (De Muyt et al. 2009). The 'counting model' (Foss and Stahl, 1995) proposes that CO distribution is regulated by a fixed number of NCOs that must occur between two CO events, and despite it was validated in both humans (Housworth and Stahl, 2003) and Arabidopsis (Copenhaver et al. 2002), results from yeast have proved to be inconsistent (De Muyt et al. 2009). This model predicts that a reduction on the initial number of DSBs would lead to a decrease on both CO and NCO events. However, results from Martini et al. (2006) show that there is a tendency to maintain the number of COs at the cost of reducing NCOs when there are less DSBs, a mechanism known as CO homeostasis, thus raising some doubts about the 'counting model'. Another model for the CO interference is the 'mechanical stress model' that suggests a distribution of COs based on the expansions and contractions suffered by chromosomes, which generates mechanical stress along the chromosome arms and would lead to the formation of COs to relieve the stress (Kleckner et al. 2004). A third model for interference is the 'Polymerization model', which describes an independent early distribution of recombination structures, each of them with an equal chance to initiate a bi-directional polymerization event. This polymer would then have the ability to further block other early structures from binding to the bivalent, thus generating interference (King and Mortimer, 1990). This model is particularly interesting since it not only explains the CO interference, but also accounts for the CO assurance (the occurrence on at least one CO per chromosome pair). Other models to explain CO interference do exist and are discussed by Berchowitz and Copenhaver (2010).

Besides interference, other factors may also control the occurrence of meiotic recombination, since COs are typically confined to certain regions of the genome. In addition, several plant species present a common display of CO distribution, characterized by a lack of COs close to the centromere region (Crismani et al. 2013). What governs the selection of some genomic regions in favor of others is not yet known, however it has been demonstrated in several species that COs from independent meioses tend to cluster in the same area, thus defining hotspots of recombination. In humans, these highly recombinogenic regions are associated with the DNA motif CCN (Myers et al. 2008), recruiting the PRDM9 protein (Berg et al. 2010), which in turn possibly triggers hotspot activity through chromatin remodeling (Paigen and Petkov, 2010). The PRDM9 protein is not present in plants (Mercier et al. 2015), however the CCN motif was recently identified in CO-rich regions of Arabidopsis (Shilo et al. 2015). Other DNA motifs, as well as specific epigenetic marks, are being associated with increased levels of CO in plants, mainly in the transcription start site (TSS) and transcription termination site (TTS) of genes (Choi and Henderson, 2015).

The abundance of COs in relative terms (COs/chromosome) is conserved between species, independently of the chromosome size, and rarely exceed three COs per bivalent. One example presented by Mercier et al. (2015) is the comparison of the giant chromosome 3B of wheat (1 Gbp), the longest chromosome of Arabidopsis (31 Mbp) and the smallest chromosome of budding yeast (0.3 Mbp). Despite the incredible difference in size, all of these chromosomes have an average of three COs per meiosis.

Such a small variation on the number of COs per chromosome, even between so unrelated species, raises the question of what forces are governing the conversion of DSBs into COs. One way to explain the relatively low number of COs per chromosome observed in the majority of species is to think about chromosome integrity and proper segregation. Would a high number of COs destabilize chromosomes and prevent meioses to proceed normally? Data from the yeast *Schizosaccharomyces pombe* and honeybees, as well as data from artificially increased CO rate in Arabidopsis (up to nine-fold), show that meioses can proceed normally in a high CO rate scenario (Mercier et al. 2015). Another

question that still needs to be answered is the fate of DSBs that do not become COs, thus probably being repaired as NCOs. The number of NCOs in plants is poorly documented, mainly because of the difficulty of detecting them. Different studies have situated the amount of NCOs per meioses in 1.7 (Mercier et al. 2015), which is much lower than what would be expected, mainly due to the 90-95% of DSBs that are not repaired as CO. One possible explanation for this discrepancy is that the majority of DSBs may be repaired using the sister chromatid, thus leaving no genetic trace or mark. Another explanation is that, even if the repair of DSBs is well balanced between using the sister chromatid or the homologous chromosome, only a small amount of homologous-repaired DSBs will include a polymorphism in the converted sequence, which will allow their detection (Mercier et al. 2015).

1.4.3 - The importance of crossovers for breeding

Understanding what controls COs may have a great impact on plant breeding, since manipulating these factors would lead to new genetic possibilities. In addition, favoring high recombinant lines or stimulating CO activity would be very useful for breaking undesired linkages, creating novel combinations of alleles or even increasing the power of genetic mapping studies (Mercier et al. 2015). Some simple methods are known to increase the recombination rates in Arabidopsis, mainly the use of flowers from secondary and tertiary branches or by elevating the temperature (Francis et al. 2007) or by using male gametes which outnumber female ones in total COs by 67% (Crismani et al. 2013). More complex methods include the control of specific genes, which proved to increase almost 10× the CO frequency in Arabidopsis (Crismani et al. 2012). However, both simple and complex methods would need validation in species of economic interest.

Wild or distantly related cultivated species are a good source of variability that could be introgressed into a species of interest. These sources of novelty are typically not used by breeding programs due to the lower recombination rate of hybrids and the consequent large number of generations needed to recover the elite genetic background. The meiotic recombination of these hybrids is termed homoeologous recombination and is controlled by naturally occurring mechanisms that prevent CO activity in order to ensure fertility

and maintain genome stability (Crismani et al. 2013). Thus new technologies that could accelerate the breeding process of interspecific hybrid lines by increasing the CO rate at meioses are highly desirable for breeders (Crismani et al. 2013).

On the other hand, breeders may also have interest in reducing or preventing recombination Two approaches have been developed through the manipulation of meiosis-related genes to take advantage of heterosis in breeding programs: reverse breeding (Dirks et al. 2009) and apomixis (Bicknell and Koltunow, 2004). In the first case, CO formation is abolished in a high performance heterozygous individual via transgenesis and double haploid plants are produced from its gametes. Crossing two complementary double haploid lines would restore the same hybrid display that was previously selected, thus allowing the commercial production of the selected heterotic line. In the second case, apomixis, meiosis is replaced by mitosis, and the genomic display of a specific individual could theoretically be perpetuated through generations. This would provide a great simplification of the production of hybrids without losing the benefits of heterosis (Crismani et al. 2013). The proof of concept of both strategies has been obtained in Arabidopsis, but not yet in crop species (Mercier et al. 2015).

2. Objectives

The main goal of this PhD thesis is to develop new DNA-based strategies and tools to improve peach fruit quality and enrich its genome with exotic genes of interest. To achieve this goal, three objectives are proposed:

1. Genetic analysis of the slow melting flesh (SMF) character in peach.

    1.1- Linkage map construction and QTL analysis in two peach × peach populations segregating for the SMF trait.

    1.2- Study of phenotyping methods for SMF and identification of the most informative for the analysis of this character.

2. Proof of concept of Marker Assisted Introgression (MAI), a fast method to mine and introgress genes from an exotic donor into peach.

    2.1- Final steps of the development of MAI using almond as a donor parent and peach as the recurrent parent.

    2.2- Progress in the development of a collection of introgression lines (ILs) in peach using almond as a donor genome.

3. Study of the recombinational landscape of peach and almond × peach hybrid using resequence data from a population (T1E) from the cross between MB1.37 (almond × peach) and ‘Earlygold’ (peach).

    3.1- Development of an in silico genotyping pipeline to detect SNP and indel variability and identify crossover breakpoints in the T1E progeny.

    3.2- Study of crossover regions at sequence level, detection of genetic marks associated with CO occurrence and identification of gene conversions.

3. Genetic analysis of the slow melting flesh character in peach

Genetic analysis of the slow melting flesh character in peach

Octávio Serra[1], Jordi Giné-Bordonaba[2], Iban Eduardo[1], Joan Bonany[3], Gemma Echeverría[2], Christian Larrigaudière[2] and Pere Arús[1§]

[1]IRTA. Centre de Recerca en Agrigenòmica CSIC-IRTA-UAB-UB, Campus UAB, Cerdanyola del Vallès (Bellaterra), 08193 Barcelona, Spain;

[2]IRTA. Fruitcentre, 25003 Lleida, Spain;

[3]IRTA. Mas Badia, 17134 La Tallada, Spain

[§]Corresponding author: e-mail: pere.arus@irta.cat; phone: +34607073923; fax: +34935636601

Submitted to Tree Genetics and Genomes

Abstract

The slow melting flesh (SMF) trait in peach [*Prunus persica* (L.) Batsch] defines a slower process of postharvest fruit softening than the prevalent melting flesh (MF) types. This gives a longer shelf-life and a delayed harvest time resulting in better fruit quality. Unlike other known fruit texture traits, SMF is difficult to measure and has a complex inheritance. We examined this character over two years in the offspring of two crosses, both with 'Big Top', a SMF nectarine, as female parent, and a melting flesh (MF) nectarine as male parent ('Armking' and 'Nectaross'). Following harvest, a texturometer was used to provide a textural profile analysis, and fruit firmness evolution was measured with a penetrometer over a period of five days storage at 20ºC. Linkage maps were constructed with a high density SNP chip, and a phenotype-genotype analysis allowed the detection of three independent genomic regions where most QTLs (quantitative trait loci) were located. Two of these, on linkage groups 4 and 5, explained the variability for two characters: maturity date and firmness loss, the QTL on linkage group 4 found in the MF parents and that on linkage group 5 in 'Big Top'. A third region on linkage group 6, identified a QTL for maturity date only in 'Armking', having no apparent association to the softening process. The relationship between maturity date and fruit firmness loss and a hypothesis on the inheritance of the SMF character are discussed.

Keywords: *Prunus persica*, postharvest behavior, marker-assisted selection, maturity date, fruit flesh texture

3.1 - Introduction

Postharvest behavior is a critical aspect of quality in the climacteric peach fruit. Depending on the postharvest conditions peaches can be kept at most for three to four weeks after harvest (Ramina et al. 2008). The key aspect determining peach fruit shelf-life is softening, with the melting flesh (MF) types, which dominate the peach fresh market, having the fastest softening rate and consequently the most reduced postharvest life. Increasing shelf-life has been one of the main objectives of breeding programs although the progress has been, to date, quite limited.

Three peach variants capable of retaining the flesh consistency after harvest have been characterized both at the genetic and physiological level. One is non-melting flesh (NMF), characteristic of canning peaches, where flesh texture remains hard for a long time. This character is determined by a major gene (*M*/*m*) located in the central region of linkage group 4 (G4) of the peach map (Peace et al. 2005), with the recessive allele (*m*) determining the non-melting character. The causal gene of *M* has been associated with the activity of an endopolygalacturonase gene in peach (Lester et al. 1996). In fact, *M* co-locates with the position of a genomic region with various endopolygalacturonase genes, two of which, *PpEndoPGM* and *PpendoPGF*, corresponding to sequences *Prupe.4G262200* in v2.0.a1of the peach genome (*ppa006857m* in v1.0) and *Prupe.4G261900* (*ppa006839m* in v1.0), respectively, are responsible for the MF vs. NMF and the clingstone vs. freestone characters (Gu et al. 2016). Another trait related with fruit firmness is the stony-hard (SH) flesh type, determined by a single gene *Hd*/*hd*, where fruit of stony-hard individuals bearing the *hdhd* genotype are characterized by the absence of ethylene production and with high firmness values during postharvest storage (Haji et al. 2001, 2005; Begheldo et al. 2008; Giné-Bordonaba et al. 2016). Nonetheless, exogenous application of ethylene to SH fruit has been shown to promote postharvest softening (Haji et al. 2001; Hayama et al. 2006). This character has been studied in detail by Pan et al. (2015), who identified a YUCCA flavin mono-oxygenase gene (*PpYUC11*, *ppa008176m* in v1.0, and *Prupe.6G157400* and *Prupe.6G157500* in v2.0.a1 of the peach genome) in the central part of chromosome 6, involved in the auxin biosynthesis pathway, as a strong candidate for the *Hd* gene. A high concentration of IAA (indol-3 acetic acid) is required for the normal climacteric ripening process: the *hd* allele of *PpYUC11* results

in a dramatic decrease of IAA accumulation in stony-hard fruit, thereby impairing ethylene production.

The third trait is for slow ripening (SR) fruit where peaches do not mature and soften at all, even after treatment with ethylene, and remain attached to the tree after leaf fall. This character is determined by a single gene, *Sr*/*sr* (Ramming 1991), located on chromosome 4 (Eduardo et al. 2015; Núñez-Lillo et al. 2015), but at a different position to the *M*/*m* gene. A candidate gene for this character is a NAC transcription factor (*ppa008301m* in v1.0; *Prupe.4G186800* in v2.0.a1) responsible for controlling the ethylene pathway and possibly involved in peach maturity date (Pirona et al. 2013; Eduardo et al. 2015). In contrast to all the above-mentioned, slow melting flesh (SMF) peaches produce fruit that are hard at physiological maturity, but melt at slower rate than MF types during postharvest (Bassi and Monet 2008). These features are appreciated by growers and retailers as fruit can be harvested at a more advanced stage of maturity and shelf-life is extended. In addition, advanced maturity at harvest may lead to fruit with improved organoleptic quality by increasing the sugar content (Iglesias and Echeverría, 2009). From a physiological point of view, no strong relationships between the rate of firmness loss and the capacity of the fruit to produce ethylene have been found when comparing SMF and MF varieties (Ghiani et al. 2011; Giné-Bordonaba et al. 2016). Several peach and nectarine cultivars from Californian breeders have this SMF character, such as the 'Rich Lady' and 'Diamond princess' peaches, and the red-skinned, yellow-fleshed nectarine 'Big Top', which has become a reference for Spanish nectarine production over the last decade (Iglesias 2012).

Unlike the other three characters (NMF, SH and SR), that have a simple Mendelian inheritance, SMF is poorly characterized at the genetic level, in part due to the difficulty of measuring this character (Bassi and Monet, 2008), and because it has a more complex genetics. A first approach has been recently provided by Zeballos et al. (2016), who measured fruit firmness in a 'Big Top' × 'Venus' population and found a QTL on G5 of the 'Big Top' map which explained approximately 20% of the total variation for this character. In this paper we analyze two segregating progenies with 'Big Top' as one of the parents, and analyze their fruits for several parameters related with flesh texture. By doing so, we attempt to provide additional evidence to help in understanding the genetic basis of this important character and to facilitate the incorporation of SMF in new varieties through breeding programs.

3.2 - Materials and Methods

3.2.1 - Plant populations

In this work we used two yellow-fleshed nectarine F_1 populations. The first population (N=75) was from the cross between 'Big Top' and 'Armking' (Bt×Ak), and the second (N=48) from crossing 'Big Top' and 'Nectaross' (Bt×Nr). 'Armking' and 'Nectaross', both regular melting (MF) varieties, were used as pollen donors, whereas 'Big Top' is a SMF variety that was the pistillate parent. Both populations and parental lines were grafted on 'Garnem' rootstock and grown using standard cultural practices at IRTA's experimental station in Gimenells, Lleida (Spain).

3.2.2 - Phenotyping and data analysis

Fruits were harvested at optimum commercial maturity when their firmness values reached 50.0 Newtons ± 5. Additionally, visual inspection of all fruits for each tree was to confirm the appropriate ripe stage of each individual and the maturity date (MD = number of Julian days till harvest day) further recorded. The apparent maturity of each individual fruit, based on the index of absorbance difference ($I_{AD} = A_{670}$-A_{720}; Ziosi et al. 2008; DA-Meter, TR Turoni, Forli, Italy), was assessed in the laboratory, and fruit with homogenous medium apparent maturity (based on a normal distribution) were selected (n=40) for further analysis and stored at 20ºC (Giné-Bordonaba et al. 2016). Samples were taken after 0, 1, 3 and 5 days storage to measure fruit firmness by means of a hand-held penetrometer, equipped with an 8mm plunger (Effegi, Milan, Italy), after removing the peel of the fruit on two opposed sides. The percentage of firmness loss was calculated at day three (FL3d) and five (FL5d) by dividing the corresponding firmness by the initial fruit firmness (F0) and multiplying by 100. Additionally, we studied the textural properties of the fruit flesh on the day of harvest, performing a texture profile analysis (TPA) on an additional ten fruits per individual. TPA was with a TA-XTplus texture analyzer (Stable Micro Systems Ltd, Godalming, Surrey, UK) equipped with a P/75 flat probe, using the following test conditions: pre-test speed, 2mm/s; test and post-test speed, 5mm/s; and 50% deformation with an activation force of 0.05N. The textural parameters obtained were Hardness, Springiness, Gumminess, Resilience, Chewiness and

Cohesiveness (see Bourne 2002, for a detailed description of TPA parameters and their relationship with sensory ratings).

All data were analyzed with R v3.2.1 (R Core Team 2015). We tested for data normality using a Shapiro-Wilk test, with data being considered not normal if p-value <0.05. Traits in different years were correlated with the Spearman's rank correlation coefficient.

3.2.3 - Genotyping and linkage map construction

Genomic DNA was extracted from young leaves using the CTAB method (Doyle and Doyle, 1990), followed by a purification step using columns from the DNeasy plant extraction kit (Qiagen, Hilden, Germany). Linkage maps were built using a set of 24 microsatellite (or simple-sequence repeats, SSR) markers (Table S3.1) and the segregating SNP markers obtained with the 9K Illumina SNP chip developed by the Peach SNP International Consortium (Verde et al. 2012). For SNPs, genotypes were scored with GenomeStudio data analysis software (Illumina Inc.) following the same criteria as in Donoso et al. (2015).

All SNPs heterozygous in one or both parents were included for mapping except those that lacked one of the expected genotypic classes (two for 1:1 and three for 1:2:1 ratios). A map was initially constructed for the parents of both progenies. Depending on the genotype of each parent, markers segregated 1:1 when only one parent was heterozygous, and 1:2:1 when both parents were heterozygous for the same alleles. Markers with 1:2:1 ratios were converted into 1:1 by discarding the heterozygous genotypes and using only the two homozygous classes. The phase for the latter markers was determined by comparing them with other 1:1 markers in the same chromosome region. SSRs segregating 1:1:1:1 (both parents heterozygous and segregations involved three or four alleles) were converted into two 1:1 segregations, one for each parent, based on their genotype. We grouped all the markers that co-segregated and selected only one as a representative of the entire region or 'bin'. The mapping data file used for map construction consisted only of one marker per bin, so reducing the complexity of map construction. Maps were built with MAPMAKER/EXP 3.0 software (Lander et al. 1987) using the Kosambi mapping function. Markers were grouped with a minimum LOD score of 4.0 and a maximum gap of 37.5 cM. Based on these criteria, we built the genetic

linkage maps for the three parental lines: ‘Big Top’ (Bt), ‘Armking’ (Ak) and ‘Nectaross’ (Nr). In the case of ‘Big Top’, we first built two different maps, one for each population. Then we constructed a consensus map of ‘Big Top’ using data from the 123 individuals of the two populations at once, since the markers that segregate for this line are expected to be the same in each population dataset, resulting in a more accurate genetic map than either of the individual ‘Big Top’ maps.

3.2.4 - QTL analysis

QTLs were detected using MapQTL 6.0 ® (Van Ooijen 2009) using interval mapping (IM). The data were initially analyzed using the maps of each parent (‘Armking’, ‘Nectaross’ and twice ‘Big Top’, one for Bt×Ak and the other for Bt×Nr) as a BC_1 population taking only 1:1 segregations, including those derived from 1:2:1 and 1:1:1:1 ratios, as described above. This made it possible to identify in which parent the QTL was heterozygous or if it was heterozygous in both parents. An additional analysis taking data from all markers (1:1 and full 1:2:1/1:1:1:1segregations) of Bt×Ak and Bt×Nr with the CP mode of MapQTL allowed a better estimation of QTLs that were heterozygous in both parents. The significance threshold for a QTL was LOD≥3.0 although, exceptionally, QTLs with 3.0≥LOD≥2.0 one year were also considered if there was a QTL with a LOD≥3.0 in the same region in the other year. A QTL was defined as consistent when it was detected at the same region of the same population in both seasons.

The regions containing the QTLs were defined by the genome fragment between the markers having the maximum LOD minus one and a broader fragment with LOD minus two. We studied in more detail the QTLs for maturity date on G5 and G6 and for them estimated the joint interval for the data from 2013 and 2014 by the most extreme markers of the regions (LOD-1) identified both years. The protein sequence of gene *Prupe.4G186800*, the candidate for the maturity date (MD) gene located on G4, was used to look for homologues of this NAC transcription factor gene in the peach genome and in these two QTL regions. Blast was performed through GDR webpage (https://www.rosaceae.org/tools/ncbi_blast) using blastp program (Altschul et al. 1997), looking for matches in the peach genome sequence v2.0.a1 with the database of all transcript peptides.

3.3 - Results

3.3.1 - Linkage maps

The 9K Illumina SNP chip and the 24 SSRs used provided information on a total of 2,201 segregating markers in Bt×Ak and 2,362 in Bt×Nr. Most of them coalesced in the eight chromosomes of peach: 2,198 (99%) in Bt×Ak and 2,318 (98%) in Bt×Nr and the rest were unmapped. For the mapped markers, the majority were heterozygous in only one of the parents and segregated 1:1; 1,707 (78%) in Bt×Ak and 1,673 (72%) in Bt×Nr. The remaining 22-28% of the markers, segregating 1:2:1, corresponded to chromosomal regions heterozygous in both parents and mapped alone occasionally, tending to cluster in specific regions of the genome, mixed with other markers segregating 1:1 that were generally much less abundant in these regions. The patterns of these regions were specific for each particular cross and represented 8% of the total physical distance of the Bt×Ak and 16% of the Bt×Nr maps.

Considering the data used for the construction of the map for each parent (Table 3.1), the map of 'Big Top' (using the ensemble of data of the 123 individuals of both populations) had 1,596 mapped markers, and those of 'Armking' 1,069 and 'Nectaross' 1,392. As expected, a high proportion of markers (88%) segregating in either population used for the construction of the 'Big Top' map were the same. The 'Big Top' map (Figure 3.1) comprised 210 bins distributed along the eight linkage groups of the *Prunus* map (G1 to G8), spanning a genetic distance of 447.0 cM (Table 3.1). The 'Armking' map had 126 bins distributed through 9 groups, as markers from G1 could not be joined as expected from the *Prunus* reference map and were split into two groups (Figure 3.2). These nine groups summed a total genetic distance of 450.0 cM (Table 3.1). For the 'Nectaross' genetic map, 94 bins coalesced into the expected eight linkage groups (Figure 3.3) for a total genetic distance of 320.4 cM (Table 3.1). The nearly double number of bins found in the 'Big Top' map compared to the other two is a logical consequence of the higher population size that allowed detection of more recombination events. Linkage maps built for the three parents, Bt, Nr and Ak presented good collinearity with the 2.0 version of the peach reference genome and were syntenic to the reference *Prunus* map based on the 'T×E' population (Donoso et al. 2015). The only exception was a fragment of 15.8 cM between markers snp_6_5294415 and SNP_IGA_618376, in the proximal end of G6 of the Ak map (Figure 3.2) that had an inverted order, which seems attributable to imprecise

mapping positioning due to the large gap without markers (28.8cM) that connects this fragment to the rest of the chromosome.

Table 3.1 - Distribution of markers in the linkage groups (G1-G8) of the maps of 'Big Top' (joint data from the Bt×Ak and Bt×Nr populations), 'Armking' and 'Nectaross' for total markers, number of bins, distance in centimorgans and physical coverage.

		G1	G2	G3	G4	G5	G6	G7	G8	Total
'Bigtop'	Mapped markers	284	274	282	194	109	169	147	137	1596
	Bins	40	15	42	25	26	21	19	22	210
	cM	90.3	25.9	60.0	49.1	49.3	71.0	45.2	56.2	447.0
	Coverage[a]	81	59	96	53	77	50	56	48	65
'Armking'	Mapped markers	65	230	92	207	23	144	194	114	1069
	Bins	17	16	9	16	5	21	26	16	126
	cM	111.2 (72.2+39.0)	78.7	37.9	36.6	9.4	82.9	64.2	29.1	450.0
	Coverage[a]	57	68	60	39	6	40	88	49	51
'Nectaross'	Mapped markers	158	219	213	261	75	243	60	163	1392
	Bins	12	12	15	15	9	12	4	15	94
	cM	58.4	32.2	50.5	46.3	24.2	52.9	11.7	44.2	320.4
	Coverage[a]	70	37	80	47	28	71	22	90	56

[a]Coverage: percentage of the total physical distance of each linkage group covered by the markers in its extremes. Large regions (>5 Mbp) without any marker were considered as identical by descent and were taken as regions not covered.

3.3.2 - Phenotyping

Data distribution for each trait was very similar across seasons in both Bt×Ak (Figure S3.1) and Bt×Nr (Figure S3.2 and Table S3.2). A substantial percentage of the characters (44.4%) did not follow a normal distribution in different populations or years. Gumminess was the only character that was normally distributed across populations and seasons, MD was not normal in all cases, and the remaining traits adjusted to normality only at certain times.

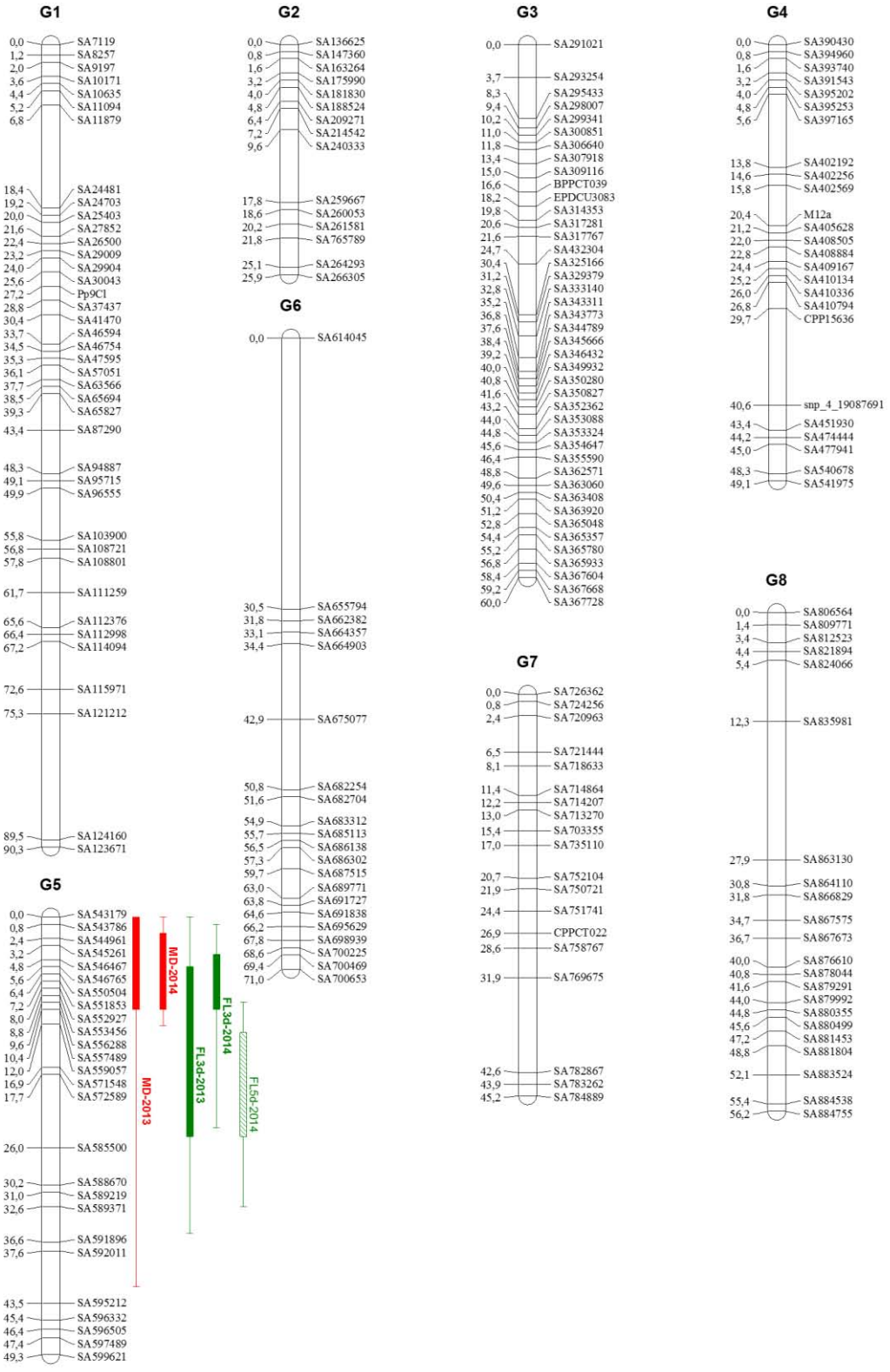


Figure 3.1 - 'Big Top' linkage map obtained with all data available from the Bt×Ak and Bt×Nr populations and the positions of the QTLs mapped in this work. Genetic distances in centimorgans are shown on the left and marker names on the right for each linkage group. Marker names have been abbreviated (SA=SNP_IGA_). Each marker corresponds to a bin (i.e. a group of markers with a same genotype for all individuals analyzed). For QTLs bars in red represent QTLs for maturity date and in green for firmness loss. Solid bars are consistent QTLs and empty bars with diagonal lines represent QTLs detected only one year.

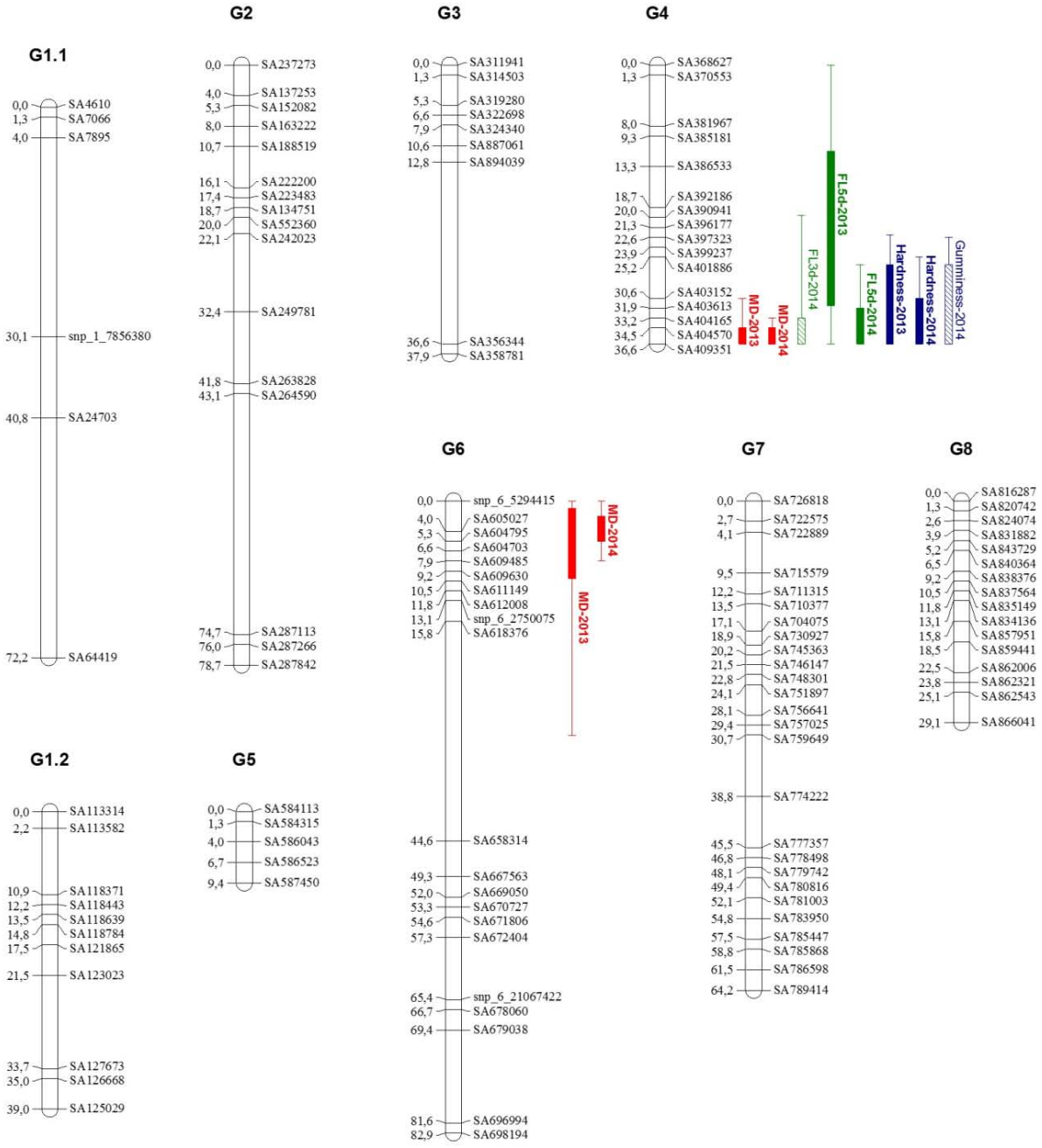


Figure 3.2 - 'Armking' linkage map obtained with positions of the QTLs mapped in this work. Genetic distances in centimorgans are shown on the left and marker names on the right for each linkage group. Marker names have been abbreviated (SA=SNP_IGA_). Each marker corresponds to a bin (i.e. a group of markers with a same genotype for all individuals analyzed). For QTLs bars in red represent QTLs for maturity date, in green for firmness loss and in blue for TPA parameters. Solid bars are consistent QTLs and empty bars with diagonal lines represent QTLs detected only one year.

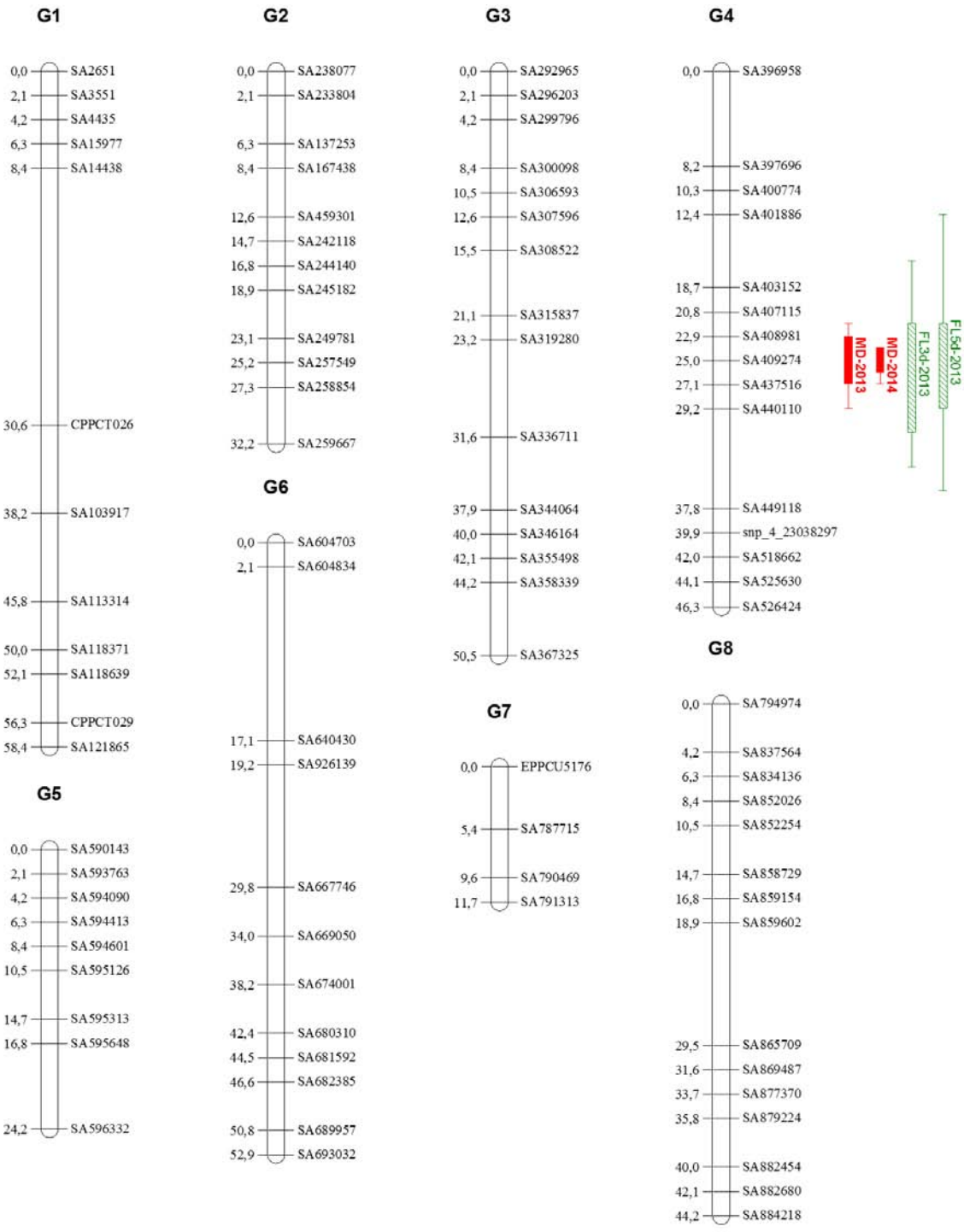


Figure 3.3 - 'Nectaross' linkage map obtained with positions of the QTLs mapped in this work. Genetic distances in centimorgans are shown on the left and marker names on the right for each linkage group. Marker names have been abbreviated (SA=SNP_IGA_). Each marker corresponds to a bin (i.e. a group of markers with a same genotype for all individuals analyzed). For QTLs bars in red represent QTLs for maturity date and in green for firmness loss. Solid bars are consistent QTLs and empty bars with diagonal lines represent QTLs detected only one year.

The Spearman correlation values (Table S3.3) for the same character between years were very high for MD (0.89 and 0.78 for Bt×Ak and Bt×Nr, respectively) and much lower for the rest. Only FL3d and FL5d had positive and highly significant values, with the exception of FL5d (0.26) in Bt×Nr. Hardness was also positively correlated between years in Bt×Ak but negative and non-significant in Bt×Nr. When looking for correlations between different traits in the same year, these were significant and positive for FL3d and FL5d for both years and populations, and negatively correlated with MD in all eight possible cases, although only significantly in five of them. Some characters, such as hardness, gumminess and chewiness, correlated positively and steadily over years and populations, whereas the correlation for the rest was much more erratic.

3.3.3 - QTL analysis

With the maps of each parent of the Bt×Ak and Bt×Nr progenies we found 20 QTLs: six in 'Armking', three in 'Nectaross', six in 'Big Top (Ak)', two in 'Big Top (Nr)' and three for the 'Big Top' map taking the data from both populations (Table 3.2). Considering the QTLs that were consistent in at least one of the three parents (in the different versions of Bt that were consistent at least once), we found eight: qP-MD5, qP-FL3d5, qP-FL5d5 in Bt, qP-MD4, qP-MD6, qP-FL5d4 and qP-Har4 in Ak and qP-MD4 in Nr. The analysis using the integrated maps of both Bt×Ak and Bt×Nr progenies yielded similar results (Table S3.4), with the detection of 32 QTLs, 11 more than with the parental maps, and with all new QTLs found only one year, except for qP-FL3d4, qP-Gum4 and qP-Har5 that were significant both years in Bt×Ak. None of the QTLs detected had a large increase in LOD and R^2 when compared to the results obtained with the parental maps, whereas such an increase would be expected for QTLs heterozygous in both parents.

Consistent QTLs localized in a few positions for each parental map, usually coinciding with the location of the QTL(s) for MD (Figure3.1-3.3). In the 'Big Top' maps all QTLs located at the same region of G5, and similar patterns occurred in 'Armking', with two MD QTLs located on G4 and G6, as all remaining QTLs. This was also the case for 'Nectaross', where a major QTL for MD explaining much of the phenotypic variance (R^2=60-87%) was on G4, and all other QTLs found in this population for other characters mapped to this position. A few exceptions were identified with the integrated maps (Table S3.4), such as three QTLs for Resilience (qP-Res3), Gumminess (qP-Gum1) and

Cohesiveness (qP-Coh1) in Bt×Ak and one for FL5d (qP-FL5d1) in Bt×Nr that mapped to other linkage groups, but they were detected in only one year.

Table 3.2 - QTLs identified in the maps of the parents 'Big Top' (Bt), 'Armking' (Ak) and 'Nectaross' (Nr). Consistent QTLs are marked with an asterisk.

Trait	Parent[a]	QTL	Year	LOD	Nearest marker	Position (cM)	R^{2b}	a
Maturity date	Bt(Ak)	qP-MD5*	2013	4.95	SNP_IGA_588670	30.7	27.1	-11.3
			2014	6.34	SNP_IGA_550504	9.2	32.3	-8.6
% firmness loss (day 3)	Bt(Ak)	qP-pFL3d5	2014	4.58	SNP_IGA_553456	13.1	24.5	26.9
% firmness loss (day 5)	Bt(Ak)	qP-pFL5d5	2014	3.49	SNP_IGA_572589	20.7	19.3	18.4
Hardness	Bt(Ak)	qP-Har5	2013	5.26	SNP_IGA_552927	11.8	28.6	-4.7
Gumminess	Bt(Ak)	qP-Gum5	2013	3.79	SNP_IGA_552927	11.8	21.5	-1.7
Chewiness	Bt(Ak)	qP-Che5	2013	3.07	SNP_IGA_552927	11.8	17.8	-2.2
% firmness loss (day 3)	Bt(Nr)	qP-FL3d5*	2013	3.61	SNP_IGA_571548	18.7	30.9	26.2
			2014	3.43	SNP_IGA_545261	3.1	29.1	35.7
% firmness loss (day 5)	Bt(Nr)	qP-FL5d5*	2013	2.41	SNP_IGA_571548	19.7	21.8	17.2
			2014	4.63	SNP_IGA_559057	10.4	37.1	24.4
Maturity date	Bt(Ak+Nr)	qP-MD5*	2013	2.73	SNP_IGA_551853	7.2	10.2	-12.4
			2014	4.00	SNP_IGA_551853	7.2	14.1	-14.8
% firmness loss (day 3)	Bt(Ak+Nr)	qP-FL3d5*	2013	2.96	SNP_IGA_572589	17.7	11	15.5
			2014	8.57	SNP_IGA_553456	8.8	27.8	31.3
% firmness loss (day 5)	Bt(Ak+Nr)	qP-FL5d5	2014	7.38	SNP_IGA_572589	19.7	24.5	20.6
Maturity date	Ak	qP-MD4*	2013	10.94	SNP_IGA_409351	36.6	50.3	15.5
			2014	13.98	SNP_IGA_409351	36.6	57.6	11.5
		qP-MD6*	2013	3.59	SNP_IGA_605027	4.0	20.5	-9.7
			2014	5.21	SNP_IGA_605027	4.0	27.4	-7.8
% firmness loss (day 3)	Ak	qP-FL3d4	2014	6.16	SNP_IGA_409351	36.5	31.5	-30.4
% firmness loss (day 5)	Ak	qP-FL5d4*	2013	2.36	SNP_IGA_401886	25.2	14	-15.1
			2014	4.84	SNP_IGA_409351	36.5	25.7	-20.9
Hardness	Ak	qP-Har4*	2013	4.69	SNP_IGA_409351	36.5	25.9	4.5
			2014	6.30	SNP_IGA_404165	33.2	32.1	8.3
Gumminess	Ak	qP-Gum4	2014	4.58	SNP_IGA_404570	34.2	24.5	3.6
Maturity date	Nr	qP-MD4*	2013	8.95	SNP_IGA_409274	24.3	60.0	-21.0
			2014	20.05	SNP_IGA_409274	24.3	86.6	-32.0
% firmness loss (day 3)	Nr	qP-FL3d4	2013	4.45	SNP_IGA_409274	24.3	36.6	28.0
% firmness loss (day 5)	Nr	qP-FL5d4	2013	3.76	SNP_IGA_409274	24.4	31.9	20.4

[a] Bt(Ak): map of Bt obtained with the BtxAk cross; Bt(Nr): map of Bt obtained with the BtxAk cross; Bt(Ak+Nr): map of Bt obtained with the data from BtxAk and Btx Nr

[b] R^2: percentage of the total phenotypic variance for this trait explained by the QTL

[c] Additive effects: a = H - B, where H is the heterozygote and B the homozygote

The MD QTLs had different effects depending on the alleles carried by the parents of the two crosses studied. One allele of 'Armking' for qP-MD4 in homozygosis reduced the maturity date by 11.5-15.5 days compared with the heterozygote for the other allele, whereas for the same QTL, one allele of 'Nectaross' extended the maturity date from 21.0-32.0 days (Table 3.2). MD was also strongly associated with the FL character, often resulting in QTLs for FL3d, FL5d or both being at the same position as those found for MD. The effects of qP-MD4 and qP-MD5 were to increase firmness loss in the early maturing genotypes and to decrease firmness loss, i.e. producing a SMF effect, in the late maturing genotypes. Nevertheless, this did not occur in qP-MD6, where the QTL was not associated with the firmness loss character.

When looking for QTLs for the textural parameters obtained with TPA we found that these traits are mainly governed by the same loci as for firmness loss or maturity date QTLs. The majority of these QTLs were found in the maps of each parent in only one of the two years studied (Table 3.2) with the exception of a QTL in G4 for Hardness (qP-Har4) coinciding with the position of other QTLs for MD and FL.

The search for genes similar to the protein sequence to the NAC transcription factor candidate gene for *MD* (Pirona et al. 2013) allowed the identification of 61 genes with high similarity (p-value$<10^{-24}$) distributed along the whole peach genome. Only one of them (*Prupe.5G006200*) fell in the region of 3.3 Mb defined by the markers SNP_IGA543179 and SNP_IGA_557566 encompassing qP-MD5, and none were found on the qP-MD6 region of 2.3 Mb defined by markers snp_6_5294415 and SNP_IGA_611891 (Table S3.5).

3.4 - Discussion

We studied the variability for SMF with different measurements, mainly the loss of firmness during the postharvest process, and a set of fruit texture parameters measured with the texturometer at the time of harvest. The parents and progeny of two populations were measured: crosses between 'Big Top', a model variety for SMF, and the two cultivars 'Armking' and 'Nectaross', having very distant maturity dates but both with

typical melting (MF) behavior. For the inheritance analysis we developed high density linkage maps of these two populations. As in other peach high density maps (Eduardo et al. 2013; Martínez-García et al. 2013; Donoso et al. 2015), the maps of the parents (Bt, Ak and Nr) had extensive fragments without segregating markers, suggesting the presence of ample genomic regions identical by descent. These regions can be estimated as 100 minus the coverage of the genome (see Table 3.1) and accounted for 35-49% of the total physical distance depending on the map. Concentration of 1:2:1 segregating markers at specific genomic regions also indicates the regions where both parents were heterozygous, which ranged between 8% and 16% of the genome of the two populations used. Genes/QTLs located at these regions may also be heterozygous in both parents, which has to be taken into account for the genetic analysis. A Bt map has recently been obtained by Zeballos et al. (2016) using also the IPGI 9k Illumina SNP chip. This map was very similar to the one we produced with the exception that we could use almost four times more markers (1,596 vs. 405) for its construction. One of the main reasons for this improvement is that we followed a mapping approach that allowed integration of the markers segregating 1:2:1 and those with skewed segregation ratios, which improved the marker density, mapping accuracy, and also made it possible to identify additional map regions, resulting in a better physical coverage (65% vs. 40%).

Of the parameters used to measure the SMF character, the percentage of firmness loss (FL) seems the most efficient. FL measures the character according to SMF definition (a slower process of fruit softening after the field maturity stage than the MF types), as exemplified by the evolution of the FL parameter in the parents used, with a steady decay in the typically MF 'Armking' and 'Nectaross', and a delayed firmness loss in 'Big Top' (Figure 3.4). FL measurements detected QTLs that each explained a substantial part of the variability (20-35%), although they were not as reproducible as those found with the MD character. Part of this may have been caused by the small size of the populations used, particularly of Bt×Nr, so larger populations are needed for accurate measurement of the character. In addition, the spring-summer climatic conditions of 2013 and 2014 were extremely different in the region of Lleida. In 2013 temperatures were atypical, with strong oscillations, being on average the lowest since 1997 and with unusually high rainfall in July, whereas 2014 was closer to average. As a consequence, maturity times were substantially different, much later in 2013 than in 2014: 14 days later in Bt×Ak and 13 in Bt×Nr on average. This is one possible explanation for the low correlations found

in some of the firmness and texture-related traits between years and the heterogeneity in the estimation of certain characters for some years leading to a less accurate QTL analysis. The fact that MD shows a very high correlation between years (0.78-0.89) indicates that the criteria employed for harvest date determination were adequate. The other characters measured with the texturometer were less reliable. QTLs were not consistent across years and, when detected, they generally fell within the same regions as the FL and MD characters, overall providing little information.

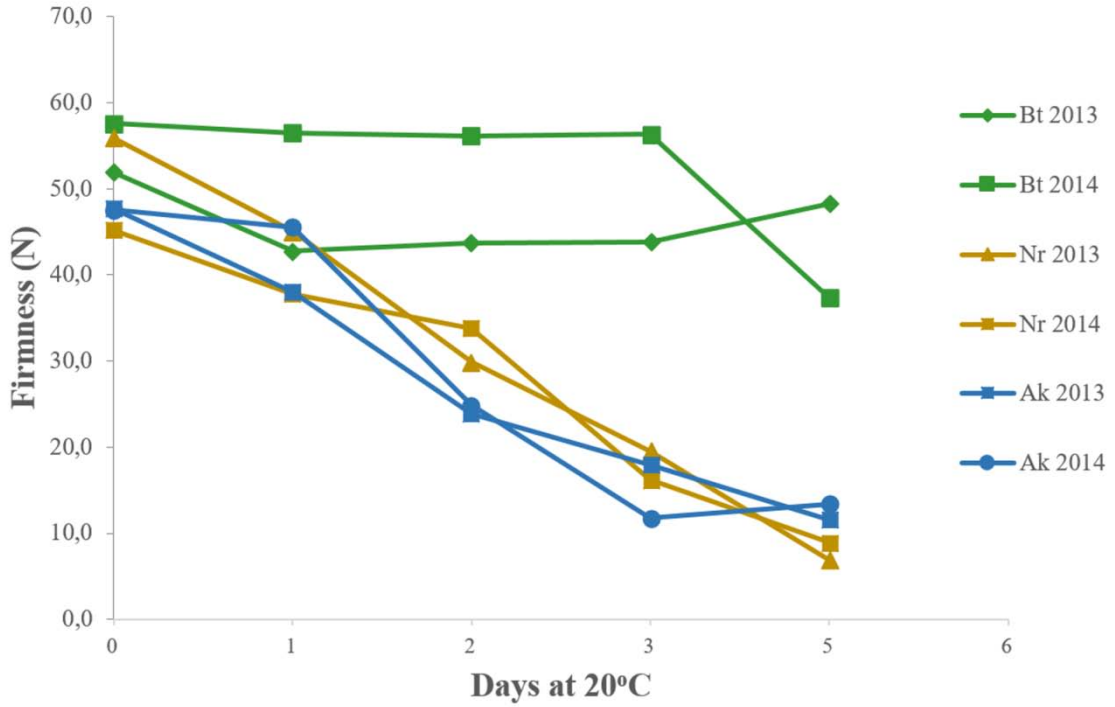


Figure 3.4 - Fruit firmness evolution for the parental lines 'Big Top' (Bt). 'Armking' (Ak) and 'Nectaross' (Nr) during a five-day period. A different behavior can be observed between 'Big Top' and the other two cultivars caused by the SMF trait. Firmness values were recorded in Newtons (N).

Our results detected essentially three regions of the genome on G4, G5 and G6, where genes involved in maturity time and flesh firmness consistently mapped. All had a QTL explaining much of the variability for the maturity date character and all had been reported previously for peach and other *Prunus*. The major QTL in the central part of G4 (qP-MD4) that we identified has been repeatedly reported by other authors in peach (Eduardo et al. 2011; Pirona et al. 2013; Verde et al. 2002) and in other *Prunus* such as

almond, cherry and apricot (Dirlewanger et al. 2012; Sánchez-Pérez et al. 2007; Donoso et al. 2016). In certain crosses (Eduardo et al. 2011; Pirona et al. 2013) it has been mapped as a single major gene (*MD*/*md*) on G4. This same region contains the slow ripening (*Sr*/*sr*) gene, and it has been hypothesized that the *sr* allele determining the SR character is one allele of *MD* (Eduardo et al. 2015; Núñez-Lillo et al 2015). The QTL on G6 (qP-MD6) has also been identified by Dirlewanger et al. (2012) in a peach F2 population ('Jalousia' × 'Fantasia'; J×F) explaining a large proportion (15.7-30.2%) of the phenotypic variability for this character. Finally, a QTL on a similar region of G5 to where we mapped qP-MD5 was also found by Dirlewanger et al. (2012) on J×F, and by Zeballos et al. (2015) although, in the former, it was not consistent over the years.

All QTLs for the firmness loss measurements fell within the same chromosomal regions as qP-MD4 and qP-MD5 and were detected on the maps of the parents that segregated for these QTLs ('Armking' and 'Nectaross' for qP-MD4 and 'Big Top' for qP-MD5), although they were identified with lower LOD scores than MD QTLs and in certain cases (in 2013 for FL3d5 and FL5d5 in Bt(Ak) and the same year for FL3d4 in Ak) were not consistent between the two years studied. The direction of the effects of these two characters was that late maturing fruit had slower firmness loss and vice-versa, which may be related to the observation of Giné-Bordonaba et al. (2016) on the lower capacity of late harvest peach fruit to produce ethylene. Similar results were obtained for qP-MD4 in relation to FL by Salgado et al. (2014). However, no QTLs for FL were associated with qP-MD6 in 'Armking'.

Two possible factors may explain the observed correlation between MD and SMF. One is that the environment determines that fruit of a tree that matures several weeks after another (with a longer period of attachment to the tree, and hence of exposure to its surrounding environment) may be at a different physiological maturity stage therefore determining a slower postharvest softening process. The second is that there is a common inheritance for these two characters where the same genes, or genes that are in the same region as those that determine maturity date in qP-MD4 and qP-MD5, are also involved in the postharvest ripening/softening process. Both factors may also act together and they are difficult to separate, but although the former may have an effect on the final result, this seems to be of minor importance considering that cultivars that are in the extremes of the production season of peach often have very similar postharvest behavior. This is

the case for ‘Armking’ and ‘Nectaross’, maturing with about 40 days difference and having similar FL parameters, or for ‘Big Top’, that, being the model for SMF, has a relatively early maturity time (first part of July under Lleida conditions). For the inheritance side, Eduardo et al. (2011) attributed the accumulation of QTLs for various fruit characters (fruit weight, skin color, soluble solids content, acidity) on the central part of G4 to a pleiotropic effect of the *MD* gene. In addition, Eduardo et al. (2015) identified the position of the *Sr* gene, that determines fruits that never ripen or soften, at the same place as *MD*, and Núñez-Lillo et al. (2015) found that the *sr* allele that determines this character co-maps with a large deletion at the position of the *Prupe.4G186800* NAC transcription factor gene that Pirona et al. (2013) identified as a possible candidate for *MD*. These observations are compatible with *MD* having different alleles that determine the speed of the maturation process, or even its arrest as in the case of *Sr*, a process that determines the maturity stage in the field, with the “slow” alleles or allele combinations maturing later than the “fast” ones. This same process may continue after harvest and determine the speed of the softening process, where again the slow or fast alleles would determine a longer or shorter shelf-life of the fruit.

While we noted the association between MD and FL on G4 in the progenies analyzed, it is clear that slight differences in softening rates may not be perceptible in practice and that a certain threshold has to be reached to be identified as having an impact on shelf-life, as with ‘Big Top’. The difference between ‘Big Top’ and the other MF cultivars may be due to the presence of the large QTL on G5 for MD and FL. This QTL has not been found in other peach progenies with the already mentioned exception of ‘Jalousia’ × ‘Fantasia’ cross (Dirlewanger et al. 2012), where the QTL had a smaller effect compared with ours and had a non-consistent behavior pattern. A QTL on G5 reported by Zeballos et al. (2016) for flesh firmness in the Bt map of a ‘Big Top’ × ‘Venus’ cross, seems to co-map with the one we detected, although these authors did not provide information on maturity date. Overall these data confirm the presence of a QTL for MD and FL on the G5 genomic region, with some inconsistencies in our case that we attribute to the small size of Bt × Nr and the contrasting climate conditions in 2013 and 2014. The allele combination of qP-MD5 in ‘Big Top’ or the interaction between the alleles of qP-MD4 and qP-MD5 may be responsible for the SMF phenotype of this cultivar. In fact, the individuals with the most favorable genotype for the SMF trait, based on the closest markers to the qP-FL and qP-MD loci (SNP_IGA_551853 in qP-MD5 and

SNP_IGA_409351 in qP-MD4), had an average FL3d=24.9 and FL5d=48.5, whereas the rest of the population had averages of FL3d=41.5 and FL5d=62.4, both significantly higher (t test with $p<0.01$).

A consequence of the suggested model of action would be that the SMF phenotype could only be found in late maturing individuals. This contradicts the relatively early-mid season maturity time of 'Big Top', about two weeks later than 'Armking' and one month before 'Nectaross'. A possible explanation could be the presence of other genes that affect maturity time but not the maturation or softening processes themselves. An example would be the qP-MD6, explaining approximately 30% of the variability of this character in 'Armking' and without an apparent effect on FL. Selection for early alleles of this QTL may produce varieties that mature early and have the SMF trait. The observation that the genomic regions of qP-MD4 and qP-MD5 contain NAC transcription factors while this is not the case in qP-MD6 (Table S3.5) is consistent with a different mode of action of these QTLs, where the first group would affect MD and FL while the gene(s) involved in qP-MD6 would only influence MD, with no detectable effect on FL. There are some examples of this in the Bt×Ak population, where plants #18, 68, 70 and 78, bearing the optimal marker genotypes for the SMF character (qP-MD4 and qP-MD5) and the early maturing genotype for the closest marker to qP-MD6 (homozygous for SNP_IGA_605027), also had a similar maturity time to 'Big Top' both seasons, and excellent SMF behavior, in some cases with even lower FL than 'Big Top' in both years (see Table S3.6).

Given that the combination of the different genes involved, including inter and intralocus interactions, determines the final phenotype, knowledge of the more precise locations of these QTLs is needed. This means analysing adequate segregating populations, where tightly-linked markers diagnostic to the important QTLs can be identified and validated. At the moment, there is information on the high heritability of the MD character (de Souza et al. 1998; Fresnedo-Ramírez et al. 2016) and the detailed position of the qP-MD4, with a marker based on the candidate gene (Meneses et al. 2016). Additionally, in this paper we have identified the qP-MD5 QTL as a key factor for SMF, the qP-MD6 that may modulate the maturity date, and we also provide a reasonable measurement of the SMF character based on the evolution of firmness loss (FL) during postharvest storage. Overall, this information is indispensable to fully understand the genetics of postharvest behavior

and establish a marker-informed breeding procedure for improvement of this crucial character in peach.

Data archiving statement

Genetic linkage maps and phenotypic data will be submitted to the Genome Database for Rosaceae (www.rosaceae.org).

Acknowledgments

We acknowledge financial support from the Spanish Ministry of Economy and Competitiveness, through the project AGL2015-68329-R, and "Severo Ochoa Programme for Centres of Excellence in R&D" 2016-2019 (SEV-2015-0533)", from the CERCA Programme-Generalitat de Catalunya and from the EU seventh Framework program by the FruitBreedomics project (FP7-KBBE-2010-265582): Integrated Approach for increasing breeding efficiency in fruit tree crop. The views expressed in this work are the sole responsibility of the authors and do not necessary reflect the views of the European Commission.

3.5 - Supplementary Material

Table S3.1 - Characteristics of the 24 microsatellites used to construct the maps of Bt, Ak and Nr. Details of the map and physical position and origin of the microsatellite can be found in Donoso et al. (2015).

		Mapped in		
Marker	Linkage group	Bt	Ak	Nr
M16a	1	X		
EPPCU5331	1	X		
EPPCU1945	1	X		
CPPCT026	1			X
BPPCT020	1	X		X
CPPCT042	1		X	
CPPCT029	1	X		X
PceGA34	2		X	
BPPCT007	3	X		X
BPPCT039	3	X		
EPDCU3083	3	X		
UDP96-008	3	X		
UDP96-003	4	X		X
M12a	4	X		X
CPP15636	4	X		X
EPPCU1775	4	X		X
Pacita021	5	X		X
EPDCU5183	5	X		X
CPSCT012	6			X
MA040a	6	X		X
CPPCT022	7	X		
CPPCT033	7		X	
EPPCU5176	7	X		X
CPPCT058	8	X		

Table S3.2 – Summary of the phenotypic data of 2013 and 2014 for the parents and the mean, minimum, maximum and standard deviation values for the Bt×Ak and Bt×Nr progenies.

2013	Parental lines			Bt×Ak					Bt×Nr				
Trait	'Bigtop'	'Armking'	'Nectaross'	mean	min	max	sd	SW test[a]	mean	min	max	sd	SW test[a]
MD	189	178	217	183.99	169	206	10.7	***	215.76	192	238	13.1	***
FLd3	15.57	62.53	65.00	43.62	-11.11	78.89	23.94	**	40.25	-4.94	78.96	22.37	ns
FLd5	7.08	75.70	87.61	59.01	0.6	87.03	20.05	***	58.82	13.95	85.83	17.45	ns
Springiness	1.23	2.09	1.23	1.01	0.62	2.63	0.26	ns	1.16	0.77	1.44	0.12	**
Hardness	8.71	8.75	17.05	11.62	3.9	21.83	4.34	*	15.54	1.98	33.47	6.69	*
Gumminess	4.31	3.70	7.87	5.18	1.78	10.16	1.83	ns	8.55	1.44	27.01	6.42	ns
Resilience	0.72	1.04	0.84	0.81	0.16	1.48	0.26	ns	0.79	0.14	1.98	0.35	ns
Chewiness	5.29	7.71	9.70	5.25	1.1	20.12	2.59	ns	10.21	1.2	33.58	8.17	ns
Cohesiviness	0.50	0.42	0.46	0.47	0.23	0.82	0.12	*	0.5	0.2	0.96	0.19	*

2014	Parental lines			BtxAk					BtxNr				
Trait	Bigtop'	Armking'	Nectaross'	mean	min	max	sd	SW test	mean	min	max	sd	SW test[a]
MD	174.00	163.00	202.00	169.52	161.00	184.00	7.45	***	201.91	177.00	226.00	16.72	***
FLd3	2.22	75.26	64.25	35.70	-34.56	83.18	26.59	ns	23.02	-61.23	74.68	32.94	ns
FLd5	35.11	71.78	80.28	58.78	-2.31	90.01	20.26	***	55.68	13.29	86.38	19.59	ns
Springiness	0.49	0.48	0.46	0.43	0.01	1.35	0.19	ns	0.46	0.37	0.62	0.05	ns
Hardness	30.39	6.42	14.05	16.54	1.98	29.02	7.20	*	13.20	6.54	32.28	4.26	ns
Gumminess	15.01	1.12	3.97	6.59	0.53	16.40	3.56	ns	3.69	1.27	14.03	2.23	ns
Resilience	0.46	0.08	0.15	0.28	-0.29	0.68	0.15	ns	0.17	-0.01	0.53	0.10	***
Chewiness	7.38	0.54	1.82	3.21	0.02	21.54	3.01	***	1.76	0.49	8.66	1.33	ns
Cohesiviness	0.53	0.15	0.26	0.42	0.13	1.14	0.18	***	0.29	0.14	1.06	0.15	ns

[a]p-value of the Shapiro-Wilk test for adjustment to a normal distribution: *p≤0.05; **p≤0.01;***p≤0.001; ns. non-significant

Table S3.3 - Spearman's Rank correlations for Bt×Ak and Bt×Nr progenies. Diagonal values shaded in grey represent correlations for the same trait between years. Values below diagonal line represent correlations between traits for 2013 and above the diagonal line are the correlations for 2014. Asterisks represent the significance of the correlation (** p-value<0.01; *** p-value<0.001)

Bt×Ak	MD	FL3d	FL5d	Springiness	Hardness	Gumminess	Resilience	Chewiness	Cohesiviness
MD	0.89***	-0.64***	-0.49***	0.4***	0.5***	0.39***	-0.02	0.47***	-0.17
FL3d	-0.07	0.34**	0.79***	-0.29	-0.65***	-0.51***	-0.2	-50***	0.08
FL5d	-0.14	0.75***	0.31**	-0.31**	-0.59***	-0.49***	-0.16	-50***	0.05
Springiness	-0.02	0.25	0.21	-0.18	0.55***	0.47***	-0.33**	0.74***	-0.29
Hardness	0.53***	-0.22	-0.16	-0.07	0.37*	0.77***	0.17	0.81***	-0.1
Gumminess	0.37**	-0.18	-0.07	0.05	0.78***	0.11	0.47***	0.91***	0.42***
Resilience	0.38***	-0.3*	-0.27	-0.32**	0.37**	0.12	0.04	0.19	0.7***
Chewiness	0.29	-0.06	0.01	0.4***	0.66***	0.9***	0.03	-0.02	0.01
Cohesiviness	-0.38***	0.13	0.21	0.11	-0.45***	0.14	-0.45***	0.17	-0.03
Bt×Nr	**MD**	**FL3d**	**FL5d**	**springiness**	**Hardness**	**Gumminess**	**Resilience**	**Chewiness**	**Cohesiviness**
MD	0.78***	-0.37	-0.41**	-0.11	-0.23	-0.39*	-0.2	-0.39***	-0.4**
FL3d	-0.65***	0.43**	0.8***	0.23	-0.01	0.12	-0.04	0.14	0.06
FL5d	-0.69***	0.68***	0.26	0.12	0.08	0.24	0.04	0.25	0.24
Springiness	-0.02	0.22	0.07	0.14	0.48***	0.23	-0.24	0.42	-0.11
Hardness	0.31	-0.22	-0.23	0.24	-0.08	0.84***	0.27	0.90***	0.52***
Gumminess	0.22	-0.13	-0.19	0.35	0.95***	-0.1	0.58***	0.97***	0.84***
Resilience	-0.34	0.05	0.35	-0.32	-0.55***	-0.67***	0.22	0.49***	0.71***
Chewiness	0.22	-0.1	-0.18	0.43**	0.94***	0.99***	-0.66***	-0.11	0.77***
Cohesiviness	0.19	0	-0.2	0.54***	0.67***	0.81***	-0.84***	0.74***	-0.07

Table S3.4 - QTLs identified in the Bt×Ak and Bt×Nr crosses obtained with genotype datasets integrating all markers (1:1 and 1:2:1). Consistent QTLs are marked with an asterisk

Trait	Cross	QTL	Year	LOD	Nearest marker	Position (cM)	R^2
Maturity date	BtxAk	qP-MD4*	2013	9,93	SNP_IGA_402256	42.9	47
			2014	9,46	SNP_IGA_401829	39.82	44,1
		qP-MD5*	2013	6,22	SNP_IGA_585500	30.75	32,8
			2014	6,62	SNP_IGA_552927	11.8	33,4
		qP-MD6*	2013	4,21	SNP_IGA_604703	12.6	23,6
			2014	5,68	SNP_IGA_605027	15.45	29,5
% firmness loss (day 3)	BtxAk	qP-FL3d4*	2013	3,02	SNP_IGA_394859	22.77	17,6
			2014	9,56	SNP_IGA_409371	61.06	44,4
		qP-FL3d5	2014	5,64	SNP_IGA_584315	27.663	29,3
% firmness loss (day 5)	BtxAk	qP-FL5d4*	2013	3,25	SNP_IGA_397228	27.5	18,8
			2014	7,25	SNP_IGA_409901	62.43	35,9
		qP-FL5d5	2014	4,66	SNP_IGA_584113	24.42	24,9
Springiness	BtxAk	qP-Spring4	2014	3,8	M12a	53.45	20,8
		qP-Spring5	2013	3,29	SNP_IGA_587708	35.53	19
Hardness	BtxAk	qP-Hard4*	2013	5,73	SNP_IGA_409901	62.43	30,7
			2014	9,96	SNP_IGA_407919	52.15	45,7
		qP-Hard5*	2013	5,46	SNP_IGA_552927	12.2	29,5
			2014	2,22	SNP_IGA_595212	53.39	12,7
		qP-Hard6	2013	3,33	SNP_IGA_605027	15.45	19,2
Gumminess	BtxAk	qP-Gum1	2014	3,15	SNP_IGA_7119	8.35*(G1.1)*	17,6
		qP-Gum4*	2013	2,18	SNP_IGA_409901	62.43	13
			2014	6,31	SNP_IGA_409544	61.43	32,1
		qP-Gum5	2013	3,9	SNP_IGA_552927	12.2	22,1
Resilience	BtxAk	qP-Resil3	2014	3,23	SNP_IGA_358781	65.52	18
		qP-Resil4	2013	5,63	SNP_IGA_408505	54.94	30,3
Chewiness	BtxAk	qP-Chew5	2013	3,85	SNP_IGA_552927	12.2	21,8
		qP-Chew6	2013	3,01	SNP_IGA_617922	2.34	17,5
Cohesiviness	BtxAk	qP-Cohesiv1	2013	3,32	SNP_IGA_24703	11.35	19,1
		qP-Cohesiv4	2013	4,79	SNP_IGA_408884	55.68	26,4
Maturity date	BtxNr	qP-MD4*	2013	9,01	SNP_IGA_410398	25.37	60,2
			2014	9,97	SNP_IGA_403152	20.03	63,1
% firmness loss (day 3)	BtxNr	qP-FL3d4	2013	4,98	SNP_IGA_409274	26.46	39,9
		qP-FL3d5*	2013	4,03	SNP_IGA_571548	19.28	33,8
			2014	3,91	SNP_IGA_557489	5.26	32,4
% firmness loss (day 5)	BtxNr	qP-FL5d1	2014	3,45	SNP_IGA_109648	25.49	29,2
		qP-FL5d4	2013	4,34	CPP15636	27.55	35,9
		qP-FL5d5*	2013	3,71	SNP_IGA_595648	51.11	31,6
			2014	5,17	SNP_IGA_571548	11.6	40,4
		qP-FL5d6	2013	3,34	SNP_IGA_695629	79,45	29
Cohesiviness	BtxNr	qP-Resil4	2013	5,15	CPP15636	28.55	41
Cohesiviness	BtxNr	qP-Cohesiv4	2014	3,91	SNP_IGA_402793	14.67	33,6

Table S3.5 - Gene sequences of peach chromosomes 5 and 6 producing a significant alignment with the NAC transcription factor gene *Prupe.4G186800*, candidate for the maturity date (MD) trait in chromosome 4, and their position on the peach genome sequence v2.0. The gene falling within the boundaries of the QTLs that we have estimated for QTLs qP-MD5 is presented with a green background

		Genome position	
Gene	E-value	chromosome	Position (bp)
Prupe.5G006200	9,00E-044	5	707.797
Prupe.5G040400	2,00E-045	5	4.430.568
Prupe.5G076100	3,00E-042	5	9011205
Prupe.5G131900	4,00E-043	5	12690115
Prupe.5G135400	1,00E-039	5	12859766
Prupe.5G146100	3,00E-039	5	13431803
Prupe.5G196000	5,00E-047	5	15882585
Prupe.5G221600	1,00E-049	5	17242546
Prupe.5G241300	4,00E-044	5	18167430
Prupe.6G098600	3,00E-055	6	6857507
Prupe.6G134400	1,00E-032	6	10621079
Prupe.6G138100	1,00E-036	6	11223633
Prupe.6G238600	3,00E-054	6	23830614

Table S3.6 - Phenotypes for maturity date (MD) and firmness loss (FL3d and FL5d) of peach fruit for the parents and progeny (nb 1-85) of 'Big Top' x 'Armking' and genotypes for closest markers to qP-MD5 in 'Big Top' (M1=SNP_IGA_589219), qP-MD4 in 'Armking' (M2=SNP_IGA_409351) and qP-MD6 in 'Armking' (M3=SNP_IGA_605368). Plants with the most favorable HAA genotype (#18, 68, 70 and 78) have phenotypes comparable to'Big Top' for the MD and FL traits.

	2013			2014					
Plant nb.	**MD**	**pFLd3**	**pFLd5**	**MD**	**pFLd3**	**pFLd5**	M1	M2 [a]	M3
'Big Top'	189	15.6	7.1	174	2.2	35.1	H	H	-
'Armking'	178	62.5	87.6	163	75.3	71.8	-	H	H
18	185	8.7	30.6	167	22.7	15.7	H	A	A
68	189	25.3	60.2	170	5.9	14.4	H	A	A
70	192	24.6	27.0	170	30.5	59.6	H	A	A
78	185	19.0	40.9	170	6.4	15.4	H	A	A
7	196	29.7	64.8	181	9.3	57.1	H	A	H
8	189	41.9	50.6	174	27.5	72.0	H	A	H
13	199	67.7	70.1	181	2.9	22.2	H	A	H
25	199	61.3	72.2	181	8.9	40.7	H	A	H
26	192	-3.3	57.9	174	21.0	58.6	H	A	H
27	199	69.8	79.4	184	-34.6	29.1	H	A	H
30	185	32.8	19.9	177	29.4	53.1	H	A	H
35	189	20.6	17.4	174	14.6	66.1	H	A	H
39	192	28.1	61.0	174	24.9	25.1	H	A	H
44	199	57.4	72.1	184	-27.2	-2.3	H	A	H
45	199	51.5	68.6	181	12.4	48.4	H	A	H
46	176	0.5	23.7	167	38.0	47.4	H	A	H
48	199	75.7	80.5	181	6.0	63.2	H	A	H
52	199	65.6	72.1	177	21.9	45.6	H	A	H
58	199	77.9	70.1	181	4.5	48.3	H	A	H
59	192	66.1	70.8	174	43.4	64.8	H	A	H
63	192	-3.6	45.1	174	11.9	42.4	H	A	H
66	192	11.1	27.8	174	33.7	41.3	H	A	H
76	203	64.1	64.8	181	15.4	61.0	H	A	H
83	189	25.4	50.4	181	35.6	82.6	H	A	H
1	185	47.0	79.2	170	41.8	67.7	A	A	A
4	192	47.0	54.3	174	22.9	50.0	A	A	A
5	178	-11.1	47.7	163	14.6	21.3	A	A	A
9	185	31.0	77.7	167	2.7	33.8	A	A	A
17	176	26.1	7.8	163	19.5	65.7	A	A	A
29	178	23.5	66.5	167	37.4	57.6	A	A	A
32	172	46.1	48.1	161	54.0	65.2	A	A	A
53	192	54.8	63.0	174	22.1	69.7	A	A	A
57	192	9.3	48.4	174	34.4	44.3	A	A	A
64	192	25.5	47.5	170	20.2	55.5	A	A	A

[a] SNP_IGA_409351 segregates 1:2:1 in BtxAk. Missing data have been imputed in M2 based on the close marker SNP_IGA_404570 that segregates 1:1

Table S3.6 - (Continued)

	2013			2014					
nr	**MD**	**pFLd3**	**pFLd5**	**MD**	**pFLd3**	**pFLd5**	M1	M2 [a]	M3
11	196	51,1	76,6	174	14,0	60,4	A	A	H
21	189	43,6	35,9	174	35,2	63,9	A	A	H
24	189	18,0	47,1	181	40,6	82,7	A	A	H
43	192	42,0	65,6	174	38,7	62,7	A	A	H
54	192	57,8	70,7	174	62,9	76,4	A	A	H
55	189	43,1	43,7	170	57,9	68,2	A	A	H
56	192	51,8	69,6	174	20,8	51,8	A	A	H
61	192	44,5	62,0	181	58,6	81,3	A	A	H
72	192	34,8	50,2	174	18,7	12,3	A	A	H
67	-	-	-	181	0,6	44,0	-	A	H
3	172	4,2	0,6	161	31,2	72,7	H	H	A
34	169	36,6	73,0	161	73,5	77,0	H	H	A
51	172	49,4	71,2	161	37,2	61,8	H	H	A
50	206	63,3	60,2	163	50,4	68,1	H	H	H
85	182	76,6	85,1	170	65,0	76,3	H	H	H
6	169	40,0	75,7	161	54,4	62,0	A	H	A
12	169	76,5	87,0	161	70,2	70,6	A	H	A
16	178	2,7	30,4	167	-7,6	31,7	A	H	A
19	178	25,7	55,7	167	-0,2	46,8	A	H	A
33	172	31,7	46,8	161	39,6	67,8	A	H	A
36	172	68,3	79,2	163	62,9	74,2	A	H	A
40	169	67,5	78,8	161	83,2	79,7	A	H	A
47	169	63,8	75,2	163	71,6	69,9	A	H	A
49	172	54,8	65,1	163	71,1	80,1	A	H	A
69	169	78,5	84,8	161	81,3	88,1	A	H	A
73	169	63,4	83,7	161	69,4	80,6	A	H	A
79	169	78,9	82,5	161	68,1	80,5	A	H	A
80	172	61,2	80,8	161	79,9	90,0	A	H	A
81	172	62,9	70,6	161	47,4	81,1	A	H	A
15	178	63,3	76,1	167	8,5	62,5	A	H	H
22	172	63,5	61,4	161	48,3	69,1	A	H	H
23	178	56,6	72,0	163	53,5	70,2	A	H	H
37	172	39,2	60,0	163	57,9	74,6	A	H	H
38	172	59,8	76,7	163	66,4	75,0	A	H	H
60	176	62,8	59,3	163	69,6	67,4	A	H	H
62	176	62,8	37,9	163	68,1	72,2	A	H	H
75	176	69,7	75,4	163	69,4	69,1	A	H	H
77	176	62,3	64,4	163	74,7	68,6	A	H	H
82	176	-9,3	19,9	163	29,8	55,9	A	H	H
14	-	-	-	161	67,5	84,0	-	H	A
31	-	-	-	161	32,3	74,2	-	H	A

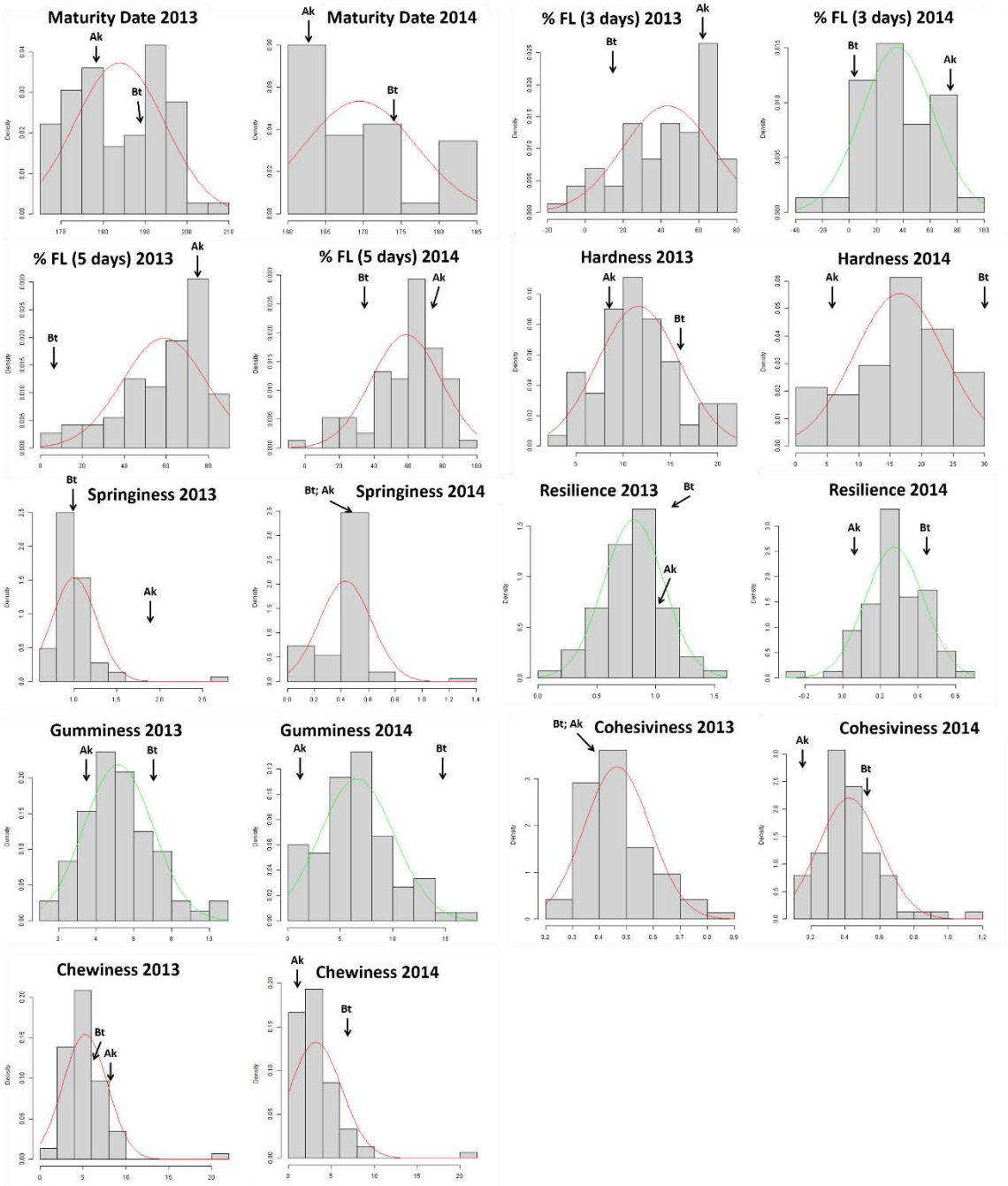


Figure S3.1 – Distribution of the traits analyzed in this work for Bt×Ak population in 2013 and 2014. Distribution curves are displayed above histograms. Green curves are for normally distributed data according to a Shapiro-Wilk normality test, red curves are not normal.

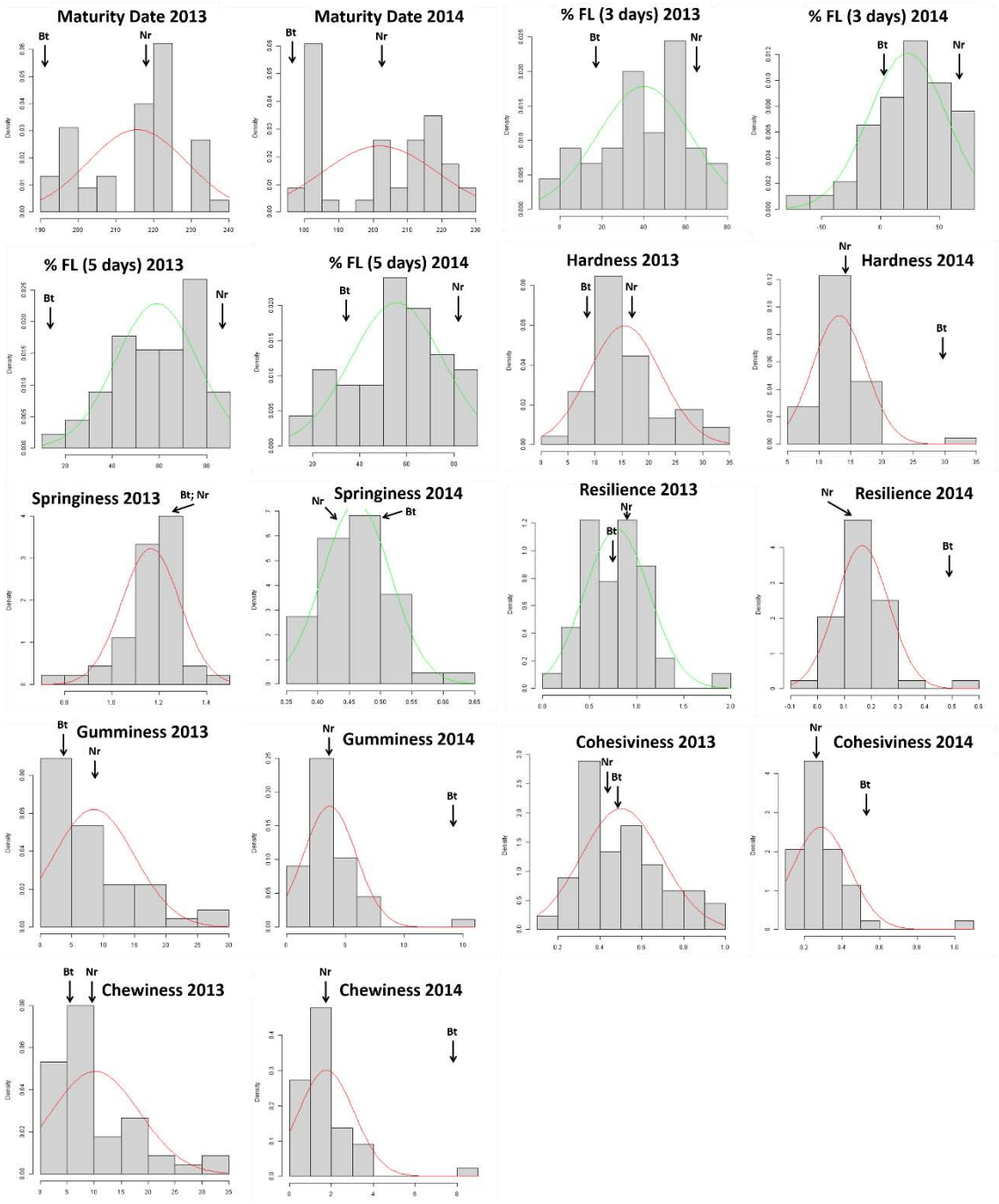


Figure S3.2 – Distribution of the traits analyzed in this work for Bt×Nr population in 2013 and 2014. Distribution curves are displayed above histograms. Green curves are for normally distributed data according to a Shapiro-Wilk normality test, red curves are not normal.

3.6 - Bibliography

Altschul, SF, Thomas LM, Alejandro AS, Jinghui Z, Zheng Z, Webb M, David JL (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25:3389-3402

Bassi D, Monet R (2008) Botany and taxonomy. In: DR Layne and D Bassi (eds) The Peach: Botany, production and uses. CABI, Wallingford, UK, pp 1-36

Begheldo M, Manganaris GA, Bonghi C, Tonutti P (2008) Different postharvest conditions modulate ripening and ethylene biosynthetic and signal transduction pathways in Stony Hard peaches. Postharvest Biol Tec 48:84-91

Bourne, M (2002) Food texture and viscosity. 2nd edn. Academic Press, San Diego

de Souza VA, Byrne DH, Taylor JF (1998) Heritability, genetic and phenotypic correlations, and predicted selection response of quantitative traits in peach: II. An analysis of several fruit traits. J Amer Soc Hortic Sci 123:604-611

Dirlewanger E, Quero-García J, Le Dantec L, Lambert P, Ruiz D, Dondini L, Illa E, Quilot-Turion B, Audergon JM, Tartarini S, Letourmy P, Arús P (2012) Comparison of the genetic determinism of two key phenological traits, flowering and maturity dates, in three *Prunus* species: peach, apricot and sweet cherry. Heredity 109:280-292

Donoso JM, Eduardo I, Picañol R, Batlle I, Howad W, Aranzana MJ, Arús P (2015). High-density mapping suggests cytoplasmic male sterility with two restorer genes in almond × peach progenies. Hortic Res 2:15016

Donoso JM, Picañol R, Serra O, Howad W, Alegre S, Arús P, Eduardo I (2016) Exploring almond genetic variability useful for peach improvement: Mapping major genes and QTLs in two inter-specific almond × peach populations. Mol Breeding 36:16

Doyle JJ, Doyle JL (1990) Isolation of plant DNA from fresh tissue. Focus 12:13–15

Eduardo I, Chietera G, Pirona R, Pacheco I, Troggio M, Banchi E, Bassi D, Rossini L, Vecchietti A, Pozzi C (2013) Genetic dissection of aroma volatile compounds from the essential oil of peach fruit: QTL analysis and identification of candidate genes using dense SNP maps. Tree Genet Genomes 9:189–204

Eduardo I, Pacheco I, Chietera G, Bassi D, Pozzi C, Vecchietti A, Rossini L (2011) QTL analysis of fruit quality traits in two peach intraspecific populations and importance of maturity date pleiotropic effect. Tree Genet Genomes 7:323–335

Eduardo I, Picañol R, Rojas E, Batlle I, Howad W, Aranzana MJ, Arús P (2015) Mapping of a major gene for the slow ripening character in peach: co-location with the maturity date gene and development of a candidate gene-based diagnostic marker for its selection. Euphytica 205:627-636

Fresnedo-Ramírez J, Frett TJ, Sandefur PJ, Salgado-Rojas A, Clark JR, Gasic K, Peace CP, Anderson N, Hartmann TP, Byrne DH, Bink MCAM, Weg E, Crisosto CH, Gradziel TM (2016) QTL mapping and breeding value estimation through pedigree-based analysis of fruit size and weight in four diverse peach breeding programs. Tree Genet Genomes 12:1-18

Ghiani A, Negrini N, Morgutti S, Baldin F, Nocito FF, Spinardi A, Mignani I, Bassi D, Cocucci M (2011) Melting of 'Big Top' nectarine fruit: some physiological, biochemical, and molecular aspects. J Amer Soc Hortic Sci 136:61-68

Giné-Bordonaba J, Cantín CM, Echeverría G, Ubach D, Larrigaudière C (2016). The effect of chilling injury-inducing storage on qualty and consumer acceptance of different *Prunus persica* cultivars. Postharvest Biol Tec 115:38-47

Gu C, Wang L, Wang W, Zhou H, Ma BQ, Zheng HY, Fang T, Ogutu C, Vimolmangkang S, Han YP (2016) Copy number variation of a gene cluster encoding endopolygalacturonase mediates flesh texture and stone adhesion in peach. J Exp Bot 67:1993-2005

Haji T, Yaegaki H, Yamaguchi M (2001) Changes in ethylene production and flesh firmness of melting, nonmelting and stony hard peaches after harvest. J Jpn Soc Hortic Sci 70:458–459

Haji T, Yaegaki H, Yamaguchi M (2005) Inheritance and expression of fruit texture melting, non-melting and stony hard in peach. Sci Hortic 105:241–248

Hayama H, Tatsuki M, Ito A, Kashimura Y (2006) Ethylene and Fruit softening in the stony hard mutation in peach. Postharvest Biol Tec 41:16-21

Iglesias I (2012) Producción, consumo e innovación varietal en el melocotón. Alimentación, equipos y tecnología 268:25-31. ISSN 0212-1689

Iglesias I, Echeverria G (2009) Differential effect of cultivar and harvest date on nectarine colour, quality and consumer acceptance. Sci Hortic 120:41–50

Lander ES, Green P, Abrahamson J, Barlow A, Daly MJ, Lincoln SE, & Newburg L (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. Genomics 1:174–181

Lester DR, Sherman WB, Atwell BJ (1996) Endopolygalacturonase and the melting flesh (M) locus in peach. J Amer Soc Hortic Sci 121:231–235

Martínez-García PJ, Parfitt DE, Ogundiwin EA, Fass J, Chan HM, Ahmad R, Lurie S, Dandekar A, Gradziel TM, Crisosto CH (2013) High density SNP mapping and QTL analysis for fruit quality characteristics in peach (*Prunus persica* L.). Tree Genet Genomes 9: 19–36

Meneses C, Ulloa-Zepeda L, Cifuentes-Esquivel A, Infante R, Cantin C, Batlle I, Arús P, Eduardo I (2016) A codominant diagnostic marker for the slow ripening trait in peach. Mol Breeding 36:77

Nuñez-Lillo G, Cifuentes-Esquivel A, Troggio M, Micheletti D, Infante R, Campos-Vargas R, Orellana A, Blanco-Herrera F, Meneses C (2015) Identification of candidate genes associated with mealiness and maturity date in peach [*Prunus persica* (L.) Batsch] using QTL analysis and deep sequencing. Tree Genet Genomes 11:86

Pan L, Zeng WF, Niu L, Lu ZH, Liu H, Cui GC, Zhu YQ, Chu JF, Li WP, Fang WF, Cai ZG, Li GH, Wang ZQ (2015) *PpYUC11*, a strong candidate gene for the stony hard phenotype in peach (*Prunus persica* L. Batsch), participates in IAA biosynthesis during fruit ripening. J Exp Bot 66:7031–7044

Peace CP, Crisosto CH, Gradziel TM (2005) Endopolygalacturonase: a candidate gene for Freestone and Melting flesh in peach. Mol Breeding 16:21–31

Pirona R, Eduardo I, Pacheco I, Da Silva Linge C, Miculan M, Verde I, Tartarini S, Dondini L, Pea G, Bassi D, Rossini L (2013) Fine mapping and identification of a candidate gene for a major locus controlling maturity date in peach. BMC Plant Biol 13:166

R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

Ramina A, Tonutti P, McGlasson B (2008) Ripening, nutrition and postharvest physiology. In: The Peach: Botany, production and uses. DR Layne and D Bassi (eds). CABI (Wallingford, UK) pp. 550-574

Ramming DW (1991) Genetic control of a slow-ripening fruit trait in nectarine. Can J Plant Sci 71:601–603

Salgado A, Clark J, Sandefur P, Frett T, Peace C (2014) Softening rate and ethylene fruit production of different flesh textures of Arkansas peach and nectarine germplasm. American Society for Horticultural Science congress, Orlando, FL, USA, july 27th to august 1st, 2014.

Sánchez-Pérez R, Howad W, Dicenta F, Arús P, Martínez-Gómez P (2007) Mapping major genes and quantitative trait loci controlling agronomic traits in almond. Plant Breeding 125:1-9

Van Ooijen JW (2009) MapQTL ® 6, Software for the mapping of quantitative trait loci in experimental populations of diploid species. Kyazma B.V., Wageningen, The Netherlands

Verde I, Bassil N, Scalabrin S, Gilmore B, Lawley CT, Gasic K Micheletti D, Rosyara UR, Cattonaro F, Vendramin E, Main D, Aramini V, Blas AL, Mockler TC, Bryant DW, Wilhelm L, Troggio M, Sosinski B, Aranzana MJ, Arús P, Iezzoni A, Morgante M, Peace C (2012) Development and evaluation of a 9K SNP array for peach by internationally coordinated SNP detection and validation in breeding germplasm. PloS One 7:e35668

Verde I, Quarta R, Cedrola C, Dettori MT (2002). QTL analysis of agronomic traits in a BC1 peach population. Acta Hortic 592:291–297

Zeballos JL, Abidi W, Giménez R, Monforte AJ, Moreno MA, Gogorcena Y (2016) Mapping QTLs associated with fruit quality traits in peach [*Prunus persica* (L.) Batsch] using SNP maps. Tree

Genet Genomes 12:37

Ziosi V, Noferini M, Fiori G, Tadiello A, Trainotti L, Casadoro G, Costa G (2008) A new index based on vis spectroscopy to characterize the progression of ripening in peach fruit. Postharvest Biol Tec 49:319–329

4. Marker-assisted introgression (MAI) of almond genes into the peach background: a fast method to mine and integrate novel variation from exotic sources in long intergeneration species

Marker-assisted introgression (MAI) of almond genes into the peach background: a fast method to mine and integrate novel variation from exotic sources in long intergeneration species

Octávio Serra[1*], Jose Manuel Donoso[1,2*], Roger Picañol[1,3*], Ignasi Batlle[4], Werner Howad[1], Iban Eduardo[1] and Pere Arús[1§]

[1]IRTA, Centre de Recerca en Agrigenòmica CSIC-IRTA-UAB-UB. Campus UAB, Cerdanyola del Vallès (Bellaterra), 08193 Barcelona, Spain

[2]Current address: INIA, Av. Salamanca s/n, Sector Los Choapinos, Rengo, Chile

[3]Current address: Rijk Zwaan, Finca La Marina-Pj. Lo Contreras s/n, 30395 La Puebla, Cartagena, Spain

[4]IRTA. Centre Mas de Bover. Crta. Reus - El Morell Km 3.8. 43120 Constantí, Tarragona, Spain

*O. Serra, JM Donoso and R. Picañol contributed equally to the research presented in this paper

[§]Corresponding author: e-mail: pere.arus@irta.cat; phone: +34607073923; fax: +34935636601

Published in Tree Genetics & Genomes, 14th September 2016

Abstract

This paper proposes a new breeding strategy, marker-assisted introgression (MAI), to obtain lines of perennial species with a single introgressed fragment from a compatible species two generations after the interspecific hybrid. MAI allows enrichment of the genome of a species with genes from a wild or exotic relative in a short timeframe and with an intermediate step that allows a first exploration of genes/QTLs that the donor species can provide to the target crop. This method has three phases: 1) creating a large backcross one (BC_1) population to select, with markers, a reduced number of individuals (15-30, called the prIL set) with a low number of introgressions; 2) phenotyping the prIL set for the traits of interest and inferring the inheritance and map position of segregating major genes/QTLs based on the known genotypes of the prILs; and 3) advancing selected lines carrying the traits of interest to a next generation of backcross or selfing to obtain individuals with a single introgression in the background of the elite commercial germplasm. The proof of concept of this strategy was implemented by using peach as the recurrent species and almond as the donor. The whole process can be done in 9-10 years as the identification of the first line with one introgression was after five years (2006-2011), and 4-5 additional years are needed for phenotypic evaluation of selected lines. The expansion of this method to other perennial clonally-propagated crops and to other species of *Prunus* compatible with peach is discussed.

Keywords: *Prunus persica*, *Prunus dulcis*, introgression lines, near-isogenic lines, fast track breeding, MAI

4.1 - Introduction

Peach (*Prunus persica*) is a species with a low level of variability (Aranzana et al. 2012), particularly when compared to other crops of the same genus such as almond, apricot, plum or cherry (Byrne 1990; Mnejja et al. 2010). This is because peach, unlike most other *Prunus*, lacks a functional self-incompatibility system, and because of major bottlenecks since its domestication more than 5,000 years ago in China (Faust and Timon 1995). Low levels of variability limit the progress towards the main challenges of current peach breeding that include longer shelf life, improved fruit quality, resistance to pests and diseases, and adaptation to climate change.

One way to increase genetic variation is by using the vast reservoir of novel alleles found in landraces, cultivated relatives, and wild germplasm (Tanksley and McCouch 1997). New sources of variability for peach varieties grown in western countries may arise from local European cultivars (Aranzana et al. 2003), Chinese accessions (Li et al. 2013; Micheletti et al. 2015) or other cultivated species of the genus that are compatible to peach, such as almond (*P. dulcis*) and Japanese plum (*P. salicina*), or wild *Prunus* species close to peach, such as *P. mira*, *P. kansuensis*, *P. davidiana* and *P. cerasifera*. Some interspecific hybrids between peach and these species are currently used as rootstocks (Bouhadida et al. 2007; Byrne et al. 2012). The first saturated molecular marker maps of peach were developed using interspecific populations between peach and almond (Foolad et al. 1995; Joobeur et al. 1998) and peach crosses with *P. davidiana* (Foulongne et al. 2002). Given the low level of variability of peach, these populations were a guarantee of high marker polymorphism to facilitate development of densely and homogeneously populated maps with coverage of the entire genome. One of these maps, an F_2 progeny between 'Texas' almond and 'Earlygold' peach was used as the reference map for the genus and taken as the basis for linkage group assignment and direction, providing data on the position of a large number of transferable markers to be used to construct maps in other populations (Dirlewanger et al. 2004).

Genes from compatible species have been a major source of useful variability in the cultivated tomato, a species that, like peach, has a low level of variability and is compatible with several wild relatives. These genes include mainly disease resistances (Zamir et al. 1994; Griffiths and Scott 2001; Bai et al. 2003; Verlaan et al. 2013), and other important traits such as soluble solids content, yield, early fruit ripening, color and

viscosity (Eshed and Zamir 1995; Tanksley and Nelson 1996; Fulton et al. 2000; Frary et al. 2003). Another example is apple (*Malus* x *domestica*), where resistance to apple scab (*Venturia inaequalis*) has been identified from various wild relatives, such as *M. floribunda* (Gessler and Pertot 2012) and *M. sieversii* (Bus et al. 2005). In peach, while several nematode resistance genes have been identified and characterized from *P. cerasifera,* almond and other peach cultivars, and are being introduced into rootstocks (Gomez-Aparisi et al. 2001; Esmenjaud et al. 2009), as yet there is no example of a disease resistance gene from wild sources that has been introgressed into peach commercial scions. The reason is that, using conventional breeding approaches such as the backcross method (Allard, 1961), the long intergeneration period of this species (3-4 years) makes this an extremely long process, far from the reach of commercial breeding operations.

Molecular markers can shorten the introgression process and make it much more efficient. The use of markers to select for the whole genome of the recurrent parent in the backcross method was first proposed by Tanksley et al. (1981), and is currently routine for marker selection in many breeding programs, cutting the process of gene introgression by at least half. Later, Tanksley and Nelson (1996) proposed a more elaborate approach, termed "advanced backcross QTL analysis", to facilitate the introgression of new genes from wide crosses and to survey and map, at the same time, the variability that the exotic donor parent can provide the elite commercial type. This method was at the origin of the development of near-isogenic line (NIL) or introgression line (IL) collections in several species, a powerful resource for fine analysis of QTLs, that has provided geneticists and breeders with important information in many species (Grandillo et al. 2007). In this paper, we adapted these approaches for the introgression of almond alleles in peach, and propose a basic scheme, termed marker-assisted introgression (MAI), that allows a) a first survey of the variability that could be introduced from almond to peach, identifying the map positions of major genes dominant for the almond or additive, and b) plants with a desired almond chromosome fragment in the peach background to be obtained only two generations after the interspecific hybrid, making the whole process feasible in a timeframe compatible with integration in a conventional fruit breeding program.

We tested this scheme using the 'Texas' almond as donor source and the 'Earlygold' peach as recurrent parent. F_2 and BC_1 populations with these two parents have been

extensively studied by our group in the past, where we have developed high density maps, described cytoplasmic male sterility with two restorer genes (Donoso et al. 2015), and analyzed the inheritance of 42 traits (Donoso et al. 2016). The first crosses of the MAI project were performed in the spring of 2006, and we currently have a set of plants with a single introgression in homozygosis or heterozygosis, demonstrating the feasibility of the method, and providing: a) a resource for genetic analysis for peach and other stone fruit, and b) a set of peach lines with genes of interest from almond that can be readily incorporated into commercial breeding programs.

4.2 - Materials and methods

4.2.1 - Plant materials, crosses, markers and maps

We used the hybrid plant 'MB1.37' from the cross between 'Texas' (syn. 'Mission') almond as female parent and 'Earlygold' peach as pollen donor, using the latter as recurrent parent to obtain the backcross one (BC_1) generation to which we will refer as T1E. Crosses to obtain T1E were performed during the spring of 2006, 2007 and 2008 until more than 1,000 seedlings were obtained. Individuals from more advanced generations were obtained from crosses carried out between 2010 and 2015, usually with 'Earlygold' as the female parent, because of the male sterility detected in individuals with the 'Texas' cytoplasm (Donoso et al. 2015). Where necessary, considering the bloom time of 'Earlygold' and other individuals used as male parents, pollen from one year was collected, kept in Parafilm sealed Petri dishes at -20°C, and used for pollination the following year. Pollen viability was confirmed using the method described by Asma (2008) with an additional 15% sucrose.

A set of 113 SSR markers mapped in the T1E population and covering its total map distance at regular intervals were selected for this study. Their map positions are those of the linkage map constructed by Donoso et al. (2015) and are shown in Table S4.1. The chromosome fragment covered by the introgression was defined by the two extreme markers within the introgression, and its size was measured as the difference between the positions in cM of the two extreme markers within the introgression plus half of the

distance between the last marker within the introgression and the first outside it. The size of each linkage group was taken as that of the last SSR marker mapped (see Table S4.1). The distance between pairs of markers was short (average of 3.2 cM), and the two most distant markers were 12.9 cM apart (M6a and UDP98-409 on linkage group eight). For this reason we considered that the chances of an undetected double recombination occurring within a given introgression were too low ($P<0.05$ in the most favorable case) to be considered.

4.2.2 - MAI breeding scheme

Given the long intergeneration period of *Prunus*, we chose a strategy that minimized the number of generations needed to obtain individuals that contained the peach genome except for a single almond DNA fragment, and allowed the discovery of DNA fragments from almond containing major genes of interest for peach. This strategy consisted of three different phases (Figure 4.1).

4.2.2.1 - Phase 1: Selection of individuals with a few introgressed almond fragments from a large BC1 progeny

Genomic DNA was extracted from young leaves of seedlings from the T1E progeny using the CTAB method (Doyle and Doyle, 1990), omitting the final RNAse treatment. Plants of T1E were selected using a three-step procedure in order to reduce the cost of the operation. In the first step, all plants were analyzed with a set of eight SSRs (set 1; Table S4.2), all at one extreme of the eight *Prunus* linkage groups selected from the T×E map of Dirlewanger et al. (2004). Only plants having three or less of these markers in heterozygosis were selected. In the second step, the selected plants were genotyped for a second SSR set (set 2; Table S4.2) with eight more SSRs located at the extreme of the linkage group opposite to those of set 1. Only the plants with three or less heterozygous loci for sets 1 and 2 were selected. The eight markers of set 1 were selected to be multiplexed at an annealing temperature (Ta) of 57°C. Selection in step two was multiplexed in two groups of four markers, one (2a) at Ta=55°C and the other (2b) at

Ta=50°C (Table S4.2). Finally, the selected plants were genotyped for 97 additional SSRs with coverage of the complete genome (Table S4.1).

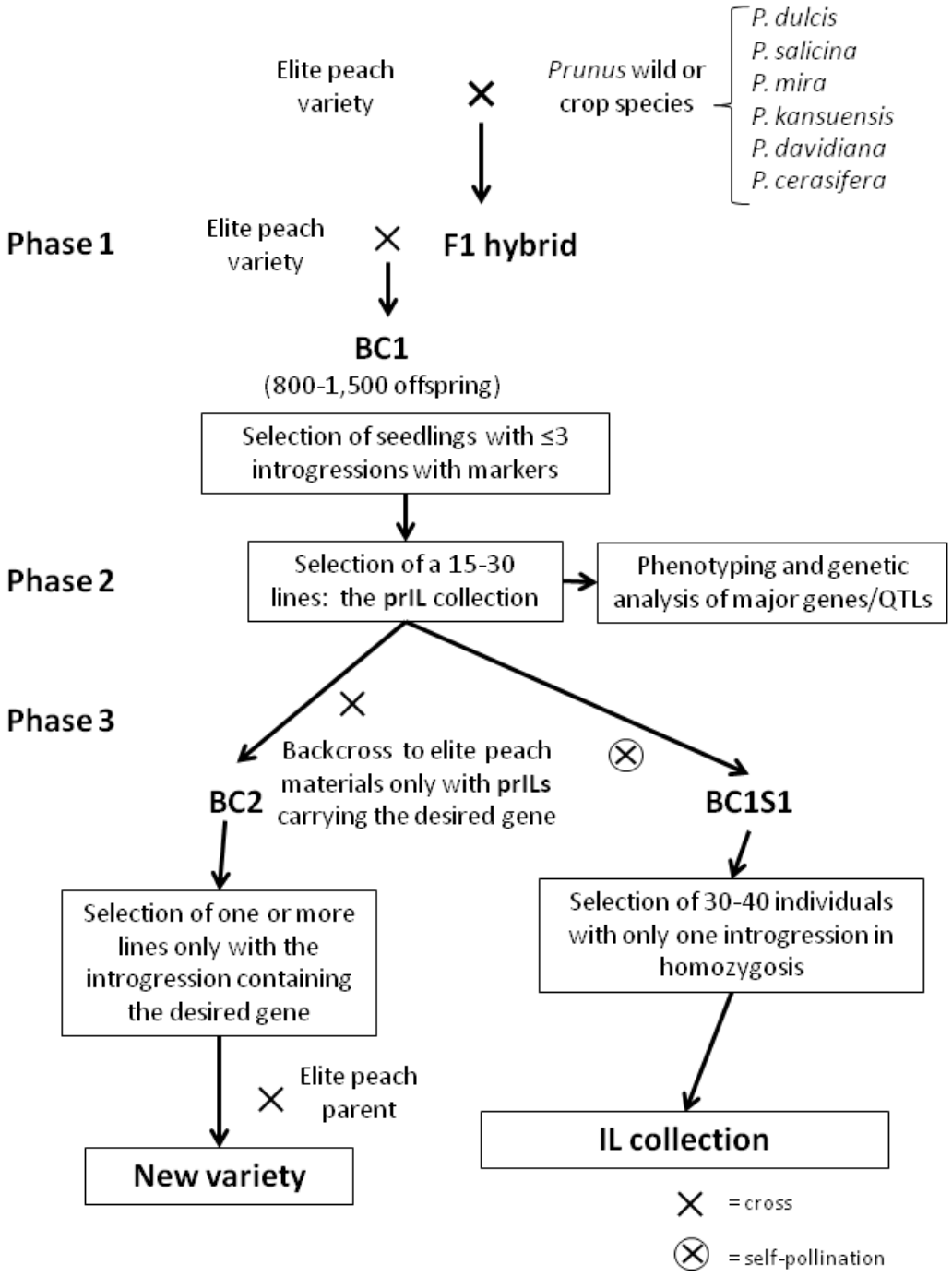


Figure 4.1 - Scheme of the marker-assisted introgression (MAI) strategy for the integration of exotic chromosome fragments into the elite peach genome

Markers were analyzed in an ABI Prism® 3130*xl* (Applied Biosystems) capillary sequencer, following the methods for PCR amplification described by Aranzana et al. (2003). Marker positions (cM) were those obtained previously in the T1E population by Donoso et al. (2015).

4.2.2.2 - Phase 2: The prIL set: Phenotyping and genetic analysis of lines with a low number of almond introgressions

The aim was to choose a set of lines with a low number of introgressions (≤3), together containing the whole almond genome, so making it possible to analyze a set of individuals with a genetic composition almost completely identical to peach but with a few well-characterized almond introgressed fragments. This set of selected lines, which we refer to as prILs (for pre-introgression lines), was the basis for the extraction of a complete set of ILs, and have been previously phenotyped for a set of 42 characters of flower, fruit, phenology and disease resistance (Donoso et al. 2016).

With the phenotyped and genotyped prIL set it was possible to determine the map positions of major genes with dominant alleles for almond or with additive effects. For proof of concept, we used six major genes segregating in T1E and already mapped in conventional populations by Donoso et al. (2015 and 2016). These were: resistance to powdery mildew (*Vr3*); flesh color (*DBF2*); flower color (*Fc2*); juiciness (*Jui*) and male sterility (*Rf1* and *Rf2*). QTLs with large effects were also analyzed, and two QTLs identified by Donoso et al. (2016), one for maturity date (*qMD4*) explaining R^2=33-58% of the phenotypic variance, and one for stone weight (*qSD6*), with R^2=28-46%, were included in this study.

For their genetic analysis, the lines of the prIL set were divided into the two phenotypic classes and their graphical genotypes compared. Chromosomal regions having one genotype in all individuals of one phenotypic class (the heterozygote almond/peach) and the alternative (genotypes with only peach alleles) in the other were identified as those that contained the gene under study. A similar approach was followed for quantitative characters, ordering the prIL individuals by the value of the trait studied.

4.2.2.3 - Phase 3: Selection of lines with a single introgression in the BC_2 and BC_1S_1 generations

T1E lines selected with markers to have three or less introgressions, and containing the gene of interest, were backcrossed (BC_2) or selfed (BC_1S_1) to obtain individuals with a single introgressed fragment in heterozygosis or homozygosis, respectively. These lines were selected with the two markers at the boundaries of these fragments. While lines with a single almond fragment in heterozygosis (ILs) can be obtained in the BC_2 progeny, a selfing generation from the BC_1 (BC_1S_1) or BC_2 (BC_2S_1) would be required to obtain homozygous ILs. The number of offspring required to obtain homozygous ILs from a line with two or three introgressions in heterozygosis would be 46 and 190 seedlings, respectively (calculated as n= log α /log (1-p), where n= number of individuals, α = 0.05 and p= probability of finding a specific homozygous IL, which is $(¼)^k$, with k=number of introgressions of the parental line). Individuals with more introgressions would require an additional backcross generation to give heterozygous lines with three or less introgressions that could be selfed and used to extract homozygous ILs. As the unexpected detection of cytoplasmic male sterility in the almond × peach (Donoso et al. 2015) progenies resulted in a large number of selected prILs being male sterile, we decided to recover the peach cytoplasm by using 'Earlygold' as the female parent. The female parent for crosses during the 2011-2013 period was usually 'MB1.37', and occasionally 'Earlygold', and from 2014 onwards only 'Earlygold'.

Seeds from T1E were germinated under standard conditions (Donoso et al. 2015). In the case of the BC_2 and BC_2S_1 plants, an embryo rescue protocol, described in Batlle et al. (2012), was employed for all seeds, since 'Earlygold' and some T1E individuals are early ripening (90-100 days after flowering), and the seeds require embryo rescue to develop into seedlings.

4.3 - Results

4.3.1 - Phase 1: Selection in the BC_1 generation

From 2006 to 2008, we performed approximately 8,500 controlled pollinations that yielded 1,720 seeds. These seeds were stratified and produced a total of 1,080 seedlings, to which we added 15 T1E plants obtained previously, giving a total of 1,095 analyzed individuals. The first step of the selection with eight SSRs allowed 213 (20%) plants that did not belong to the T1E population to be identified: they originated from self-pollination (14%) or from cross-pollination (5%) with neighboring trees, which gave a final number of 882 BC_1 plants. In this first set of markers, 3,899 data points (55%) heterozygous with one allele of almond and another of peach were interpreted as introgressions (Table 4.1). The remaining data points were considered as homozygous for the peach alleles (heterozygous markers for the two peach alleles were considered homozygous too). In all, linkage group 2 (G2) had the lowest percentage of heterozygous individuals (49%) and G8 had the highest (66%). After marker examination, 226 plants (26% of the initial 882) with three or less detected introgressions were carried to the next stage of selection.

Eight additional SSRs were genotyped in these 226 plants, where we found 1,084 heterozygous data, 60% of the total analyzed. Considering the 16 markers together and assuming that individuals with both markers in heterozygosis for the almond allele in a particular linkage group had a single introgression, the proportion of individuals with at least one introgression ranged from 38% in G3 to 50% in G8 (Table 4.1). There were 22 plants selected to have three or less introgressions at this stage (10% of the 226, or 3% of the 882), one with two introgressions and the other 21 with three.

In the final step, 97 additional SSRs analyzed in the 22 selected plants allowed identification of undetected introgressions and genotyping errors in the previous steps, giving a final selection of nine plants with three or less introgressions (eight plants with three and one plant with two). Three of these had low vigor and were discarded for further use.

Table 4.1 - Percentage of individuals having at least one introgression in each chromosome and on average in the different stages of selection in the BC1 generation of 'Texas' almond x 'Earlygold' peach.

Selection stage	Nº of plants[a]	Linkage group								
		G1	G2	G3	G4	G5	G6	G7	G8	Average
Stage 1	882	57.2	48.7	54.4	58.5	54.2	50.4	52.6	66.3	55.2
(8 SSRs)	226	37.4	25.2	28.5	37.5	28.5	33.8	28.6	44.5	33.0
Stage 2	226	45.3	38.0	37.7	46.4	41.8	39.7	42.3	50.3	42.6
(16 SSRs)	22	45.4	31.8	18.1	63.6	22.7	31.8	40.9	40.9	36.3
Stage 3	22	54.5	45.5	27.3	72.7	31.8	36.4	45.5	45.5	45.5
(113 SSRs)	9	66.7	11.1	11.1	77.8	33.3	11.1	33.3	44.4	33.3

[a]On the top line, the initial number of plants, the bottom line shows the number of plants selected at the end of each stage to obtain plants with three or less introgressions.

4.3.2 - Phase 2: Generation of the prIL set and genetic analysis of major genes and QTLs

Due to the low number of vigorous plants with three or less introgressions (six), we completed the set of prILs with 12 individuals with four introgressions to obtain a final set of 18 plants: one with two introgressions (prIL 67), five with three introgressions (prILs 87, 147, 306, 630, 695) and 12 with four introgressions (prILs: 3, 27, 31, 79, 110, 193, 333, 412, 498, 505, 694, 769). The genotypes of these plants are described in Table S4.1 and together cover, at least threefold, all regions of the almond genome except for the distal end of G5, where coverage is only double (Figure 4.2).

The prIL set contained 66 introgressed fragments from almond in the peach genetic background (Table 4.2). About a third of them (21 introgressions) correspond to the entire chromosome (at least one case per chromosome), and the remaining two thirds (45 introgressions) exhibited a single recombination event. In no case did more than one recombination event occur in a given chromosome. The distribution of the different fragments varied from five in G3 to 15 in G4, although most linkage groups had six to eight introgressions. The average length of the introgression was 30.4 cM, representing an average of 67.6 % of the total length of a linkage group. Four small introgressions were found: one of 0.7 cM in G2 (prIL769), one of 6.4 cM in G6 (prIL 412), and two in G8 (prILs 110 and 769) of 3.9 cM each.

We focused on six major genes (*Vr3, DBF2*, *Fc2*, *Jui*, *Rf1* and *Rf2)* and two major QTLs (*qMD4* and *qSD6*) identified in the T1E population (Donoso et al. 2016). Five prILs (3, 27, 31, 333, and 412) were phenotyped as resistant to powdery mildew. These plants were all heterozygous almond/peach in a region of G2 (Figure 4.3). In contrast, the other prILs were susceptible to the disease and homozygous for the peach allele at this region. With the introgressions of the prILs 498 and 769 in G2, the resistance gene (*Vr3*) was located to a region of 10.5 cM between markers AMPA93 and BPPCT004. This genomic region coincides with the location of the gene described by Donoso et al. (2016) between CPDCT044 and BPPCT004 in both TxE and T1E, although the resolution was lower (10.5 vs. 2.7 cM) in the prIL collection. Similar results were obtained for *DBF2*, *Fc2* and *Jui*, where it was also possible to identify a specific genomic region including the gene responsible for the trait (Table 4.3; Table S4.3).

Table 4.2 - Characteristics of the introgressions of the 18 prILs selected in the T1E offspring.

	Linkage group								
	G1	G2	G3	G4	G5	G6	G7	G8	Average
Number of introgressions	7	8	5	15	7	8	6	10	8.3
Average length (cM) per introgression	34.3	35.8	33.1	38.0	23.9	28.3	26.0	24.2	30.4
% introgression[a]	66.9	71.8	77.7	82.9	62.7	65.8	57.3	55.9	67.6

[a] Proportion of the genetic map of each T1E linkage group covered by the average almond introgression.

A special case was that of the fertility restorers *Rf1* and *Rf2*. These two genes interact with each other, where individuals with any of the dominant alleles are fertile and only those homozygous for both recessive alleles are sterile. It was possible also to identify their position on the map: in the subset of male sterile lines, only the expected two fragments in G2 and G6 were homozygous for the peach allele whereas the fertile lines had heterozygous and homozygous individuals (Figure 4.4; Table 4.3). Mapping the two genes was, however, less efficient than mapping one, as only the male sterile individuals were useful for the identification of the map positions; the use of the genotypes of the fertile lines for mapping required previous knowledge on the genetics of the interaction between both loci.

Different results were inferred from the data of the QTLs examined. In the case of maturity date, it was not possible to consistently identify the position of *qMD4*. Looking specifically at the region of G4 where the QTL occurs, we found that all lines with late maturation had the almond genotype in the region of G4, consistent with what we expected, but the group with early maturation had lines with both genotypes (Figure S4.1). When the prILs were ordered by stone weight, we found that the set of lines with stones of 4.1 g or lighter had a fragment of G6 associated with the peach genotype, whereas this fragment was heterozygous for the almond allele for the heavier stones (Figure 4.5). While this is consistent with the position of the *qSD6* QTL identified in T1E by Donoso et al. (2016), a different interpretation is possible where seeds with stones of 6.3 g or lighter could be determined by a QTL at the end of G5 that was not identified by T1E data (Figure 4.5).

Table 4.3 - Mapping of major genes segregating in the T1E offspring using the 18 prIL set and comparison with results obtained with the entire T1E mapping population.

Major gene/QTL	**symbol**	**LG**	**marker interval**	**cM prIL set[a]**	**cM T1E[b]**
Powdery mildew resistance	*Vr3*	G2	AMPA93-BPPCT004	10.5	2.7
Fruit flesh color	*DBF2*	G1	BPPCT016-BPPCT028	11.7	2.0
Juiciness	*Jui*	G1	EPDCU3122-EPPCU3489	39.5	8.3
Flower color	*Fc*	G4	CPDCT011-CPPCT005	8.4	6.0
Fertility restorer 1	*Rf1*	G2	CPPCT044-BPPCT004	11.2	1.1
Fertility restorer 2	*Rf2*	G6	CPP21413-UDP96-001	13.3	5.4
Maturity date	*qMD4*	G4	-[c]	-	6.3[d]
Stone weight	*qSD6*	G6	CPSCT012-CPPCT021	26.5	15.2[d]

[a] Distance of flanking markers in the prIL set.
[b] Distance of flanking markers in the T1E map.
[c] Not determined.
[d] According to the QTL analysis with -1 LOD criterion.

4.3.3 - Phase 3: Generation of lines with a single introgression

Although our initial purpose was to obtain lines with a single introgression from the BC_2 and BC_1S_1 generation, the appearance of the cytoplasmic male sterility described by Donoso et al. (2015) determined that BC_1S_1 individuals could rarely be obtained. As only the plants carrying one or both of the G2 and G6 fragments containing the *Rf1* and *Rf2* fertility restorers from almond were fertile, most of the selected prIL set of plants (11 of 18) were sterile. The sterile plants could not be selfed and, crossed with pollen of

'Earlygold', would always produce sterile progeny. As a result, we discarded the sterile plants of the prIL set and complemented the seven fertile prILs with a selected set of T1E lines with 4-5 introgressions, that were fertile and that together covered the complete almond genome. By backcrossing these lines with 'Earlygold' as female parent, all BC_2 progeny were fertile. From these crosses lines were selected with only one heterozygous fragment of almond and others with two heterozygous fragments and almost full genome coverage (Figure S4.2). Some of these lines started to produce fruit in 2015, and were selfed to extract the BC_2S_1 homozygous and heterozygous lines with a single introgression.

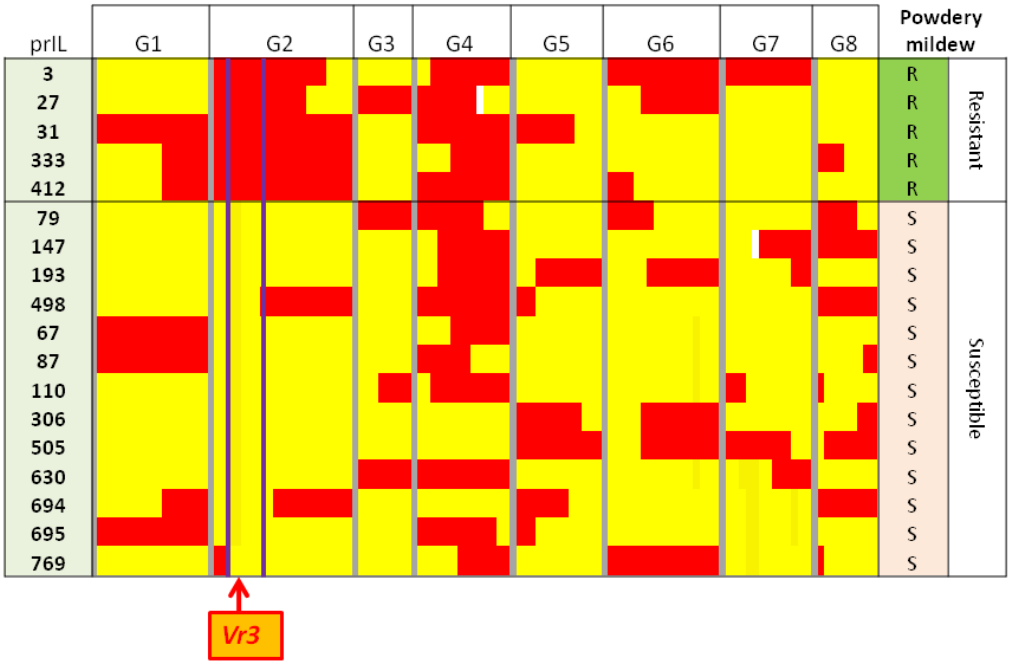


Figure 4.3 - Identification of the map position of the powdery mildew resistance gene (*Vr3*) using the prIL set. The only region that fits a single gene hypothesis is in G2 (between violet bars). The red color corresponds to The almond/peach heterozygote is in red and the peach homozygote in yellow. The box below with a red arrow indicates *Vr3* position according to previous mapping information

Over five pollination seasons (2011-2015), almost 31,000 controlled crosses were done between 'Earlygold' and T1E individuals in order to obtain BC_2 progeny with few almond introgressions in the peach background. These crosses produced 2,264 fruits, giving 529 seedlings after embryo rescue in vitro (Table S4.4). From these seedlings, 28 lines with a single introgression in heterozygosity were extracted. The outcome from the progenies of

individuals with two introgressions selfed in 2015 was much higher: from 1,287 fruits harvested, 528 seedlings were recovered from which 109 individuals with a single introgression were extracted, 28 in homozygosis and 81 in heterozygosis. In total, 109 ILs in heterozygosis (Figure S4.3) and 28 in homozygosis (Figure S4.4) are available, covering 64% and 16% of the almond genome, respectively (Table 4.4). The average size of the introgressed fragments in the ILs in heterozygosis is 15.7 cM (Table S4.5). The smallest introgression for this collection of lines has 0.5 cM and is located on G4. The largest introgression is 43.0 cM long and covers the whole G6. For the collection of lines in homozygosis, the average size of introgressions is 11.3 cM, where the smallest is also located on G4 (2.5 cM) and the largest on G2 (26.8 cM).

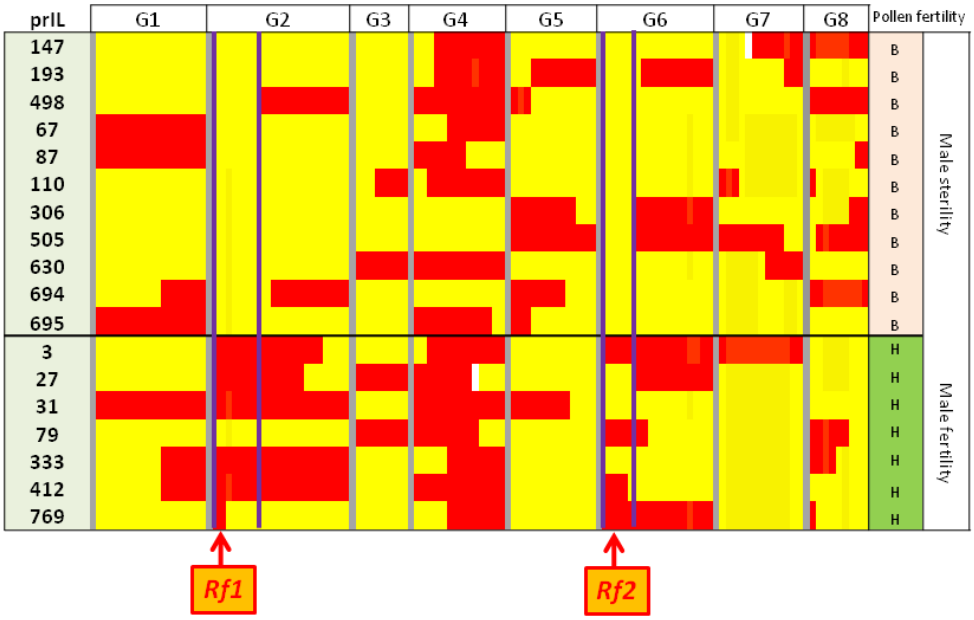


Figure 4.4 - Map positions of the fertility restorer genes as inferred from the prIL set. The almond/peach heterozygote is in red and the peach homozygote in yellow. The violet bars indicate the boundaries of the gene positions according to the prIL set of plants. The arrow indicates the position of the gene in the T1E map

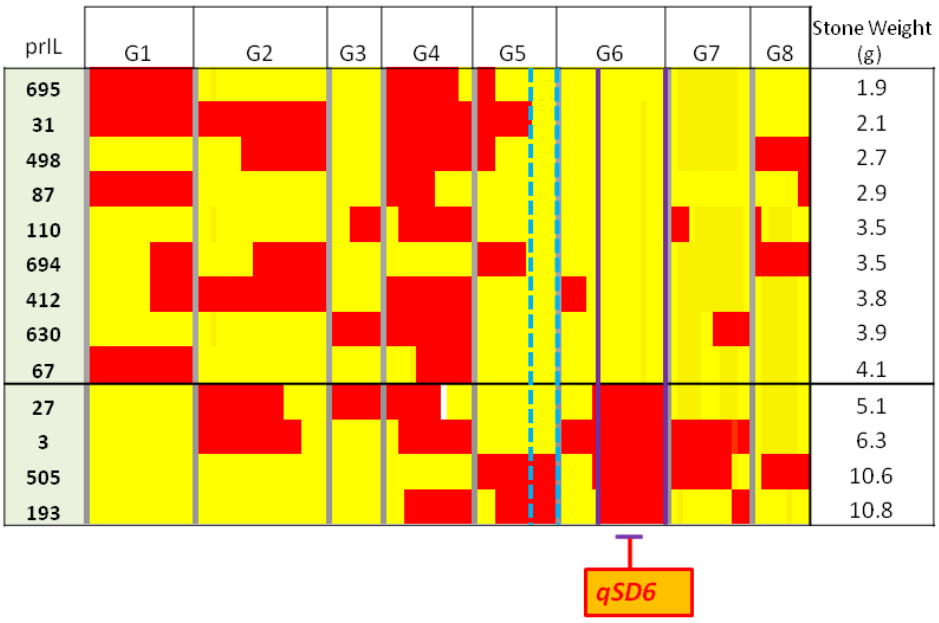


Figure 4.5 - Map position of the stone weight QTL (*qSD6*) as inferred with the prIL set. The almond/peach heterozygotes are in red and the peach homozygote in yellow. The violet bars indicate the position of the QTL according to the prIL set of plants. Another possible position for the QTL is indicated by the blue broken vertical lines. The purple horizontal line indicates the *qSD6* position based on the complete T1E map

Table 4.4 - Distribution of almond fragments per linkage group and their respective coverage in the collection of 137 ILs with one almond introgression in homo- and heterozygosis.

		Introgression lines in heterozygosis			**Introgression lines in homozygosis**		
Linkage group	**cM Total**	**nº fragments**	**cM covered**	**coverage (%)**	**nº fragments**	**cM covered**	**coverage (%)**
G1	51.2	7	39.7	78%	0	0.0	0%
G2	49.9	17	26.8	54%	4	26.8	54%
G3	42.6	12	22.5	53%	2	11.3	27%
G4	45.8	13	45.8	100%	7	2.5	5%
G5	38.1	2	27.5	72%	0	0.0	0%
G6	43.0	48	43.0	100%	12	6.1	14%
G7	45.3	7	10.0	22%	3	10.0	22%
G8	43.4	3	14.5	33%	0	0.0	0%
Total	**359.3**	**109**	**229.8**	**64%**	**28**	**56.6**	**16%**

4.4 - Discussion

Donoso et al. (2016) discovered genes and QTLs for fruit quality and disease resistance from almond that could enrich the peach genome. In this paper, we evaluated a marker-based breeding strategy for introgressing these genes into the peach commercial gene pool in a relatively short period of time. A side result of this was the selection of a collection of ILs that could be a useful resource for genetic analysis in the *Prunus* genus. In the following paragraphs we discuss the results obtained and their consequences on various aspects of the process of gene introgression.

4.4.1 - Selection of plants with a low number of introgressions

Nine individuals with three or less introgressions were selected, representing 1.02% of the 882 offspring from backcross one. Assuming that a single crossover is produced in every pair of homologous chromosomes during meiosis in the almond x peach hybrid individual, the proportion of individuals with one, two or three introgressions is 2.73%, giving an expected 24 plants from the original 882, significantly higher than the nine plants (χ^2=9.7; P<0.01) extracted. One possible explanation is that more than one recombination per chromosome per meiosis occurred, although in this case the scenario of one recombination per chromosome seems realistic based on the mapping data of the T1E population (Donoso et al. 2015). Another explanation could be that individuals with homozygous fragments of the peach genome would produce less seed or germinate less frequently than others, or result in weak or deleterious phenotypes, causing them to die or be unsuitable for further analysis. This would be more frequent in the individuals that we sought, with a large fraction of their genome homozygous for the peach alleles. Examples are the three plants known to have three introgressions that we discarded because of their low vigor, or the fact that certain genomic regions appeared much more frequently in heterozygous state, particularly the central and proximal parts of G4 and G8, respectively (Table 4.1).

The six prIL survivors with three or less introgressions allowed full coverage of the peach genome in six out the eight linkage groups of the genus (G1, G2, G3, G4, G7, and G8)

and 66% in the remaining two groups (G5 and G6), together covering a 92% of the almond genome. MAI could have proceeded from only these six lines because the almond genome coverage was almost complete, but we preferred to add 12 additional T1E plants with four introgressions each (Figure 4.2) for full representation of the almond genome, and to facilitate phenotypic characterization and genetic analysis at the prIL stage.

The three-step selection plan with SSR markers proved to be an efficient strategy. We obtained 13,451 data points (including those from the 213 plants dismissed as selfing or open pollinated), 10,568 in the first two steps and 2,883 in the third step. If we had chosen to run 16 markers in the first step, we would have generated a total of 17,520 datapoints, 60.3% more, leading to considerable savings of time and effort without loss of data quality. Full genotyping of the prILs improved the precision on the definition of the boundaries of the introgressed fragments of each line. The average size of the introgressions was high, 42.2 cM for introgression, equivalent to 12% of the total genetic map distance of T1E. In a third of the cases the introgression included the entire chromosome. This was not surprising as the selection was in the first backcross generation, representing a single meiosis: smaller introgressions would be expected if it had occurred in later generations. The low recombination rate of the almond × peach hybrid compared to that of peach (Donoso et al. 2015), also observed in other interspecific crosses in *Prunus* (Arús et al. 2005) and other species (Gebhardt et al. 1991; Tanksley et al. 1992), is an additional reason for the large size of the introgression fragments. The advantage of an early selection is that small introgressed fragments are likely to be detected by the set of markers used, improving the quality of the final IL set. Moreover, the offspring of heterozygous ILs with large introgressions or covering a whole chromosome may produce a series of sub-ILs suitable for more detailed genetic analysis and identification of genes in small chromosomal fragments (Koumproglou et al. 2002). Nevertheless, it has the disadvantage that the phenotype of each line is affected by many almond genes simultaneously, which in some cases may complicate genetic analysis.

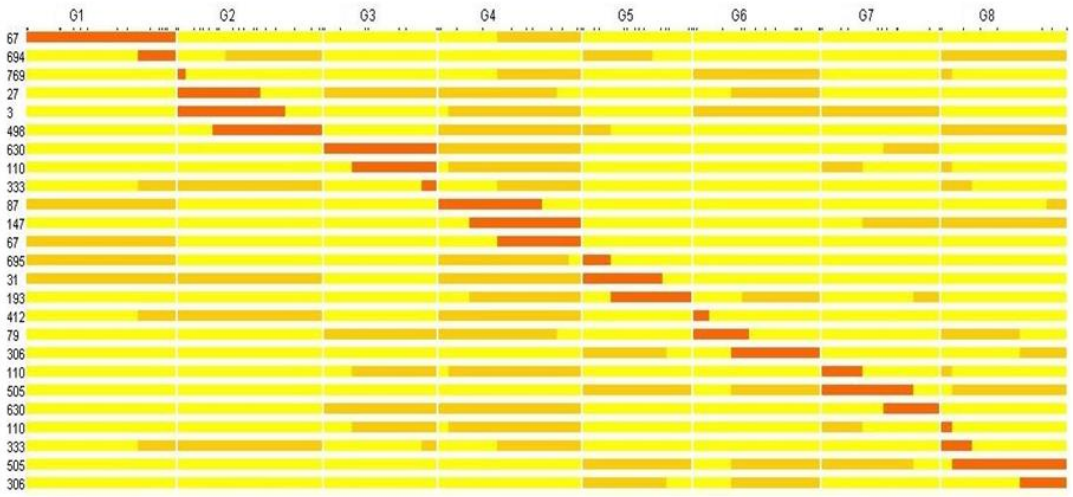


Figure 4.2 - Graphical genotypes of 18 pre-introgression lines (prILs), i.e. backcross one lines that cover the whole peach genome with 2-4 almond introgressions each. Horizontal bars represent the prIL genotypes identified by their number on the left column for the eight linkage groups (G1 to G8) of *Prunus*. The chromosomes are segmented by each SSR used in this study and the scale is in cM. Each chromosomal fragment has a background color, where yellow corresponds to genotypes with the peach allele(s), and orange or red to the heterozygous peach/almond genotypes, indicating the presence of the almond introgression. The difference between red and orange is to highlight (red fragments) the almond introgressions that allowed covering the complete genome (those in the main diagonal). Some of the prILs (10) are repeated to show this feature.

4.4.2 - Mapping major genes and QTLs with the prIL set

This study demonstrates that the prIL collection can be used for mapping traits determined by major genes and for identifying molecular markers flanking them. By analysis of Mendelian trait loci using the phenotype and the graphical genotype of the prILs, a set of six genes was located at the same genome positions as in the full T1E map although, as expected, with lower precision. The proof of concept showed that this approach can be used in other prIL collections without previous knowledge of the genetics of the characters under study. PrILs carrying the character of interest may be used for further selection of generations of introgression allowing marker-assisted selection of the fragment of interest given that its genome position is known, including that of markers nearby.

When we performed this analysis with major QTLs the result was less clear than with major genes. We were able to identify the position of the seed weight QTL, but a spurious additional position was identified. This may be due to errors in QTL identification

occurring more easily because, unlike the qualitative classification of major genes, quantitative characters have a continuous distribution, and the separation between two classes is not evident. For the maturity date QTL, we were not able to identify a clear map position. A reason may be that phenotypes for this character could be masked by the segregation in the 'Earlygold' background. Indeed, a significant QTL explaining 19% of the phenotypic variance was identified on the 'Earlygold' map at this region (Donoso et al. 2016), complicating the interpretation of the almond allele effects.

In all, these results indicate that many of the genes with major effects, generated by the variability introduced in peach by the almond genome, can be identified and mapped in a prIL collection of only 18 plants. Excluded from this analysis are all major genes where the allele of the donor parent is recessive, and the characters which are determined by various QTLs or by one that has minor effects. This analysis, although partial, provides a genome-wide survey of the genes of interest carried by exotic lines or wild species in the BC_1 generation, and requires maintaining only 15-30 plants per species, i.e., with little time and field space. Moreover, the genetic evaluation in this phase of the process does not interfere or slow MAI, as crosses to advance to the BC_2 or BC_1S_1 could be done at the same time or before the phenotyping for the traits.

4.4.3 - Towards a complete peach-almond IL collection

As an additional result of the MAI project, we are developing a collection of introgression lines of almond chromosome fragments in the peach background. Given the clonally reproducing nature of peach, it will be possible to develop two independent collections: one with the introgressed fragments in homozygosis and the other in heterozygosis. The latter will be useful for estimating the effects of a single dosage of specific genes, and also to provide a permanent source of heterozygous individuals to obtain sub-ILs with smaller DNA introgressions when needed for the characterization of specific genes or other uses. In this paper we present a first set of lines, where 64% of the genome is covered in heterozygosis and 14% in homozygosis. We expect that, in one or two years, we will have sets with >95% of the almond genome covered. This is because most lines of the BC_2 collection in the peach cytoplasm and only two introgressions (see Figure S4.2) will start to produce a substantial selfed progeny in 2016 and the following years, meaning

more seedlings and easier to obtain than when working with offspring from BC1 individuals. This collection is likely to become a powerful resource for genetic analysis in peach, particularly for the analysis of characters with complex inheritance, facilitating mendelization and eventual cloning of QTLs of interest.

IL collections developed in different crops have been obtained on the background of an inbred line of the recurrent parent (Eshed and Zamir, 1995, Pestsova et al. 2001, Eduardo et al. 2005, Monforte and Tanksley, 2000, Fletcher et al. 2013, Urrutia et al. 2015). In our case the individual we chose as recurrent parent, 'Earlygold', is a peach commercial cultivar being partially heterozygous. Although peach is a species with a low level of variability (Aranzana et al. 2003; Li et al. 2013) and, as shown by Donoso et al. (2015), approximately half of the 'Earlygold' physical distance consists of large DNA fragments identical by descent, there is an element of distortion in the genetic analysis of an IL collection because the genetic background of peach in each prIL (or later in the IL) will not be completely fixed, due to the segregation of the alleles in 'Earlygold'. Considering that 50% of the genome of 'Earlygold' is identical by descent and that the probability of any map region heterozygous in 'Earlygold' would be also heterozygous in subsequent backcross generations is 50%, the proportion of the genome expected to be identical as 'Earlygold' in any of the ILs is of 75% on average. This problem could have been avoided by using a fully homozygous peach parent, such as the few dihaploid lines that exist in this species (Germanà 2006), or highly homozygous, such as the traditional non-melting Spanish varieties (Aranzana et al. 2003). However, construction of the IL collection would have been considerably delayed, since it would have required the development of the hybrid and then 3-5 additional years in order to begin the process as with the hybrid 'MB 1.37'. The project was begun knowing this limitation, taking into consideration that: a) we expected that many of the important genes that almond may bring into peach should have a clear qualitative effect, and b) the effects of the variation in the 'Earlygold' background could be partially estimated using its map derived from the T1E population, as by Donoso et al. (2016).

4.4.4 - A model for marker-assisted introgression (MAI)

The essential reason why peach breeders have almost never used other *Prunus* species to develop new varieties is the extremely high cost, in time and effort, to integrate the exotic alleles. Here we proposed and validated a strategy (MAI) to make introgressed fragments of almond available for peach breeding in a short period of time, with a marker-based approach to monitor what part of the peach and what of the almond genome are present in each line. Indeed, the usual 5-8 generations needed to extract a collection of ILs (Eshed and Zamir 1994; Monforte and Tanksley 2000; Eduardo et al. 2005) or to obtain a new cultivar with the backcross method (Allard 1961) in herbaceous species, makes the process practically unfeasible in a species of long intergenerational period: peach would require 25-35 years, three years per generation plus five for evaluation at the end of the process. An example is the introduction in apple of the *Vf* gene for resistance to apple scab from *M. floribunda*. The initial interspecific crosses were produced more than 100 years ago, with the first resistant variety, 'Prima', released 55 years later (Gessler and Pertot, 2012). In our case, the first lines with a single introgression in heterozygosis were obtained in 2011, five years after the first crosses were made, so assuming that the interspecific hybrid is available, the whole process could be completed in 9-12 years with MAI. This method also allows a first survey of the genes provided by the donor parent in the prIL set and, in certain cases, their genome positions can be inferred. Plants with the desired phenotypes may then be advanced to the next backcross or selfing generation and selected only for the desired introgression to obtain varieties with commercial value.

Several circumstances may make the process longer, more complex or more laborious. The number of individuals to be analyzed in the BC_1 generation to reach the objective of prILs with ≤ 3 introgressions increases with chromosome number. For species such as apple (x=17) and grape (x=19), the number of BC_1 plants needed for a reasonable prIL set would be much higher, or two backcross generations may be needed. In the case of almond × peach hybrids, the unforeseen circumstance of cytoplasmic male sterility (Donoso et al. 2015) implied the addition of one generation to recover the peach cytoplasm and obtain selfed seed from all prILs. This could be avoided by starting the process with a hybrid individual from the reciprocal cross (peach × almond), or by obtaining individuals with the peach cytoplasm as soon as possible, ideally in fertile plants of the BC1 generation as here, although at the cost of delaying the extraction of

homozygous ILs to the BC_2S_1 generation. Self-incompatibility, as in the case of apple or stone fruit other than peach, does not preclude the use of MAI, but if the introgressed fragment has to be recovered in homozygosis, only crosses between different plants may generate such lines. One alternative is to choose a self-compatible recurrent parent, which is possible for most stone fruit. Finally, for backcross breeding, the large size of the introgression may result in linkage drag, i.e. the unwanted effects of other genes included in the introgressed fragment. These may be eliminated by reducing the size of these fragments, which requires additional backcross steps that can be monitored with markers developed within the introgressed fragment (Tanksley et al 1989). Saturating an already defined introgression fragment with additional markers is currently an easy task in many fruit species using the data from SNP chips already developed or searching for other variants with the information of the whole genome sequences.

When considering its use in providing novel variability for peach, MAI can be expanded to species other than almond. Interspecific hybrids are available between peach and other species (*P. davidiana, P. salicina, P. cerasifera, P. kansuensis, P. mira* and others) that could be backcrossed with peach, generate offspring for marker selection, and individuals with low numbers of introgressions identified with markers. The discovery of genes in Phase 2 of our model provides the information to allow only individuals carrying major genes of interest for a specific breeding program to be advanced to the next generation. The number of plants in the prIL set is low (15-30 individuals), so that the entire genome of the exotic gamete that contains the F_1 hybrid can be maintained with a small population, requiring relatively minor effort and making it possible to survey several donor species or different gametes from the same donor species with a moderate investment. Moreover, prILs are one generation away from lines that contain all the genome of elite commercial cultivars with a single fragment from the donor species, meaning that they can realistically be integrated in current breeding programs. In this way, a wave of materials with foreign DNA fragments, but in the background of the cultivated species, will be available for breeding and become a new source of variability that may be essential to incorporate needed genes for pest and disease resistance, climate change adaptation and fruit quality in peach over the next decades.

Data archiving statement

Genotypes and phenotypes used in this paper are provided in Figures 4.3, 4.4 and 4.5, and Tables S4.1 and S4.3. Additional information on the TxE and T1E maps can be found at the Genome Database for Rosaceae (http://www.rosaceae.org/)

Acknowledgments

We acknowledge financial support from the Spanish Ministry of Economy and Competitiveness, through the "Severo Ochoa Programme for Centres of Excellence in R&D" 2016-2019 (SEV-2015-0533)" and project AGL2012-40228.

4.5 - Supplementary Material

Table S4.1 - The two sets of SSR markers used for the selection of the T1E lines with low numbers of introgressions. Sets with the same reference (1, 2a and 2b) were multiplexed.

Linkage group	Set	Marker	Annealing Temp. (°C)	Origin
G1	1	BPPCT028	57	Dirlewanger et al. (2002)
G2	1	CPPCT044	57	Aranzana et al. (2002)
G3	1	BPPCT007	57	Dirlewanger et al. (2002)
G4	1	EPDCU5060	57	Howad et al. (2005)
G5	1	CPPCT040	57	Aranzana et al. (2002)
G6	1	CPPCT021	57	Aranzana et al. (2002)
G7	1	CPSCT004	57	Mnejja et al. (2004)
G8	1	EPPCU3117	57	Howad et al. (2005)
G1	2a	UDP96-018	55	Testolin et al. (2000)
G2	2b	UDA023	50	Messina et al. (2004)
G3	2a	EPDCU0532	55	Howad et al. (2005)
G4	2a	CPPCT051	55	Aranzana et al. (2002)
G5	2a	BPPCT014	55	Dirlewanger et al. (2002)
G6	2b	Ps7a2	50	Joobeur et al. (2000)
G7	2b	Ps5c3	50	Cantini et al. (2001)
G8	2b	CPSCT018	50	Mnejja et al. (2004)

Table S4.2 - Genotypes of the set of selected prILs. The background is highlighted in orange for heterozygous peach/almond genotypes (H), and in yellow for peach homozygous (B) genotypes.

Linkage group	Marker (SSR)	cM[a]	prIL set																	
			3	27	31	67	79	87	110	147	193	306	333	412	498	505	630	694	695	769
	G1																			
G1	EPDCU3122	0,8	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	UDP96-018	1,4	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	CPPCT004	7,9	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	EPPCU5331	12,2	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	PACITA005	12,2	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	EPPCU1090	19,1	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	UDP96-005	19,1	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	CPPCT003	21,2	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	CPPCT026	32,1	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	EPDCU3489	35,3	B	B	H	H	B	H	B	B	B	B	B	B	B	B	B	B	H	B
	BPPCT016	43,6	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	CPPCT019	46,7	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	CPPCT042	47,7	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	EPDCU2862	47,7	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	CPPCT053	48,7	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	CPPCT029	49,2	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	BPPCT028	51,2	B	B	H	H	B	H	B	B	B	B	H	H	B	B	B	H	H	B
	G2																			
G2	CPPCT044	0,0	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	H
	CPP8062	0,0	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	H
	AMPA93	1,4	H	H	-	B	-	B	-	B	B	-	H	-	B	B	-	B	-	B
	MA024a	5,4	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	B
	UDP98-025	6,5	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	B
	CPDCT044	7,5	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	B
	BPPCT004	10,1	H	H	H	B	B	B	B	B	B	B	H	H	B	B	B	B	B	B
	EPDCU4017	12,2	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	B	B	B
	BPPCT001	12,7	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	B	B	B
	CPSCT044	17,5	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	M1a	19,6	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	UDP96-013	21,6	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	CPDCT004	25,2	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	UDP98-411	26,3	H	H	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	pchgms1	28,6	H	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	BPPCT030	32,2	H	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	CPPCT043	32,2	H	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	CPSCT021	38,1	B	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	PceGA34	45,1	B	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	CPSCT034	48,8	B	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	UDA-023	49,9	B	B	H	B	B	B	B	B	B	B	H	H	H	B	B	H	B	B
	G3																			
G3	EPPCU5990	0,0	B	H	B	B	H	B	B	B	B	B	B	B	B	B	H	B	B	B
	EPPCU4610	0,0	B	H	B	B	H	B	B	B	B	B	B	B	B	B	H	B	B	B
	UDP97-403	6,5	B	H	B	B	H	B	B	B	B	B	B	B	B	B	H	B	B	B
	BPPCT007	6,5	B	H	B	B	H	B	B	B	B	B	B	B	B	B	H	B	B	B
	BPPCT039	15,5	B	H	B	B	H	B	H	B	B	B	B	B	B	B	H	B	B	B
	EPDCU3083	16,6	B	H	B	B	H	B	H	B	B	B	B	B	B	B	H	B	B	B
	CPPCT002	28,2	B	H	B	B	H	B	H	B	B	B	B	B	B	B	H	B	B	B
	UDP96-008	30,6	B	H	B	B	H	B	H	B	B	B	B	B	B	B	H	B	B	B
	EPDCU0532	42,6	B	H	B	B	H	B	H	B	B	B	H	B	B	B	H	B	B	B
	G4																			
G4	BPPCT010	0,0	B	H	H	B	H	H	B	B	B	B	B	H	H	B	H	B	H	B
	EPDCU5060	1,1	B	H	H	B	H	H	B	B	B	B	B	H	H	B	H	B	H	B
	pchgms2	7,9	H	H	H	B	H	H	H	B	B	B	B	H	H	B	H	B	H	B
	CPPCT011	14,6	H	H	H	B	H	H	H	H	H	B	B	H	H	B	H	B	H	B
	CPPCT005	14,6	H	H	H	B	H	H	H	H	H	B	B	H	H	B	H	B	H	B
	CPDCT045	24,7	H	H	H	H	H	H	H	H	H	B	H	H	H	B	H	B	H	H
	UDP96-003	29,5	H	H	H	H	H	H	H	H	H	B	H	H	H	B	H	B	H	H
	M12a	31,6	H	H	H	H	H	H	H	H	H	B	H	H	H	B	H	B	H	H
	EPPCU2000	37,0	H	H	H	H	H	B	H	H	H	B	H	H	H	B	H	B	H	H
	BPPCT015	37,0	-	-	H	H	H	-	-	H	-	B	H	-	H	B	-	B	H	-
	UDA-021	40,6	H	B	H	H	B	B	H	H	H	B	H	H	H	B	H	B	H	H
	CPPCT046	40,6	H	B	H	H	B	B	H	H	H	B	H	H	H	B	H	B	H	H
	UDA-027	44,3	H	B	H	H	B	B	H	H	H	B	H	H	H	B	H	B	B	H
	Ps12e2	44,9	H	B	H	H	B	B	H	H	H	B	H	H	H	B	H	B	B	H

[a]The distances of the genetic map correspond to the map described by Donoso et al (2015) in cM. [1]"-" no data; these genotypes occurred in SSRs where the peach alleles were dominant and heterozygous for the 'Earlygold' parent resulting in segregations with two of the four expected classes being indistinguishable

Table S4.2 - (continued)

Linkage group	Marker (SSR)	cM[a]	prIL set																	
			3	27	31	67	79	87	110	147	193	306	333	412	498	505	630	694	695	769
	G5																			
G5	CPPCT040	0,0	B	B	H	B	B	B	B	B	B	H	B	B	H	H	B	H	H	B
	BPPCT026	4,0	B	B	H	B	B	B	B	B	B	H	B	B	H	H	B	H	H	B
	UDP97-401	5,4	B	B	H	B	B	B	B	B	B	H	B	B	H	H	B	H	H	B
	BPPCT017	15,3	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	H	B	B
	CPSCT006	15,3	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	H	B	B
	BPPCT037	17,9	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	H	B	B
	pchgms4	19,5	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	H	B	B
	CPPCT013	21,2	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	H	B	B
	EPDCU5183	26,9	B	B	H	B	B	B	B	B	H	H	B	B	B	H	B	B	B	B
	EPDCU4658	28,4	B	B	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	B
	BPPCT038	29,4	B	B	B	B	B	B	B	B	H	B	B	B	B	H	B	B	B	B
	CPSCT022	36,9	B	B	B	B	B	B	B	B	H	B	B	B	B	H	B	B	B	B
	BPPCT014	38,1	B	B	B	B	B	B	B	B	H	B	B	B	B	H	B	B	B	B
	G6																			
G6	CPP21413	0,0	H	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B	B	H
	Ps7a2	1,1	H	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B	B	H
	CPP21245	1,1	H	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B	B	H
	CPP20836	1,1	H	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B	B	H
	UDP96-001	10,6	H	B	B	B	H	B	B	B	B	B	B	B	B	B	B	B	B	H
	BPPCT008	16,0	H	H	B	B	H	B	B	B	B	H	B	B	B	H	B	B	B	H
	CPSCT012	17,0	H	H	B	B	H	B	B	B	H	H	B	B	B	H	B	B	B	H
	pchcms5	20,7	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	BPPCT025	24,1	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	CPPCT047	27,6	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	UDP98-412	34,4	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	MA040a	34,4	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	AMPA130	34,9	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	MA14a	34,9	-	H	-	-	B	-	-	B	H	-	B	B	B	H	-	B	B	-
	EPPCU4092	37,0	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	CPPCT030	43,0	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	CPPCT021	43,0	H	H	B	B	B	B	B	B	H	H	B	B	B	H	B	B	B	H
	G7																			
G7	CPSCT004	5,0	H	B	B	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B
	CPPCT039	6,7	-	-	-	B	-	B	-	-	-	B	-	-	-	H	B	-	B	-
	pchgms6	11,4	H	B	B	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B
	UDP98-408	12,9	H	B	B	B	B	B	H	B	B	B	B	B	B	H	B	B	B	B
	CPPCT057	18,0	-	-	B	-	B	B	-	-	B	B	-	-	B	-	-	-	-	B
	CPPCT033	24,1	H	B	B	B	B	B	B	H	B	B	B	B	B	H	B	B	B	B
	MA20a	25,8	H	B	B	B	B	B	B	H	B	B	B	B	B	H	B	B	B	B
	PMS02	27,9	H	B	B	B	B	B	B	H	B	B	B	B	B	H	H	B	B	B
	EPPCU5176	30,6	H	B	B	B	B	B	B	H	B	B	B	B	B	H	H	B	B	B
	pchcms2	31,1	H	B	B	B	B	B	B	H	B	B	B	B	B	H	H	B	B	B
	CPPCT017	39,1	-	-	B	-	B	B	-	-	H	B	-	-	B	-	-	-	-	B
	EPDCU3392	41,9	H	B	B	B	B	B	B	H	H	B	B	B	B	B	H	B	B	B
	Ps5c3	45,3	H	B	B	B	B	B	B	H	H	B	B	B	B	B	H	B	B	B
	G8																			
G8	CPSCT018	0,0	B	B	B	B	H	B	H	H	B	B	H	B	H	B	B	H	B	H
	BPPCT006	7,8	B	B	B	B	H	B	B	H	B	B	H	B	H	H	B	H	B	B
	CPPCT058	7,8	B	B	B	B	H	B	B	H	B	B	H	B	H	H	B	H	B	B
	CPDCT034	7,8	-	-	B	B	-	B	-	-	B	B	-	B	H	-	B	-	B	B
	CPPCT006	13,7	B	B	B	B	H	B	B	H	B	B	B	B	H	H	B	H	B	B
	M6a	21,9	-	B	B	-	H	B	-	-	-	-	-	-	H	H	-	-	B	-
	UDP98-409	34,7	B	B	B	B	B	B	B	H	B	H	B	B	H	H	B	H	B	B
	EPDCU3454	38,4	B	B	B	B	B	H	B	H	B	H	B	B	H	H	B	H	B	B
	EPDCU3117	43,3	B	B	B	B	B	H	B	H	B	H	B	B	H	H	B	H	B	B

Table S4.3 - Summary of the number of crosses, fruits harvested, seedlings and introgression lines in heterozygosis (IL HET) and in homozygosis (IL HET) obtained during the period 2011-2015.

Year	crosses	fruits	seedlings	cross pollinations	undesired selfing	selected individuals	IL HET	IL HOM	2I HET	≥3I HET
2011	4.300	184	15	0	0	15	1	-	4	10
2012	8.000	657	145	5	9	145	8	-	31	93
2013	1.900	230	42	7	4	42	3	-	6	22
2014	10.000	696	180	1	9	96	7	-	23	140
2015	6.700	497	147	0	17	10	9	-	28	92
Total BC2	30.900	2.264	529	13	39	308	28	-	92	357
2015 BC2S1	n/a	1.287	528	-	-	528	81	28	381	
Total		3.551	1.057	13	39	836	109	28		

Table S4.4 - Size of the almond fragments in the peach Introgression lines in heterozygosity (IL HET) and homozygosity (IL HOM).

		IL HET		IL HOM
Linkage group	**Smallest fragment**	**Largest fragment**	**Average fragment size**	**Fragment size**
G1	2	32	16.3	-
G2	4	26.8	22.7	26.8
G3	3.25	22.5	13.1	11.25
G4	0.5	35	11.5	2.5
G5	27.5	27.5	27.5	-
G6	4.95	43	10.9	6.05
G7	1.5	10	8.8	10.0
G8	14.5	14.5	14.5	-
Average			15.7	11.3

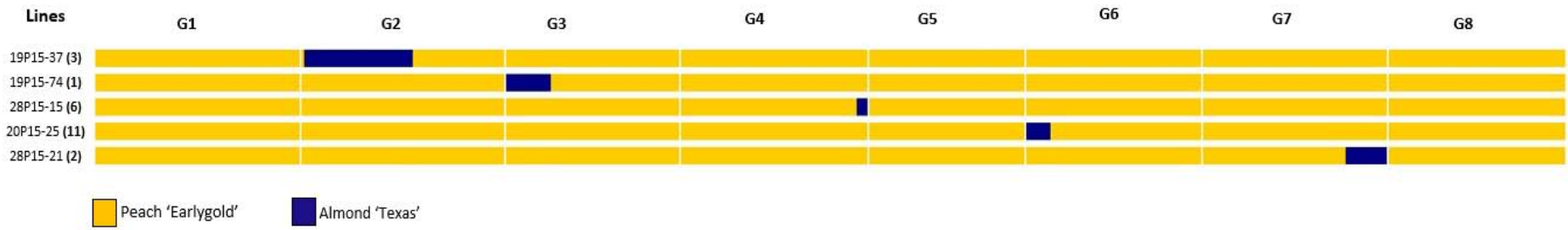
Figure S4.1 - Graphical genotype of the prIL set compared with the data of fruit maturity date. The almond/peach heterozygotes are in red and the peach homozygote in yellow. The approximate map position of the *qMD4* QTL is indicated below. PrILs with late maturity date had the expected heterozygous genotype, but those with earlier dates had homozygous and heterozygous genotypes.

[1]Maturity date is measured in number of Julian days from blooming to commercial maturity.

Figure S4.2 - Graphical genotype of the collection of BC_2 lines from the 'Texas' almond × 'Earlygold' peach hybrid with only two peach introgressions in heterozygosity. Yellow indicates homozygous regions for peach alleles and green heterozygous peach/almond regions. Regions with unknown genotype are in grey. G1-G8 are linkage groups of *Prunus*. All genome is covered with the exception of the distal part of G3.

Figure S4.3 - Collection of 109 introgression lines with one almond introgression in heterozygosis corresponding to 36 different almond fragments. On the left, in parentheses, is the number of lines that share the same almond fragment but differ in their peach genome composition. G1-G8 are linkage groups of *Prunus*. The 'Earlygold' genetic background is shown in yellow, the peach/almond heterozygous fragment in green.

Figure S4.4 - Collection of 28 introgression lines with a single almond introgression in homozygosis corresponding to five different almond genome fragments. On the left, in parentheses, is the number of lines that share the same almond fragment but differ in their peach genome composition. The 'Earlygold' genetic background is shown in yellow, and the almond homozygous fragment in blue.



4.6 - Bibliography

Allard RW (1961) Principles of plant breeding. J Willey and Sons Ltd. London, New York.

Aranzana MJ, Carbó J, Arús P (2003) Microsatellite variability in peach [*Prunus persica* (L) Batsch.]: cultivar identification, marker mutation, pedigree inferences and population structure. Theor Appl Genet 106:1341-1352

Aranzana MJ, Garcia-Mas J, Carbó J, Arús P (2002) Development and variability analysis of microsatellite markers in peach. Plant Breed 121:87-92

Aranzana MJ, Illa E, Howad W, Arús P (2012) A first insight into peach [*Prunus persica* (L.) Batsch] SNP variability. Tree Genet Genomes 8:1359-1369

Arús P, Yamamoto T, Dirlewanger E, Abbott AG (2005) Synteny in the Rosaceae. In: Jules Janick (ed.) Plant Breeding Reviews. Wiley, Hoboken, New Jersey, volume 27, pp 175-211

Asma BM (2008) Determination of pollen viability, germination ratios and morphology of eight apricot genotypes. Afr J Biotechnol 7:4269-4273

Bai Y, Huang CC, van der Hulst R, Meijer-Dekens F, Bonnema G, Lindhout P (2003) QTLs for tomato powdery mildew resistance (Oidium lycopersici) in Lycopersicon parviflorum G1.1601 co-localize with two qualitative powdery mildew resistance genes. Mol Plant Microbe In 16:169-176

Batlle I, Fontich C, Lozano L, Iglesias I, Reig G, Alegre S, Echeverría G, de Herralde F, Claveria E, Dolcet-Sanjuan R, Carbó J, Bonany J, Maillard A, Maillard L (2012) The peach breeding programme IRTA-ASF: aiming for high fruit quality. Acta Hortic 940:75–78

Bouhadida M, Martín JP, Eremin G, Pinochet J, Moreno MA, Gogorcena Y (2007) Chloroplast DNA diversity in *Prunus* and its implications on genetic relationships. J Amer Soc Hort Sci 132:670-679

Bus VGM, Laurens FND,Van deWeg WE, Rusholme RL, Rikkerink EHA, Gardiner SE, Bassett HCM, Plummer KM (2005) The Vh8 locus of a new gene-for-gene interaction between Venturia inaequalis and the wild apple Malus sieversii is closely linked to the Vh2 locus in Malus pumila R12740-7A. New Phytol 166:1035–1049

Byrne DH (1990) Isozyme variability in four diploid stone fruits compared with other woody perennial plants. J Hered 81:68-71

Byrne DH, Raseira MB, Bassi D, Piagnani MC, Gasic K, Reighard GL, Moreno MA, Pérez S (2012) Peach. In: Badenes ML, Byrne DH (eds) Fruit Breeding (Handbook of Plant Breeding). Springer, New York, pp 513-569

Cantini C, Iezzoni AF, Lamboy WF, Bortizki M, Struss D (2001) DNA fingerprinting of tetraploid cherry germplasm using simple sequence repeats. J Amer Soc Hort Sci 126:205-209

Dirlewanger E, Cosson P, Tavaud M, Aranzana MJ, Poizat C, Zanetto A, Arús P, Laigret F (2002) Development of microsatellite markers in peach (*Prunus persica* (L.) Batsch) and their use in genetic diversity analysis in peach and sweet cherry (*Prunus avium* L.). Theor Appl Genet 105:127-138.

Dirlewanger E, Graziano E, Joobeur T, Garriga-Calderé T, Cosson P, Howad W, Arús P (2004) Comparative mapping and marker assisted selection in Rosaceae fruit crops. Proc Natl Acad Sci USA 101:9891–9896

Donoso JM, Picañol R, Eduardo I, BatlIe I, Howad W, Aranzana MJ, Arús P (2015) High-density mapping suggests a cytoplasmic male sterility system with two restorer factors in almond x peach progenies. Hortic Res 2:15016

Donoso JM, Picañol R, Serra O, Howad W, Alegre S, Arús P, Eduardo I (2016) Exploring almond genetic variability useful for peach improvement: Mapping major genes and QTLs in two inter-specific almond × peach populations. Mol Breeding 36:16

Doyle JJ, Doyle JL (1990) Isolation of plant DNA from fresh tissue. Focus 12:13–15

Eduardo I, Arús P, Monforte AJ (2005) Development of a genomic library of near isogenic lines (NILs) in melon (Cucumis melo L.) from the exotic accession PI161375. Theor Appl Genet 112:139-148

Eshed Y, Zamir D (1994) Introgressions from Lycopersicon pennellii can improve the soluble-solids yield of tomato hybrids. Theor Appl Genet 88:891-897

Eshed Y, Zamir D (1995). An introgression-line population of Lycopersicon pennelli in the cultivated tomato enables the identification and fine mapping of yield-associated QTLs. Genetics. 141:1147–1162

Esmenjaud D, Voisin R, Van Ghelder C, Bosselut N, Lafargue B, Di Vito M, Dirlewanger E, Poessel JL, Kleinhentz M (2009) Genetic dissection of resistance to root-knot nematodes Meloidogyne spp. in plum, peach, almond, and apricot from various segregating interspecific *Prunus* progenies. Tree Genet Genomes 5:279-289

Faust M, Timon B (1995) Origen and dissemination of peach. Hortic Rev 17:331–379

Fletcher R, Mullen JL, Yoder S, Bauerle WL, Reuning G, Sen S, Meyer E, Juenger T, McKay JK (2013) Development of a next-generation NIL library in *Arabidopsis thaliana* for dissecting complex traits. BMC Genomics:14:655–668

Foolad MR, Arulsekar S, Becerra V, Bliss FA (1995) A genetic map of *Prunus* based on an intraspecific cross between peach and almond. Theor Appl Genet 91:262-269

Foulongne M, Pascal T, Pfeiffer F, Kervella J (2002) Introgression of a polygenic resistance to powdery mildew from a wild species *Prunus* davidiana into peach [*Prunus persica* (L.) Batsch], a case study of marker assisted selection in fruit tree. Acta Hortic 592:259–265

Frary A, Doganlar S, Frampton A, Fulton TM, Uhlig J, Yates HE, Tanksley SD (2003) Fine Mapping of quantitative trait loci for improved fruit characteristics from Lycopersicon chmielewskii chromosome 1. Genome 46:235–243

Fulton TM, Grandillo S, Beck-Bunn T, Fridman E, Frampton A, Lopez J, Petriard V, Uhlig J, Zamir D, Tanksley SD (2000) Advanced backcross QTL analysis of a Lycopersicon esculentum × Lycopersicon parviflorum cross. Theor Appl Genet 100:1025–1042

Gebhardt C, Ritter E, Barone A, Debener T, Walkemeier B, Schachtschabel U, Kaufmann H, Thompson RD, Bonierbale MW, Ganal MW, Tanksley SD, Salamini F (1991) RFLP maps of potato and their alignment with the homoeologous tomato genome. Theor Appl Genet 83:49–57

Germanà MA (2006) Double haploid production in fruit trees. Plant Cell Tiss Org 86:131-146

Gessler C, Pertot I (2012) Vf scab resistance in Malus. Trees 26:95-108

Gómez-Aparisi J, Carrera M, Felipe AJ, Socias i Company R (2001) 'Garnem', 'Monegro' and 'Felinem': new almond × peach hybrid rootstocks, nematode resistant and red leaved for stone fruits. ITEA-Inf Tec Econ Ag 97:282–288

Grandillo S, Tanksley SD, Zamir D (2007) Exploitation of natural biodiversity through genomics. In: RK Varshney and R Tuberosa (eds) Genomics Assisted Crop Improvement: Vol I Genomics Approaches and Platforms. Springer, The Netherlands, pp 121-150

Griffiths P, Scott J (2001) Inheritance and linkage of tomato mottle virus resistance genes derived from Lycopersicon chilense accession LA 1932. J Amer Soc Hort Sci 126:462–467

Howad W, Yamamoto T, Dirlewanger E, Testolin R, Cosson P, Cipriani G, Monforte AJ, Georgi L, Abbott AG, Arús P (2005) Mapping with a few plants:using selective mapping for microsatellite saturation of the *Prunus* reference map. Genetics 171:1305-1309

Joobeur T, Periam N, de Vicente MC, King GJ, Arús P (2000) Development of a second generation linkage map for almond using RAPD and SSR markers. Genome 43:649-655

Joobeur T, Viruel MA, de Vicente MC, Jáuregui B, Ballester J, Dettori MT, Verde I, Truco MJ, Messeguer R, Batlle I, Quarta R, Dirlewanger E, Arús P (1998) Construction of a saturated linkage map in *Prunus* using an almond x peach F2 progeny. Theor Appl Genet 97:1034-1041

Koumproglou R, Wilkes TM, Townson P, Wang XY, Beynon J, Pooni HS, Newbury HJ, Kearsey MJ (2002) STAIRS: a new genetic resource for functional genomic studies of Arabidopsis. Plant J 31:355-364

Li X, Meng X, Jia H, Yu M, Ma R, Wang L, Cao K, Shen Z, Niu L, Tian J, Chen M, Xie M, Arús P, Gao Z, Aranzana MJ (2013) Peach genetic resources: diversity, population structure and linkage disequilibrium. BMC Genet 14:84

Messina R, Lain O, Marrazzo MT, Cipriani G, Testolin R (2004) New set of microsatellite loci isolated in apricot. Molecular Ecology Notes 4:432-434

Micheletti D, Dettori MT, Micali S, Aramini V, Pacheco I, da Silva C, Foschi S, Banchi E, Barreneche T, Quilot-Turion B, Lambert P, Pascal T, Iglesias I, Carbó J, Wang LR, Ma RJ, Li XW, Gao ZS, Nazzicari N, Troggio M, Bassi B, Rossini L, Verde I, Laurens F, Arús P, Aranzana MJ (2015) Whole-genome analysis of diversity and SNP-major gene association in Peach Germplasm. PLoS ONE 10(9):e0136803

Mnejja M, Garcia-Mas J, Audergon JM, Arús P (2010) *Prunus* microsatellite marker transferability across rosaceous crops. Tree Genet Genomes 6:689-700

Mnejja M, Garcia-Mas J, Howad W, Badenes ML, Arús P (2004) Simple-sequence repeat (SSR) markers of Japanese plum (Prunus salicina Lindl.) are highly polymorphic and transferable to peach and almond. Molec Ecol Notes 4:163-166

Monforte AJ, Tanksley SD (2000) Development of a set of near isogenic and backcross recombinant inbred lines containing most of the Lycopersicon hirsutum genome in a L. esculentum genetic background: a tool for gene mapping and gene discovery. Genome 43:803-813

Pestsova EG, Borner A, Roder MS (2001) Development of a set of Triticum aestivum–Aegilops tauschii introgression lines. Hereditas 135(2–3):139–143

Tanksley SD, Ganal MW, Prince JP, de Vicente MC, Bonierbale MW, Broun P, Fulton TM, Giovannoni JJ, Grandillo S, Martin GB, Messeguer R, Miller JC, Miller L, Paterson AHD, Pineda O, Röder MS, Wing RA, Wu W, Young ND (1992) High density molecular linkage maps of the tomato and potato genomes. Genetics 132:1141-1160

Tanksley SD, McCouch SR (1997) Seed banks and molecular maps:Unlocking genetic potential from the wild. Science 277:1063-1066

Tanksley SD, Medina-Filho H, Rick CM (1981) Effect of isozyme selection on metric characters in an interspecific backcross of tomato:basis for an early screening procedure. Theor Appl Genet 60:291-296

Tanksley SD, Nelson JC (1996) Advanced backcross QTL analysis:a method for the simultaneous discovery and transfer of valuable QTLs from unadapted germplasm into elite breeding lines. Theor Appl Genet 92:191–203

Tanksley SD, Young ND, Paterson AH, Bonierbale MW (1989) RFLP mapping in plant breeding:new tools for an old science. Nat Biotechnol 7:257-264

Testolin R, Marrazzo T, Cipriani G, Quarta R, Verde I, Dettori MT, Pancaldi M, Sansavini S (2000) Microsatellite DNA in peach [*Prunus persica* (L.) Batsch] and its use in fingerprinting and testing the genetic origin of cultivars. Genome 43:512-520

Urrutia M, Bonet J, Arús P, Monfort A (2015) A near-isogenic line (NIL) collection in diploid strawberry and its use in the genetic analysis of morphologic, phenological and nutritional characters. Theor Appl Genet 128:1261–1275

Verlaan MG, Hutton SF, Ibrahem RM, Kormelink R, Visser RGF, Scott JW, Edwards JD, Bai Y (2013) The tomato yellow leaf curl virus resistance genes Ty-1 and Ty-3 are allelic and code for DFDGD-class RNA–dependent RNA polymerases. PLoS Genet 9(3):e1003399

Zamir D, Ekstein Michelson I, Zakay Y, Navot N, Zeidan M, et al. (1994) Mapping and introgression of a tomato yellow leaf curl virus tolerance gene, Ty-1. Theor Appl Genet 88:141–146

5. Whole genome sequencing uncovers the recombinational landscape of intraspecific (peach) and interspecific (almond x peach) meioses

5.1 - Introduction

Meiosis is a key cellular mechanism that determines chromosomal reshuffling via recombination and ensures the proper formation of gametes in the majority of eukaryotic species. During meiosis, homologous chromosomes exchange genetic material, as a consequence of repairing double strand breaks (DSBs). Homologous recombination can produce crossovers (COs) or noncrossovers (NCOs), depending on the way that DSBs are repaired (Mercier et al. 2015). The reciprocal exchange of large amounts of DNA between homologous chromosomes at a DSB site produces a CO, while the repair of DSB by copying a small sequence tract either from intact homologous chromosome or from the sister chromatid produces a NCO.

The number of COs per chromosome is known to be highly regulated and usually dependent on chromosome size for a given species (Mercier et al. 2015). The occurrence of at least one CO is necessary for the formation of physical connections between homologous chromosomes, known as chiasmata, which ensure proper orientation and segregation of chromosomes. When one CO takes place, the next one is generally further apart than expected by chance due to a phenomenon called CO interference. For the great majority of eukaryotic species, the number of COs per chromosome ranges from one to three (76% for species studied by Mercier et al. 2015), which is much lower than the initial number of DSBs generated at the beginning of meiosis (Terasawa et al. 1995; Franklin et al. 1999; Chelysheva et al. 2005; Chelysheva et al. 2007; Zhang et al. 2015), thus suggesting that the majority of DSBs are probably repaired as NCOs. The detection of NCOs is a challenging task since it can only be performed if the DSB was repaired using the homologous chromosome (not the sister chromatid) and at least one DNA polymorphism, commonly a SNP (single nucleotide polymorphism) exists in the converted fragment. In addition, the ratio of DSBs that are repaired using the sister chromatid or the homologous chromosome is unknown, thus making it even harder at the time of estimating the number of NCOs per meiosis. Some studies investigated the occurrence of NCOs in plants, mainly in maize (Dooner 2002; Dooner and He, 2014), Arabidopsis (Lu et al. 2012; Yang et al. 2012; Drouaud et al. 2013; Wijnker et al. 2013,

Qi et al. 2014), rapeseed and chickpea (Bayer et al. 2015), and despite that their occurrence has been proven, the amount of NCOs detected in the great majority of these works is low (typically one NCO per chromosome or less).

The distribution of both COs and NCOs is not uniform for almost all species. At the chromosome level, the distribution of COs in plants tends to occur closer to the telomeres and far from centromeric regions in wheat (*Triticum aestivum*), maize (*Zea mays*) or barley (*Hordeum vulgare*), but other distributions are possible for different species (Mézard et al. 2007). At the physical scale, both COs and NCOs are known to cluster in small regions, typically 1-10 kb in size, called hotspots of recombination, in contrast to the adjacent regions where no recombinations occur (Choi and Henderson, 2015). The non-random distribution of recombination events, which reflects the non-random distribution of DSBs, is thought to be caused by genetic and epigenetic marks present in some chromosomal regions (Choi and Henderson, 2015; Melemed-Bessudo et al. 2016). DNA motifs have been associated with higher levels of recombination. In humans, one single DNA motif (CCN)is responsible for recruiting PRDM9, a SET domain protein with a zinc-finger motif that controls over 40% of all recombination hotspots in the human genome (Myers et al. 2008). PRDM9 is not present in plants (Mercier et al. 2015) although the majority of SET domain proteins are expressed during meiosis in Arabidopsis (Zhang and Ma, 2012). In plants, some DNA motifs as well as some epigenetic marks have been associated with higher CO occurrence (Choi and Henderson, 2015; Shilo et al. 2015). Recent studies in *A. thaliana* have shown that CTT, poly-A (Wijnker et al. 2013; Choi et al. 2013) and CCN (Shilo et al. 2015) DNA motifs are significantly enriched at CO locations, most likely playing a similar role to the human CCN motif in recruiting recombination machinery (Shilo et al. 2015). Some of these studies also associated epigenetic marks H3K4me3 and H2A.Z with recombination in *A. thaliana* (Choi et al. 2013; Shilo et al. 2015). Overall, despite the fact that the majority of proteins involved in meiotic recombination are identified, there are still many obscure aspects on the genetic basis of recombination landscape, CO interference and in general in how the overall recombination process proceeds at the molecular level (Mercier et al. 2015).

Peach [*Prunus persica* (L.) Batsch] is a diploid species with 2n=*2x=16*closely related with other cultivated *Prunus*, such as apricot (*P. armeniaca* L.), plum (*P. salicina* Lindl, *P. domestica* L.), cherry [*P. avium*L.(L.)] and almond (*P. dulcis* D.A. Webb). Peach has a low level of variability (Mnejja et al. 2010; Aranzana et al. 2012), a fact that is mainly explained by the absence of a functional gametophytic self-incompatibility system, which is active in most of the other *Prunus*. The self-pollinating behavior of peach, together with its history after domestication, contributed to the eroded gene pool available for the occidental breeding programs (Aranzana et al. 2010). Almond is the most variable cultivated *Prunus* (Byrne 1990; Mnejja et al. 2010) with approximately 7× more variability than peach (Velasco et al. 2016) and the production of fertile interspecific hybrids between both species is often possible. The richness of the almond genome may be very useful for peach, mainly to contribute with new alleles for disease resistance, extended shelf-life and organoleptic traits, as recently evidenced by Donoso et al. (2016). However, introgression of new almond genes into elite peach cultivars can be a lifetime challenging task. New methodologies for interspecific breeding in long intergeneration species are arising (Serra et al. 2016), but a better understanding of the driving forces that control CO frequency and location are needed, in order to facilitate the breakage of undesired linkages and help the creation of new allelic combinations.

In the last decade, the massive use of next generation sequencing allowed the development of reference genomes for a wide variety of species and many major crop species genomes have become available. These new genomic tools have a great value, as the access to the genome sequence at a base pair resolution for one entire segregating population, opens a new world of possibilities for the development of new markers, genetic and genomic studies, genetic mapping, gene discovery and breeding. To date, only a few studies have explored some of these possibilities using whole genome sequencing data for segregating plant populations, such as for Arabidopsis (Yang et al. 2012; Shilo et al. 2015), rice (Huang et al. 2009),radish (Mun et al. 2015),rapeseed and chickpea (Bayer et al. 2015) and also in peach (Wang et al. 2016; Cao et al. 2016).Most of these works have used segregating populations of low genetic complexity, where only two different alleles exist and three possible genotypes, at most, are present in the progeny.

Here we present the whole genome sequencing and *in silico* genotyping of T1E, a large backcross one interspecific population between almond and peach. To overcome the genetic complexity of T1E, with both parents being heterozygous and four possible genotypes expected in the progeny, we designed a genotyping pipeline suited for this kind of population. The detection of a high number of variants allowed us to precise and confidently map CO and NCO events. This population allowed studying the recombination at both intra (from the recurrent peach parent) and interspecific (from the almond x peach hybrid plant) levels, and to compare their rates and distribution along the *Prunus* genome. This information is crucial to understand the effects of genetic distance on recombination and reproductive isolation between closely related species and to design appropriate strategies for fast and efficient introgression of almond genes into the peach genome, which is one of our main goals.

5.2 - Materials and Methods

5.2.1 - Plant material and sequencing

Individuals from the interspecific population T1E obtained from the cross between an F1 hybrid plant between almond 'Texas' and peach 'Earlygold' (named MB1.37), and 'Earlygold' as the recurrent parent, were used in this work. A SNP high density map was previously constructed (Donoso et al.2015) and the inheritance of a diverse set of agronomic traits was studied in this population (Donoso et al. 2016). In total, 128 T1E individuals from the total of N=190 available were resequenced as well as their ancestors 'Earlygold', 'Texas' and MB1.37. DNA was extracted from young leaves using DNeasy® Plant Mini Kit (Qiagen) following manufacturer instructions.

Whole genome sequencing was performed using the HiSeq2000 sequencer (Illumina Inc.) The standard Illumina protocol with minor modifications was followed for the creation of short-insert paired-end libraries. In brief, 2.0 micrograms of genomic DNA were sheared on a Covaris™ E220, size selected and concentrated using AMPure XP beads (Agencourt, Beckman Coulter) in order to reach the fragment size of 220-480bp. The

fragmented DNA was end-repaired, adenylated and ligated to Illumina specific paired-end adaptors. Each library was sequenced using HiSeq SBS Kit v3, in paired end mode, 2×100bp, according to standard Illumina operation procedures. Primary data analysis was carried out with the standard Illumina pipeline.

5.2.2 - Raw sequence processing, read mapping and SNP calling

All raw sequence data underwent the same processing protocol. Using AdapterRemoval v. 1.5.2 (Lindgreen, 2012), adaptor sequences were removed and only those reads with a minimum size of 35bp and a mean quality of 25 were kept. Orphan reads were identified at this step and filtered out. Clean reads were then mapped against the peach reference genome (The International Peach Genome Initiative, 2013) Version 2.0[2] using BWA v. 0.7.5 (Li and Durbin, 2009) with default parameters. The obtained SAM file was converted to BAM using SAMTools v. 0.1.19 (Li et al. 2009)and reads mapping to more than one position or reads that originated from PCR duplication events were excluded from the alignment. SNPs were called using SAMTools mpileup, adjusting mapping quality for reads with excessive mismatches (-C 50). SNPs detected with SAMTools close to indels were re-verified with GenomeAnalysisTK (GATK) v3.1.1 (van der Auwera et al., 2013), performing a local *de novo* assembly. SNPs were filtered depending on their purpose in downstream analysis. The details for obtaining reference SNPs for genotyping the progeny are presented in the next section. In the case of SNPs from individuals of the population, a minimum coverage of 8×, a genotype quality of 90 and global quality of 90 was needed to consider a valid SNP. Annotation and functional analysis of SNPs were performed using SnpEff (Cingolani et al. 2012) with default parameters. A local database for version 2.0 of peach genome was built following the documentation available at SnpEff webpage (http://snpeff.sourceforge.net/SnpEff_manual.html#databases).

5.2.3 - Creation of SNP core files and establishment of regions identical by descent (IBD) in 'Earlygold'

Parents for T1E population are both partly heterozygous. The hybrid MB1.37 has one allele coming from almond ('T' allele) and one allele coming from peach ('E1' allele). On the other hand, 'Earlygold' has the same peach allele present in the hybrid ('E1' allele)

[2]https://www.rosaceae.org/species/prunus_persica/genome_v2.0.a1

and another allele different from E1 (the ‘E2’ allele). Possible genotypes for the T1E progeny are: E1T, E2T, E1E2 and E1E1. This is a more complex situation than the populations analyzed so far with resequencing data that include only two alleles, and where two homozygous genotypes (RILs or DHs) or three genotypes, both homozygotes and the heterozygote (F2s) occur, with all genotypes present in the parents. To overcome this problem, we took advantage of two properties of our data: (1) the existence of genomic regions with the homozygous genotype (E1E1) in the individuals of the T1E population, and (2) the genetic linkage map for the population previously developed using the peach SNP array (Donoso et al. 2015). Based on the genetic map, we selected a subset of T1E progeny that together covered all the genome with E1E1 fragments. Then, SNPs were called between E1E1 (a merged BAM file for E1E1 from a set of T1E individuals) and the reference genome ‘Lovell’. For each SNP passing the filter criteria, the ‘Lovell’ reference allele was replaced by the ‘E1’ allele using VCFconsensus (Danecek et al. 2011). For those SNPs that did not fulfill the quality criteria (minimum depth of >20x and global quality >60), the reference allele was replaced by ‘N’. Base pairs with zero coverage (not covered by any read) were also replaced by ‘N’. In the end, a reference genome for ‘E1’ was obtained (RefE1), which was identical to ‘Lovell’ in coordinates and size, but incorporated the ‘E1’ SNPs. Following the strategy described above, subsets of T1Eindividuals with entire genome coverage for genotypes E1E2, E1T and E2T were also selected, using the information of the genetic map. The RefE1 was used to align reads from E1E2 and E1T in order to obtain the E1vsE2 SNPs and E1vsT SNPs, respectively. Then, a second reference genome for allele ‘E2’ (RefE2) was created following a similar approach: starting with the RefE1, the allele of ‘E1’ was replaced by the alternative allele of ‘E2’ whenever a SNP was found between ‘E1’ and ‘E2’. Finally, the reads from genotype E2T were aligned with RefE2 to obtain the E2vsT SNPs. A core file with all SNP information (LovellvsE1, E1vsE2, E1vsT and E2vsT) was created (Figure S5.1).

The variability observed for the peach parent (E1 vs E2) was not uniformly distributed. Instead, large regions of the ‘Earlygold’ genome had no or very low frequencies of SNP markers. No marker variability in certain regions was previously described in several genetic maps for the species. In peach these regions are considered identical by descent (IBD) since it is common that two different founder lines share the same homozygous

fragments (Donoso et al. 2015). We considered an IBD region in 'Earlygold' whenever there was ≤ 10 SNPs between E1 and E2 in a window of 100 Kbp.

5.2.4 - In silico genotyping pipeline and estimation of crossover regions (COR)

In silico genotyping pipeline

For genotyping *in silico*, the SNP data of each individual of the progeny were compared with all the SNPs present in the SNP core file. All genotyping steps were performed using custom Python scripts. The first Python script 'compare_table_and_vcf.py' (Annex 1) compares the SNP information of a given individual of the progeny with the SNP core file and defines the allele represented by each SNPs (E1, E2 or T). After this step, a second custom script 'find_regions_from_genotyping.py' (Annex 2) reads the output of the first script and defines which regions have the almond allele ('T regions') and which regions only have peach alleles ('E regions'). Using the advantage of the previous information of 'E regions' and 'T regions', this second script makes a comparison between the SNPs from the individual and the SNP core file, similar to what 'compare_table_and_vcf.py' does. This allows the detection of the phase and the attribution of a complete genotype (E1E1, E1E2, E1T or E2T) for every SNP. After the previous classification, this script groups consecutive genotypes into windows. When a change on the genotype between adjacent windows is observed, a CO is called. In order to get as close as possible to the CO breakpoint, a third script was implemented, 'Fill_recpoints.py' (FRP; Annex 3), which looks for additional positions in the CO region using a second SNP core file that includes additional SNPs (of medium quality as described in the variant calling section below) that were not present in the first SNP core file (and represent a 40% increase in total number of SNPs available). The procedure applied is similar to what the second script does, but is only targeted to the breakpoint regions.

Further study of crossover regions

Only for the estimation of the length of the CORs we applied a less conservative approach than the FRP described in the previous paragraph using a new custom Python script,

‘Genotype_by_absence.py’ (GBA; Annex 4). This script implements the same procedure as ‘compare_table_and_vcf.py’, attributing one of the three possible alleles (E1, E2 or T) to each SNP detected in one given T1E individual. Then, adjacent SNPs identified as T are grouped together if the distance between them is< 10 Kbp. The biggest T window created is used as ‘seed’ to sum up smaller adjacent windows, when they were closer than one Mbp, giving rise to one unique window in the end. When this window starts later or ends earlier than what is expected by the chromosome size, a CO event is called for the hybrid. If the adjacent window is longer than one Mbp, a double recombination is called. To attribute the exact boundaries of the CO, the first flanking position is defined by the last T SNP of the window and the second flanking position is defined by the first T SNP that was expected to be present but is absent for the individual being genotyped. This same procedure was applied using SNPs classified as E2, grouping adjacent SNPs when the distance between them was < 500Kbp. Windows were consecutively added to the larger E2 ‘seed’ window when they were closer than two Mbp. As for the hybrid, a CO was called for ‘Earlygold’ when the E2 window did not cover all the expected range. Regions IBD (≤ 10 SNPs between E1 and E2 in a window of 100 Kbp) were excluded from this analysis.

Two parameters were calculated to characterize the distribution of COs in the chromosomes: the average of COs per chromosome (mean CO) and the rate between the genetic distance in Haldane centimorgans (cM) and physical distance in Mbp (CO rate). The number of cM of each chromosome was calculated as 50× the mean CO, and the physical distance was that of each chromosome in the peach genome version 2.0.a1 (https://www.rosaceae.org/species/prunus_persica/genome_v2.0.a1).

5.2.5 - Gene conversions

The high diversity between the T and E1 haplotypes present in the MB1.37 hybrid individual are a favorable situation for the study of gene conversions during its meioses, particularly when a DSB in the E1 allele was repaired with DNA from the homologous T chromosome. We looked for the presence of almond SNPs (T allele) in regions of all T1E individuals studied only in genome regions genotyped as peach/peach (E1E1 or E1E2)

using a custom Python script 'Get_gene_conversions.py' (Annex 5). All filtering steps were performed as proposed by Qi et al. (2014): all reads that mapped in more than one location in the genome were excluded, and for those that mapped uniquely, a minimum mapping quality score of 30 was required. SNPs from the progeny were only considered if they had at least 8 reads supporting them. Since we looked for T SNPs only in those regions peach/peach, no T indels exist there. However, to prevent the identification of false T SNPs that may have been caused by any intraspecific indel, the allelic ratio for each SNP was also examined. SNPs outside the range of 30% to 70% were excluded, since only heterozygous T SNPs were expected, and indels could produce false homozygous T SNPs (allelic ratio > 70%). We used Tandem Repeats Finder Version 4.09 (Benson, 1999) with a minimum alignment score of 10 and a maximum period size of 20, the remaining parameters as default, to look for tandem repeats in the RefE1 genome. T SNPs overlapping with a tandem repeat, or within the range of 50 bp upstream or downstream of a tandem repeat, were excluded. The remaining T SNPs were then analyzed and a putative gene conversion was called whenever a T SNP in a peach/peach region was found. Consecutive T SNPs were grouped together in the same putative gene conversion if they were close (< 1 Kbp) and if no other SNP (i.e. E2 SNP) existed between them. Putative gene conversions were then manually inspected and only those with unique positions were kept.

5.2.6 - Indel calling

In the case of indels, a different approach was implemented. Since RefE1 was created with the only purpose of *in silico* genotyping using SNPs, it does not consider 'E1' indels and other structural variations (SVs) in its sequence. Thus, in order to add these polymorphisms to the already existing RefE1, reads from E1E1 (the same reads used to generate RefE1) were aligned against the 'Lovell' reference genome. From the crude BAM file we discarded only reads that were annotated as PCR duplicates and the output was used as input for SV identification. Small indels were obtained by running GATK and the output was filtered with the following parameters: minimum genotype quality: 99, minimum depth: 10×; maximum length: 10bp. For the detection of larger deletions we implemented the following strategy: (1) we used Platypus v0.8.1 (Rimmer et al. 2014) filtering for deletions ≥ 10bp and (2) MATE-CLEVER (clever-toolkit) (Marschall et al.

2013) filtering for deletions between 10 bp and 5 Kbp, with at least 20 reads supporting it. Regions that fulfilled these requirements were annotated as large deletions. All deletions obtained with GATK, Platypus, MATE-CLEVER and manually detected were combined and were annotated as E1_vs_Lovell deletions. For insertions, we also used GATK for detecting small polymorphisms (< 10bp) with the same parameters as for deletions. Larger insertions were detected with Platypus and Pindel v0.2.5a4 (Ye et al. 2009), selecting for insertions larger than 10bp. At the end, both insertions and deletions were combined and a single VCF file was generated. This VCF file was then used to incorporate all polymorphisms into RefE1, generating RefE1_plus_indels.fa genome. This new E1 genome was then used as reference to call SVs for 'Earlygold' (E1E2), which represents the intraspecific SV variability and forMB1.37 (E1T; interspecific SV's), following the same procedure as described above.

5.2.7 - Search of DNA motifs associated with crossovers

We used MEME (Bailey et al. 2009) to look for DNA motifs that could be enriched at breakpoint sites. We selected the DNA sequence of all crossovers mapped in 5Kbp or less to look for motif enrichment. For every DNA fragment selected, we extended the region 1kbp upstream and downstream in order to ensure the inclusion of any DNA motif that could be relevant for the CO. The sequence of RefE1 was used to look for these motifs (in both intra and interspecific crossovers). MEME was used with 'anr' model. The Markov background model was built with the tool 'fasta-get-markov' available at MEME webpage (http://meme-suite.org/) using random sequences of 5 Kbp in size from RefE1 genome (excluding centromeric regions). MEME was configured to search for the five more significant motifs, with a size between six and 25 base pairs. All remaining parameters were kept as default.

5.2.8 - Validation of SNPs and Indels

For SNP validation, we looked for the presence of the same SNPs previously used in the construction of the genetic linkage maps of the parental lines (Donoso et al. 2015) in the resequencing data that we have developed in this work. The set of intraspecific filtered SNPs (E1E2 SNPs) were used to look for the SNPs mapped in 'Earlygold' genetic linkage

map. Positions matching with ‘N’s’ in the RefE1 genome or positions removed from E1E2 SNPs due to filtering criteria were excluded from the analysis. The same procedure was implemented for those SNPs mapping in the MB1.37 genetic linkage map, using the interspecific SNPs (E1T) identified in this work.

For validation of indels, we selected specific regions close to genes of interest for parallel studies. Primers were designed from the flanking sequence using Primer3 (Untergasser et al. 2012). For indels with less than 40 bp in size, we developed primers with generic non-complementary nucleotide sequences at their 5'-ends (*tag primers*; Hayden et al., 2008) and results were visualized in an Abi Prism 3130xl automated sequencer (Applied Biosystems). For indels bigger than 40 bp, standard primers were designed around the polymorphism and results were observed directly in agarose gels.

5.2.9 - Pollen viability determination

Flowers from parents ‘Earlygold’, ‘Texas’ and the hybrid MB1.37 were collected in the field just before anthesis. Anther dehiscence was induced in the lab using a 60 watt lamp over the anthers overnight. Opened anthers were transferred to a slide and were dyed with a jelly aceto-carmine solution (Marks, 1954), excluding ferric acetate. Pollen grain viability was observed in a light microscope Olympus BH-2.

5.2.10 - Statistical analysis

For correlation analysis between inter and intraspecific CO numbers we divided the genome in 500 Kbp, 1 Mbp and 5 Mbp domains. The position of each CO was considered that of the midpoint of its COR, and CORs longer than the size of the domain were discarded. Spearman rank correlation was used to test these relationships. Comparisons of SNP or CO parameters measured in the chromosomes of the maps constructed with different parents were performed with two-tailed paired t tests. R version 3.2.5 (R development core team, 2016) was used to perform correlation and statistical tests.

Graphical representation of SNP density, CO and GC distributions and pollen viability were performed using ggplot2 package (Wickham, 2009).

5.3 - Results

5.3.1 - Variant calling and construction of SNP core files

In this work we sequenced the genome 'Texas', 'Earlygold', and their hybrid MB1.37, with a genome depth of 80× on average. Additionally, we sequenced 128 individuals from their backcross one progeny to the 'Earlygold' parent (T1E) with a mean genome depth of 42×, ranging from 27.6× for individual T1E-93 to 65.6× for individual T1E-97 (Table S5.2). A total of 1,300Gbp of Illumina whole genome sequence data was generated, which is equivalent to 5,730 peach genomes (considering the size of peach genome v2.0.a1 of 227Mbp; https://www.rosaceae.org/species/prunus_persica/genome_v2.0.a1).

Clean reads from both parents and progeny were mapped to the peach reference genome. After applying our filtering criteria, depth reduction ranged from 21.2% of excluded reads for 'Earlygold' to 38.4% in 'Texas', being the hybrid and all 128 T1E individuals between these boundaries. In the end an average of 32× of genome depth was obtained for the T1E progeny, ranging from 20× for individual T1E93 to 50× for individual T1E97 (Table S5.2). The genome coverage in the parents and T1E offspring (the proportion of the genome covered with at least one read in each individual studied) was 95.12% on average. The minimum values were found for 'Texas' (86.44%) and the maximum for T1E199 (95.89%) (Table S5.2).

For the elaboration of the RefE1 genome we selected 26 individuals that together provided complete genome coverage with E1E1 fragments (Table S5.3). The reads of the E1E1 regions were merged into a large alignment file, covering all the peach genome with average depth of 107×, (Table S5.4). After applying our filtering criteria we

identified 189,586 SNPs between E1E1 and 'Lovell'. (Table S5.1). Following the same strategy, we grouped reads for the other three genotypes (E1E2, E1T and E2T) using different subsets of individuals from the T1E population (Table S5.3) generating at the end three new merged alignment files with a massive increase in average depth (Table S5.4).Merged reads for E1E2 and E1T genotypes were mapped against RefE1, which allowed to obtain the E2 and T SNPs. We finally created a RefE2 genome (replacing E1 reference allele from RefE1 by E2 alternative allele) that was used for the SNP calling of the E2T genotype. In the end we found 162,224 SNPs between E1 and E2 (which represents 'Earlygold' variability), and 1,171,268 and 1,139,179 SNPs between E1 and T and E2 and T, respectively (which represents the interspecific almond/peach distance). 'Earlygold' had on average 1 SNP each 1,399 bp, whereas the hybrid MB1.37 (E1 vs. T haplotypes) had 1 SNP each 194 bp, approximate seven-fold higher. We calculated also the numbers of medium quality SNPs and obtained an additional 21% for E1 vs E2 and 41-43% for the comparison between T and the two 'Earlygold' haplotypes (Table S5.1).

The number of SNPs found between the two species (E1 allele vs T allele) was highly correlated with the size of the chromosome (Spearman correlation ρ=0.98, $p<3.3\times10^{-5}$), whereas in the case of 'Earlygold' SNPs (E1 allele vs E2 allele) there is no correlation (ρ=0.06, n.s.). These results occurred because 'Earlygold' SNPs were not uniformly distributed in contrast with the even distribution of the MB1.37 interspecific parent (Figure 5.1). There were several regions in the genome of 'Earlygold' either completely homozygous or with very low variability (less than 10 SNPs/100 Kbp) along with other regions with much higher variability levels. These regions accounted for 123.2 Mbp of genome sequence in 'Earlygold', which correspond to 54% of its genome, meaning that 160,721 SNPs between E1 and E2 (99%) are located in only 46% of the genome (103.8 Mbp), which represent one SNP every 624 bp on average in the polymorphic regions.

In the case of indels we found a total of 64,051 polymorphisms in 'Earlygold' and 577,851 in MB1.37, which represent on indel every 3,544 bp (1,608 bp excluding IBD regions) and one indel every 395 bp, respectively. The size of deletions ranged from 1-4,998 bp in the peach parent whereas for the hybrid it ranged from 1 bp to 4,994 bp. In the case of

insertions, the smallest was 1 and the largest 63 in 'Earlygold' and in MB1.37 was 1 and 81 bp, respectively. SNP and indel distributions were significantly correlated for both parents: ρ=0.87 for 'Earlygold' and ρ=0.81 for MB1.37 (P-value < $1.0x10^{-22}$ in both cases). The overall distribution of SNPs and indels, for both parents, is presented in Figure S5.2.

5.3.2 - SNP and indel validation

We validated the SNPs discovered in this work by searching for previously mapped SNPs in the genetic maps of both parents (Donoso et al. 2015). In total we looked for all 1,037 SNPs from 'Earlygold' genetic map in the E1 vs E2 SNP dataset developed in this work. At the end 907 SNPs from the map were present in our data. For the remaining 130, 53 overlapped with 'Ns' in the RefE1 and 16 were lost due to filtering. Thus only 61 SNPs from 'Earlygold' genetic map (94.1% accuracy) were not called as expected. The same procedure was used for the hybrid. From the initial 1,904 SNPs in the genetic map, 1,636 were present in E1 vs T dataset. For the remaining 268, 35 overlapped in 'N's' and 67 were filtered out. In the end only 166 SNPs (91.3% accuracy) were not detected. These accuracy levels were in the range of those found before in peach and related species: better than Cao et al. (2014) (63.6%), similar to Cao et al. (2016) (83.8-93.6) and worse than Xie et al. (2016) (100%).

For indels, we tested for 44 different polymorphisms, 35 interspecific and nine intraspecific, ranging from 1bp to 812 bp. In the case of interspecific indels, 34out of 35 produced an amplification product, and from these 34, 32 (94%) amplified the expected size and the other two were monomorphic or amplified a different size than expected. In the case of intraspecific indels, one did not amplify any detectable fragment, whereas the remaining eight indels (100%) amplified the expected polymorphism. These accuracy values are comparable with previous validations in Arabidopsis (95.8%; Păcurar et al. 2012), rice (90.5%; Lü et al. 2015), chickpea (95%; Das et al. 2015), barley (95% Zhou et al. 2015), watermelon (78.2%; Liu et al. 2016) or soybean (68%; Song et al. 2015). Altogether, the accuracy of the polymorphisms called in this work is high, reinforcing the idea that high sequence depth and strict filtering criteria are major factors for quality of *in silico* polymorphism analysis.

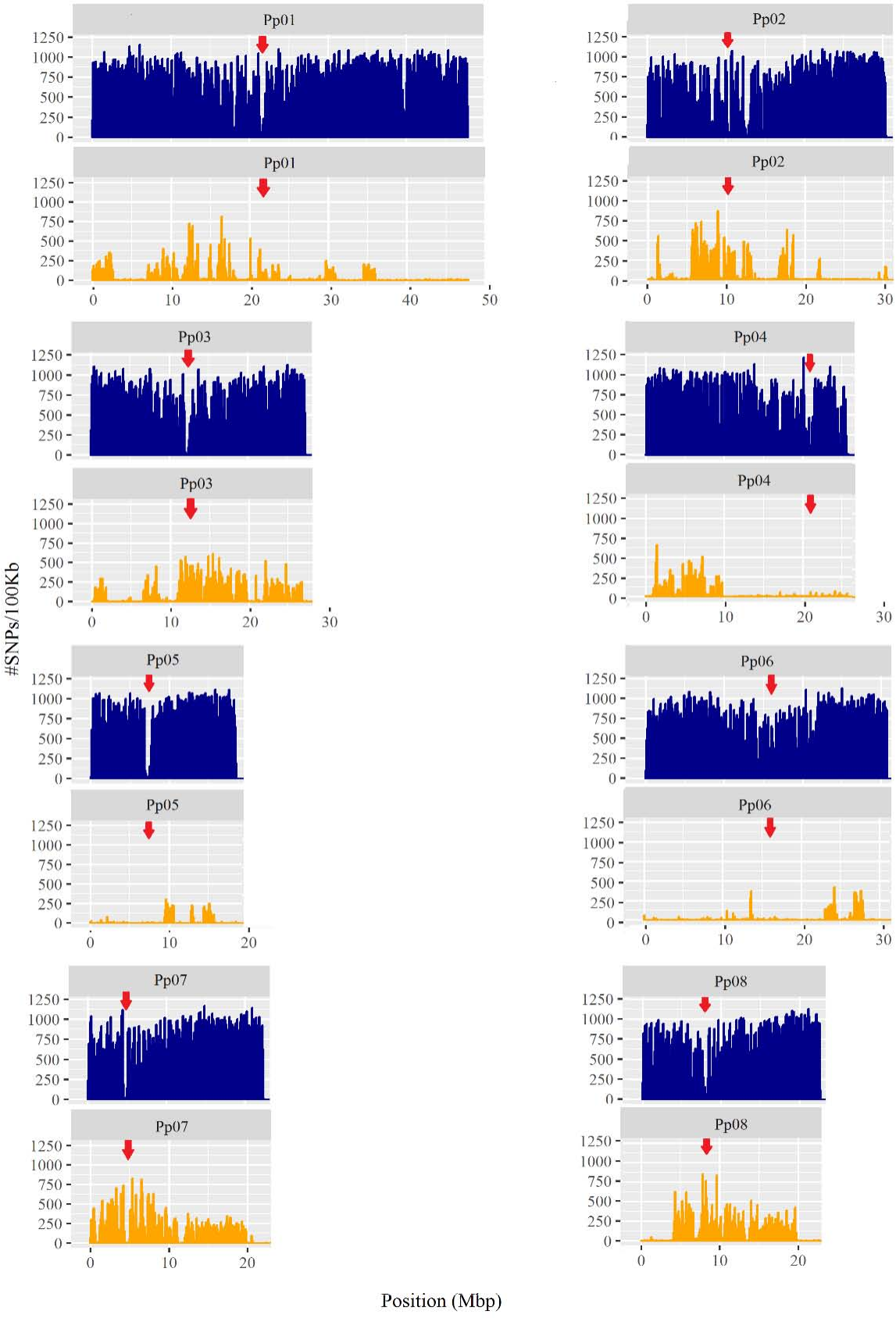


Figure 5.1 - Distribution of E1 vs E2 SNPs ('Earlygold' intraspecific variability; orange) and E1 vs T SNPs (MB1.37 interspecific almond × peach variability; blue) for the eight chromosomes of *Prunus* (Pp01-Pp08). Approximate positions of centromeres are marked with red arrows.

5.3.3 - Distribution of SNP polymorphisms and estimation of their effects

We investigated the exact location of SNP polymorphisms in the genome with SnpEff software to see how many of them occurred inside genes in order to predict their possible effect at protein level. Overall, the majority of SNPs fell outside of genes (83% in E1E2 SNPs; 73% in E1T SNPs) and only 4% (E1vsE2) and 6% (E1vsT) fell in exons. Within the SNP variants detected between peach alleles, in 232 occasions (0.036% of total) the polymorphism was predicted as causal for a disruptive effect on the protein (typically by causing changes in start or stop codons), in one or more transcripts, for a total 142 genes affected. In the case of peach/almond SNPs, this disruptive effect was observed 3,417 times (0.049% of total), affecting 2,480 different genes (9.2% of peach annotated genes). This represents an increase of 17 fold on the number of genes affected when moving from intraspecific peach SNP variability into peach/almond SNP variability.

5.3.4 - In silico genotyping, CO detection and relationship between CO occurrence and pollen fertility

In this work we present for the first time a pipeline to successfully and accurately detect breakpoint sites in a population with both parents partly heterozygous and four possible genotypes across the progeny. This task was achieved by (1) creating a SNP core file based on individuals of the population instead of parents (Figure S5.1), and (2) constructing a bioinformatics pipeline that, in a conservative way, attributes a genotype to windows or groups of SNPs (instead of each SNP individually) in order to reduce the background noise caused by false SNPs, thus preventing the erroneous call of crossover events. One of the pioneering resequencing papers (Huang et al. 2009) used a similar strategy, with the difference that we used an adjacent-like windows approach (where SNPs sharing the same genotype are grouped in windows, and these windows are then compared with adjacent ones) instead of an overlapping windows approach (where a window of fixed size moves along the SNPs calculating at each step the genotypic frequency). Windows were also used in Arabidopsis (Yang et al. 2012; Choi et al. 2013; Wijnker et al. 2013, Rowan et al. 2015) and in peach (Wang et al. 2016), where SNPs were grouped into blocks for CO identification. For other works in plants the genotype was attributed directly to each SNP *per se* (Lu et al. 2012; Bayer et al. 2015), also with good performance on CO detection. Overall, the filtering criteria applied in this work to

call variants is more strict than those in other studies, mainly because the majority of variants here presented are heterozygous, which requires additional filtering to confidently call a variant.

By implementing this pipeline we examined recombination in 128 individuals of the T1E population. Three individuals from the initial set of 128 were excluded from the analysis (T1E199, T1E201 and T1E694) since they presented unexpected patterns of CO for all chromosomes, suggesting a possible contamination of samples during sample processing. We detected a total of 1,081 CO events across 125 individuals of the T1E population, 612 CO coming from 'Earlygold' meioses and 469 from the almond/peach hybrid MB1.37 meioses (Figure 5.2). While the number of CO events detected in the hybrid are a precise estimation of their actual number due to the even and dense SNP coverage of the genome for this parent, the COs detected for 'Earlygold' are likely an underestimation, since there are several chromosomes with large homozygous regions at their extremes (Figure 5.2) precluding the identification of recombination events.

Overall, these regions in the extremes of the chromosomes account for a total of 46 Mbp (20.3% of the genome), being chromosomes one, four, five and six (Pp01, Pp04, Pp05 and Pp06) the ones with largest homozygous regions (>10 Mbp) (Figure 5.1). Assuming that COs occur at random along chromosomes, an estimation of the real number of COs for 'Earlygold' would be 736 events for the 125 individuals analyzed, 57% higher than in the almond × peach hybrid MB1.37, showing that the level of intraspecific recombination was much higher than that of interspecific recombination (Table 5.1, Figure 5.3). In 'Earlygold' we observed an overall average of 1.47 COs/chromosome (11.79 COs/meiosis, equivalent to 2.66 cM/Mbp), ranging from 2.25 CO/chromosome (2.35 cM/Mbp) in Pp01 to 0.92 (2.49) in Pp05, whereas the peach/almond hybrid parent had only 0.94 COs/chromosome on average (7.50 COs/meiosis or 1.75 cM/Mbp), ranging from a maximum of 1.10 COs/chromosome (1.15 cM/Mbp) in Pp01 to a minimum of 0.78 (2.12 cM/Mbp) in Pp08 (Table 5.1).

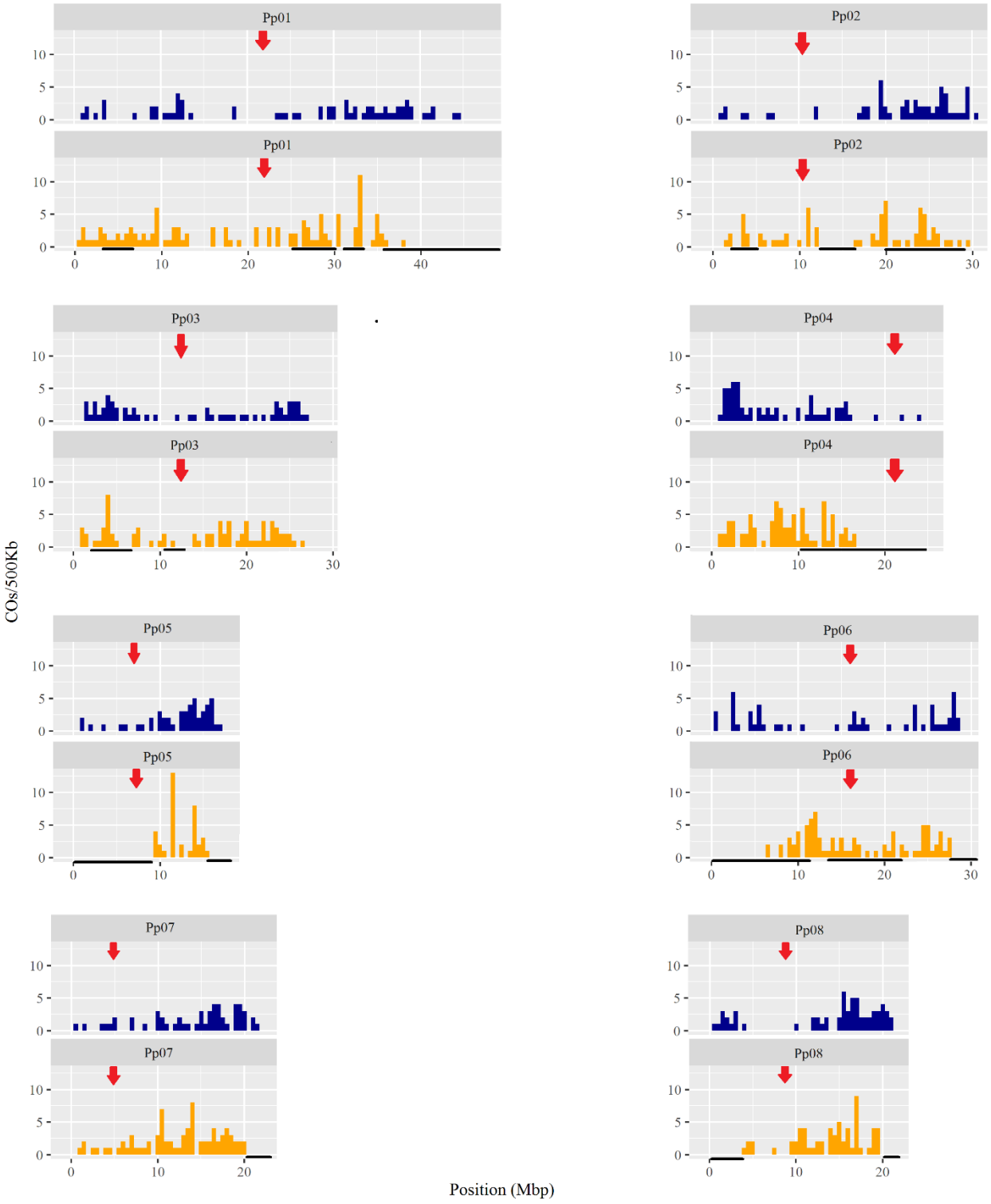


Figure 5.2 - Distribution of intraspecific (orange bars) and interspecific (blue bars) crossovers across the genome in the T1E (MB1.37 x 'Earlygold') progeny. Large regions of low variability in 'Earlygold' are represented with black lines at the bottom of the intraspecific graphic. Peach chromosomes are Pp01-Pp08. Approximate positions of centromeres are marked with red arrows.

CO numbers and positions were in agreement with those previously described in the genetic map for this population (Donoso et al. 2015), where 11.19 and 7.34 CO/meiosis for intra and interspecific recombination, respectively, were found. These values are slightly lower than those obtained in this work, with an overall increase of 2.2-5.4% COs (Figure 5.3), as expected from the higher resolution of the resequencing data. The distribution of COs was similar for both parents and for all eight chromosomes, being characterized by reduced CO occurrence at centromeric regions and a higher CO number near the telomeres (Figure 5.2).Considering windows of 500 Kbp, the numbers of intra and interspecific COs were correlated (ρ=0.36; p=3.6×10^{-15}). With longer regions of 1 Mbp and 5 Mbp, the correlation increased to ρ=0.46 (p=8.9×10^{-14}) and 0.69 (p=2.8×10^{-09}). CO number per chromosome and chromosome size were positively but not significantly correlated (ρ=0.52and ρ=0.57 for 'Earlygold' and MB1.37, respectively). However, the CO rate (cM/Mbp) was negatively correlated with chromosome size, in intraspecific (ρ=-0.45, ns) and significantly in interspecific (ρ=-0.93; p=0.0009) recombination, indicating that smaller chromosomes have more COs per distance unit than longer chromosomes, which in turn may be related in species with small chromosomes, such as peach, by the need of one crossover event per chromosome to guarantee gamete viability. This inverse correlation has been detected in peach and peach x *P. davidiana* (Wang et al. 2016) and in other species (Kaback et al. 1992; Kaback 1996).

Table 5.1 – Mean number of crossovers (COs) and CO rate for each chromosome Pp01-Pp08 in both intraspecific ('Earlygold') and interspecific almond × peach (MB1.37) meioses. Data for 'Earlygold' were adjusted considering the homozygosity of chromosome ends.

Samples	**Pp01**	**Pp02**	**Pp03**	**Pp04**	**Pp05**	**Pp06**	**Pp07**	**Pp08**	**All**
Physical distance (bp)	47.85	30.41	27.37	25.84	18.50	30.77	22.39	22.57	225.69
Intraspecific 'Earlygold'									
mean COs	2.25	1.14	1.25	1.90	0.92	1.67	1.40	1.26	11.79
CO rate (cM Mbp^{-1})	2.35	1.87	2.28	3.68	2.49	2.71	3.12	2.80	2.66
Interspecific 'MB1.37'									
mean COs	1.10	0.96	0.94	0.96	0.78	0.90	0.86	0.99	7.50
CO rate (cM Mbp^{-1})	1.15	1.58	1.72	1.86	2.12	1.46	1.93	2.20	1.75

The average CO per chromosome for MB1.37 was lower than the minimum expected of one (0.94 overall and all chromosomes but Pp01 lower than 1.00), which led us to suspect that the microsporogenesis on this line could be producing some unviable gametes. To prove this, we observed the pollen grains at mature stage for the three ancestors of T1E: 'Earlygold', 'Texas' and MB1.37 and found that, in the hybrid, almost half of the pollen grains (46%) were unviable (Figure S5.3), whereas its two parents had a normal pollen behavior with only 3% (peach) and 1%(almond)pollen grains that were not viable.

5.3.5 - Recombination hotspots in intraspecific and interspecific meioses and associated DNA motifs

The distribution of COs along chromosomes was not uniform, with regions that accumulated more COs than others. Hotspots of recombination are typically regions of 1-10 kb in size that have higher CO rates than adjacent regions (Choi and Henderson 2015). We looked for recombination in windows of 10 kb across the entire genome and found a range of 0-3 COs in the same window. Assuming that the number (k) of CO events follows a Poisson distribution with average of m=0.034 CO/10 kb fragment for intraspecific and m= 0.021 for interspecific COs in the population examined, where the probability of k recombination events is $p(k)=m^k e^{-m}/k!$. This probability is $< 1\times10^{-3}$for finding two COs in the same 10 kb or $<1x10^{-5}$ in the case of three COs. Thus we defined these 10 kb regions with two or more COs as probable hotspots of recombination.

In total we found 19 hotspots, 11 detected in the meiosis of 'Earlygold' and eight coming from MB1.37. Pp01 and Pp08 accumulated the majority of hotspots detected (seven and five hotspots, respectively), whereas no hotspots were found for Pp02 or Pp05 (Table S5.5). COs with a distance between flanking markers higher than 10 kb were excluded from this analysis, reducing from 1,081 to 245 (23%) the number of CO used. Of these 245, 118 COs were intraspecific and 127 interspecific. The fact that these 19 hotspot regions were found using just a subset of all COs detected raises the possibility that some additional hotspots were not detected.

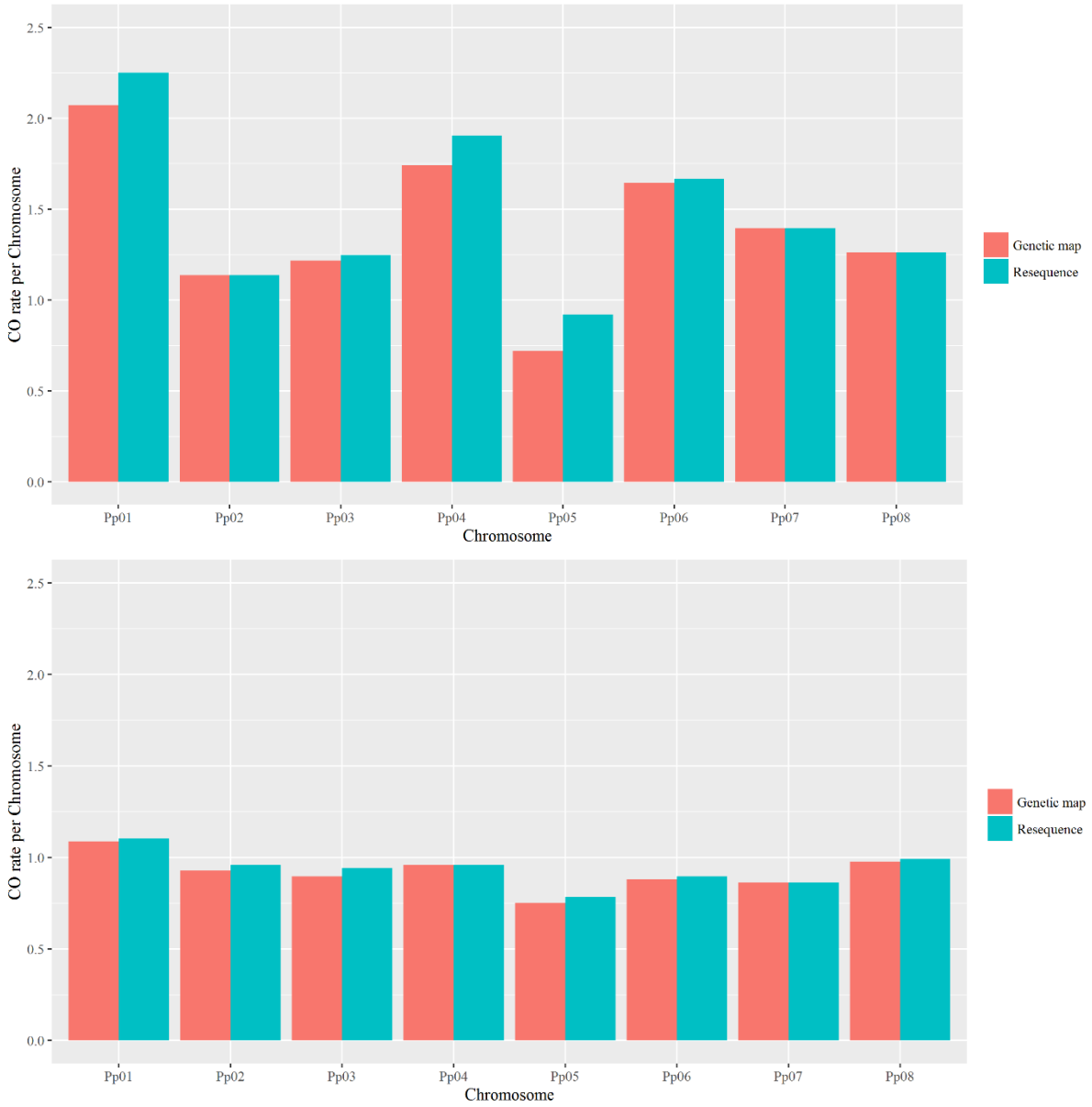


Figure 5.3 - Comparison of number of crossovers per chromosome between genetic map (orange) and resequence data (blue) in the T1E (MB1.37 x 'Earlygold') progeny. The crossover rate in 'Earlygold' is presented at the top and for the hybrid is presented at the bottom. Data for 'Earlygold' were adjusted considering the homozygosity of chromosome ends

In order to study the presence of DNA motifs that could be enriched at CO sites, we selected only those with a resolution of 5 kb or less (193 COs in total; 76 in 'Earlygold', 117 in MB1.37). We found several DNA motifs (Figure 5.4A-B), the most significant of them for intraspecific were the CT_n and poly-A microsatellites (e-value: 7.5e-132 and 2.4e-051, respectively), whereas for interspecific were $(CT)_n$, AGTATAGCCGCGCGGCTATACTATT and $(CCN)_n$ (e-value: 8.6e-210, 1.9e-200 and 5.6e-088, respectively). Further we decided to investigate the presence of any DNA motif

associated with hotspot occurrence (i.e. DNA motifs that could be overrepresented in the 19 regions classified as hotspots in this work), where we identified again the $(CT)_n$ [or $(AG)_n$] motif as the most significant for intra and interspecific hotspots. (Figure 5.4C-D)

5.3.6 - Estimation of the crossover region (COR) length

The resolution of detection of a COR (i.e. the distance between the two closest SNPs flanking the CO) depends on the marker density available. High resolution CORs allow a more precise study of the DNA sequence where the recombination breakpoint took place. In this work, the high variability found between peach and almond species contrasted with the low variability found within peach 'Earlygold'. As a consequence, the average distance between the two flanking markers was very different in the two parents of T1E. As a first approach we used the more conservative FRP strategy, and found that intraspecific CORs had an average length of 1.5 Mbp (median of 340 Kbp), whereas for interspecific CORs that distance decreased to0.9 Mbp (median 170 Kbp). To estimate these data more accurately, we developed the GBA approach, that allowed us to estimate the COR length as the distance between the closest SNP markers flanking 977 COs, 521 intraspecific and 456 intraspecific (Table S5.6). On average, interspecific COs had 679 bp, almost twice the size expected assuming that they were produced at random (388 bp =2 × 194 bp of mean distance between two SNPs in MB1.37). Data obtained for intraspecific CORs yielded much longer values (avg. 84,200 bp), which compared with their expected size of 2,798 (1399×2) is ~30 times longer. In part this can be attributable to the effect of the long IBD zones of the 'Earlygold' genome, so we separated the CORs that were in chromosomal regions >100 Kbp with SNP frequencies <10/100Kbp, with the result of obtaining shorter average CORs (29,906 bp), which is still more than an order of magnitude (24×) above the expected size of 1,248 bp (624×2; because in the non-IBD fragment of the genome the SNP density is higher) although with most of the CORs skewed towards the smaller sizes, with a median of 4,472 bp (Table S5.6, Figure S5.4).

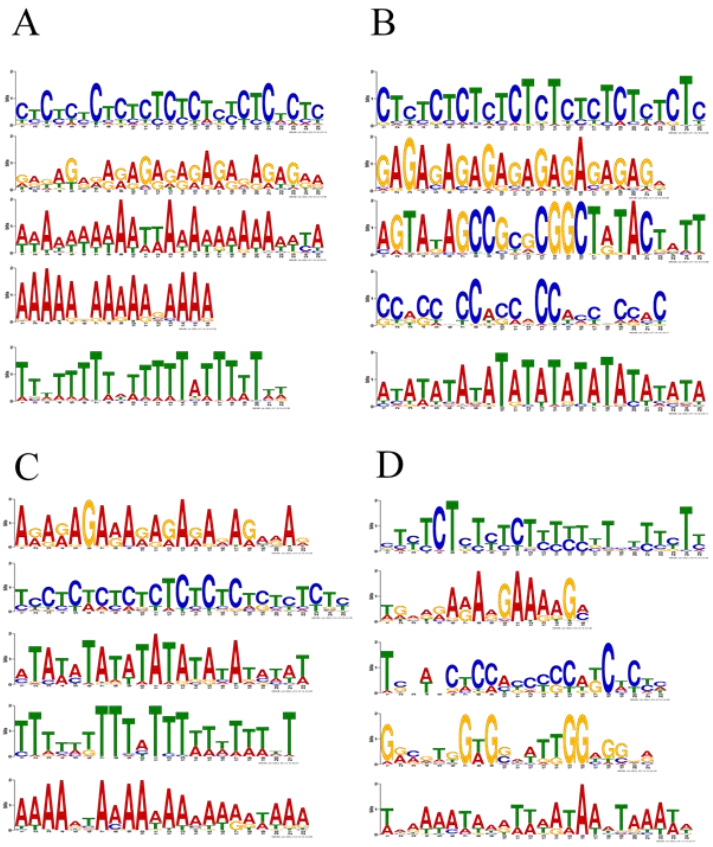


Figure 5.4 - The five DNA motifs more significantly enriched at crossover (CO) sites. A) Motifs found for intraspecific COs, B) motifs for interspecific COs, C) motifs for intraspecific CO hotspots, and D) motifs for interpsecific CO hotspots.

5.3.7 - Detection of gene conversions

Gene conversions (GCs) are a change in DNA sequence occurring when a NCO is resolved by the substitution of a DSB by a short stretch of the DNA from the homologous chromosome pair. GCs are detectable in the progeny when polymorphisms are abundant, since their size typically ranges from a few bp to a few Kbp. In this work we took advantage of the high variability in the hybrid MB1.37 to detect GCs across T1E individuals that might have occurred in hybrid meiosis. The detection of GCs was limited to regions genotyped previously as peach/peach (E1E1 or E1E2 genotype), since that was the only way to unequivocally identify one insertion of a small fragment coming from almond (T allele) into E1 allele at hybrid meiosis. For all 125 T1E individuals analyzed, peach/peach regions accounted for a total of 9,412 Mbp of sequence, approximately 75.3 Mbp per T1E individual, which represents 32% of the genome on average, a lower proportion than the 50% expected. The detection of T SNPs in these peach/peach regions

led us to discover 790 potential GCs for the 125 individuals analyzed, representing approximately 6.3 GCs per individual, which corresponds to 0.8 GCs per chromosome. We expect this result to be a conservative approximation to the real number of gene conversions that indeed took place at hybrid meiosis, since we applied very stringent filtering criteria. The largest GC found has 3,266bp and was detected in Pp02 for individual T1E474. The actual GC size is expected to be longer than that, with boundaries starting before the first T SNP and ending after the last T SNP. The highest accumulation of T SNPs inside a GC was 18 consecutive SNPs for the largest GC found (3,266bp) in Pp02, which corresponds to one T SNP every 181 bp on average. The average number of GCs per individual (6.3) was obtained for only 32% of the genome that was available to us (regions peach/peach) and our estimation was done only for one of the two gametes. An estimation for the entire genome would then translate into 39.4 GCs per meiosis, or 4.92 GCs/chromosome. The location of the GCs in the genome was not uniform, with a tendency of accumulation of GCs in the pericentromeric regions where interspecific COs were scarce (Figure S5.5). However, there is no significant correlation between the location of both genetic events with 500Kbp windows (ρ=-0.07; P-value: 0.14), 1Mbp (ρ=-0.14; P-value: 0.0288) or 5Mbp (ρ=-0.01; P-value: 0.9528).

5.4 - Discussion

To fulfill our objective of comparing the recombination landscape of intra vs. interspecific crosses, we chose one population (T1E) that has an interspecific almond × peach hybrid (MB1.37) and a partly heterozygous peach cultivar (‘Earlygold’) as parents. The high level of polymorphism of the interspecific hybrid and high depth DNA coverage (average of 42×) guaranteed a high density of markers to accurately find CO breakpoints, and the large number of plants resequenced (N=125) insured that a sufficiently large set of COs would be surveyed. T1E was more complex than the populations typically used for resequencing analysis, and required a specific strategy for *in silico* genotyping that we developed in this chapter.

5.4.1 - High and uniform variability in the almond × peach hybrid vs. lower and uneven variability in peach

The SNP distribution of the 162,224 SNPs identified in 'Earlygold' was irregular, with polymorphisms accumulating in certain locations whereas extensive regions were completely depleted of variability. These homozygous regions in 'Earlygold' were previously seen as large gaps in the genetic map (Donoso et al. 2015) and they are commonly found in peach modern cultivars (Donoso et al. 2015; Chapter 4) probably corresponding to fragments identical by descent due to their high level of coancestry (Micheletti et al. 2015) and the self-pollinating nature of the species. Aranzana et al. (2012) found a 2.3× higher SNP variability using the sequence of various genomic regions in a collection of peach cultivars, and Wang et al. (2016) estimated almost twice more SNPs in the resequence of a Chinese accession. This difference may be explained by the fact that more than half of the 'Earlygold' genome (54%) was essentially homozygous, whereas the other works were based in a set of cultivars (Aranzana et al. 2012) or in the results of a Chinese accession (Wang et al. 2016), likely to have a higher level of variability compared to Occidental breeding materials (Li et al. 2013; Cao et al. 2014; Micheletti et al. 2015).

The peach and almond gametes that generated MB1.37differed by 1,171,268 SNPs (one SNP each 194 bp), 7.22× more SNPs than 'Earlygold'. Unlike peach, the number of SNPs detected was high and uniform along the whole genome (Figure 5.1), as expected considering that this is a wide cross between two distant taxons. Wang et al. (2016) found and intermediate 4.4x increase on diversity when comparing peach and a peach ×*P. davidiana* hybrid, which confirms that this species is a closer relative to peach than almond (Chin et al. 2014).

Considering the effects of the SNPs identified, the great majority of 'Earlygold' SNPs fell outside of genes, and from the ones that fell within a gene, only a very small amount had a predicted disruptive effect at the protein level, for a total of 142 genes affected. On the other hand, peach/almond SNPs, with a much more homogeneous chromosome-wide distribution, do concentrate more inside of genes, and have a greater impact at protein level, with 2,480 genes being theoretically disrupted. Taking into account that the total genome size non identical by descent in 'Earlygold' is approximately 50%, we estimate a total of 284 genes possibly disrupted in peach, meaning that in peach × almond we

found an 8.7× more of these genes, approximately the same proportion as 7.0× difference in variability between peach and almond (Velasco et al. 2016). It is interesting to note that 2,480 disrupted genes correspond to 9.2% of the total genes annotated in the second version of the peach genome, which suggests that almond carries a high level of mutation load.

5.4.2 - Higher CO frequency in intraspecific vs interspecific meioses but similar genomic CO distribution

We found a total of 1,081 COs and estimated a higher (57%) mean number of COs per meiosis in 'Earlygold' (11.79) than in the peach/almond hybrid MB1.37 (7.50). Our results for overall CO rates were in agreement with previous data from the genetic map (Donoso et al. 2015), with a slight increase on CO resolution when using resequence data (Figure 5.3). Our figures for peach were very similar to those found by Wang et al. (2016) (11.92 CO per meiosis; paired t test for the values of the eight chromosomes t=0.09; ns), suggesting that the recombination rates within peach are relatively uniform. These data are consistent with a low level of recombination in peach compared to other species such as rice (Si et al. 2015) or Arabidopsis (Yang et al. 2012), which more than double the recombination rate of peach and suggest that low recombination may be part of a specific reproductive strategy of perennial species (Yang et al. 2016; Luo et al. 2015).

The mean CO/chromosome for the almond × peach hybrid was significantly lower than that of the intraspecific level (paired t test 4.02; p=0.005). Recombination is often limited in interspecific hybrids, one of the possible causes being sequence divergence between homologous chromosomes as demonstrated in yeast (Datta et al. 1997), humans (Waldman and Liskay 1988), fungi (Seplyarskiy et al. 2016) and Arabidopsis (Emmanuel et al. 2006). A strong reduction in recombination was also observed in interspecific hybrids of tomato (Tanksley et al. 1992) and potato (Gebhardt et al. 1991). Examples that variability scale inversely with recombination rates exist also in *Prunus*, where almond has a lower recombination rate compared to peach based on linkage map information (Arús et al. 2005), being almond a much more variable species than peach (Mnejja et al. 2010; Velasco et al. 2016), and the peach x *P. davidiana* hybrid studied by Wang et al. (2016) that had a mean number of COs/meiosis of 9.47, lower than our estimation for 'Earlygold' (11.79, paired t test 2.08; n.s.) and significantly higher than the peach ×

almond hybrid (7.50, paired t test 4.54; p=0.003); the order of divergence for peach, peach x *P. davidiana* and almond x peach, being the reverse (1 SNP every 1.399, 204, and 194 bp, respectively), overall confirming that nucleotide diversity is a key factor in CO formation.

The difference of recombination rates observed between peach and almond could be also attributable to heterochiasmy, i.e. the differences between recombination rates in males and females, which occurs in several plant species, including Arabidopsis (Vizir and Korol, 1990),tomato (de Vicente and Tanksley, 1991) and certain Brassicas(Lenormand and Dutheil, 2005). Map comparison in various rosaceous crops did not allow to find clear patterns of heterochiasmy (Arús et al. 2005). If this phenomenon had played a role in the reduced recombination of MB1.37 due to its use as female parent, one would expect that in its F2 progeny (TxE) recombination rates would be higher overall. This progeny exists and a map was constructed (Donoso et al. 2015) that is longer than that of T1E (370 cM in T1E vs 472 in TxE). This difference was significant considering the heterogeneity of the different chromosomes (paired t test 2.53; p=0.04), suggesting a role, probably minor, of heterochiasmy in the low recombination rates of MB1.37.

When the CO distribution across the chromosomes was compared between inter and intraspecific meioses, the results were significantly correlated, showing higher CO rates near the chromosome ends and lower recombination in the pericentromeric regions and suggesting that CO distribution was similar at intra and interspecific level. Similar results were found by Wang et al.(2016) comparing peach with peach ×*P. davidiana*, overall indicating an irregular pattern of distribution comparible with observations in other plant species (Melamed-Bessudo et al. 2016; Mercier et al. 2015).

The mean length of the crossover region (COR) for interspecific meioses was estimated as 678 bp, almost twice its expected size. Longer CORs than expected were also found in the hypervariable fungus *Schyzophyllum commune*(Seplyarskiy et al. 2016), with COR lengths almost four times those expected, showing that COs occurred preferentially in regions where the genomes of the two parents were similar, i.e., avoiding divergent sequences. Given that the number of these regions is less frequent in highly diverse parents, this may also determine a reduction in CO number, providing a causal explanation for the lower CO frequence. For *S. commune*the extreme value of genetic distance of 0.14 resulted in an average of 0.76 CO/chromosome, compared to a much

lower distance (0.005 SNPs/bp) for almond × peach corresponding to a higher recombination rate 0.93 CO/chromosome. When looking at the intraspecific COR lengths, our estimate was surprisingly much higher, almost 30 kb or 24× its expected size. This can be explained assuming that the probability of occurrence of a CO is directly proportional to the length of the genome region with no or very low nucleotide divergence, so longer regions with these characteristics will have more opportunity for CO occurrence. In highly divergent pairs of homologous chromosomes the number of regions with optimal conditions for the occurrence of a CO diminishes, along with the size of the fragment able to recombine, resulting in smaller COR lengths between polymorphic markers and a progressive decline in the CO number.

5.4.3 - Low frequency of recombination hotspots and similar DNA motifs associated with CO events than Arabidopsis

Based on the 245 COs with COR lengths ≤10 Kbp we located 19 recombination hotspot regions (9% of the CORs are hotspots) containing a total of 41 CO events (a 17% of the 245 sampled). No obvious differences in number and distribution of the hotspots were identified at the inter or intraspecific levels (Table S5.5).These are low rates compared to existing data in yeast (10% of CORs correspond to ~50% of COs) (Pan et al. 2011), human (~2% of CORs account for 30% COs) (Kong et al. 2008) and in the fungus *S. commune*, where COs occurred mostly at recombination hotspots (Seplyarskiy et al. 2016). Wang et al. (2016) found only a few hotspots more than us (26) in peach and peach × *P. davidiana*, although their results are difficult to compare due to the difference of window size used (500 kb vs 10 kb). Various reasons can be proposed for the discrepancy in the frequency of hotspots, including the broad set of organisms analyzed with enormous differences in genome sizes, sequence variability, diversity of recombination mechanisms and strategies, and finally different approaches to the identification of CORs in the genome sequence.

Our analysis of sequences associated with recombination events showed that there are two main DNA motifs associated with COs, the poly-A sequence and the dinucleotide CT, as both of them are enriched in intra- and interspecific CO sites. Recent studies have investigated the presence of DNA motifs associated with COs in Arabidopsis(Wijnker et al. 2013; Choi et al. 2013; Shilo et al. 2015), revealing that CTT and poly-A motifs are

enriched at CO sites. The motifs found in this work with peach and those found in Arabidopsis are similar, suggesting that despite being two different species, the CO marks at DNA sequence were maintained over evolution. In humans it is known that the CCN motif is required for the recruitment of *PRDM9*, a SET domain protein with a zinc finger array that controls over 40% of all hotspots. No homologs of *PRDM9*have been found in plants to date, and the CO machinery for plant species is thought to be controlled differently, with some degree of re-use of transcriptional start sites (TSS) and transcription termination sites (TTS) of genes, where the transcriptional machinery is often attached (Choi and Henderson 2015). However, despite that *PRDM9* is not present in plants, the majority of PRDM9-like SET domain proteins in Arabidopsis are expressed during meiosis (Zhang and Ma, 2012). Also intriguing is the fact that CCN, the DNA motif to which *PRDM9* attaches to, was recently detected for the first time at CO sites in Arabidopsis (Shilo et al. 2015) and also in our work (only for interspecific COs). This reveals that much is still unknown concerning CO regulation and that the study of these DNA motifs in other plant species would be of great interest to understand the role of these sequences in the recombination landscape.

5.4.4 - Gene conversions were about five times more frequent than COs

Non crossovers (NCO) can only be detected in the form of gene conversions (GCs), where the double strand break (DSB) is repaired using the homologous chromosome as template and provided that there is sufficient polymorphism between the corresponding fragments of the homologous chromosomes. A few studies have recently addressed the presence of GCs on plant genomes, mainly in maize (Dooner 2002; Dooner and He, 2014), Arabidopsis (Lu et al. 2012; Yang et al. 2012; Drouaud et al. 2013; Wijnker et al. 2013, Qi et al. 2014), rapeseed and chickpea (Bayer et al. 2015), being its rate estimated in less than one GC per chromosome. The only exception was the work by Yang et al. (2012) in which GCs in Arabidopsis were estimated to be 90% to 99% of all DSBs (about 3,000 GCs per meiosis). Later Qi et al. (2014) re-analyzed Yang's data and concluded that the high amount of GCs was related with mistakes on the filtering criteria employed, concluding that the actual number of GCs was much lower, about one GC per chromosome and meiosis. Our analysis suggests the occurrence of at least five GCs per chromosome, on average, in peach/almond hybrid meiosis. The high density of markers

allowed by the almond/peach diversity may be the main explanation for the identification of a higher number of GCs, since in other studies the SNP density is commonly much lower. In Arabidopsis, there are approximately 200 DSBs per meiosis (Chelysheva et al. 2007), from which ~10 (5%) result in COs which suggests that the remaining 95% are repaired as NCOs. Our conservative identification of790 GCs in the T1E population (that we estimate correspond to 4.9 GCs/chromosome), and 7.50 COs per meiosis indicates that, if all DSBs were repaired with the homologous chromosome sequence, 19% of the DSBs would result in a CO, and these data closer to the estimation of Chelysheva et al. (2007) in Arabidopsis, particularly considering that we are underestimating the actual number of GCs, because many of them would be undetected due to lack of polymorphism. If these results are general in the plant kingdom, they suggest that the importance of GCs as a source of chromosome reshuffling may be high and that a large number of additional polymorphisms can be incorporated via gene conversion when using wide crosses, with important genetic implications. Knowledge of the extent and distribution of the GCs along the genome and its variability across species would be critical to understand its effects in evolution and to incorporate this source of variability in plant breeding strategies.

5.4.5 - Low CO frequency in the almond ×peach hybrid is associated with reduced pollen viability

In this paper we investigated the recombination landscape for both parents of T1E and found as one of the most relevant differences that there are approximately 57% more COs in the peach parent than in the hybrid parent. This result was expected, since it was previously predicted with the genetic map (Donoso et al. 2015). It is known that at least one CO must occur in every chromosome during meiosis for proper chromosome segregation. However, our results indicate that, for the hybrid, only Pp01 (the larger chromosome) had a CO rate higher than 1.0, with all other chromosomes having less than one CO per meiosis. This suggests that part of the meioses would include chromosomes without a CO event leading to unbalanced gametes that would be unviable. We demonstrated that this was the case for the almond × peach hybrid where almost half of the pollen grains were unviable compared to almost 100% viability of the peach and almond parents.

These results provide information on the key elements of a process gradually leading to reproductive isolation between two species. Accumulation of mutations between peach and almond results in genomic divergence and determines that fewer recombination events take place in their hybrid. This decrease originates from the fact that recombination occurs less frequently in divergent genomic regions as we have shown by the sizes of the CORs, which were higher than expected. A progressively lower opportunity for the production of COs increases the chance of having chromosome pairs that do not go to opposite poles at meiosis, leading to gametes with unbalanced chromosome dosages that in turn result in unviable pollen grains. Based on our estimations, it seems that almond and peach are halfway in the process of developing sexual incompatibility, so an increase of divergence should probably lead to additional unviability and eventually to complete reproductive isolation.

5.4.6 - Supplementary Material

Table S5.1 - Single nucleotide polymorphisms (SNPs) found in the progeny of the cross between MB1.37 (E1T) and 'Earlygold' (E1E2) (in parenthesis medium quality SNPs). E1 vs 'Lovell' are the SNPs found between the E1E1 genotype and the 'Lovell' peach reference genome. E1 vs E2, E1 vs T and E2 vs T are the SNPs found between all alleles present in the T1E population. Peach chromosomes are Pp01-Pp08.

	E1 vs 'Lovell'	**E1 vs E2**	**E1 vs T**	**E2 vs T**
Chromosome	**SNPs**	**SNPs**	**SNPs**	**SNPs**
Pp01	34.769 (36.508)	23.498 (28.488)	261.742 (361.274)	247.156 (358.952)
Pp02	37.197 (40.415)	22.638 (26.843)	143.514 (202.712)	130.991 (199.983)
Pp03	14.341 (15.008)	28.975 (35.675)	139.637 (194.950)	130.115 (193.462)
Pp04	9.036 (9.832)	11.110 (14.126)	129.844 (182.013)	130.890 (184.488)
Pp05	12.107 (12.653)	4.191 (5.042)	108.962 (147.477)	108.858 (146.526)
Pp06	21.049 (22.120)	5.898 (7.822)	154.512 (227.520)	163.763 (228.029)
Pp07	25.270 (26.475)	37.476 (43.948)	119.736 (170.996)	123.010 (172.621)
Pp08	35.817 (37.781)	28.438 (34.583)	113.321 (161.473)	104.396 (149.696)
Total	189.586 (200.792)	162.224 (196.527)	1.171.268 (1.648.415)	1.139.179 (1.633.757)

Table S5.2 - Summary of sequencing depth and alignment depth for all the grandparents ('Texas' and 'Earlygold'), parents (MB1.37 and 'Earlygold') and offspring of the T1E population.

Individual	Sequencing depth	Alignment depth	Coverage
'Earlygold'	67,98	53,54	95,79%
'Texas'	97,69	60,12	86,44%
MB1.37	82,32	57,37	95,52%
T1E2	42,4	32,13	95,57%
T1E3	48,78	36,57	95,48%
T1E4	36,45	26,96	95,36%
T1E5	50,13	37,57	95,18%
T1E6	45,24	34,06	94,94%
T1E7	45,98	33,7	95,11%
T1E8	38,97	29,44	94,60%
T1E10	45,76	34,75	95,32%
T1E11	34,48	25,71	95,11%
T1E13	31,38	23,54	95,31%
T1E14	39,95	30,62	88,48%
T1E15	43,29	32,19	95,66%
T1E16	39,77	28,9	95,49%
T1E17	30,06	21,97	94,94%
T1E19	32,86	23,99	95,26%
T1E20	36,05	26,16	94,99%
T1E21	33,95	24,85	95,15%
T1E22	36,36	26,3	94,66%
T1E23	46,43	35,23	95,61%
T1E24	44,38	34,05	95,43%
T1E25	42,66	31,06	95,34%
T1E26	46,77	34,8	94,99%
T1E27	30,46	22,68	95,42%
T1E28	62,62	47,48	95,80%
T1E30	43,7	31,83	95,10%
T1E31	44,18	32,92	94,93%
T1E32	37,97	27,1	95,22%
T1E33	41,4	30,62	95,28%
T1E34	34,71	25,59	95,27%
T1E35	39,21	28,18	95,07%
T1E36	45,08	33,99	95,32%
T1E37	44,05	33,46	95,34%
T1E38	41,84	30,33	95,40%
T1E40	46,84	34,33	95,15%
T1E42	45,05	34,93	95,14%
T1E43	44,9	33,85	95,41%
T1E44	48,39	35,67	95,22%
T1E45	43,44	32,47	95,01%
T1E46	43,44	31,43	95,37%
T1E47	42,45	31,7	95,01%

Individual	Sequencing depth	Alignment depth	Coverage
T1E82	42,39	32,22	94,80%
T1E87	42,86	33,15	95,17%
T1E89	43,07	33,22	95,17%
T1E90	40,59	29,21	95,29%
T1E91	63,17	45,11	95,61%
T1E92	45,32	34,37	95,28%
T1E93	27,6	20,46	94,66%
T1E96	45,37	33,64	95,18%
T1E97	65,59	50,93	95,79%
T1E98	43,35	31,69	95,33%
T1E102	45,94	34,06	94,95%
T1E107	45,92	33,72	95,14%
T1E110	45,6	34,57	95,21%
T1E117	65,33	48,28	95,53%
T1E137	36,43	26,85	95,34%
T1E139	30,26	22,35	95,14%
T1E143	44,1	33,76	95,40%
T1E148	34,43	25,2	95,26%
T1E152	35,25	26,5	94,88%
T1E153	46,89	35,41	95,57%
T1E155	31,31	23,77	95,23%
T1E183	39,47	29,5	95,08%
T1E189	45,3	34,31	95,42%
T1E193	33,01	25,09	95,02%
T1E197	31,52	23,19	95,41%
T1E198	46,35	35,22	94,95%
T1E199	46,17	34,65	95,89%
T1E201	48,14	36,24	95,84%
T1E220	41,81	30,22	94,90%
T1E226	39,05	28,37	95,34%
T1E241	44,01	32,48	95,51%
T1E248	47,6	35,84	95,11%
T1E282	47,83	35,62	95,40%
T1E287	40,68	30,95	94,94%
T1E304	33,12	24,94	94,62%
T1E306	43,11	33,47	95,06%
T1E333	46,61	33,7	95,53%
T1E343	44,82	33,5	95,42%
T1E364	37,06	26,74	94,77%
T1E410	36,05	26,51	94,94%
T1E412	49,2	37,6	94,99%
T1E417	42,96	32,23	95,07%
T1E418	44,97	33,25	94,93%

Table S5.2 (continued)

Individual	Sequencing depth	Alignment depth	Coverage
T1E49	41,72	31,03	95,07%
T1E50	45,21	33,64	95,28%
T1E51	42,03	32,47	95,27%
T1E55	42,02	31,96	95,24%
T1E56	43,78	33,18	95,34%
T1E57	42,87	30,13	94,87%
T1E62	43,71	32,74	95,24%
T1E63	45,37	33,91	95,37%
T1E64	46,1	33,51	95,48%
T1E65	42,96	32,23	95,35%
T1E66	41,03	29,93	95,03%
T1E67	42,44	32,16	94,99%
T1E68	42,26	31,23	95,45%
T1E69	43,36	32,48	95,49%
T1E70	45	34,29	95,10%
T1E71	43,53	32,69	95,12%
T1E72	42,29	32,09	95,42%
T1E74	42,84	32,18	95,29%
T1E75	41,32	30,25	95,49%
T1E76	38,83	28,09	94,75%
T1E78	38,26	28,17	95,44%
T1E79	44,42	34,08	95,15%
T1E80	36,17	27,21	95,57%

Individual	Sequencing depth	Alignment depth	Coverage
T1E421	43,1	32,86	95,36%
T1E424	44,63	32,88	95,61%
T1E463	46,89	32,93	95,13%
T1E464	41,11	31,4	94,98%
T1E467	48,59	34,87	95,39%
T1E468	43,92	32,13	95,36%
T1E474	49,99	36,6	95,12%
T1E478	44,53	34,27	95,12%
T1E479	43,21	32,93	95,16%
T1E487	47,6	35,06	95,63%
T1E492	28,1	21,28	95,02%
T1E494	42,89	32,68	95,22%
T1E500	51,42	37,49	95,20%
T1E505	43,69	32,87	95,60%
T1E548	43,64	33,28	94,83%
T1E568	42,85	33,15	94,88%
T1E630	42,65	32,72	94,77%
T1E691	45,5	35,11	95,61%
T1E694	46,78	35,96	95,70%
T1E695	46,61	35,72	94,88%
T1E724	46,34	35,26	95,60%
T1E769	49,56	37,98	95,39%

Table S5.3 – Boundaries (in bp) of the peach genome regions covered by genotypes E1E1, E1E2, E1T and E2T from selected subsets of individuals of the T1E progeny [MB1.37 (E1T) × 'Earlygold' (E1E2)] to generate bulked alignment files.

E1E1				E1E2				E1T				E2T			
Chr	**Ind**	**Start**	**End**	**Chr**	**Ind**	**Start**	**End**	**Chr**	**Ind**	**Start**	**End**	**Chr**	**Ind**	**Start**	**End**
Pp01	T1E8	10000000	34000000	Pp01	T1E37	1	47851208	Pp01	T1E89	15000000	47851208	Pp01	T1E16	1	30000000
Pp01	T1E10	1	20000000	Pp01	T1E148	1	47851208	Pp01	T1E287	1	25000000	Pp01	T1E49	35000000	47851208
Pp01	T1E11	32000000	47851208	Pp01	T1E193	1	47851208	Pp01	T1E474	10000000	47851208	Pp01	T1E67	10000000	47851208
Pp01	T1E13	1	47851208	Pp01	T1E306	1	20000000	Pp01	T1E49	1	20000000	Pp01	T1E117	1	20000000
Pp01	T1E17	1	47851208	Pp01	T1E2	15000000	47851208	Pp01	T1E56	15000000	47851208	Pp01	T1E463	25000000	47851208
Pp02	T1E8	1	30405870	Pp02	T1E197	1	30405870	Pp01	T1E65	1	20000000	Pp01	T1E47	10000000	47851208
Pp02	T1E11	1	30405870	Pp02	T1E505	1	30405870	Pp01	T1E220	1	47851208	Pp01	T1E56	1	10000000
Pp02	T1E22	1	28000000	Pp02	T1E97	1	15000000	Pp02	T1E152	1	30405870	Pp02	T1E148	1	30405870
Pp02	T1E76	1	20000000	Pp02	T1E494	1	25000000	Pp02	T1E226	1	30405870	Pp02	T1E492	1	30405870
Pp02	T1E193	15000000	27368013	Pp02	T1E15	10000000	30405870	Pp02	T1E44	1	20000000	Pp02	T1E65	1	30405870
Pp02	T1E42	21000000	27368013	Pp02	T1E79	10000000	30405870	Pp02	T1E56	15000000	30405870	Pp02	T1E72	1	30405870
Pp03	T1E67	1	27368013	Pp03	T1E15	1	27368013	Pp02	T1E63	15000000	30405870	Pp03	T1E42	1	14000000
Pp03	T1E76	1	27368013	Pp03	T1E13	1	27368013	Pp02	T1E137	1	15000000	Pp03	T1E148	1	15000000
Pp03	T1E87	1	27368013	Pp03	T1E80	1	27368013	Pp03	T1E5	1	27368013	Pp03	T1E8	1	15000000
Pp03	T1E467	10000000	27368013	Pp03	T1E417	1	27368013	Pp03	T1E51	1	27368013	Pp03	T1E65	1	18000000
Pp03	T1E548	1	20000000	Pp04	T1E51	1	25843236	Pp03	T1E63	1	27368013	Pp03	T1E343	10000000	20000000
Pp04	T1E11	1	25843236	Pp04	T1E505	1	25843236	Pp03	T1E79	1	27368013	Pp03	T1E20	10000000	20000000
Pp04	T1E22	1	25843236	Pp04	T1E505	1	25843236	Pp04	T1E2	1	25843236	Pp03	T1E474	18000000	27368013
Pp04	T1E306	1	25843236	Pp04	T1E287	1	25843236	Pp04	T1E14	1	25843236	Pp03	T1E4	20000000	27368013
Pp04	T1E42	1	25843236	Pp05	T1E137	1	18496696	Pp04	T1E56	1	25843236	Pp03	T1E17	20000000	27368013
Pp05	T1E8	1	18496696	Pp05	T1E155	1	18496696	Pp04	T1E68	1	25843236	Pp04	T1E197	1	25843236
Pp05	T1E35	1	18496696	Pp05	T1E226	1	18496696	Pp05	T1E4	1	18496696	Pp04	T1E7	1	25843236
Pp05	T1E3	1	18496696	Pp05	T1E92	1	18496696	Pp05	T1E7	1	18496696	Pp04	T1E10	1	25843236
Pp05	T1E492	1	18496696	Pp06	T1E36	1	30767194	Pp05	T1E49	1	18496696	Pp04	T1E62	1	25843236
Pp06	T1E5	1	10000000	Pp06	T1E14	1	30767194	Pp05	T1E117	1	18496696	Pp05	T1E474	1	18496696
Pp06	T1E67	1	30767194	Pp06	T1E487	1	30767194	Pp06	T1E8	1	30767194	Pp05	T1E50	1	18496696
Pp06	T1E97	1	30767194	Pp06	T1E548	1	30767194	Pp06	T1E34	1	30767194	Pp05	T1E2	1	18496696
Pp06	T1E282	1	30767194	Pp07	T1E13	1	22388614	Pp06	T1E137	1	30767194	Pp05	T1E10	1	18496696
Pp06	T1E110	8000000	30767194	Pp07	T1E306	1	22388614	Pp06	T1E3	1	30767194	Pp06	T1E183	1	30767194
Pp07	T1E65	1	22388614	Pp07	T1E47	1	22388614	Pp07	T1E364	1	22388614	Pp06	T1E220	1	30767194
Pp07	T1E492	1	10000000	Pp07	T1E87	1	22388614	Pp07	T1E3	1	22388614	Pp06	T1E364	1	20000000
Pp07	T1E494	8000000	22388614	Pp08	T1E14	1	22573980	Pp07	T1E467	1	15000000	Pp06	T1E467	1	30767194
Pp07	T1E22	1	10000000	Pp08	T1E68	1	22573980	Pp07	T1E14	10000000	22388614	Pp06	T1E193	18000000	30767194
Pp07	T1E42	1	15000000	Pp08	T1E148	1	18000000	Pp07	T1E110	1	10000000	Pp07	T1E487	1	22388614
Pp07	T1E56	14000000	22388614	Pp08	T1E478	1	18000000	Pp07	T1E14	5000000	22388614	Pp07	T1E20	1	22388614
Pp07	T1E412	8000000	22388614	Pp08	T1E79	17000000	22573980	Pp08	T1E152	1	22573980	Pp07	T1E137	1	22388614
Pp08	T1E17	1	22573980	Pp08	T1E226	17000000	22573980	Pp08	T1E155	1	22573980	Pp07	T1E148	1	22388614
Pp08	T1E3	12000000	22573980					Pp08	T1E43	1	22573980	Pp08	T1E11	1	22573980
Pp08	T1E36	1	15000000					Pp08	T1E47	1	22573980	Pp08	T1E137	1	22573980
Pp08	T1E193	1	22573980									Pp08	T1E492	1	22573980
Pp08	T1E548	1	22573980									Pp08	T1E2	1	22573980

Table S5.4 - Summary of global parameters of large alignment files for the four different genotypes of the T1E progeny from the cross MB1.37 (E1T) × 'Earlygold' (E1E2)

	E1E1	E1E2	E1T	E2T
Mapped reads	247,557,610	290,579,836	260,635,283	227,775,931
Read mean length (bp)	99.28	99.61	99.8	99.53
Coverage (mean + sd)	107 ± 276	126 ± 279	112 ± 325	98 ± 204
GC percentage	36.99	37.02	37.04	36.99
Mean mapping quality	37.78	37.75	37.35	37.27

Table S5.5 - Summary of hotspots found in T1E (MB1.37 x 'Earlygold') backcross one population. The location of the hotspots is shown as a window of 10 kbp, in which 2 or 3 crossovers (COs) from different individuals took place. Interspecific hotspots (those that occurred in the meiosis of the almond × peach MB1.37 hybrid) are separated from intraspecific ones. Pp01-Pp08 represent the eight chromosomes of peach.

Interspecific Hotspots			
Chromosome	bin (10kbp)		CO number
Pp01	11,980,000	11,990,000	3
Pp01	18,340,000	18,350,000	2
Pp04	1,290,000	1,300,000	2
Pp04	1,410,000	1,420,000	2
Pp04	2,410,000	2,420,000	2
Pp07	10,640,000	10,650,000	2
Pp08	11,810,000	11,820,000	2
Pp08	15,430,000	15,440,000	2
Intraspecific Hotspots			
Chromosome	bin (10kbp)		CO number
Pp03	23,050,000	23,060,000	3
Pp08	17,110,000	17,120,000	3
Pp01	11,920,000	11,930,000	2
Pp01	16,020,000	16,030,000	2
Pp01	17,550,000	17,560,000	2
Pp01	21,230,000	21,240,000	2
Pp01	23,570,000	23,580,000	2
Pp03	19,510,000	19,520,000	2
Pp04	8,350,000	8,360,000	2
Pp08	10,850,000	10,860,000	2
Pp08	13,830,000	13,840,000	2

Table S5.6 - Average distance between flanking markers estimated at crossover regions in the T1E progeny. Intraspecific recombinations were separated into those falling on regions identical by descent (IBD) in 'Earlygold' and those that were not (no IDB)

RECOMBINATION TYPE[1]	N	Avg. size	N (no IBD)	Avg. Size (no IBD)	N (IBD)	Avg. Size (IBD)
E1E1↔E1E2	246	78,727	159	29,411	87	168,855
E1T↔E2T	275	89,096	183	30,351	92	205,948
Total intraspecific	521	84,200	342	29,906	179	187,919
E1E1↔E1T	222	751	-	-	-	-
E1E2↔E2T	234	610	-	-	-	-
Total interspecific	456	679	-	-	-	-
TOTAL	977	45,217	-	-	-	-

[1]Genotype before and after the crossover event.

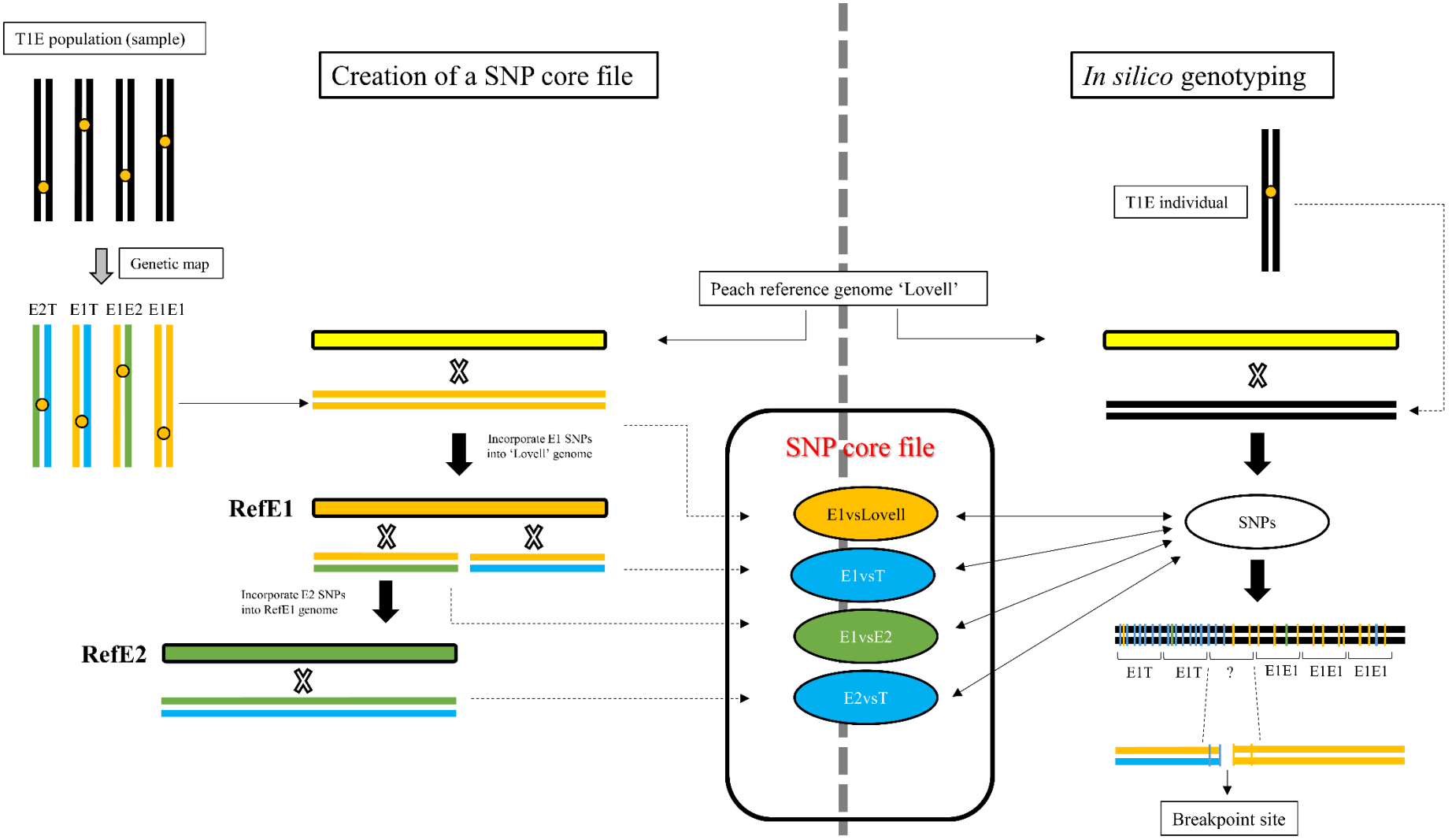


Figure S5.1 – Overview of the genotyping strategy employed in this work to genotype and identify crossover regions in the T1E (MB1.37 × 'Earlygold') progeny.

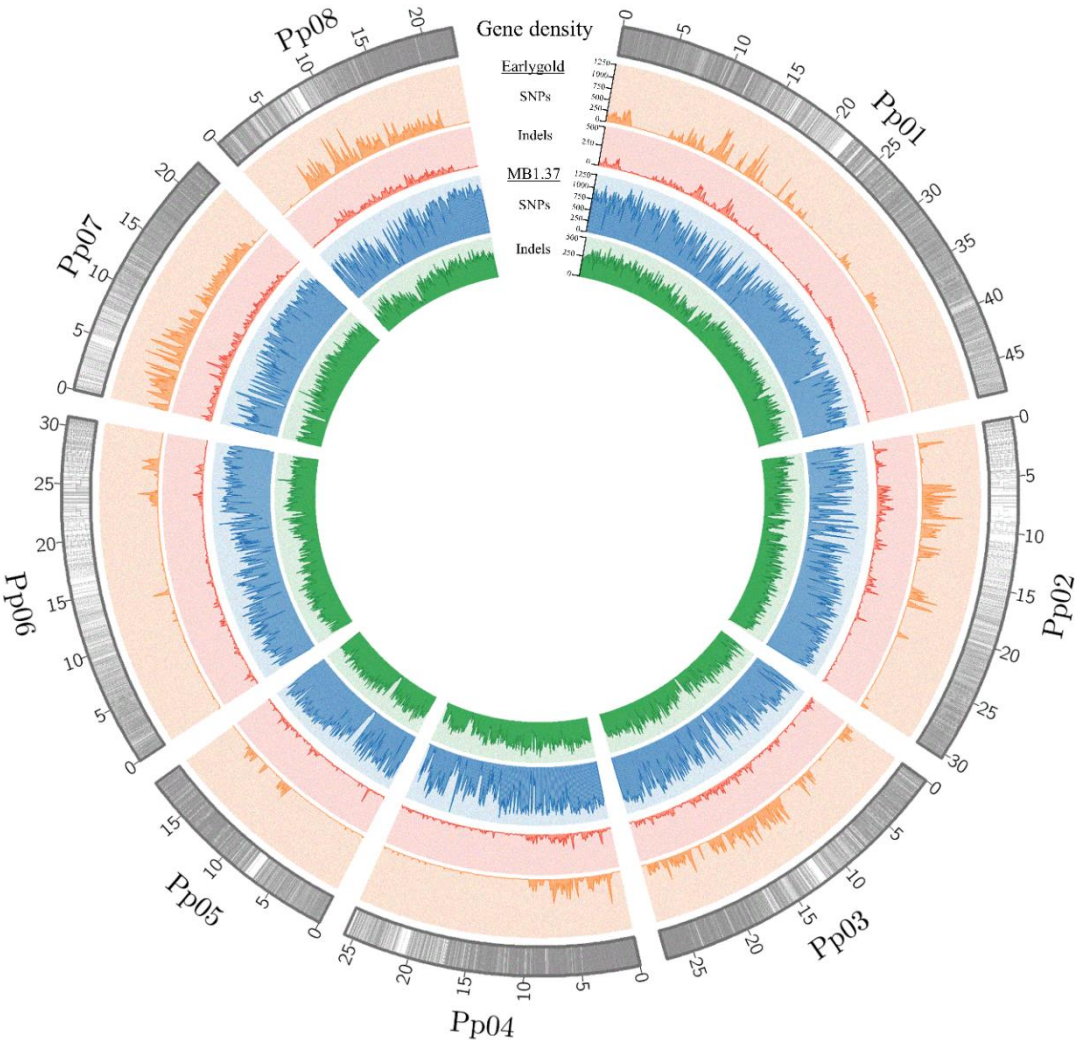


Figure S5.2- Distribution of SNP and indel frequencies in the intra (peach) and interspecific (almond × peach) genome. The values for SNPs and indels are presented considering bins of 100 Kbp. Peach chromosomes are Pp01-Pp08

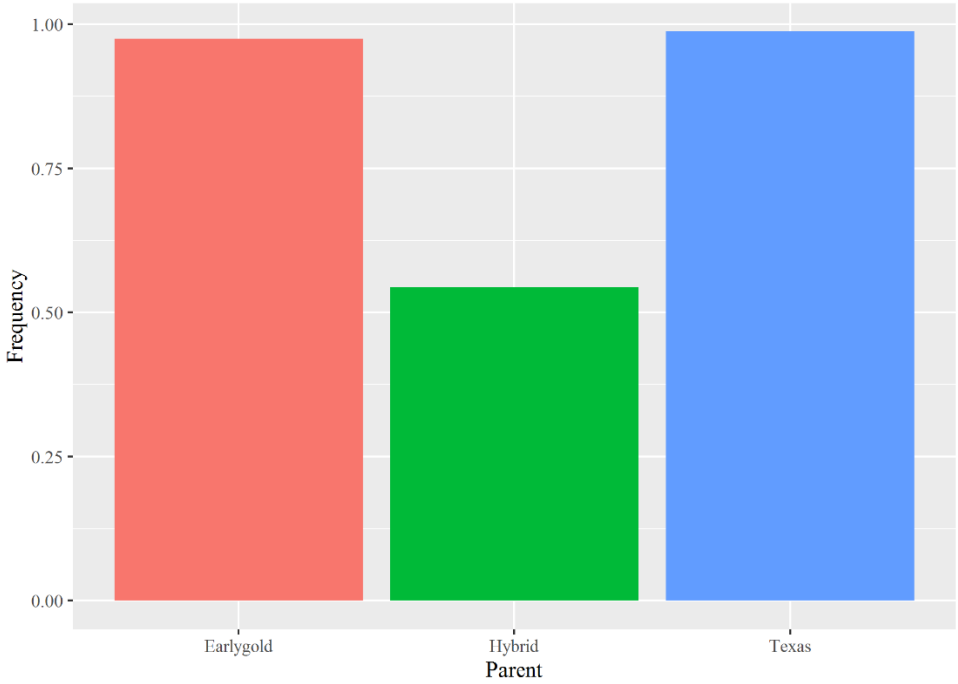


Figure S5.3 - Frequency of viable pollen grains in 'Earlygold', 'Texas' and their hybrid MB1.37.

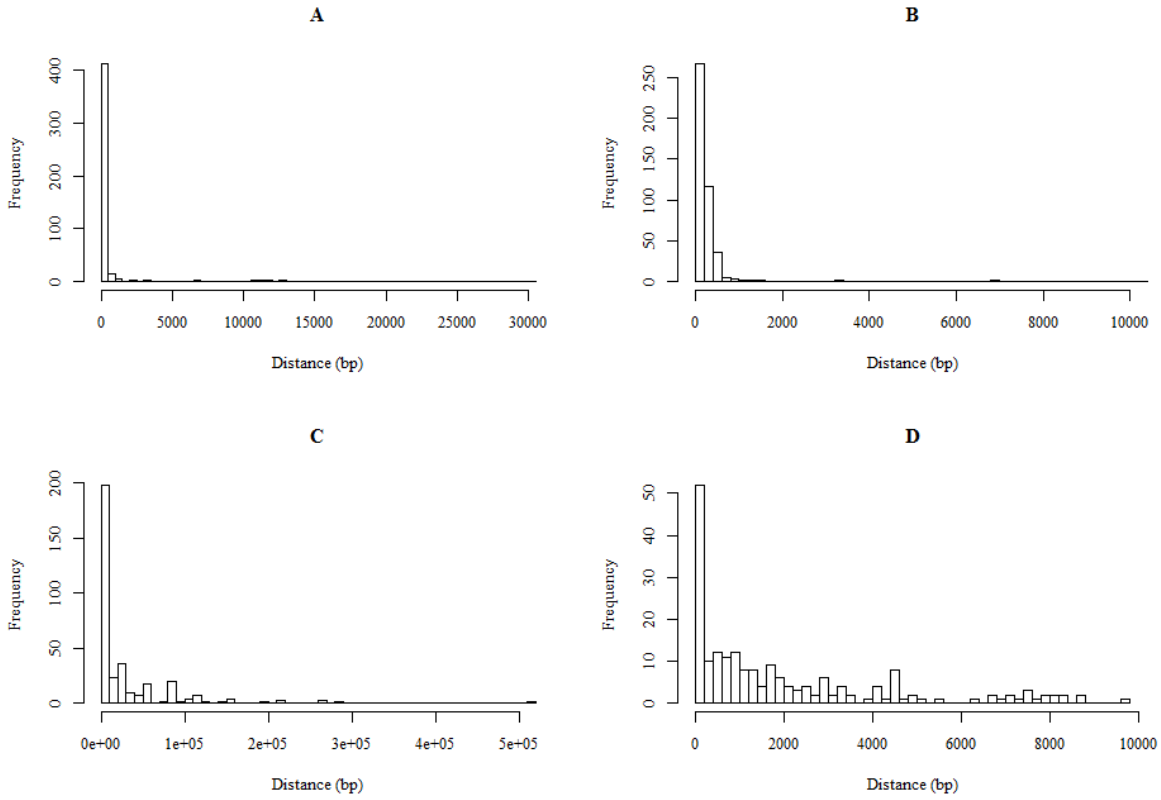


Figure S5.4 - Distribution of the distances between the last SNP marker detected and the first SNP marker absent at the crossover (CO) region. At the top are the distribution for hybrid COs (A) and at the bottom for peach COs (C). Plots B and D represent a zoom-in on the first 10 Kbp of A and C, respectively.

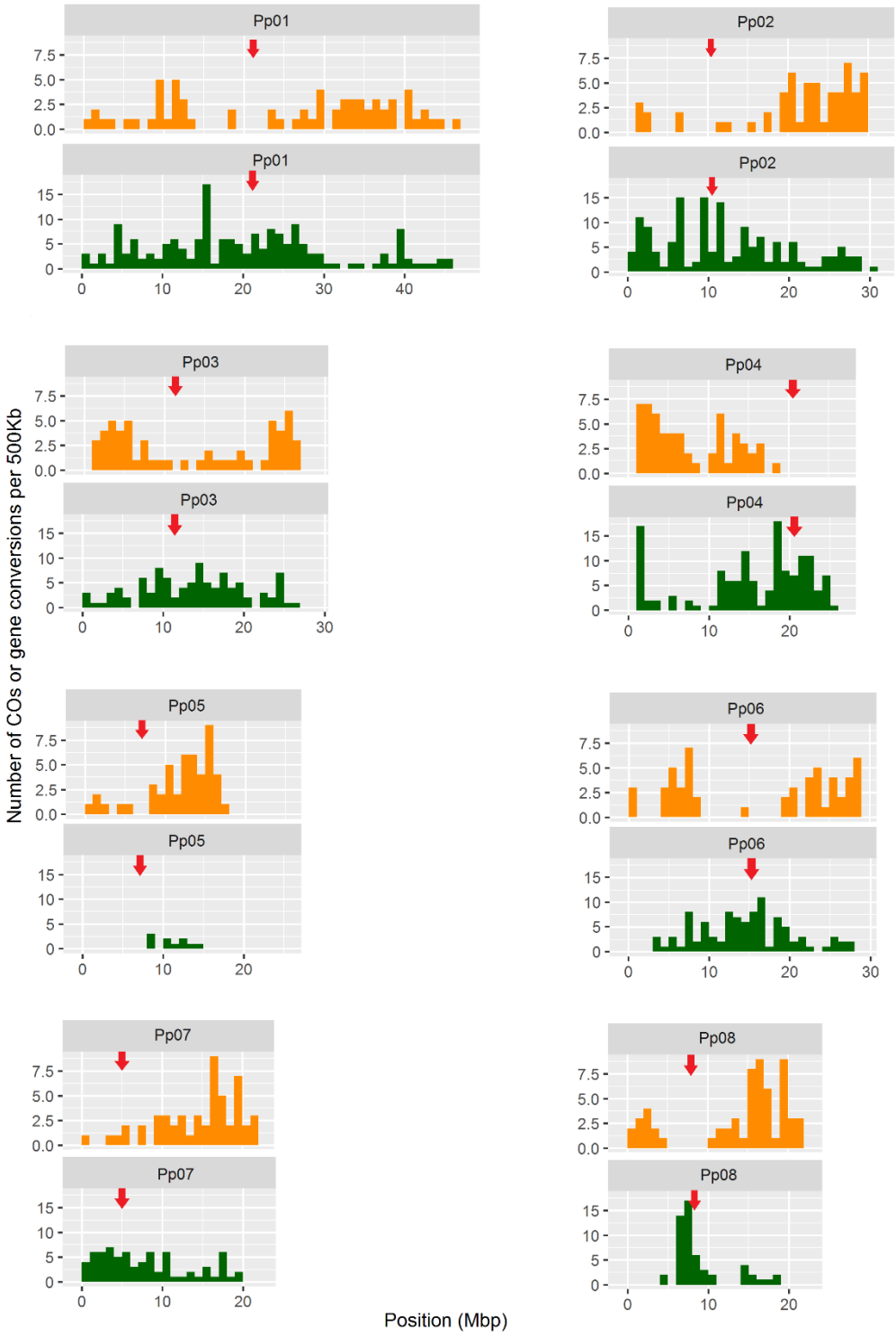


Figure S5.5 - Distribution of gene conversions (green) and crossovers (orange) found for the interspecific almond × peach (MB1.37) meioses for the eight chromosomes of *Prunus* (Pp01-Pp08). Approximate position of centromeres is presented with red arrows.

5.4.7 - Bibliography

Aranzana MJ, El Kadri A., Howad W, Arús P (2010) Genetic variation, population structure and linkage disequilibrium in peach commercial varieties. BMC Genet 11:69

Aranzana MJ, Illa E, Howad W, Arús P (2012) A first insight into peach [*Prunus persica* (L.) Batsch] SNP variability. Tree Genet Genomes 8:1359–1369

Arús P, Yamamoto T, Dirlewanger E, Abbott AG (2005) Synteny in the Rosaceae. Plant Breeding Reviews, volume 27. Jules Janick (ed.) pp. 175-211

Bailey TL, Bodén M, Buske FA, Frith M, Grant CE, Clementi L, Ren J, Li WW, Noble WS (2009) MEME SUITE: tools for motif discovery and searching. Nucleic Acids Research 37:W202-W208

Bayer P, Ruperao P, Mason A, Stiller J, Chan CK, Hayashi S, Long Y, Meng J, Sutton T, Visendi P, Varshney RK, Batley J, Edwards D (2015). High-resolution skim genotyping by sequencing reveals the distribution of crossovers and gene conversions in Cicer arietinum and Brassica napus. Theor appl genet 128:1039–1047

Benson G (1999) Tandem repeats finder: a program to analyze DNA sequences" Nucleic Acids Research 27:573-580

Byrne DH (1990) Isozyme variability in four diploid stone fruits compared with other woody perennial plants. J Hered 81:68–71

Cao K, Zheng Z, Wang L, Liu X, Zhu G, Fang W, Cheng S, Zeng P, Chen C, Wang X, Xie M, Zhong X, Wang X, Zhao P, Bian C, Zhu Y, Zhang J, Ma G, Chen C, Li Y, Hao F, Li Y, Huang G, Li Y, Li H, Guo J, Xu X, Wang J (2014) Comparative population genomics reveals the domestication history of the peach, *Prunus persica*, and human influences on perennial fruit crops. Genome Biol 15:415

Cao K, Zhou ZK, Wang Q, Guo J, Zhao P, Zhu GR, Fang WC, Chen CW, Wang XW, Wang XL, Tian ZX, Wang LR (2016) Genome-wide association study of 12 agronomic traits in peach. Nat Commun 7:13246

Chelysheva L, Diallo S, Vezon D, Gendrot G, Vrielynck N, Belcram K, Rocques N, Márquez-Lema A, Bhatt AM, Horlow C, Mercier R, Mézard C, Grelon M (2005) AtREC8 and AtSCC3 are essential to the monopolar orientation of the kinetochores during meiosis. Journal of Cell Science 118:4621–4632

Chelysheva L, Gendrot G, Vezon D, Doutriaux M- P, Mercier R, Grelon M (2007) ZIP4/SPO22 is required for class I CO formation but not for synapsis completion in *Arabidopsis thaliana*. PLoS Genetics 3:e83

Chin SW, Shaw J , Haberle R , Wen J, Potter D (2014) Diversification of almonds, peaches, plums and cherries - Molecular systematics and biogeographic history of *Prunus* (Rosaceae) Mol Phylogenet Evol 76:34-48

Choi K and Henderson IR (2015) Meiotic recombination hotspots—a comparative view. Plant J 83:52–61

Choi K, Zhao X, Kelly KA, Venn O, Higgins JD, Yelina NE, Hardcastle TJ, Ziolkowski PA, Copenhaver GP, Franklin FCH, McVean G, Henderson IR (2013) Arabidopsis meiotic crossover hot spots overlap with H2A.Z nucleosomes at gene promoters. Nat Gen 45:1327–1336

Cingolani P, Platts A, Wang le L, Coon M, Nguyen T, Wang L, Land SJ, Lu X, Ruden DM (2012) A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3.Fly (Austin) 6:80-92

Danecek P, Auton A, Abecasis G, Albers CA, Banks E, DePristo MA, Handsaker R, Lunter G, Marth G, Sherry ST, McVean G, Durbin R, 1000 Genomes Project Analysis Group (2011) The Variant Call Format and VCFtools. Bioinformatics 27:2156-2158

Das S, Upadhyaya HD, Srivastava R, Bajaj D, Gowda CLL, Sharma S, Singh S, Tyagi AK, Parida SK (2015). Genome-wide insertion-deletion (InDel) marker discovery and genotyping for genomics-assisted breeding applications in chickpea. DNA Research 22:377–386

Datta A, Hendrix M, Lipsitch M, Jinks-Robertson S (1997) Dual roles for DNA sequence identity and the mismatch repair system in the regulation of mitotic crossing-over in yeast. Proc Natl Acad Sci USA 94:9757–9762

de Vicente MC, Tanksley SD (1991) Genome-wide reduction in recombination of backcross progeny derived from male versus female gametes in an interspecific cross of tomato. Theor Appl Genet 83:173-178

Donoso JM, Picañol R, Eduardo I, BatlIe I, Howad W, Aranzana MJ, Arús P (2015) High-density mapping suggests a cytoplasmic male sterility system with two restorer factors in almond x peach progenies. Horticulture Research 2:15016

Donoso JM, Picañol R, Serra O, Howad W, Alegre S, Arús P, Eduardo I (2016) Exploring almond genetic variability useful for peach improvement: mapping major genes and QTLs in two inter-specific almond × peach populations. Mol Breeding 36:16

Dooner HK (2002) Extensive interallelic polymorphisms drive meiotic recombination into a crossover pathway. Plant Cell 14:1173–1183

Dooner HK and He L (2014) Polarized gene conversion at the bz locus of maize. PNAS USA 111:13918–13923

Drouaud J, Khademian H, Giraut L, Zanni V, Bellalou S, Henderson IR, Falque M, Mézard C (2013) Contrasted patterns of crossover and non-crossover at *Arabidopsis thaliana* meiotic recombination hotspots. PLoS Genetics 9:e1003922

Emmanuel E, Yehuda E, Melamed-Bessudo C, Avivi-Ragolsky N, Levy A (2006) The role of AtMSH2 in homologous recombination in *Arabidopsis thaliana*. EMBO Rep 7:101–105

Franklin AE, McElver J, Sunjevaric I, Rothstein R, Bowen B, Cande WZ (1999) Three-dimensional microscopy of the Rad51 recombination protein during meiotic prophase. Plant Cell 11:809–824

Gebhardt C, Ritter E, Barone A, Debener T, Walkemeier B, Schachtschabel U, Kaufmann H, Thompson RD, Bonierbale MW, Ganal MW, Tanksley SD, Salamini F (1991) RFLP maps of potato and their alignment with the homoeologous tomato genome. Theor Appl Genet 83:49-57

Genet 13:20–21

Hayden MJ, Nguyen TM, Waterman A, Chalmers KJ (2008) Multiplex-ready PCR: a new method for multiplexed SSR and SNP genotyping. BMC Genomics 9:80–85

Huang X, Feng Q, Qian Q, Zhao Q, Wang L, Wang A, Guan J, Fan D, Weng Q, Huang T, Dong G, Sang T, Han B (2009) High-throughput genotyping by whole-genome resequencing. Genome Research 19:1068-1076

Kaback DB (1996) Chromosome–size dependent control of meiotic recombination in humans. Nat

Kaback DB, Guacci V, Barber D, Mahon JW (1992) Chromosome size-dependent control of meiotic recombination. Science 256:228–232

Kong A, Gudbjartsson DF, Sainz J, Jonsdottir GM, Gudjonsson SA, Richardsson B, Sigurdardottir S, Barnard J, Hallbeck B, Masson G, Shlien A, Palsson ST, Frigge ML, Thorgeirsson TE, Gulcher JR, Stefansson K (2002) A highresolution recombination map of the human genome. Nat Genet 31:241–247

Lenormand T, Dutheil J (2005) Recombination difference between sexes: A role for haploid selection. PLOS Biol 3:396-403

Li H and Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics 25:1754-60

Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R, 1000 Genome Project Data Processing Subgroup (2009) The sequence alignment/map format and SAMtools. Bioinformatics 25:2078–9

Li XW, Meng XQ, Jia HJ, Yu ML, Ma RJ, Wang LR, Cao K, Shen ZJ, Niu L, Tian JB, Chen MJ, Xie M, Arús P, Gao ZS, Aranzana MJ (2013) Peach genetic resources: diversity, population structure and linkage disequilibrium. BMC Genet 14:84

Lindgreen (2012) AdapterRemoval: Easy Cleaning of Next Generation Sequencing Reads, BMC Research Notes 5:337

Liu G, Xu JH, Zhang M, Li PF, Yao XF, Hou Q, Zhu LL, Ren RS, Yang XP (2016) Exploiting Illumina sequencing for the development of InDel markers in watermelon (Citrullus lanatus). J Hortic Sci Biotechnol 91:220-226

Lu P, Han X, Qi J, Yang J, Wijeratne AJ, Li T, Ma H (2012) Analysis of Arabidopsis genome-wide variations before and after meiosis and meiotic recombination by resequencing Landsberg erecta and all four products of a single meiosis. Genome Research 22:508–18

Lü Y, Cui X, Li R, Huang P, Zong J, Yao D, Li G, Zhang D, Yuan Z (2015) Development of genome-wide insertion/deletion markers in rice based on graphic pipeline platform. J Integr Plant Biol 57:980-991

Luo MC, You FM, Li P, Wang JR, Zhu T, Dandekar AM, Leslie CA, Aradhya M, McGuire PE, Dvorak J (2015) Synteny analysis in rosids with a walnut physical map reveals slow genome evolution in long-lived woody perennials. BMC Genomics 16:1

Marks G (1954) An aceto-carmine glycerol jelly for use in pollen-fertility counts. Biotechnic & Histochemistry 29:277-277

Marschall T, Hajirasouliha I, Schönhuth A (2013) MATE-CLEVER: Mendelian-inheritance-aware discovery and genotyping of midsize and long indels. Bioinformatics 29:3143–3150

Melamed-Bessudo C, Shilo S, Levy, AA (2016) Meiotic recombination and genome evolution in plants. Curr Opin Plant Biol 30:82-87

Mercier R, Mezard C, Jenczewski E, Macaisne N, Grelon M (2015) The molecular biology of meiosis in plants. Annu Rev Plant Biol 66:297–327

Mézard C, Vignard J, Drouaud J, Mercier R (2007) The road to crossovers: plants have their say. Trends in Genetics 23:91–99

Micheletti D, Dettori MT, Micali S, Aramini V, Pacheco I, da Silva C, Foschi S, Banchi E, Barreneche T, Quilot-Turion B, Lambert P, Pascal T, Iglesias I, Carbó J, Wang LR, Ma RJ, Li XW, Gao ZS, Nazzicari N, Troggio M, Bassi B, Rossini L, Verde I, Laurens F, Arús P, Aranzana MJ (2015). Whole-genome analysis of diversity and SNP-major gene association in Peach Germplasm. PLoS ONE 10(9): e0136803

Mnejja M, Garcia-Mas J, Audergon J-M, Arús P (2010) *Prunus* microsatellite transferability across Rosaceae species. Tree genet genomes 6:689–700

Mun JH, Chung H, Chung WH, Oh M, Jeong YM, Kim N, Ahn BO, Park BS, Park S, Lim KB, Hwang YJ, Yu HJ (2015) Construction of a reference genetic map of Raphanus sativus based on genotyping by whole-genome resequencing. Theor appl genet 128:259-272

Myers S, Freeman C, Auton A, Donnelly P, McVean G (2008) A common sequence motif associated with recombination hot spots and genome instability in humans. Nat Gen 40:1124–1129

Pacurar DI, Pacurar ML, Street N, Bussell JD, Pop TI, Gutierrez L, Bellini C (2012) A collection of INDEL markers for map-based cloning in seven Arabidopsis accessions. J. Exp. Bot 63:2491–2501

Pan J, Sasaki M, Kniewel R, Murakami H, Blitzblau HG, Tischfield SE, Zhu X, Neale MJ, Jasin M, Socci ND, Hochwagen A, Keeney S (2011) A hierarchical combination of factors shapes the genome-wide topography of yeast meiotic recombination initiation. Cell 144:719–731

Qi J, Chen Y, Copenhaver GP, Ma H (2014) Detection of genomic variations and DNA polymorphisms and impact on analysis of meiotic recombination and genetic mapping. PNAS USA 111:10007–10012

R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rimmer A, Phan H, Mathieson I, Iqbal Z, Stephen R, Twigg F, WGS500 Consortium, Andrew O, Wilkie M, McVean G, Lunter G (2014) Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. Nat Gen 46:912-8

Rowan BA, Patel V, Weigel D, Schneeberger K (2015) Rapid and inexpensive whole-genome genotyping-by-sequencing for crossover localization and fine-scale genetic mapping. G3 (Bethesda) 5:385–398

Seplyarskiy VB, Logacheva MD, Penin AA, Baranova MA, Leushkin EV, Klepikova AV, Kondrashov FA, Kondrashov AS, James TY (2014) Crossing-Over in a hypervariable species preferentially occurs in regions of high local similarity. Mol Biol Evol 31:3016-3025

Serra O, Donoso JM, Picañol R, Batlle I, Howad W, Eduardo I, Arús P (2016) Marker-assisted introgression (MAI) of almond genes into the peach background: a fast method to mine and integrate novel variation from exotic sources in long intergeneration species. Tree Genet Genomes 12:96

Shilo S, Melamed-Bessudo C, Dorone Y, Barkai N, Levy AA (2015) DNA Crossover Motifs Associated with Epigenetic Modifications Delineate Open Chromatin Regions in Arabidopsis. Plant Cell 27:2427-2436

Si W, Yuan Y, Huang J, Zhang X, Zhang Y, Zhang Y, Tian D, Wang C, Yang Y, Yang S (2015) Widely distributed hot and cold spots in meiotic recombination as shown by the sequencing of rice F2 plants. New Phytol 206:1491-1502

Song X, Wei H, Cheng W, Yang S, Zhao Y, Li X, Luo D, Zhang H, Feng X. (2015) Development of INDEL Markers for Genetic Mapping Based on Whole Genome Resequencing in Soybean. G3 (Bethesda) 5:2793-9

Tanksley SD, Ganal MW, Prince JP, de Vicente MC, Bonierbale MW, Broun P, Fulton TM, Giovannoni JJ, Grandillo S, Martin GB, Messeguer R, Miller JC, Miller L, Paterson AH, Pineda O, Röder MS, Wing RA, Wu W, Young ND (1992) High density molecular linkage maps of the tomato and potato genomes. Genetics 132:1141-1160

Terasawa M, Shinohara A, Hotta Y, Ogawa H, Ogawa T (1995) Localization of RecA-like recombination proteins on chromosomes of the lily at various meiotic stages. Genes and Development 9:925–934

The International Peach Genome Initiative (2013) The high-quality draft genome of peach (*Prunus persica*) identifies unique patterns of genetic diversity, domestication and genome evolution. Nat Gen 45:487-494

Untergasser A, Cutcutache I, Koressaar T, Ye J, Faircloth BC, Remm M, Rozen SG (2012) Primer3--new capabilities and interfaces. Nucleic Acids Research 40:e115

van der Auwera GA, Carneiro M, Hartl C, Poplin R, del Angel G, Levy-Moonshine A, Jordan T, Shakir K, Roazen D, Thibault J, Banks E, Garimella K, Altshuler D, Gabriel S, DePristo M (2013) From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline. Current Protocols In Bioinformatics 43:11.10.1-11.10.33

Velasco D, Hough J, Aradhya M, Ross-Ibarra J (2016) Evolutionary genomics of peach and almond domestication. G3 (Bethesda) 6:3985-3993

Vizir IY, Korol AB (1990) Sex difference in recombination frequency in Arabidopsis. Heredity 65:379-383

Waldman A, Liskay M (1988) Dependence of intrachromosomal recombination in mammalian cells on uninterrupted homology. Mol Cell Biol 8:5350–5357

Wang L, Zhang Y, Qin C, Tian D, Yang S, Hurst LD (2016) Mutation rate analysis via parent – progeny sequencing of the perennial peach. II. No evidence for recombination-associated mutation. P Roy Soc B-Biol Sci 283:20161785

Wickham H (2009) ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York.

Wijnker E, Velikkakam James G, Ding J, Becker F, Klasen JR, Rawat V, Rowan BA, de Jong DF, de Snoo CB, Zapata L, Huettel B, de Jong H, Ossowski S, Weigel D, Koornneef M, Keurentjes JJ, Schneeberger K (2013) The genomic landscape of meiotic crossovers and gene conversions in *Arabidopsis thaliana*. eLife 2:e01426

Xie ZQ, Wang L, Wang LR, Wang ZQ, Lu ZH, Tian DC, Yang SH, Hurst LD (2016) Mutation rate analysis via parent-progeny sequencing of the perennial peach. I. A low rate in woody perennials and a higher mutagenicity in hybrids. P Roy Soc B-Biol Sci 283:20161016

Yang S, Yuan Y, Wang L, Li J, Wang W, Liu H (2012) Great majority of recombination events in Arabidopsis are gene conversion events. PNAS USA 109:20992–20997

Ye K, Schulz MH, Long Q, Apweiler R, Ning Z (2009) Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads. Bioinformatics 25:2865-2871

Zhang B, Wang M, Tang D, Li Y, Xu M, Gu M, Cheng Z, Yu H (2015) XRCC3 is essential for proper double-strand break repair and homologous recombination in rice meiosis. J. Exp. Bot 66:5713–5725

Zhang L and Ma H (2012) Complex evolutionary history and diverse domain organization of SET proteins suggest divergent regulatory interactions. New Phytologist 195:248–263

Zhou GF, Zhang QS, Tan C, Zhang XQ, Li CD (2015) Development of genome-wide InDel markers and their integration with SSR, DArT and SNP markers in single barley map. BMC Genomics 16:804

Ziolkowski PA, Berchowitz LE, Lambing C, Yelina NE, Zhao X, Kelly KA, Choi K, Ziolkowska L, June V, Sanchez-Moran E, Franklin C, Copenhaver GP, Henderson IR (2015) Juxtaposition of heterozygosity and homozygosity during meiosis causes reciprocal crossover remodeling via interference. Elife 4:e0370

6. Main Discussion

This thesis aims to explore the intraspecific and interspecific variability in *Prunus* using both peach × peach and peach × almond segregating populations, in order to develop new tools that may help to improve fruit quality, enrich the peach gene pool and increase overall peach breeding potential. Peach is a genetically impoverished species (Aranzana et al. 2012), the less variable of the cultivated *Prunus* (Byrne 1990; Mnejja et al. 2010). Large homozygous chromosome fragments are commonly found in peach varieties and sometimes these regions may account for more than half of the genome (Donoso et al. 2015; Chapter 4; Chapter 5) These homozygous fragments are likely to be identical by descent (IBD) due to the high kinship among occidental breeding materials (Micheletti et al. 2015). The existence of such regions generates big gaps without polymorphic markers in genetic linkage maps, and determines that no gene located in them will segregate, which has to be taken into account by breeders as it can be a limit to the development of progeny with novel characteristics.

There are three main factors for the limited variability seen in cultivated peach, especially in western countries: (1) The self-pollinating behavior of peach, since it lacks a functional self-incompability system like most others *Prunus*, thus facilitating inbreeding, (2) the long process of westwards migration of peach materials since its domestication in China almost 5,000 years ago, which implied selection for adaptation to specific climatic conditions that may have additionally eroded its gene pool, and (3) the use of a small group of varieties, which shared the same parents, as the founders of the majority of breeding programs in the United States that generated bottleneck effects on the variability of occidental modern varieties. To achieve the major objectives of ongoing breeding programs, including longer shelf life, better fruit quality and resistance to diseases, the gene pool of peach must be enriched (Serra et al. 2016).

One way to bring novelty to cultivated peach is to explore landraces and wild relatives of *P. persica*, since they represent a source of variability for the species (Aranzana et al. 2003b; Li et al. 2013). Another possibility is to consider other cultivated *Prunus*, which are sex compatible with peach, as a source of novel variability. This second option is much more challenging, since the production of interspecific hybrids in a perennial species like peach, and the following backcross generations needed to recover the peach

genomic background, may take several decades when using conventional approaches. However, because of their genetic distance to peach, other *Prunus* are probably the most interesting source of novel genes for disease resistance or fruit quality, among many other traits (Donoso et al. 2016).

In this work we aim to contribute in several ways to improve peach breeding. First we investigated the genetic bases behind a character considered to have a complex genetic basis, the slow-melting fruit flesh (SMF), using two intraspecific populations. Then we developed a model for the introgression of exotic material, using molecular markers, into cultivated peach, which allows the introduction of new and unseen phenotypes from other *Prunus* in a much shorter period of time than using classical backcross approaches. Finally we studied in depth the landscape of recombination events that take place in both peach and almond × peach hybrids, to understand which factors are controlling meiotic recombination, thus improving our knowledge on how to disrupt or keep certain linkages.

6.1 - New genomic tools to improve peach fruit quality

In Chapter 3 of this thesis we investigated the genetic bases of the slow melting flesh (SMF) trait in peach using two F_1 populations. The SMF variety 'Big Top' was used as female parent in both populations, being the male parents 'Armking' or 'Nectaross', both melting flesh (MF) varieties. SMF is a trait difficult to phenotype since, by definition, it is characterized by a slower pace of fruit softening after the field maturity stage. Assessing the firmness of fruits at different time points after harvest seems to be the most reliable and reproducible methodology to classify the SMF trait, where fruits with this phenotype are expected to lose firmness at a slower rate than MF fruits. Indeed, this methodology was proven effective using fruits from the three parents, where firmness loss (FL) for 'Big Top' fruits was lower than both 'Armking' and 'Nectaross'. After accessing the fruit FL for the progeny of both populations for two seasons, we performed a quantitative trait loci (QTL) analysis. Results showed that three genomic regions account for maturity date (MD) QTLs, all of them previously described in other populations: qP-MD4, qP-MD5, and qP-MD6 (Eduardo et al. 2011; Dirlewanger et al. 2012). A single QTL for MD segregated in 'Big Top' (qP-MD5), whereas 'Nectaross' was segregating only for qP-

MD4 and ‘Armking’ for both qP-MD4 and qP-MD6. Two of the QTLs for MD, qP-MD4 and qP-MD5, co-localized consistently with QTLs for FL. Given that qP-MD5 was exclusive to the SMF parent ‘Big Top’, qP-MD4 segregated only in the other (MF) parents, and all FL QTLs for each parent co-mapped with a QTL for MD, it is reasonable to state: (1) There is an evident association between MD and FL, and (2) qP-MD5 is key for the SMF behavior of ‘Big Top’, whereas qP-MD4 alone results in the MF behavior of ‘Armking’ and ‘Nectaross’. On the contrary, no other QTLs were found in the genomic region of qP-MD6 in ‘Armking’.

The relationship between MD and FL was checked when looking for the most favorable genotypes for SMF trait both in the region of qP-MD4 and qP-MD5 (i.e. selecting those individuals with the best allele combinations for SMF). Following this strategy we found that the majority of the best performing individuals for SMF matured late. Two conclusions could be retained from this result. First, the genes that control the maturity time of fruits are probably the same that control their softening after harvest. Second, the production of SMF fruits leads to late maturing varieties. While the first conclusion seems plausible, since there are genes (NAC genes) known to act in overall ripening process within the boundaries of qP-MD4 and qP-MD5 regions, the second conclusion is objectionable, because ‘Big Top’ itself is not a late maturing variety. The presence of SMF fruits in other harvest windows besides late maturing periods may be explained by the existence of other QTLs besides qP-MD4 and qP-MD5 that may anticipate the maturity of fruits without affecting their FL. Indeed, the qP-MD6 of ‘Armking’ affected the MD without any predictable effect over the FL. Furthermore, ‘Armking’ was the earliest variety of the three parents, and the only one having qP-MD6. When selecting for the best performing SMF individuals accounting also for the selection of the early alleles of qP-MD6, we obtained fruits with the same ripening period as ‘Big Top’ and a clear SMF behavior both years, sometimes even better than ‘Big Top’.

The selection of traits such as SMF and their incorporation into breeding programs can contribute strongly to the development of new varieties with improved performance. SMF allows a broader harvest window, since fruits mature very slowly *in planta*, representing a clear advantage to producers. Furthermore, due to their slower firmness loss after

harvest, SMF fruits can be kept for longer periods of time without losing fruit quality or attractiveness, which is appreciated by both retailers and consumers. In order to select early for SMF it is necessary to develop molecular markers associated with the trait. With the work developed in this thesis we have contributed to Mendelize a character that was initially considered of quantitative inheritance, have found three QTLs explaining a large fraction of the SMF and MD variability, and provided possible markers for their selection. Further studies using a broader germplasm collection, including other materials known to be SMF, will be needed to validate our results and to develop if necessary additional markers closer to the causal genes.

6.2 - New DNA-based strategies to facilitate the enrichment of peach gene pool

In the fourth Chapter of this thesis we present a new methodology, marker assisted introgression (MAI), which is useful to introduce novel variability from exotic materials into long intergeneration species. The proof of concept of this model was performed using peach as the recurrent species and almond as the donor (Serra et al. 2016). Assuming that a hybrid line already exists between the two species involved, the entire process can take 9-10 years and has three major phases: (1) The selection of individuals with a low number of introgressions in the BC1, (2) the creation of a set of 15-30 lines with a low number of introgressions (the prIL set), and the analysis of major genes and QTLs in this small set, and (3) the generation of lines with one single introgression from the donor genome.

This model presents several advantages compared with other strategies. First, the use of molecular markers in phase one allows the early selection of a small subset of individuals, with only a few introgression of the donor species, thus reducing the number of plants to subsequent analysis from >1,000 to only two or three tens. This decrease on overall number of plants makes the strategy affordable without losing any information, since the prIL set covers the entire genome with introgressions of the donor parent. Second, the small group of plants of the prIL set permits the genetic analysis of traits of simple inheritance coming from the donor genome, which represents an overview of the variability that may be introgressed into the recipient species. In addition, at this step it is also possible to map certain quantitative traits, without any previous knowledge of the

genetics of the characters being studied. Third, the individuals from the prIL set that present interesting phenotypes are just one generation away of being suitable for use in breeding programs, since they have a small number of introgressions from the donor genome and the recipient genome is already an elite cultivar. Alternatively, or together with breeding-targeted applications, the prIL set is also one generation away to form introgression lines (ILs) that can be generated and kept for future research uses, mainly for genetic studies, generating sub-ILs for fine mapping of genes of interest or simply as a reservoir of variability that can be accessed whenever needed.

This model is a reference for the construction of ILs in peach using other *Prunus*. In our example we used almond as a donor genome, but several hybrids between peach and other species already exist, thus opening other possibilities for MAI. Furthermore this model is applicable to other perennial species, taking into account that the chromosome number or larger intergeneration periods can delay the overall process. If the species of interest has a higher chromosome number than peach, a larger BC_1 population must be developed to allow the selection of a sufficient number of lines with a low number of introgressions. Alternatively, there is the possibility to perform an additional backcross generation, which would delay the overall process, but may be needed for those species like grape (n=*19*) or apple (n=*17*) that would require an impractical population size to find suitable individuals in the BC_1. In addition, other factors like cytoplasmic male sterility as it happened in the almond × peach cross we used, or self incompatibility may also force extra backcrosses or complicate the model. In the scenario of an additional backcross generation, or when applying MAI to higher recombination rate crosses than almond x peach, small sized introgressions can be frequent. For these situations a higher marker density must be used in order to accurately detect all the introgressions. To do it, other genotyping strategies than SSRs can be used, mainly SNP chips, which in certain cases could represent a faster, cheaper and more reliable way to genotype large populations. Furthermore, these genotyping strategies permit the application of MAI at an early stage and only the selected plants can be moved from the greenhouse to the field making the process less space and labor consuming. Since the prIL set is small, it requires very little space and cost to be maintained, making possible the production of several IL collections from distinct crosses, allowing for a first survey of the potential of different lines of the same donor species or of different species. MAI can contribute to accelerate the

introgression of novelty from genetically distant sources into cultivated species, by providing a necessary intermediate step that facilitates the use of valuable alleles existing in the currently locked gene pool of related *Prunus* species into commercial peach breeding.

As a side result of applying MAI in our (almond × peach) × peach cross, we developed already several ILs with a single introgression in heterozygosis at BC_2. Taken together, all these ILs cover 64% of peach genome. A second IL collection with almond introgressions in homozygosis, which already covers 14% of peach genome, is also being constructed. Based on the BC_2 progeny developed during this thesis, we estimate that these collections may be finished in the next two years. Both collections will be very useful, since they will allow the comparison of single dosage vs double copy of almond genes, facilitate the study of complex traits and permit the fine mapping of almond genes by the development of sub-ILs for further genetic studies. To our knowledge, these will be the first IL collections developed for a tree species.

6.3 - The study of the recombinational landscape in peach and almond × peach

In this thesis we studied the landscape of recombination in both peach and almond × peach scenarios. We developed a strategy to map crossovers (COs) *in silico* using the resequence data of 125 individuals from T1E population. Since both parents of T1E are heterozygous (the partially heterozygous peach cultivar 'Earlygold' and the interspecific almond × peach hybrid MB1.37), four different genotypes exist across the progeny, which is a more complex situation than that of previous similar works with resequence data.

The study of the variability for the parents of T1E have shown two completely different landscapes. The SNP polymorphisms observed in 'Earlygold' were clustered in specific heterozygous regions, which accounted for less than half of its genome. The remaining portion of 'Earlygold' genome (54%) was highly homozygous, a characteristic previously observed in this parent and other modern cultivars that evidences the presence of chromosome fragments identical by descent (Donoso et al. 2015). On the other hand the variability observed in the almond × peach hybrid was much higher than that of 'Earlygold' and was uniformly distributed along chromosomes. These results translate

into one SNP every 1,399 bp between ‘Earlygold’ alleles and one SNP every 194 bp between the almond and peach allele present in the MB1.37. The difference in the levels of variability (~7×) reflects the major genetic distance between almond and peach, and is much higher than the diversity between peach and peach × *Prunus davidiana* hybrid (4.4×) evidenced recently (Wang et al. 2016), corroborating the good choice of almond as variability donor for peach (Serra et al. 2016).

After mapping a total of 1,081 COs events along the progeny, we found that the peach parent accumulated more COs than the hybrid line. In ‘Earlygold’ we estimated the mean number of COs per meiosis to be 11.79 whereas in almond × peach it was much lower (7.50). The CO frequency for ‘Earlygold’ is in agreement with previous knowledge from the genetic map (Donoso et al. 2015) and with the results obtained recently for peach species by Wang et al. (2016). Our results, together with Wang’s data, evidenced that (1) recombination rates within peach are very similar (2) peach has a low level of recombination when compared with other plant species, and (3) the CO rate in interspecific hybrids decreases with the increase of sequence divergence. Despite the differences observed in CO rate, intraspecific and interspecific COs distributed similarly, overall with lower CO numbers in pericentromeric regions in both cases. A deeper study on the crossover regions (CORs) allowed us to estimate the mean size of interspecific CORs in 678 bp and intraspecific CORs in almost 30 Kbp. These results show that COs take place in regions with less variability than expected, since the predictable COR size would be 388 bp for interspecific COs and 1,248 bp for intraspecific COs, considering an even distribution of SNP markers (2× and 24× shorter than observed, respectively). The difference between the intra and interspecific observed COR sizes can be explained considering that the probability of CO occurrence is: (1) more likely when a minimal homozygous DNA fragment exists, which determines the small, but higher than expected COR length in the hybrid, and (2) is inversely correlated with the abundance of polymorphism, thus determining that in ‘Earlygold’, where large homozygous regions exist, COs take place in much longer homozygous regions than expected.

To elucidate the existence of DNA sequences that could be associated with CO presence we looked for the enrichment of DNA motifs in CORs smaller than 5 Kbp. Two motifs, the poly-A and the dinucleotide $(CT)_n$ were found significantly enriched in both

‘Earlygold’ and MB1.37 CORs, both of them similar to previous DNA motifs identified in Arabidopsis CO sites. We also found the DNA motif CCN enriched in interspecific COs, the same found in human hotspots of recombination (Myers et al. 2008) that has been also identified in plants (Shilo et al. 2015). The role of these motifs is not yet understood, but it is evident that these marks are conserved over different species at CO sites. In this work we have also provided evidence for the existence of recombination hotspots, which were detected at a lower rate than expected by data from other species. Hotspots occurred in similar numbers for both hybrid and peach meioses. The lower number of hotspots may be explained because we used only a subset of all COs available (the ones with COR lower than 10 Kbp) to look for these regions, based on the definition of hotspots as regions up to 10Kbp in size (Choi and Henderson, 2015). Besides the study of CO events, we also analyzed the presence of gene conversions (GCs) produced during the meiosis of the hybrid MB1.37. Our results show that GCs are at least five times more frequent than COs. The study of GCs was performed in other plants like maize, Arabidopsis, rapeseed and chickpea (Dooner and He, 2014; Wijnker et al. 2013; Qi et al. 2014; Bayer et al. 2015) and for the majority of cases their rate was estimated in less than one GC per chromosome. The higher GC rate detected here when comparing with other works may be related with the higher density and uniform distribution of markers in our data, which permits the detection of small sized GCs that would not be detectable with lower SNP densities or with irregular distributions of polymorphisms. The relevance of GCs for breeding should not be ignored, since they could allow certain desired phenotypes without the drawbacks of linkage drag associated with the introgression of chromosome fragments. This idea is reinforced by the fact that in our data 6% of all almond SNPs fell in exons, theoretically generating a disruptive effect on proteins for a total of 2,480 genes. Gene conversions may then have a more important role than initially considered as a source of genetic diversity, in addition to other essential sources, such as mutation and recombination.

6.4 - Future perspectives

Peach breeding progress is threatened by the low variability existing in modern cultivars. New sources of variability are needed as well as new strategies that may help the efficient incorporation of such variability in the cultivated peach gene pool. The peach breeding

of tomorrow must consider the combination of new and conventional tools developed in the past as a way to leverage the development of new and more adapted varieties to the environmental and commercial challenges. To generate faster, cheaper and more accurate breeding processes, new technologies must be included. The experience acquired during this thesis leads to the proposal of a new breeding strategies for peach that contemplate such new technologies (Figure 6.1). First, new markers must be developed, mainly for early selection of complex traits such as SMF in peach × peach segregating progenies. Due to the routinary use of whole genome sequencing, such markers are currently more efficiently selected using resequence data and the sequence of the peach genome. Second, the introgression of new variability coming from other *Prunus* will be feasible due to the application of the MAI model proposed in this thesis. This will strongly contribute to an increase in the available variability for the species, which will be reflected into resistances for several peach diseases, new organoleptic traits and overall increase in fruit quality and diversity. Third, the understanding of the factors governing recombination (COs and NCOs) may have a tremendous impact for breeding, through the development of strategies allowing increasing or decreasing recombination rates at the whole genome level where needed, or by enhancing recombination in regions where CO events are rare, allowing for the generation of new genetic combinations.

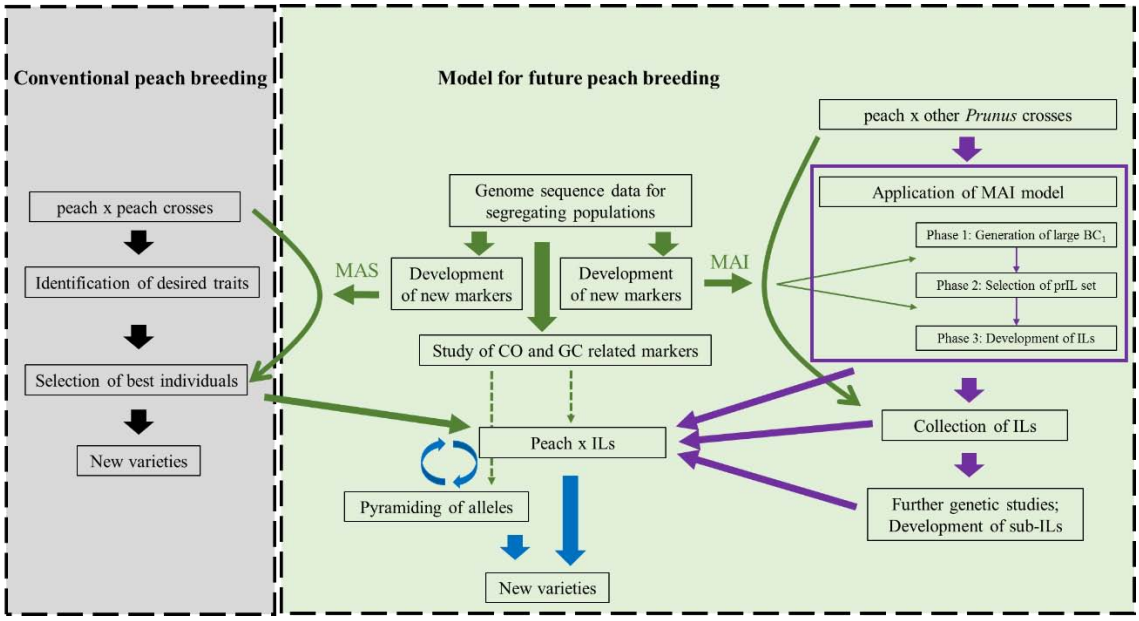


Figure 6.1 - Model for future peach breeding based on the experience and results obtained in this thesis.

7. Conclusions

1- The assessment of fruit firmness loss after harvest during two independent seasons has proven to be an effective method to measure the slow melting flesh (SMF) phenotype in the segregating populations 'Bigtop' × 'Armking' and 'Bigtop' × Nectaross'.

2- The QTLs affecting the firmness loss (FL) of fruits are also responsible for controlling the maturity date (MD) trait. The SMF phenotype seen in 'Big Top' was mainly explained by the joint action of two QTLs, one for fruit firmness loss on chromosome five, exclusive of 'Big Top' and the other on chromosome four, present in melting flesh (MF) cultivars 'Armking' and 'Nectaross'.

3- A collection of 18 pre-introgression lines (prILs) with 2-4 almond introgressions, and together containing several times the almond genome, was selected in phase one of the marker-assisted introgression (MAI) model with a small set of molecular markers in a large BC1 population.

4- The prIL set allowed to map all major genes tested to their expected locations, as well as certain quantitative traits, demonstrating the appropriateness of prILs for a preliminary detection of genes provided by the donor parent during the application of MAI.

5- A second backcross allowed the generation of 137 individuals with a single almond introgression (ILs), 109 with the introgression in heterozygosis and 28 in homozygosis, covering 64% and 16% of peach genome, respectively. Both IL collections are expected to be finished in the next two years.

6- The marker assisted introgression (MAI) model proposed in this thesis has demonstrated that the incorporation of foreign material into cultivated perennial species is feasible in a short period of time. Our proof of concept using almond as donor genome and peach as recipient species can be adopted for the introgression of genes from other *Prunus* into peach as well as for introgressing exotic materials in other species than peach.

7- A new *in silico* strategy to genotype the T1E population, where both parents are heterozygous and four different genotypes exist across the progeny, was developed using SNP information obtained from resequencing data.

8- The resequencing of the T1E population and its ancestors revealed important features of peach and almond genomes. The fact that 99% of 'Earlygold' variability (160,721 SNPs) was restricted to only 46% (103.8 Mbp) of its genome uncovered with high precision which regions were identical by descent in this cultivar. This irregular pattern of distribution of variability contrasted with the uniform and constant high level of polymorphisms detected in the almond × peach hybrid MB1.37.

9- The *in silico* genotyping strategy developed allowed the detection of 1,081 crossovers (COs) along 125 individuals of the T1E population. The peach intraspecific mean number of COs per meiosis was similar to other peach studies (11.79) and much higher than in the interspecific almond × peach hybrid (7.50).

10- DNA motifs $(CT)_n$, poly-A and $(CCN)_n$ where the most significantly associated with CO occurrence, all of them previously associated with plant CO position in other species.

11- Nineteen small genomic regions (10 Kbp each), 11 in 'Earlygold' and eight in MB1.37, were classified as hotspots of recombination since they were targeted in at least two independent meiosis in the population surveyed.

12- The study of gene conversion (GC) events in the meiosis of the almond × peach hybrid MB1.37 led to an estimation of almost five GCs per chromosome, which is about five times higher than previous reports in other plant species.

13- The crossover regions (CORs) in both 'Earlygold' and MB1.37 were larger than what was expected by the density of polymorphisms available, meaning that COs tend to occur mainly in homozygous regions in both peach and almond × peach.

14- The low mean number of COs observed in the hybrid MB1.37 (less than one CO per chromosome), together with the 46% pollen unviability observed in this line, suggests that the absence of sequence similarity characteristic of genetically distant genomes tends to prevent CO formation leading to unviable gametes as a consequence of unbalanced chromosome composition.

8. Main Bibliography

Abbott A, Georgi L, Yvergniaux D, Wang Y, Blenda A, Reighard G, Inigo M and Sosinski B (2002) Peach, the model genome for Rosaceae. Acta Hortic 575:145–155

Aranzana M, Abbassi E-K, Howad W, Arús P (2010) Genetic variation, population structure and linkage disequilibrium in peach commercial varieties. BMC Genet 11:69

Aranzana MJ, Carbó J, Arús P (2003a) Using amplified fragment length polymorphisms (AFLPs) to identify peach cultivars. J Am Soc Hortic Sci 128:672–677

Aranzana MJ, Carbó J, Arús P (2003b) Microsatellite variability in peach [*Prunus persica* (L.) Batsch]: cultivar identification, marker mutation, pedigree inferences and population structure. Theor Appl Genet 106:1341–1352

Aranzana MJ, Eduardo I, Vilanova S, Romero C, Martín-Hernández AM (2012). Genomics of Temperate Fruit Trees. In: Schnell RJ, Priyadarshan PM (Eds) *Genomics of Tree Crops*, Springer, New York, pp 155-208

Arús P, Ballester J, Jáuregui B, Joobeur T, Truco M and Carmen de Vicente M (1998) The European *Prunus* mapping project: update on marker development in almond. Acta Hortic 484:331–338

Arus P, Verde I, Sosinski B, Zhebentyayeva T and Abbott AG (2012) The peach genome. Tree Genet Genomes 8: 531-547

Arús P, Yamamoto T, Dirlewanger E and Abbott AG (2005) Synteny in the Rosaceae. In: Janick J (ed) Plant breeding reviews, vol 27, Wiley, Hoboken, pp. 175–211

Bai Y, Huang CC, van der Hulst R, Meijer-Dekens F, Bonnema G, Lindhout P (2003) QTLs for tomato powdery mildew resistance (Oidium lycopersici) in Lycopersicon parviflorum G1.1601 co-localize with two qualitative powdery mildew resistance genes. Molecular Plant-Microbe Interactions 16:169–176

Baird WV, Estager AS and Wells JK (1994) Estimating nuclear DNA content in peach and related diploid species using laser flow cytometry and DNA hybridization. J Am Soc Hortic Sci 119:1312–1316

Ballester J, Socias i Company R, Arús P and de Vicente MC (2001) Genetic mapping of a major gene delaying blooming time in almond. Plant Breeding 120:268–270

Bassi D and Monet R (2008). Botany and taxonomy. In: Layne DR and Bassi D (Eds) The Peach: Botany, Production and Uses. CAB International, Wallingford, pp 1–36

Bayer P, Ruperao P, Mason A, Stiller J, Chan CK, Hayashi S, Long Y, Meng J, Sutton T, Visendi P, Varshney RK, Batley J and Edwards D (2015). High-resolution skim genotyping by sequencing reveals the distribution of crossovers and gene conversions in Cicer arietinum and Brassica napus. Theor Appl Genet 128:1039–1047

Berchowitz LE and Copenhaver GP (2010) Genetic interference: don't stand so close to me. Current Genomics 11:91–102

Berchowitz LE, Francis KE, Bey AL and Copenhaver GP (2007) The role of AtMUS81 in interference-insensitive crossovers in *A. thaliana*. PLOS Genetics 3:e132

Berg IL, Neumann R, Lam KW, Sarbajna S, Odenthal-Hesse L, May CA and Jeffreys AJ (2010) PRDM9 variation strongly influences recombination hot-spot activity and meiotic instability in humans. Nat Genet 42:859–863

Bicknell RA and Koltunow AM (2004) Understanding apomixis: Recent advances and remaining conundrums. Plant Cell 16:S228–S245

Bielenberg D, Gasic K and Chaparro JX (2009) Genetics and genomics of Rosaceae, plant genetics and genomics: crops and models. Springer, Berlin, pp 223–234

Blake MA (1937) Progress in peach breeding. P Am Soc Hortic SCI 35:49–53

Bliss FA, Arulsekar S, Foolad MR, Becerra V, Gillen AM, Warburton ML, Dandekar AM, Kocsisne GM and Mydin KK (2002) An expanded genetic linkage map of *Prunus* based on an interspecific cross between almond and peach. Genome 45:520–529

Börner GV, Kleckner N and Hunter N (2004) Crossover/noncrossover differentiation, synaptonemal complex formation, and regulatory surveillance at the leptotene/zygotene transition of meiosis. Cell 117:29–45

Bretó MP, Cantín CM, Iglesias I, Arús P and Eduardo I (2017) Mapping a major gene for red skin color suppression (highlighter) in peach. Euphytica 213:14

Byrne D, Raseira M, Bassi D, Piagnani M, Gasic K, Reighard G, Moreno M and Pérez S (2012) Peach. In: Badenes ML, Byrne DH, (Eds) Fruit Breeding. New York: Springer Verlag pp 505–569

Byrne DH (1990) Isozyme variability in four diploid stone fruits compared with other woody perennial plants. J Hered 81:68–71

Byrne DH (2005) Trends in stone fruit cultivar development. Horticultural Technology 15:494–500

Byrne DH (2007) Molecular marker use in perennial plant breeding. Acta Hortic 751:163–168

Byrne DH, Sherman WB and Bacon TA (2000) Stone Fruit Genetic Pool and Its Exploitation for Growing under Warm Winter Conditions. In Erez A (Ed) Temperate Fruit Crops in Warm Climates, Springer, Netherlands pp 157–230

Cao K, Zheng Z, Wang L, Liu X, Zhu G, Fang W, Cheng S, Zeng P, Chen C, Wang X, Xie M, Zhong X, Wang X, Zhao P, Bian C, Zhu Y, Zhang J, Ma G, Chen C, Li Y, Hao F, Li Y, Huang G, Li Y, Li H, Guo J, Xu X and Wang J (2014) Comparative population genomics reveals the domestication history of the peach, *Prunus persica*, and human influences on perennial fruit crops. Genome Biol 15:415

Carrillo-Mendoza O, Chaparro JX and Williamson J (2013) Branching and blind node incidence in interspecific backcross families of peach. HortScience 48:1119–1124

Chagné D, Crowhurst RN, Pindo M, Thrimawithana A, Deng C, Ireland H, Fiers M, Dzierzon H, Cestaro A, Fontana P, Bianco L, Lu A, Storey R, Knäbel M, Saeed M, Montanari S, Kim YK, Nicolini D, Larger S, Stefani E, Allan AC, Bowen J, Harvey I, Johnston J, Malnoy M, Troggio M, Perchepied L, Sawyer G, Wiedow C, Won K, Viola R, Hellens RP, Brewer L, Bus VG, Schaffer RJ, Gardiner SE, Velasco R (2014) The draft genome sequence of European pear (Pyrus communis L. 'Bartlett'). PLoS ONE 9:e92644

Chaparro JX, Werner DJ, O'Malley D and Sederoff RR (1994) Targeted mapping and linkage analysis of morphological isozyme, and RAPD markers in peach. Theor Appl Genet 87:805–815

Chin S-W, Shaw J, Haberle R, Wen J and Potter D (2014) Diversification of almonds, peaches, plums and cherries – Molecular systematics and biogeographic history of *Prunus* (Rosaceae). Mol. Phylogenet. Evol 76:34–48

Choi K and Henderson IR (2015) Meiotic recombination hotspots—a comparative view. Plant J 83:52–61

Claverie M, Bosselut N, Lecouls AC, Voisin R, Lafargue B, Poizat C, Kleinhentz M, Laigret F, Dirlewanger E and Esmenjaud D (2004) Location of independent root-knot nematode resistance genes in plum and peach. Theor Appl Genet 108:765–773

Connors CH (1922) Peach breeding. A summary of results. Proceedings of the American Society for Horticultural Science 19:108–115

Copenhaver GP, Housworth EA and Stahl FW (2002) Crossover interference in Arabidopsis. Genetics 160:1631–1639

Crismani W, Girard C and Mercier R (2013) Tinkering with meiosis. J. Exp. Bot 64:55-65

Crismani W, Girard C, Froger N, Pradillo M, Santos JL, Chelysheva L, Copenhaver GP, Horlow C and Mercier R (2012) FANCM limits meiotic crossovers. Science 336:1588–1590

De Candolle A (1885) Origin of Cultivated Plants. D Appleton and Company, New York, pp36-38

De Muyt A, Mercier R, Mézard C and Grelon M (2009) Meiotic recombination and crossovers in plants.Genome Dynamics 5:14–25

Dettori MT, Quarta R and Verde I (2001) A peach linkage map integrating RFLPs, SSRs, RAPDs, and morphological markers. Genome 44:783–790

Dirks RK, van Dun CB, de Snoo M, Vandenberg CL, Lelivelt C, Voermans W, Woudenberg L, de Wit JPC, Reinink K, Schut JW, van der Zeeuw E, Vogelaar A, Freymark G, Gutteling EW, Keppe MN, van Drongelen P, Kieny M, Ellu P, Touraev A, Ma H, de Jong H and Wijnker E (2009): Reverse breeding: a novel breeding approach based on engineered meiosis. Plant Biotechnol J 7:837-845

Dirlewanger E, Cosson P, Boudehri K, Renaud C, Capdeville G, Tauzin Y, Laigret F and Moing A (2006) Development of a second generation genetic linkage map for peach [*Prunus persica* (L.) Batsch] and characterization of morphological traits affecting flower and fruit. Tree Genet Genomes 3:1–13

Dirlewanger E, Graziano E, Joobeur T, Garriga-Caldere F, Cosson P, Howard W and Arús P (2004) Comparative mapping and marker assisted selection in Rosaceae fruit crops. PNAS USA 101:9891-9896

Dirlewanger E, Moing A, Rothan C, Svanella L, Pronier V, Guye A, Plomion C and Monet R (1999) Mapping QTL controlling fruit quality in peach (*Prunus persica* (L.) Batsch Theor Appl Genet 98:18–31

Dirlewanger E, Pascal T, Zuger C and Kervella J (1996) Analysis of molecular markers associated with powdery mildew resistance genes in peach (*Prunus persica* (L.) Batsch) × *Prunus* davidiana hybrids. Theor Appl Genet 93:909–919

Dirlewanger E, Pronier V, Parvery C, Rothan C, Guye A, Monet R (1998) Genetic linkage map of peach [*Prunus persica* (L.) Batsch] using morphological and molecular markers. Theor Appl Genet 97:888–895

Dirlewanger E, Quero-García J, Le Dantec L, Lambert P, Ruiz D, Dondini L, Illa E, Quilot-Turion B, Audergon JM, Tartarini S, Letourmy P, Arús P (2012) Comparison of the genetic determinism of two key phenological traits, flowering and maturity dates, in three *Prunus* species: peach, apricot and sweet cherry. Heredity 109:280-292

Donoso JM, Picañol R, Eduardo I, BatlIe I, Howad W, Aranzana MJ, Arús P (2015) High-density mapping suggests a cytoplasmic male sterility system with two restorer factors in almond x peach progenies. Horticulture Research 2:15016

Donoso JM, Picañol R, Serra O, Howad W, Alegre S, Arús P and Eduardo I (2016) Exploring almond genetic variability useful for peach improvement: mapping major genes and QTLs in two inter-specific almond × peach populations. Mol Breed 36:16

Dooner HK and He L (2014) Polarized gene conversion at the bz locus of maize. PNAS USA 111:13918–13923

Duval H, Hoerter M, Polidori J, Confolent C, Masse M, Moretti A, Ghelder CV and Esmenjaud D (2014) High-resolution mapping of the RMia gene for resistance to root-knot nematodes in peach. Tree Genet Genomes 10:297–306

Eduardo I, Arus P, Monforte AJ (2005) Development of a genomic library of near isogenic lines (NILs) in melon (Cucumis melo L.) from the exotic accession PI161375. Theor Appl Genet 112:139-148

Eduardo I, Pacheco I, Chietera G, Bassi D, Pozzi C, Vecchietti A and Rossini L (2011) QTL analysis of fruit quality traits in two peach intraspecific populations and importance of maturity date pleiotropic effect. Tree Genet Genomes 7:323–335

Eduardo I, Picañol R, Rojas E, Batlle I, Howad W, Aranzana MJ and Arús P (2015) Mapping of a major gene for the slow ripening character in peach: co-location with the maturity date gene and development of a candidate gene-based diagnostic marker for its selection. Euphytica 205:627–636

Eshed Y and Zamir D (1994) A genomic library of *Lycopersicon pennellii* in *L. esculentum*: a tool for fine mapping of genes. Euphytica 79:175-179

Eshed Y and Zamir D (1995) An introgression-line population of Lycopersicon pennelli in the cultivated tomato enables the identification and fine mapping of yield-associated QTLs. Genetics 141:1147–1162

Esmenjaud D, Voisin R, Van Ghelder C, Bosselut N, Lafargue B, Di Vito M, Dirlewanger E, Poessel JL and Kleinhentz M (2009) Genetic dissection of resistance to root-knot nematodes Meloidogyne spp. in plum, peach, almond, and apricot from various segregating interspecific *Prunus* progenies. Tree Genet and Genomes 5:279–289

Etienne C, Rothan C, Moing A, Plomion C, Bodenes C, Dumas LS, Cosson P, Pronier V, Monet R and Dirlewanger E (2002) Candidate genes and QTL for sugar and organic acid content in peach (*Prunus persica* (L.) Batsch). Theor Appl Genet 105:145–159

Fan S, Bielenberg DG, Zhebentyayeva TN, Reighard GL, Okie WR, Holland D and Abbott AG (2010) Mapping quantitative trait loci associated with chilling requirement, heat requirement and bloom date in peach (*Prunus persica*). New Phytologist 185:917–930

FAOSTAT (2017) Food and Agriculture Organization of the United Nations. FAOSTAT Statistics Database. URL: http://www.fao.org/faostat/en/#home. (Accessed 25/02/2017)

Faust M and Timon B (1995) Origin and dissemination of peach. Horticultural Reviews 17:331–379.

Fernández i Martí A, Alonso JM, Espiau MT, Rubio-CabetasMJ, and Socias i Company R (2009) Genetic diversity in Spanish and foreign almond germplasm assessed by molecular characterization with SSRs. J Am Soc Hortic Sci 134:535–542

Foss EJ and Stahl FW (1995) A test of a counting model for chiasma interference. Genetics. 139:1201–1209

Foulongne M, Pascal T, Pfeiffer F and Kervella J (2003) QTLs for powdery mildew resistance in peach x *Prunus davidiana* crosses: consistency across generations and environments. Mol Breed 12:33–50

Francis KE, Lam SY, Harrison BD, Bey AL, Berchowitz LE and Copenhaver GP (2007) Pollen tetrad-based visual assay for meiotic recombination in Arabidopsis. PNAS USA 104:3913–3918

Frary A, Doganlar S, Frampton A, Fulton TM, Uhlig J, Yates HE and Tanksley SD (2003) Fine mapping of quantitative trait loci for improved fruit characteristics from Lycopersicon chmielewskii chromosome 1. Genome 46:235–243

Fulton TM, Grandillo S, Beck-Bunn T, Fridman E, Frampton A, Lopez J, Petriard V, Uhlig J, Zamir D and Tanksley SD (2000) Advanced backcross QTL analysis of a Lycopersicon esculentum × Lycopersicon parviflorum cross. Theor Appl Genet 100:1025–1042

Garcia-Mas J, Benjak A, Sanseverino W, Bourgeois M, Mir G, González VM, Hénaff E, Câmara F, Cozzuto L, Lowy E, Alioto T, Capella-Gutiérrez S, Blanca J, Cañizares J, Ziarsolo P, Gonzalez-Ibeas D, Rodríguez-Moreno L, Droege M, Du L, Alvarez-Tejado M, Lorente-Galdos B, Melé M, Yang L, Weng Y, Navarro A, Marques-Bonet T, Aranda MA, Nuez F, Picó B, Gabaldón T, Roma G, Guigó R, Casacuberta JM, Arús P, Puigdomènech P (2012) The genome of melon (Cucumis melo L.). PNAS USA 109:11872-11877

Ghiani A, Onelli E, Aina R, Cocucci M, Citterio S (2011) A comparative study of melting and non-melting flesh peach cultivars reveals that during fruit ripening endo-polygalacturonase (endo-PG) is mainly involved in pericarp textural changes, not in firmness reduction. J Exp Bot 62:4043–4054

Gómez-Aparisi J, Carrera M, Felipe AJ and Socias i Company R (2001) 'Garnem', 'Monegro' and 'Felinem': new almond × peach hybrid rootstocks, nematode resistant and red leaved for stone fruits. Informacion Tecnica Economica Agraria 97:282–288

Gradziel TM (2003) Interspecific hybridizations and subsequent gene introgression within *Prunus* subgenus *Amygdalus*. Acta Hortic 622:249–255

Grandillo S, Tanksley SD and Zamir D (2007) Exploitation of natural biodiversity through genomics. In: Varshney RK and Tuberosa R (Eds) Genomics assisted crop improvement: vol I genomics approaches and platforms. Springer, Dordrecht, pp. 121–150

Griffiths P and Scott J (2001) Inheritance and linkage of tomato mottle virus resistance genes derived from Lycopersicon chilense accession LA 1932. J Am Soc Hortic Sci 126:462–467

Haji T, Yaegaki H and Yamaguchi M (2005) Inheritance and expression of fruit texture melting, non-melting and stony hard in peach. Sci Hort 105:241–248

Hancock JF, Scorza R and Lobos GA (2008). Peaches. In: Hancock JF (Ed.) *Temperate fruit crop breeding*, Springer, Netherlands, pp 265-298

Higgins JD, Buckling EF, Franklin FCH and Jones GH (2008) Expression and functional analysis of AtMUS81 in Arabidopsis meiosis reveals a role in the second pathway of crossing-over. Plant J 54:152–162

Housworth EA and Stahl FW (2003) Crossover interference in humans. The American Journal of Human Genetics 73:188–197

Huang H, Cheng Z, Zhang Z and Wang Y (2008) History of cultivation and trends in China. In: Layne DR

and Bassi D (Eds) The Peach: Botany, Production and Uses.Wallingford: CABI. pp 37–60

Hummer KE and Janick J (2009) Rosaceae: taxonomy, economic importance, genomics. In: Folta KM and Gardiner SE (Eds) Genetics and Genomics of Rosaceae, Springer, New York, pp 1-17

Hunter N and Kleckner N (2001) The single-end invasion: an asymmetric intermediate at the double-strand break to double-holliday junction transition of meiotic recombination. Cell 106:59–70

Hurtado MA, Romero C, Vilanova S, Abbott AG, Llácer G and Badenes ML (2002) Genetic linkage maps of two apricot cultivars (*Prunus armeniaca* L.), and mapping of PPV (sharka) resistance. Theor Appl Genet 105:182–191

Iglesias I (2013) Peach production in Spain: Current Situation and Trend, from Production to Consumption. In: Proceedings of the 4th Conference, Innovation in Fruit Growing, Belgrade, 75-98

Illa E, Sargent DJ, Lopez Girona E, Bushakra J, Cestaro A, Crowhurst R, Pindo M, Cabrera A, van der Knaap E and Iezzoni A (2011) Comparative analysis of rosaceous genomes and the reconstruction of a putative ancestral genome for the family. BMC Evol Biol11:9-10

Jaillon O, Aury JM, Noel B, Policriti A, Clepet C, Casagrande A, Choisne N, Aubourg S, Vitulo N, Jubin C, Vezzi A, Legeai F, Hugueney P, Dasilva C, Horner D, Mica E, Jublot D, Poulain J, Bruyère C, Billault A, Segurens B, Gouyvenoux M, Ugarte E, Cattonaro F, Anthouard V, Vico V, Del Fabbro C, Alaux M, Di Gaspero G, Dumas V, Felice N, Paillard S, Juman I, Moroldo M, Scalabrin S, Canaguier A, Le Clainche I, Malacrida G, Durand E, Pesole G, Laucou V, Chatelet P, Merdinoglu D, Delledonne M, Pezzotti M, Lecharny A, Scarpelli C, Artiguenave F, Pè ME, Valle G, Morgante M, Caboche M, Adam-Blondon AF, Weissenbach J, Quétier F, Wincker P, French-Italian Public Consortium for Grapevine Genome Characterization (2007) The grapevine genome sequence suggests ancestral hexaploidization in major angiosperm phyla. Nature 449:463–467

Jáuregui B (1998) Identification of molecular markers linked to agronomic characters in an interspecific almond × peach progeny. University of Barcelona, Spain

Jeuken MJW and Lindhout P (2004) The development of lettuce backcross inbred lines (BILs) for exploitation of the Lactuca saligna (wild lettuce) germplasm. Theor Appl Genet 109:394–401

Joobeur T, Viruel MA, de Vicente MC, Jauregui B, Ballester J, Dettori MT, Verde I, Truco MJ, Messeguer R, Batlle I, Quarta R, Dirlewanger E and Arus P (1998) Construction of a saturated linkage map for *Prunus* using an almond × peach F-2 progeny. Theor Appl Genet 97:1034–1041

Keeney S, Giroux CN and Kleckner N (1997) Meiosis-specific DNA double-strand breaks are catalyzed by Spo11, a member of a widely conserved protein family. Cell 88:375–384

King JS and Mortimer RS (1990) A polymerization model of chiasma interference and corresponding computer simulation. Genetics 126:1127–1138

Kleckner N, Zickler D, Jones GH, Dekker J, Padmore R, Henle J and Hutchinson J (2004) A mechanical basis for chromosome function. PNAS USA 101:12592–7

Konopacka D, Jesionkowska K, Kruczynska D, Stehr R, Schoorl E, Buehler A, Eddger S, Codarin S, Hilaire C, H6l1er I, Guerra w, Uverani A, Donati E, Sansavini S, Martinelli A, Petiot C, Carbó J, Echeverria G, Iglesias I and Bonany J (2010) Apple and peach consumption habits across European countries. Appetite 55:478-483

Kurzbauer M-T, Uanschou C, Chen D and Schlögelhofer P (2012) The recombinases DMC1 and RAD51 are functionally and spatially separated during meiosis in Arabidopsis. Plant Cell 24:2058–70

Lambert P, Campoy JA, Pacheco I, Mauroux J-B, Da Silva Linge C, Micheletti D, Bassi D, Rossini L, Dirlewanger E, Pascal T, Troggio M, Aranzana MJ, Patocchi A and Arús P (2016) Identifying SNP markers tightly associated with six major genes in peach [*Prunus persica* (L.) Batsch] using a high-density SNP array with an objective of marker-assisted selection (MAS). Tree Genet Genomes 12:121

Li XW, Meng XQ, Jia HJ, Yu ML, Ma RJ, Wang LR, Cao K, Shen ZJ, Niu L, Tian JB, Chen MJ, Xie M, Arús P, Gao ZS and Aranzana MJ (2013) Peach genetic resources: diversity, population structure and linkage disequilibrium. BMC Genet 14:84

Llácer G, Alonso JM, Rubio MJ, Batlle I, Iglesias I, Vargas FJ, García-Brunton J and Badenes ML (2009).Situación del material vegetal de melocotonero utilizado en España. Informacion Tecnica

Economica Agraria 195:67–83

Lu ZX, Reighard GL, Nyczepir AP, Beckman TG and Ramming DW (1998) Inheritance of resistance to root-knot nematodes in peach rootstocks. Acta Hortic 465:111–116

Malik SB, Ramesh MA, Hulstrand AM and Logsdon JM Jr. (2007) Protist homologs of the meiotic Spo11 gene and topoisomerase VI reveal an evolutionary history of gene duplication and lineage-specific loss. Mol Biol Evol 24:2827–2841

Martinez-Garcia PJ, Parfitt DE, Bostock RM, Fresnedo-Ramirez J, Vazquez-Lobo A, Ogundiwin EA, Gradziel TM and Crisosto CH (2013) Application of genomic and quantitative genetic tools to identify candidate resistance genes for brown rot resistance in peach. PLoS One 8:12

Martínez-Gómez P, Rubio M, Dicenta F and Gradziel TM (2004) Resistance to Plum Pox Virus (RB3.30 isolate) in a group of California almonds and transfer of resistance to peach. J Am Soc Hortic Sci 129:544–548

Martini E, Diaz RL, Hunter N and Keeney S (2006) Crossover homeostasis in yeast meiosis. Cell 126:285–295

McNeill J, Barrie FR, Buck WR, Demoulin V, Greuter W, Hawksworth DL, Herendeen PS, Knapp S, Marhold K, Prado J, Prud'homme van Reine WF, Smith GF, Wiersema JH, Turland NJ (2012) International code of nomenclature for algae, fungi and plants (Melbourne code) adopted by the 18th International Botanical Congress Melbourne, Australia, July 2011

Meneses C, Ulloa-Zepeda L, Cifuentes-Esquivel A, Infante R, Cantin CM, Batlle I, Arús P and Eduardo I (2016) A codominant diagnostic marker for the slow ripening trait in peach. Mol Breed 36:77

Mercier R, Mezard C, Jenczewski E, Macaisne N and Grelon M (2015) The molecular biology of meiosis in plants. Annu Rev Plant Biol 66:297–327

Micheletti D, Dettori MT, Micali S, Aramini V, Pacheco I, da Silva C, Foschi S, Banchi E, Barreneche T, Quilot-Turion B, Lambert P, Pascal T, Iglesias I, Carbó J, Wang LR, Ma RJ, Li XW, Gao ZS, Nazzicari N, Troggio M, Bassi B, Rossini L, Verde I, Laurens F, Arús P, Aranzana MJ (2015). Whole-genome analysis of diversity and SNP-major gene association in Peach Germplasm. PLoS ONE 10(9): e0136803

Ming R1, Hou S, Feng Y, Yu Q, Dionne-Laporte A, Saw JH, Senin P, Wang W, Ly BV, Lewis KL, Salzberg SL, Feng L, Jones MR, Skelton RL, Murray JE, Chen C, Qian W, Shen J, Du P, Eustice M, Tong E, Tang H, Lyons E, Paull RE, Michael TP, Wall K, Rice DW, Albert H, Wang ML, Zhu YJ, Schatz M, Nagarajan N, Acob RA, Guan P, Blas A, Wai CM, Ackerman CM, Ren Y, Liu C, Wang J, Wang J, Na JK, Shakirov EV, Haas B, Thimmapuram J, Nelson D, Wang X, Bowers JE, Gschwend AR, Delcher AL, Singh R, Suzuki JY, Tripathi S, Neupane K, Wei H, Irikura B, Paidi M, Jiang N, Zhang W, Presting G, Windsor A, Navajas-Pérez R, Torres MJ, Feltus FA, Porter B, Li Y, Burroughs AM, Luo MC, Liu L, Christopher DA, Mount SM, Moore PH, Sugimura T, Jiang J, Schuler MA, Friedman V, Mitchell-Olds T, Shippen DE, dePamphilis CW, Palmer JD, Freeling M, Paterson AH, Gonsalves D, Wang L, Alam M (2008) The draft genome of the transgenic tropical fruit tree papaya (Carica papaya Linnaeus). Nature 452:991-996

Mnejja M, Garcia-Mas J, Audergon J-M and Arús P (2010) *Prunus* microsatellite transferability across Rosaceae species. Tree Genet Genomes 6:689–700

Myers S, Freeman C, Auton A, Donnelly P and McVean G (2008) A common sequence motif associated with recombination hot spots and genome instability in humans. Nat Genet 40:1124–1129

Ogundiwin EA, Peace CP, Gradziel TM, Parfitt DE, Bliss FA and Cristosto CH (2009) A fruit quality gene map of *Prunus*. BMC Genom 10:587–600

Okie WR and Scorza R (2002) Breeding peach for narrow leaf width. Acta Hortic 592:137–141

Olmstead JW, Sebolt AM, Cabrera A, Sooriyapathirana SH, Iriarte G, Wang D, Chen CY, Knaap van der E and Iezzoni AF (2008) Construction of an intra-specific sweet cherry (*Prunus persica* L.) genetic linkage map and synteny analysis with the *Prunus* reference map. Tree Genet Genomes 4897-910

Osman K, Higgins JD, Sanchez-Moran E, Armstrong SJ and Franklin FC (2011) Pathways to meiotic recombination in *Arabidopsis thaliana*. New Phytologist 190:523–544

Paigen K and Petkov P (2010) Mammalian recombination hot spots: properties, control and evolution.

Nature Reviews Genetics 11:221–33

Pascal T, Pfeiffer F and Kervella J (2010) Powdery mildew resistance in the peach cultivar Pamirskij 5 is genetically linked with the Gr gene for leaf color. Hortscience 45:150–152

Potter D, Eriksson T, Evens RC, Oh S, Smedmark JEE, Morgan DR, Kerr M, Robertson KR, Arsenault M, Dickinson TA and Campbell CS (2007) Phylogeny and classification of Rosaceae. Plant Syst Evol 266:5-43

Pozzi C and Vecchietti A (2009) Peach structural genomics. In: Folta KM and Gardiner SE (Eds) *Genetics and genomics of Rosaceae*, Springer, Nueva York, pp 235-257

Qi J, Chen Y, Copenhaver GP and Ma H (2014) Detection of genomic variations and DNA polymorphisms and impact on analysis of meiotic recombination and genetic mapping. PNAS USA 111:10007–10012

Quilot B, Wu BH, Kervella J, Génard M, Foulongne M and Moreau K (2004) QTL analysis of quality traits in an advanced backcross between *Prunus persica* cultivars and the wild relative species *P. davidiana*. Theor Appl Genet 109:884–897

Rajapakse S, Belthoff LE, He G, Estager AE, Scorza R, Verde I, Ballard RE, Baird WV, Callahan A, Monet R and Abbott AG (1995) Genetic linkage mapping in peach using morphological, RFLP and RAPD markers. Theor Appl Genet 90:503-510

Rehder A (1940) A manual of cultivated trees and shrubs hardy in North America exclusive of the subtropical and warmer temperate regions, 2nd ed. Macmillan, New York, New York, USA

Ru S, Main D, Evans K and Peace C (2015) Current applications, challenges, and perspectives of marker-assisted seedling selection in Rosaceae tree fruit breeding. Tree Genet Genomes 11:8

Rubio M, Pascal T, Bachellez A and Lambert P (2010) Quantitative trait loci analysis of Plum pox virus resistance in *Prunus davidiana* P1908: new insights on the organization of genomic resistance regions. Tree Genet Genomes 6:291–304

Sanchez-Moran E, Santos J-L, Jones GH, Franklin FCH (2007) ASY1 mediates AtDMC1-dependent interhomolog recombination during meiosis in Arabidopsis. Genes & Development 21:2220–33

Sansavini S, Gamberini A, Bassi D (2006) Peach breeding, genetics and new cultivar trends. Acta Hortic 713:23-48

Sauge MH, Lambert P and Pascal T (2012) Co-localisation of host plant resistance QTLs affecting the performance and feeding behaviour of the aphid *Myzus persicae* in the peach tree. Heredity 108:292–301

Schmutz J, Cannon SB, Schlueter J, Ma J, Mitros T, Nelson W, Hyten DL, Song Q, Thelen JJ, Cheng J, Xu D, Hellsten U, May GD, Yu Y, Sakurai T, Umezawa T, Bhattacharyya MK, Sandhu D, Valliyodan B, Lindquist E, Peto M, Grant D, Shu S, Goodstein D, Barry K, Futrell-Griggs M, Abernathy B, Du J, Tian Z, Zhu L, Gill N, Joshi T, Libault M, Sethuraman A, Zhang XC, Shinozaki K, Nguyen HT, Wing RA, Cregan P, Specht J, Grimwood J, Rokhsar D, Stacey G, Shoemaker RC, Jackson SA (2010) Genome sequence of the palaeopolyploid soybean. Nature 463:178-183

Scorza R and Okie WR (1991) Peaches (*Prunus*). In: Moore JN and Ballington RB (Eds) Genetic Resources of Template Fruit and Nut Crops. Acta Hortic 290:177–234

Scorza R, Mehlenbacher SA and Lightner GW (1985) Inbreeding and coancestry of freestone peach cultivars of the Eastern United States and implications for peach germplasm improvement. J Am Soc Hortic Sci 110:547–552

Scorza R, Melnicenco L, Dang P and Abbott AG (2002) Testing a microsatellite marker for selection of columnar growth habit in peach (*Prunus persica* (L.) Batsch). Acta Hortic 592:285–289

Scorza R, Miller S, Glenn DM, Okie WR and Tworkoski T (2006) Developing peach cultivars with novel growth habits. Acta Hortic 713:61–64

Scott D and Cullinan F (1942) The inheritance of wavy leaf character in the peach. J Hered 33:293–295

Serra O, Donoso JM, Picañol R, Batlle I, Howad W, Eduardo I and Arús P (2016) Marker-assisted introgression (MAI) of almond genes into the peach background: a fast method to mine and integrate novel variation from exotic sources in long intergeneration species. Tree Genet Genomes

12:96

Shen Z, Confolent C, Lambert P, Poëssel J-L, Quilot-Turion B, Yu M, Ma R and Pascal T (2013) Characterization and genetic mapping of a new blood-flesh trait controlled by the single dominant locus DBF in peach. Tree Genet Genomes 9:1435–1446

Shilo S, Melamed-Bessudo C, Dorone Y, Barkai N and Levy AA (2015) DNA Crossover Motifs Associated with Epigenetic Modifications Delineate Open Chromatin Regions in Arabidopsis. Plant Cell 27:2427-2436

Shulaev V, Korban SS, Sosinski B, Abbott AG, Aldwinckle HS, Folta KM, Iezzoni A, Main D, Arús P, Dandekar AM, Lewers K, Brown SK, Davis TM, Gardiner SE, Potter D, Veilleux RE (2008) Multiple models for Rosaceae genomics. Plant Phys. 147:985-1003

Shulaev V, Sargent DJ, Crowhurst RN, Mockler TC, Folkerts O, Delcher AL, Jaiswal P, Mockaitis K, Liston A, Mane SP, Burns P, Davis TM, Slovin JP, Bassil N, Hellens RP, Evans C, Harkins T, Kodira C, Desany B, Crasta OR, Jensen RV, Allan AC, Michael TP, Setubal JC, Celton JM, Rees DJ, Williams KP, Holt SH, Ruiz Rojas JJ, Chatterjee M, Liu B, Silva H, Meisel L, Adato A, Filichkin SA, Troggio M, Viola R, Ashman TL, Wang H, Dharmawardhana P, Elser J, Raja R, Priest HD, Bryant DW Jr, Fox SE, Givan SA, Wilhelm LJ, Naithani S, Christoffels A, Salama DY, Carter J, Lopez Girona E, Zdepski A, Wang W, Kerstetter RA, Schwab W, Korban SS, Davik J, Monfort A, Denoyes-Rothan B, Arus P, Mittler R, Flinn B, Aharoni A, Bennetzen JL, Salzberg SL, Dickerman AW, Velasco R, Borodovsky M, Veilleux RE, Folta KM (2011) The genome of woodland strawberry (*Fragaria vesca*). Nat Genet 43:109-116

Su T, Wilf P, Huang Y, Zhang S and Zhou Z (2015) Peaches preceded humans: Fossil evidence from SW China. Scientific Reports 5:16794–16794

Szalma S, Hostert B, LeDeaux J, Stuber C, Holland J (2007) QTL mapping with near-isogenic lines in maize. Theor Appl Genet 114:1211–1228

Tanksley SD and Nelson JC (1996) Advanced backcross QTL analysis:a method for the simultaneous discovery and transfer of valuable QTLs from unadapted germplasm into elite breeding lines. Theor Appl Genet 92:191–203

Tanksley SD, McCouch SR (1997) Seed banks and molecular maps: unlocking genetic potential from the wild. Science 277:1063-1066

The Arabidopsis Genome initiative (2000) Analysis of the genome sequence of the flowering plant *Arabidopsis thaliana*. Nature 408:796–815

The International Peach Genome Initiative (2013) The high-quality draft genome of peach (*Prunus persica*) identifies unique patterns of genetic diversity, domestication and genome evolution. Nat Genet 45:487-494

The international Rice Genome Sequencing Project (2005) The map-based sequence of the rice genome. Nature 436:793–800

Tsuruta M and Mukai Y (2015) Hybrid seedling inviability locus (HIs1) mapped on linkage group 4 of the Japanese flowering cherry, *Cerasus* × *yedoensis* 'Somei-yoshino'. Tree Genet Genomes 11:1–11

Urrutia M, Bonet J, Arús P, Monfort A (2015) A near-isogenic line (NIL) collection in diploid strawberry and its use in the genetic analysis of morphologic, phenological and nutritional characters. Theor Appl Genet 128:1261–1275

Velasco D, Hough J, Aradhya M and Ross-Ibarra J (2016) Evolutionary genomics of peach and almond domestication. G3 (Bethesda) g3.116.032672

Velasco R, Zharkikh A, Affourtit J, Dhingra A, Cestaro A, Kalyanaraman A, Fontana P, Bhatnagar SK, Troggio M, Pruss D, Salvi S, Pindo M, Baldi P, Castelletti S, Cavaiuolo M, Coppola G, Costa F, Cova V, Dal Ri A, Goremykin V, Komjanc M, Longhi S, Magnago P, Malacarne G, Malnoy M, Micheletti D, Moretto M, Perazzolli M, Si-Ammour A, Vezzulli S, Zini E, Eldredge G, Fitzgerald LM, Gutin N, Lanchbury J, Macalma T, Mitchell JT, Reid J, Wardell B, Kodira C, Chen Z, Desany B, Niazi F, Palmer M, Koepke T, Jiwan D, Schaeffer S, Krishnan V, Wu C, Chu VT, King ST, Vick J, Tao Q, Mraz A, Stormo A, Stormo K, Bogden R, Ederle D, Stella A, Vecchietti A, Kater MM, Masiero S, Lasserre P, Lespinasse Y, Allan AC, Bus V, Chagné D, Crowhurst RN, Gleave AP, Lavezzo E, Fawcett JA, Proost S, Rouzé P, Sterck L, Toppo S, Lazzari B, Hellens RP, Durel CE, Gutin A, Bumgarner RE, Gardiner SE, Skolnick M, Egholm M, Van de Peer Y, Salamini F, Viola

R (2010) The genome of the domesticated apple (Malus × domestica Borkh.). Nat Gen 42:833–839

Verde I, Bassil NV, Scalabrin S, Gilmore B, Lawley CT, Gasic K, Micheletti D, Rosyara UR, Cattonaro F, Vendramin E, Main DS, Aramini V, Blas AL, Mockler T, Bryant DW, Wilhelm L, Troggio M, Sosinski B, Aranzana MJ, Arus P, Iezzoni A, Morgante M and Peace C (2012) Development and Evaluation of a 9K SNP Array for Peach by Internationally Coordinated SNP Detection and Validation in Breeding Germplasm. PLoS One 7:e35668-10.1371/journal.pone.0035668

Verde I, Quarta R, Cerdrola C and Dettori MT (2002) QTL analysis of agronomic traits in a BC1 peach population. Acta Hortic 592:291– 297

Verlaan MG, Hutton SF, Ibrahem RM, Kormelink R, Visser RGF, Scott JW, Edwards JD and Bai Y (2013) The tomato yellow leaf curl virus resistance genes Ty-1 and Ty-3 are allelic and code for DFDGD-class RNA–dependent RNA polymerases. PLoS Genetics 9:e1003399

Vilanova S, Romero C, Abbott AG, Llacer G and Badenes ML (2003) An apricot (*Prunus armeniaca* L.) F2 progeny linkage map based on SSR and AFLP markers, mapping plum pox virus resistance and self-incompatibility traits. Theor Appl Genet 107:239-247

Wang L, Zhang Y, Qin C, Tian D, Yang S and Hurst LD (2016) Mutation rate analysis via parent – progeny sequencing of the perennial peach. II. No evidence for recombination-associated mutation. P Roy Soc B-Biol Sci 283:20161785

Wang Y (1985) Peach growing and germplasm. Acta Hortic 175:3–55

Wang Y, Georgi LL, Reighard GL, Scorza R and Abbott AG (2002) Genetic mapping of the evergrowing gene in peach (*Prunus persica* (L.) Batsch). J Hered 93:352–358

Warburton ML, Becerra-Velasquez VL, Goffreda JC and Bliss FA (1996) Utility of RAPD markers in identifying genetic linkages to genes of economic interest in peach. Theor Appl Genet 93:920–925

Werner DJ, Creller MA and Chaparro JX (1998) Inheritance of the blood-flesh trait in peach. HortScience 33:1243– 1246

Wijnker E, Velikkakam James G, Ding J, Becker F, Klasen JR, Rawat V, Rowan BA, de Jong DF, de Snoo CB, Zapata L, Huettel B, de Jong H, Ossowski S, Weigel D, Koornneef M, Keurentjes JJ and Schneeberger K (2013) The genomic landscape of meiotic crossovers and gene conversions in *Arabidopsis thaliana*. eLife 2:e01426

Yamamoto T, Mochida K and Hayashi T (2003) Shanhai Suimitsuto, one of the origins of Japanese peach cultivars. J Jpn Soc Hortic Sci 72:116–121

Yamamoto T, Shimada T, Imai T, Yaegaki H, Haji T, Matsuta N, Yamaguchi M and Hayashi T (2001) Characterization of morphological traits based on a genetic linkage map in peach. Breed Sci 51:271–278

Yamamoto T, Yamaguchi M and Hayashi T (2005) An integrated genetic linkage map of peach by SSR, STS, AFLP and RAPD. J Jpn Soc Hortic Sci 74:204–213

Yazbek MM and Al-Zein MS (2014) Wild almonds gone wild: revisiting Darwin's statement on the origin of peaches. Genet Resour Crop Ev 61:1319

Zamir D, Ekstein Michelson I, Zakay Y, Navot N, Zeidan M, Sarfatti M, Eshed Y, Harel E, Pleban T, van-Oss H, Kedar H, Rabinowitch HD and Czosnek H (1994) Mapping and introgression of a tomato yellow leaf curl virus tolerance gene, Ty-1. Theor Appl Genet 88:141–146

Zhang Q, Chen W, Sun L, Zhao F, Huang B, Yang W, Tao Y, Wang J, Yuan Z, Fan G, Xing Z, Han C, Pan H, Zhong X, Shi W, Liang X, Du D, Sun F, Xu Z, Hao R, Lv T, Lv Y, Zheng Z, Sun M, Luo L, Cai M, Gao Y, Wang J, Yin Y, Xu X, Cheng T, Wang J (2012) The genome of *Prunus mume*. Nat Commun 3:1318

Zhebentyayeva TN, Swire-Clark G, Georgi LL, Garay L, Jung S, Forrest A, Blackmon B, Horn R, Howad W, Arús P, Main D, Sosinski B, Baird WV, Reighard GL and Abbott AG (2008) A framework physical map for peach, a model Rosaceae species. Tree Genet Genomes 4:745–756

Zohary D, Hopf M and Weiss E (2012) Domestication of Plants in the Old World. Oxford University Press, Oxford

9. Annexes

Annex 1

‘compare_table_and_vcf.py’

This script compares the SNP information of a given individual of the progeny with the SNP core file and defines the allele represented by each SNPs. In detail:

If a position from the individual VCF file exists in the SNP core file:

- If it exists only in E1E2 and E1T and is heterozygous: **E1**
- If it exists only in E1E2 and E1T and is homozygous: **T**
- If it exists only in E1T and E2T and is heterozygous: **T**
- If it exists only in E1E2 and E2T and is heterozygous: **E2**
- If it exists in all, except E2T: **E1**
- If it exists in all, except E1T, and is heterozygous: **E2**
- If it exists in all, except E1T and is homozygous: **E1**
- If it exists in all, except E1E2 and is heterozygous: **T**

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
Created on Mon Sep  9 14:02:11 2013

@author: mtormo
"""

import argparse
import sqlite3
import vcf

def InputPar():#input parameters
    parser = argparse.ArgumentParser(description='Extract SNPs positions and genotype from a table (e1e1,e1e2,e1t,e2t) comparing with vcf')
    parser.add_argument('--vcf', metavar='FILE', required=True,
                        help='Single genotype VCF file (Required)')
    parser.add_argument('--snp_list', metavar='FILE', required=True,
                        help='List with SNPs in columns to compare (Required)')
    return parser.parse_args()


def SaveVcf(param):
    vcf_set=set()

    vcf_reader = vcf.Reader(open('.'.join([param.vcf]), 'r')) ### vcf file

    ###save chrom,pos from vcf to a dict of list
    for record in vcf_reader:
        if record.var_type=='snp':
            vcf_set.add('|'.join([record.CHROM,str(record.POS)]))

    return vcf_set


def vcf2bd(file_in,conn):
    c = conn.cursor()

    c.execute("CREATE TABLE indv (crompos PRIMARY KEY,crom,pos,geno)")

    v = vcf.Reader(open(file_in, 'r'))
    for record in v:
        if record.var_type == 'snp':
            genotype = record.samples[0]['GT']

            if len(record.ALT)>1:
                continue

            crompos = '/'.join([record.CHROM,str(record.POS)])
            c.execute("INSERT INTO indv VALUES (?,?,?,?)"\
                      ,(crompos,record.CHROM,int(record.POS),genotype))
            conn.commit()


def main():
    #take arguments
    param=InputPar()

    conn = sqlite3.connect(':memory:')
    vcf2bd(param.vcf,conn)

#    vcf_set = SaveVcf(param)

    with open(param.snp_list) as lst:

        for line in lst:
            items = line.rstrip().split('\t')

            chrom = items[0]
            pos = items[1]
```

```
69.
70.             e1e1 = items[2]
71.             e1e2 = items[3]
72.             e1t = items[4]
73.             e2t = items[5]
74.
75.             query = "'"'"'"+'/'.join((chrom,pos))+"'"
76.
77.
78.             c = conn.cursor()
79.
80.             guion=0
81.             for each in items[2:6]:
82.                 if each == '-':
83.                     guion += 1
84.
85.
86.             for row in c.execute("SELECT indv.crom,indv.gen
    o FROM indv \
87.                 WHERE {xcrompos}=indv.crompos"\
88.                 .format(xcrompos=query)):
89.
90.                 cr,gt = row
91. #                 print items,guion,gt
92.
93.
94.                 if guion==2 and e1e2!='-' and e1t!='-
    ' and gt=='0/1':
95.                     geno='E1'
96.                 elif guion==2 and e1e2!='-' and e1t!='-
    ' and gt=='1/1':
97.                     geno='T'
98.                 elif guion==2 and e1t!='-' and e2t!='-
    ' and gt=='0/1':
99.                     geno='T'
100.                    elif guion==1 and e2t=='-':
101.                        geno='E1'
102.                    elif guion==2 and e1e2!='-
    ' and e2t!='-' and gt=='0/1':
103.                        geno='E2'
```

```
104.                    elif guion==1 and e1t=='-
    ' and gt=='0/1':
105.                        geno='E2'
106.                    elif guion==1 and e1t=='-
    ' and gt=='1/1':
107.                        geno='E1'
108.                    elif guion==1 and e1e2=='-
    ' and gt=='0/1':
109.                        geno='T'
110.                    else:
111.                        continue
112.
113.
114.                    print '\t'.join((chrom,pos,geno))
115.
116.
117.
118.        if __name__ == "__main__":
119.            main()
```

Annex 2

'find_regions_from_genotyping.py'

This script reads the output of compare_table_and_vcf.py and defines which regions have the almond allele ('T regions') and which regions only have peach alleles ('E regions'), using the procedure as follows:

Section 1

1.1- From the output of 'compare_table_and_vcf.py', SNPs with the same allele information are grouped together whenever there are more than 10.

1.2- Those groups of 11+ markers are grouped into bigger windows of at least 100 markers.

1.3- This bigger windows are examined and all chromosomes classified as 'E region' (peach/peach) or 'T region' (peach/almond). Whenever a change in the genotype is observed, chromosomes are classified as 'E region' → 'T region' or 'T region' → 'E region' using the coordinates of the last SNP from the left window and the first SNP from the right window as boundaries.

Using the advantage of the previous information of 'E regions' and 'T regions', this second script makes a comparison between the SNPs from the individual and the SNP core file, similar to what 'compare_table_and_vcf.py' does. This allows the detection of the phase and the attribution of a complete genotype:

Section 2

If a SNP from a given individual is homozygous and the region is 'E', then:

- If the SNP exists only in E1E1 and E1E2, the final genotype is **E1E1**
- If the SNP exists only in E1E1, E1E2 and E1T, the final genotype is **E1E1**
- In the SNP exists only in E1E1, E1E2 and E2T, the final genotype is **E1E1**

If the SNP is heterozygous and the region is 'E':

- If the SNP exists only in E1E1 and E1E2, the final genotype is **E1E2**
- If the SNP exists only in E1E1, E1E2 and E1T, the final genotype is **E1E2**
- If the SNP exists only in E1E1, E1E2 and E2T, the final genotype is **E1E2**
- If the SNP exists only in E1E2 and E1T, the final genotype is **E1E2**
- If the SNP exists only in E1E2 and E2T, the final genotype is **E1E2**

If the SNP is homozygous and the region is 'T':

- If the SNP exists only in E1E2 and E1T, the final genotype is **E2T**

- If the SNP exists only in E1E1, E1E2 and E2T, the final genotype is **E1T**

If the SNP is heterozygous and the region is ‘T region’:

- If the SNP exists only in E1E2 and E1T, the final genotype is **E1T**
- If the SNP exists only in E1E1, E1E2 and E2T, the final genotype is **E2T**
- If the SNP exists only in E1E1, E1E2 and E1T, the final genotype is **E1T**
- If the SNP exists only in E1E2 and E2T, the final genotype is **E2T**
- If the SNP exists only in E1E2, E1T and E2T, the final genotype is **-/T**
- If the SNP exists only in E1E1, E1T and E2T, the final genotype is **-/T**

After the previous classification, this script groups the genotypes into windows in order to confidently detect CO events:

Section 3

3.1- Windows of E1E1 and E1E2 genotypes are generated whenever there are at least 10 adjacent genotypes; E1T and E2T windows are generated when there are at least one.

3.2- From these windows, only those with at least 30 SNPs are considered reliable. Starting with these reliable windows as ‘seeds’, adjacent windows with less than 30 SNPs are joined if they have the same genotype. When a reliable window with a different genotype is found, a CO is declared. The boundaries for the CO are defined by the position of the last SNP of the left window and the first SNP on the right window.

3.3- Genotype information and CO events are outputted in an independent file for each individual.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
Created on Tue Oct 14 12:53:57 2014

@author: mtormo
"""

import argparse
from collections import defaultdict
from collections import OrderedDict as od
import sqlite3
import sys
import vcf

def InputPar():
    ####Introducing arguments
    parser = argparse.ArgumentParser(description='Find recombination points from genotyping results. \n \
                    Extract 1 or 2 files: 1. Recomb.tab -> Per chromosome recombinations 2.(optional) New_results.tab -
> adding comparisons with E1E2.vcf,E1T.vcf and E2T.vcf ')
    parser.add_argument('--file', metavar='FILE', required=True,
                        help='Genotyping results file (Required)')
    parser.add_argument('--genos_file', metavar='FILE', required=True,
                        help='File with table of positions from 4 genotypes (Required)')
    parser.add_argument('--vcf', metavar='FILE',
                        help='Vcf file (Required if "fill=True")')
    parser.add_argument('--e1e2', metavar='FILE',default='-' ,
                        help='File with density per MegaBase from E1E2 vcf [None]')
    parser.add_argument('--out', metavar='BASENAME', required=True,
                        help='Output basename file obtaining OUT.recomb.tab and OUT.new_results.tab (Required)')
    parser.add_argument('--win', metavar='INT', type=int,default=100,
                        help='Minimum window size to accept it [100]')
    parser.add_argument('--lim', metavar='INT', type=int,default=10,
                        help='Limit of nucleotides to accept T windows [10]')
    parser.add_argument('--fill', action='store_true',
                        help='Create a XXX.new_results.tab filling E regions with SNPs from E1E2.vcf,E1T.vcf,E2T.vcf and Individual.vcf')
    parser.add_argument('--only_rec', action='store_true',
                        help='Only create a OUT.recomb.tab file with regions of genotypes')
    return parser.parse_args()

###join windows with the same Gt and below the limit
def JoinWin(scafXposgtXn,win):
    join_scafXposgtXn=defaultdict(set)##dict [crom] [pos1-pos2 - gt] = N_of occurrencies

    for crom,dict1 in sorted(scafXposgtXn.iteritems()):
        posgtXn=od()
        pos_list=[]
        gt_count=0

        for index,info in enumerate(dict1.iteritems()):
            posgt,n = info
            pos1, pos2, gt = posgt.split('-')
            gt_count += n

            try:
                next_key = dict1.keys()[index+1]
                next_gt = next_key.split('-')[2]
            except:
                next_gt = None
```

```

            if not next_gt and not crom in join_scafXposgtX
n.keys() and gt_count <= win:###unique window below limit
                posgtXn['NA'] = gt_count
                join_scafXposgtXn[crom] = posgtXn
                continue

            if gt == next_gt:###same window
                pos_list.extend((pos1,pos2))

            else:###close window
                pos_list.extend((pos1,pos2))

                if gt_count > win:
                    posgtXn['-
'.join([pos_list[0],pos_list[-1],gt])] = gt_count
                    join_scafXposgtXn[crom] = posgtXn

                pos_list=[]
                gt_count=0

    return join_scafXposgtXn

###make windows
def MakeWin(filein,win,n_recomb):
    scafXposgtXn=defaultdict(set)##dict [crom] [pos1-
pos2 - gt] = N_of occurrencies
    prev_scaf=None
    pos_list=[]
    gt_count=0

    all_set = set()###set to compare all file with new line
s (later)

    with open(filein,'r') as infile:
        all_file = infile.readlines()

        for n in  range(len(all_file)):
            crom,pos,gt = all_file[n].rstrip().split()
            all_set.add(crom+'/'+pos+'/'+gt)

            if n_recomb == 1:
                gt_index = 'E'
                if gt == 'T':
                    gt_index = 'T'
                elif n_recomb == 2:
                    gt_index = gt

                try:
                    next_crom, next_pos, next_gt = all_f
ile[n+1].rstrip().split()
                except IndexError:
                    next_crom = None
                    next_gt = None

                if crom != prev_scaf:
                    posgtXn=od()

                if crom == next_crom and gt_index in nex
t_gt:###same window
      #            if crom == next_crom and gt in next_gt:
###same window
                    pos_list.append(pos)
                    gt_count += 1

                else:###close window
                    pos_list.append(pos)
                    gt_count += 1

                    if gt_count > 10:###window is bigger
 than 10
                        posgtXn['-
'.join([pos_list[0],pos_list[-1],gt_index])] = gt_count
                        scafXposgtXn[crom] = posgtXn


                    pos_list=[]
                    gt_count=0
                prev_scaf = crom

```

```
130.           join_scafXposgtXn = JoinWin(scafXposgtXn,win)
131.
132.           return join_scafXposgtXn,all_set
133.
134.       def RedefineT(n, distance,join_scafXposgtXn,all_set)
   :
135.           redef_scafXposgtXn=defaultdict(list)
136.           for cr,v in sorted(join_scafXposgtXn.iteritems()
   ):
137.               pos_list=[]
138.
139.               prev_gt=None
140.               for index,info in enumerate(v.iteritems()):

141.                   ori_pos1,ori_pos2,gt = info[0].split('-
   ')
142.       #            print '@@@@@@@@@@@@@@@@',cr,ori_pos1,o
   ri_pos2,gt,v[info[0]]
143.
144.                   try:###check next line
145.                       next_key = v.keys()[index+1]
146.                       next_pos1,next_pos2,next_gt = next_k
   ey.split('-')
147.
148.                   except:
149.                       next_key = None
150.                       next_gt = None
151.                   if prev_gt != gt:
152.                       pos_list=[]
153.                   prev_gt = gt
154.                   pos_list.extend((ori_pos1,ori_pos2))
155.
156.                   if next_key and gt == next_gt:
157.                       continue
158.
159.                   if gt == 'E':
160.                       redef_scafXposgtXn[cr].append('-
   '.join([str(pos_list[0]),str(pos_list[-1]),'E']))
161.       #                print cr,str(ori_pos1),str(ori_pos2
   ),'E'
```

```
162.                   elif gt == 'T':
163.                       pos1=pos_list[0]
164.                       pos2=pos_list[-1]
165.
166.                       reg=[]
167.                       for elem in sorted(all_set, key=lamb
   da x: (x.split('/')[0],int(x.split('/')[1]))):
168.                           elem = elem.split('/')
169.                           if elem[0] == cr and int(pos1) <
   = int(elem[1]) <= int(pos2) and elem[2] == 'T':
170.                               reg.append(int(elem[1]))
171.
172.                       whole_wind=[]
173.                       for ps in range(len(reg)):
174.                           posi = reg[ps]
175.
176.                           slid_win=[]
177.
178.                           init=posi
179.                           end=posi+distance
180.
181.                           for p in reg[ps:]:
182.                               if init <= p <= end:
183.                                   slid_win.append(p)
184.                               elif p > end:
185.                                   break
186.                           whole_wind.append(slid_win)
187.
188.                       pos_list=[]
189.                       cc=0
190.                       prev_check=''
191.                       for w in range(len(whole_wind)):
192.                           resu=whole_wind[w]
193.                           try:
194.                               next_w = whole_wind[w+1]
195.                           except:
196.                               next_w = None
197.
198.                           if len(resu) < n and next_w:
199.                               check='>limit'
```

```
                            elif not next_w:
                                check='last'
                            else:
                                check='ok'

                            if check==prev_check or prev_che
    ck=='':
                                cc+=1
                                pos_list.extend([resu[0],res
    u[-1]])
                            elif not next_w:
                                pos_list.extend([resu[0],res
    u[-1]])
                                if pos_list[-1]-
    pos_list[0] > 4000000:
                                    redef_scafXposgtXn[cr].a
    ppend('-'.join([str(pos_list[0]),str(pos_list[-1]),'E']))
        #                            print cr,str(pos_list[0
    ]),str(pos_list[-1]),'E'

                            else:
                                if prev_check == 'ok':
                                    redef_scafXposgtXn[cr].a
    ppend('-'.join([str(pos_list[0]),str(resu[0]),'T']))
        #                            print cr,pos_list[0],st
    r(resu[0]),'T'
                                elif prev_check != 'ok' and
    pos_list[-1]-pos_list[0] > 4000000:
                                    redef_scafXposgtXn[cr].a
    ppend('-'.join([str(pos_list[0]),str(pos_list[-1]),'E']))
        #                            print cr ,str(pos_list[
    0]),str(pos_list[-1]),'E'

                                pos_list=[]
                                pos_list.extend([resu[0],res
    u[-1]])
                                cc=1
                            prev_check=check

        #                print 'total',redef_scafXposgtXn
        #    sys.exit()
            return redef_scafXposgtXn


        def RecPoint(scafXposgtXn):
            recomb = []
            recomb.append(('chrom', 'recombination', 'first_
    start', 'first_end', 'second_start', 'second_end'))

            for crom,list1 in sorted(scafXposgtXn.iteritems(
    )):

                part1=[]###gt-pos1-pos2 of first part
                part2=[]###gt-pos1-pos2 of second part

                for index in range(len(list1)):
                    posgt = list1[index]
                    try:
                        pos1, pos2, gt = posgt.split('-')
                    except ValueError:
                        recomb.append((crom,'NA','-','-','-
    ','-'))
        #                print '\t',crom,'NA','-','-','-','-
    '
                        continue
        #            print index,crom,pos1,pos2,gt,n

                    if index != 0:###it's not the first line

                        prev_key = list1[index-1]
                        prev_gt = prev_key.split('-')[2]
                    else:
                        prev_key = None
                        prev_gt = None

                    try:###check next line
                        next_key = list1[index+1]
                        next_pos1,next_pos2,next_gt = next_k
    ey.split('-')
                    except:
```

```
                        next_key = None
                        next_gt = None

                    if not prev_gt and not next_gt:###not prev not next, only one
                        recomb.append((crom,'only_'+gt,pos1,pos2,'-','-'))
                        continue

                    elif not prev_gt and next_gt:###not prev, next
                        part1.extend((gt,pos1,pos2))
                        continue

                    if part1 and part1[0] == gt:###prev with same genotype
                        prev_gt,prev_pos1,prev_pos2 = part1

                        part1=[]
                        part1.extend((gt,prev_pos1,pos2))

                    elif part1 and part1[0] != gt:###prev with other genotype
                        if not part2:###starts second part
                            part2.extend((gt,pos1,pos2))
                        else:###extend second part
                            prev_gt,prev_pos1,prev_pos2 = part2
                            part2=[]
                            part2.extend((gt,prev_pos1,pos2))

                    if next_gt:
                        if part2 and part2[0] != next_gt:###different second genotype, double recombination
                            p1_gt,p1_pos1,p1_pos2 = part1
                            p2_gt,p2_pos1,p2_pos2 = part2

                            recomb.append((crom,p1_gt+'>'+p2_gt,p1_pos1,p1_pos2,p2_pos1,p2_pos2))

                            part1=[]
                            part1=part2
                            part2=[]
                        elif part2 and part2[0] == next_gt:###extend second part
                            prev_gt,prev_pos1,prev_pos2 = part2
                            part2=[]
                            part2.extend((gt,prev_pos1,pos2))
                    else:###last
                        if part1 and part2:
                            p1_gt,p1_pos1,p1_pos2 = part1
                            p2_gt,p2_pos1,p2_pos2 = part2

                            recomb.append((crom,p1_gt+'>'+p2_gt,p1_pos1,p1_pos2,p2_pos1,p2_pos2))

                        elif part1 and not part2:
                            p1_gt,p1_pos1,p1_pos2 = part1
                            recomb.append((crom,'only_'+p1_gt,p1_pos1,p1_pos2,'-','-'))

            return recomb


        def FinalRecomb(infile,win,n_recomb):
            join_scafXposgtXn,all_set = MakeWin(infile,win,n_recomb)
            n=10
            distance=100000
            redef_scafXposgtXn = RedefineT(n, distance,join_scafXposgtXn,all_set)

            recomb = RecPoint(redef_scafXposgtXn)

            return recomb,all_set
```

```
324.
325.       def vcf2bd(file_in,conn,iden):
326.           c = conn.cursor()
327.
328.           c.execute("DROP TABLE IF EXISTS {d}".format(d=id
    en))
329.           c.execute("CREATE TABLE {d} (crompos PRIMARY KEY
    ,crom,pos,geno,ref,alt)".format(d=iden))
330.
331.           v = vcf.Reader(open(file_in, 'r'))
332.           for record in v:
333.               if record.var_type == 'snp':
334.                   genotype = record.samples[0]['GT']
335.
336.                   if len(record.ALT)>1:
337.                       continue
338.
339.                   crompos = '/'.join([record.CHROM,str(rec
    ord.POS)])
340.                   c.execute("INSERT INTO {xiden} VALUES (?
    ,?,?,?,?,?)"\
341.                                     .format(xiden=iden),(crompos
    ,record.CHROM,int(record.POS),genotype,record.REF,str(recor
    d.ALT[0])))
342.                   conn.commit()
343.       #    print iden,'done'
344.
345.       def CreateBd(file_in,conn):
346.           c = conn.cursor()
347.           c.execute("CREATE TABLE info (crompos PRIMARY KE
    Y,e1e1_ref,e1e1_alt,e1e2_ref,e1e2_alt,e1t_ref,e1t_alt,e2t_r
    ef,e2t_alt)")
348.           conn.commit()
349.
350.           with open(file_in,'r') as geno_f:
351.               for line in geno_f:
352.                   line = line.rstrip().split('\t')
353.                   if line[0]=='crom':
354.                       continue
355.                   crompos = '/'.join([line[0],line[1]])
```

```
356.                   try:
357.                       e1e1_ref,e1e1_alt=line[2]
358.                   except:
359.                       e1e1_ref='-'
360.                       e1e1_alt='-'
361.                   try:
362.                       e1e2_ref,e1e2_alt=line[3]
363.                   except:
364.                       e1e2_ref='-'
365.                       e1e2_alt='-'
366.                   try:
367.                       e1t_ref,e1t_alt=line[4]
368.                   except:
369.                       e1t_ref='-'
370.                       e1t_alt='-'
371.                   try:
372.                       e2t_ref,e2t_alt=line[5]
373.                   except:
374.                       e2t_ref='-'
375.                       e2t_alt='-'
376.
377.                   c.execute("INSERT INTO info VALUES (?,?,
    ?,?,?,?,?,?,?)",(crompos,e1e1_ref,e1e1_alt,e1e2_ref,e1e2_al
    t,e1t_ref,e1t_alt,e2t_ref,e2t_alt))
378.                   conn.commit()
379.
380.       def FillET(coord,out_gt,conn):###fill E-T regions
381.           crompos=set()
382.           c = conn.cursor()
383.
384.           for crom,dict_et_coord in coord.iteritems():
385.               crom="'"'"'"+crom+"'"'"###quote chromosome for
    comparing
386.               for et_coord in dict_et_coord:
387.                   try:
388.                       et,init,fin = et_coord.split('/')
389.                   except:###NA
390.                       continue
391.       #            print 'FillET',crom,et,init,fin
392.                   if et == 'E':
```

```
393.       #                for row in c.execute("SELECT * FROM
    indv,info \
394.
395.       #                    AND ((info.e1e1_ref != '-
    ' AND info.e1e2_ref = '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref = '-')\
396.                         for row in c.execute("SELECT indv.cr
    om,indv.pos FROM indv,info \
397.                             WHERE indv.crom = {xcrom} AND in
    dv.pos >= {xinit} AND indv.pos <= {xfin} AND indv.crompos =
     info.crompos AND indv.geno='1/1' \
398.                             AND (  (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref = '-')\
399.                             OR   (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-')\
400.                             OR   (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-') )"\
401.                             .format(xcrom=crom,xinit=init,xf
    in=fin)):
402.       #                     print 'E1E1',row
403.                             cr,pos = row
404.                             pos = str(pos)
405.                             out_gt.add('/'.join([cr,pos,'E1E
    1']))
406.                             crompos.add('/'.join([cr,pos]))
407.
408.       #                for row in c.execute("SELECT * FROM
    indv,info \
409.
410.       #                    AND ((info.e1e1_ref = '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref = '-')\
411.                         for row in c.execute("SELECT indv.cr
    om,indv.pos FROM indv,info \
```

```
412.                             WHERE indv.crom = {xcrom} AND in
    dv.pos >= {xinit} AND indv.pos <= {xfin} AND indv.crompos =
     info.crompos AND indv.geno='0/1' \
413.                             AND (  (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref = '-')\
414.                             OR   (info.e1e1_ref = '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-')\
415.                             OR   (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-')\
416.                             OR   (info.e1e1_ref = '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-')\
417.                             OR   (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-') )"\
418.                             .format(xcrom=crom,xinit=init,xf
    in=fin)):
419.       #                     print 'E1E2',row
420.                             cr,pos = row
421.                             pos = str(pos)
422.                             out_gt.add('/'.join([cr,pos,'E1E
    2']))
423.                             crompos.add('/'.join([cr,pos]))
424.
425.
426.                     elif et == 'T':
427.       #                 for row in c.execute("SELECT * FROM
    indv,info \
428.
429.       #                    AND ((info.e1e1_ref = '-
    ' AND info.e1e2_ref = '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-')\
430.                         for row in c.execute("SELECT indv.cr
    om,indv.pos,indv.geno FROM indv,info \
```

```
                            WHERE indv.crom = {xcrom} AND in
    dv.pos >= {xinit} AND indv.pos <= {xfin} AND indv.crompos =
     info.crompos \
                            AND (  (info.e1e1_ref = '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-') )"\
                            .format(xcrom=crom,xinit=init,xf
    in=fin)):
                            cr,pos,geno = row
                            pos = str(pos)
                            if geno == '1/1':
        #                        print 'E1T',row
                                out_gt.add('/'.join([cr,pos,
    'E2T']))
                                crompos.add('/'.join([cr,pos
    ]))
                            elif geno == '0/1':
        #                        print 'E2T',row
                                out_gt.add('/'.join([cr,pos,
    'E1T']))
                                crompos.add('/'.join([cr,pos
    ]))

        #                for row in c.execute("SELECT * FROM
     indv,info \

        #                    AND ((info.e1e1_ref = '-
    ' AND info.e1e2_ref = '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-')\
                           for row in c.execute("SELECT indv.cr
    om,indv.pos,indv.geno FROM indv,info \
                            WHERE indv.crom = {xcrom} AND in
    dv.pos >= {xinit} AND indv.pos <= {xfin} AND indv.crompos =
     info.crompos \
                            AND (  (info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-') )"\
                            .format(xcrom=crom,xinit=init,xf
    in=fin)):
                            cr,pos,geno = row
                            pos = str(pos)
                            if geno == '1/1':
        #                        print 'E1T',row
                                out_gt.add('/'.join([cr,pos,
    'E1T']))
                                crompos.add('/'.join([cr,pos
    ]))
                            elif geno == '0/1':
        #                        print 'E2T',row
                                out_gt.add('/'.join([cr,pos,
    'E2T']))
                                crompos.add('/'.join([cr,pos
    ]))


        #                for row in c.execute("SELECT * FROM
     indv,info \
                           for row in c.execute("SELECT indv.cr
    om,indv.pos,info.e1t_ref,info.e2t_ref FROM indv,info \
                            WHERE indv.crom = {xcrom} AND in
    dv.pos >= {xinit} AND indv.pos <= {xfin} AND indv.crompos =
     info.crompos AND indv.geno='0/1' \
                            AND ((info.e1e1_ref != '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref = '-')\
                            OR (info.e1e1_ref = '-
    ' AND info.e1e2_ref = '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref != '-')\
                            OR   (info.e1e1_ref != '-
    ' AND info.e1e2_ref = '-' AND info.e1t_ref != '-
    ' AND info.e2t_ref != '-')\
                            OR   (info.e1e1_ref = '-
    ' AND info.e1e2_ref != '-' AND info.e1t_ref = '-
    ' AND info.e2t_ref != '-') )"\
                            .format(xcrom=crom,xinit=init,xf
    in=fin)):
        #                    print 'E1E2',row
                            cr,pos,e1t,e2t = row
                            pos = str(pos)
                            if e1t!='-' and e2t!='-':
```

```
                                out_gt.add('/'.join([cr,pos,'T']))
                                crompos.add('/'.join([cr,pos]))
                            elif e1t!='-' and e2t=='-':
                                out_gt.add('/'.join([cr,pos,'E1T']))
                                crompos.add('/'.join([cr,pos]))
                            else:
                                out_gt.add('/'.join([cr,pos,'E2T']))
                                crompos.add('/'.join([cr,pos]))
        #             print out_gt
            conn.commit()

            return out_gt,crompos

        def FillT(coord,all_set,out_gt,crompos):###fill with positions of results file
            for line_all in all_set:
                crom,pos,gt = line_all.split('/')


                if crom in coord:
                    for et_coord in coord[crom]:
                        try:
                            et,init,fin = et_coord.split('/')
                        except:###NA
                            out_gt.add('/'.join([crom,pos,gt]))
                            continue

                        if et == 'T' and int(init) <= int(pos) <= int(fin):
                            if not '/'.join([crom,pos]) in crompos:
                                if gt=='E1E2' or gt=='E1E1':

                                    continue
                                elif gt.find('E')==0 and gt.find('T')==-1:
                                    gt = gt+'T'
                                out_gt.add('/'.join([crom,pos,gt]))

            return out_gt

        def Fill(param,recomb,all_set,out1):

            conn = sqlite3.connect(':memory:')

            vcf2bd(param.vcf,conn,'indv')
            CreateBd(param.genos_file,conn)

            out2_file = '.'.join([param.out,'new_results','tab'])
            out2 = open(out2_file, 'w')
            out_gt = set()

            coord=defaultdict(set)

            for line in recomb:
                chrom, recombination, first_start, first_end, second_start, second_end = line


                ###obtain regions
                if recombination.find('E')==0:###E>T
                    e_start = first_start
                    e_end = second_start
                    t_start = second_start
                    t_end = second_end

                    coord[chrom].add('/'.join(['E',e_start,e_end]))
```

```
                    coord[chrom].add('/'.join(['T',t_start,t_end]))

                elif recombination.find('T')==0:###T>E
                    e_start = first_end
                    e_end = second_end
                    t_start = first_start
                    t_end = first_end

                    coord[chrom].add('/'.join(['E',e_start,e_end]))
                    coord[chrom].add('/'.join(['T',t_start,t_end]))

                elif recombination == 'only_E':###only_E
                    e_start = first_start
                    e_end = first_end

                    coord[chrom].add('/'.join(['E',e_start,e_end]))

                elif recombination == 'only_T':###only_T
                    t_start = first_start
                    t_end = first_end

                    coord[chrom].add('/'.join(['T',t_start,t_end]))

                elif recombination == 'NA':###NA
                    coord[chrom].add('NA')

            if not param.only_rec:

            #        print coord
                out_gt,crompos = FillET(coord,out_gt,conn)
                out_gt = FillT(coord,all_set,out_gt,crompos)

```

```
                for newline in sorted(out_gt, key=lambda x: (x.split('/')[0],int(x.split('/')[1]))):###sort by chrom and pos
                    out2.write('\t'.join(newline.split('/'))+'\n')

                out2.close()
                return out2_file,coord
            else:
                return coord

        def JoinWinGenos(scafXposgtXn,lim,coord):
            join_scafXposgtXn=defaultdict(set)##dict [crom][pos1-pos2 - gt] = N_of occurrencies

            for crom,dict1 in sorted(scafXposgtXn.iteritems()):
                posgtXn=od()
                pos_list=[]
                gt_count=0

                for index,info in enumerate(dict1.iteritems()):
                    posgt,n = info
                    try:
                        pos1, pos2, gt = posgt.split('-')
                    except:
                        posgtXn['NA'] = 0
                        join_scafXposgtXn[crom] = posgtXn
                        continue
        #             print crom,pos1,pos2,gt,n

                    try:
                        next_key = dict1.keys()[index+1]
                        next_gt = next_key.split('-')[2]
        #                next_pos = int(next_key.split('-')[1])
                    except:
                        next_gt = None
        #                next_pos = 999999999
```

```
602.
603.                     gt_count += n
604.
605.                     if gt == next_gt:###same window
606.                         pos_list.extend((pos1,pos2))
607.
608.                     else:###close window
609.                         pos_list.extend((pos1,pos2))
610.
611.
612.                         if gt_count > lim:
613.       #                     print 'ok','-
    '.join([pos_list[0],pos_list[-1],gt]), gt_count
614.                             posgtXn['-
    '.join([pos_list[0],pos_list[-1],gt])] = gt_count
615.                             join_scafXposgtXn[crom] = posgtX
    n
616.       #                     prev_pos = int(pos_list[-1])
617.
618.                         elif gt_count <= lim and not next_gt
     and not crom in join_scafXposgtXn.keys():###unique window
    below limit
619.       #                     print 'NA','-
    '.join([pos_list[0],pos_list[-1],gt]), gt_count
620.                             posgtXn['NA'] = 0
621.                             join_scafXposgtXn[crom] = posgtX
    n
622.
623.
624.       #                 elif gt_count > 1:
625.                         else:###below limit
626.       #                     print 'below','-
    '.join([pos_list[0],pos_list[-1],gt]), gt_count
627.                             posgtXn['-
    '.join([pos_list[0],pos_list[-1],gt])] = gt_count
628.                             join_scafXposgtXn[crom] = posgtX
    n
629.       #                     prev_pos = int(pos_list[-1])
630.
631.       #                 print join_scafXposgtXn[crom]
```

```
632.                         pos_list=[]
633.                         gt_count=0
634.
635.             return join_scafXposgtXn
636.
637.         ###make windows
638.         def MakeWinGenos(filein,lim,coord):
639.             scafXposgtXn=od()##dict [crom] [pos1-
    pos2 - gt]
640.             prev_scaf=None
641.             pos_list=[]
642.             gt_count=0
643.
644.             with open(filein,'r') as infile:
645.                 all_file = infile.readlines()
646.
647.                 for n in  range(len(all_file)):
648.                     crom,pos,gt_index = all_file[n].rstrip()
    .split()
649.       #             print crom,pos,gt_index,gt_count
650.                     if len(gt_index) == 2:###if it's E1 or E
    2
651.                         gt_index = gt_index+'T'
652.
653.                     try:
654.                         next_crom, next_pos, next_gt = all_f
    ile[n+1].rstrip().split()
655.                         if len(next_gt) == 2:###if it's E1T
    or E2T
656.                             next_gt = next_gt+'T'
657.                     except IndexError:
658.                         next_crom = None
659.                         next_gt = None
660.
661.                     if crom != prev_scaf:
662.                         posgtXn=od()
663.
664.                     if gt_index == 'T' and (not next_gt or c
    rom != next_crom) and crom not in scafXposgtXn.keys():
665.                         posgtXn['NA'] = gt_count
```

```
                        scafXposgtXn[crom] = posgtXn
                        continue
                    elif gt_index == 'T' and next_gt:

                        prev_scaf = crom
                        continue


                    if crom == next_crom and gt_index == nex
t_gt:###same window
                        pos_list.append(pos)
                        gt_count += 1

                    else:###close window
                        pos_list.append(pos)
                        gt_count += 1

                        if gt_index == 'E1E1' or gt_index ==
 'E1E2':
                            if gt_count > 10:###window is bi
gger than 10
                                posgtXn['-
'.join([pos_list[0],pos_list[-1],gt_index])] = gt_count
                                scafXposgtXn[crom] = posgtXn

      #                          print '-
'.join([pos_list[0],pos_list[-1],gt_index]), gt_count
                        else:
                            if gt_count > 1:###window is big
ger than 1
                                posgtXn['-
'.join([pos_list[0],pos_list[-1],gt_index])] = gt_count
                                scafXposgtXn[crom] = posgtXn

      #                          print '-
'.join([pos_list[0],pos_list[-1],gt_index]), gt_count
      #                          print scafXposgtXn[crom]
                            else:
                                pos_list=[]
                                gt_count=0
```

```
                                prev_scaf = crom
                                continue

                        pos_list=[]
                        gt_count=0
                    prev_scaf = crom

            join_scafXposgt = JoinWinGenos(scafXposgtXn,lim,
coord)

            return join_scafXposgt


        def SecondRec(out2_file,lim,coord):
            sec_rec=[]
            join_scafXposgtXn = MakeWinGenos(out2_file,lim,c
oord)

            for crom,dict1 in sorted(join_scafXposgtXn.iteri
tems()):
                pos_list=[]
                prev_gt=None
                for index,info in enumerate(dict1.iteritems(
)):
                    posgt,n = info

                    if posgt=='NA':
                        sec_rec.append('\t'.join([crom,'NA',
'-','-']))
                        pos_list=[]
                        continue

                    pos1, pos2, gt = posgt.split('-')
      #              print crom,posgt

                    try:###check next line
                        next_key = dict1.keys()[index+1]
                        next_pos1,next_pos2,next_gt = next_k
ey.split('-')

```

```
                    except:
                        next_key = None
                        next_gt = None

                    try:
                        second_key = dict1.keys()[index+2]
                        sec_pos1,sec_pos2,sec_gt = second_ke
    y.split('-')
                    except:
                        second_key = None
                        sec_gt = None

                    if prev_gt!=gt and n<lim:###below limit

                        continue

                    pos_list.extend((pos1,pos2))


                    if next_key and second_key and gt != nex
    t_gt and dict1[next_key]<lim and gt == sec_gt and dict1[sec
    ond_key]>=10:
                        prev_gt=gt
                        continue
                    else:
        #                print 'tanca'+'\t'.join([crom,str(n
    ),gt,pos_list[0],pos_list[-1]])
                        sec_rec.append('\t'.join([crom,gt,po
    s_list[0],pos_list[-1]]))
                        pos_list=[]
                        prev_gt=None

            return sec_rec


        def GetRecpoint(sec_rec,crom_size):###get recombinat
    ions points between genotypes, or empty regions
            regions=[]###crom,pos1,pos2,recomb

            pos_list=[]
            first_pos=None

            for n in  range(len(sec_rec)):
                rec = sec_rec[n].rstrip().split('\t')
        #         print rec
                regions.append('/'.join(['final',rec[0],rec[
    2],rec[3],rec[1]]))
                try:
                    next_rec = sec_rec[n+1].rstrip().split('
    \t')
                except:
                    next_rec = None

                if not first_pos:
                    first_pos=rec[2]
                    regions.append('/'.join(['final',rec[0],
    '0',first_pos,'Undef']))
        #             print '/'.join(['recpoint',rec[0],'0',f
    irst_pos,'Empty'])

                pos_list.append(rec[3])

                if next_rec and rec[0] == next_rec[0]:###if
    it's the same chromosome
                    if rec[1] == next_rec[1]:
                        continue
                    else:
                        regions.append('/'.join(['recpoint',
    rec[0],pos_list[-1],next_rec[2],rec[1]+'>'+next_rec[1]]))
        #                print '/'.join(['recpoint',rec[0],p
    os_list[-1],next_rec[2],rec[1]+'>'+next_rec[1]])
                else:###other chromosome
                    first_pos=None
                    regions.append('/'.join(['final',rec[0],
    rec[3],str(crom_size[rec[0]]),'Undef']))
        #             print '/'.join(['recpoint',rec[0],rec[3
    ],str(crom_size[rec[0]]),'Empty'])
                pos_list=[]

            return regions
```

```
794.
795.       def VarRegions(regions,density):###get variable/unva
    riable regions from recombination points and density of snp
    s' file
796.       #    pos_w_dens=od()
797.           pos_w_dens=[]
798.           min_snp_mb=100
799.
800.           for line in regions:
801.               definition,crom,pos1,pos2,rec = line.split('
    /')
802.
803.       #        print crom,pos1,pos2,rec
804.               pos1 = int(pos1)
805.               pos2 = int(pos2)
806.
807.               done = False
808.               if definition == 'recpoint':
809.       #            if rec == 'E1T>E2T' or rec == 'E2T>E1T'
     or rec == 'E1E1>E1E2' or rec == 'E1E2>E1E1':
810.
811.                   for d in range(len(density)):
812.
813.                       crom_d,pos1_d,pos2_d,count = density
    [d]
814.
815.
816.                       if crom == crom_d:
817.                           if pos1 >= int(pos1_d) and pos2
    <= int(pos2_d) and int(count) >= min_snp_mb:
818.       #                           pos_w_dens = FillDict(c
    rom,pos1,pos2,pos_w_dens)
819.                               done = True
820.                           elif pos1 >= int(pos1_d) and pos
    2 <= int(pos2_d) and int(count) < min_snp_mb:
821.                               xpos1=str(pos1)
822.                               xpos2=str(pos2)
823.                               pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Low_var']))
824.
825.       #                           pos_w_dens = FillDict(c
    rom,xpos1,xpos2,pos_w_dens)
826.                               done = True
827.
828.                           elif int(pos1_d) <= pos1 < int(p
    os2_d) and pos2 > int(pos2_d) and int(count) >= min_snp_mb:
829.       #                           pos_w_dens = FillDict(c
    rom,pos1,int(pos2_d),pos_w_dens)
830.                               done = True
831.                           elif int(pos1_d) <= pos1 < int(p
    os2_d) and pos2 > int(pos2_d) and int(count) < min_snp_mb:
832.                               xpos1=str(pos1)
833.                               xpos2=pos2_d
834.                               pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Low_var']))
835.       #                           pos_w_dens = FillDict(c
    rom,xpos1,xpos2,pos_w_dens)
836.                               done = True
837.
838.                           elif pos1 < int(pos1_d) and int(
    pos1_d) < pos2 <= int(pos2_d) and int(count) >= min_snp_mb:
839.       #                           pos_w_dens = FillDict(c
    rom,int(pos1_d),pos2,pos_w_dens)
840.                               done = True
841.                           elif pos1 < int(pos1_d) and int(
    pos1_d) < pos2 <= int(pos2_d) and int(count) < min_snp_mb:
842.                               xpos1=pos1_d
843.                               xpos2=str(pos2)
844.                               pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Low_var']))
845.       #                           pos_w_dens = FillDict(c
    rom,xpos1,xpos2,pos_w_dens)
846.                               done = True
847.
848.                           elif pos1 < int(pos1_d) and pos2
     > int(pos2_d) and int(count) >= min_snp_mb:
```

```
849.       #                               pos_w_dens = FillDict(c
    rom,int(pos1_d),int(pos2_d),pos_w_dens)
850.                                   done = True
851.                               elif pos1 < int(pos1_d) and pos2
     > int(pos2_d) and int(count) < min_snp_mb:
852.                                   xpos1=pos1_d
853.                                   xpos2=pos2_d
854.                                   pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Low_var']))
855.       #                               pos_w_dens = FillDict(c
    rom,xpos1,xpos2,pos_w_dens)
856.                                   done = True
857.
858.                               else:###it's another region
859.                                   continue
860.
861.                       if done == False:###no information about
     the region = unvariable
862.                           xpos1=str(pos1)
863.                           xpos2=str(pos2)
864.       #                    pos_w_dens.append('/'.join([crom,xp
    os1,xpos2,'Low_var']))
865.                           pos_w_dens.append('/'.join([crom,xpo
    s1,xpos2,'Undef']))
866.       #                       pos_w_dens = FillDict(crom,xpos
    1,xpos2,pos_w_dens)
867.                           done = True
868.
869.
870.                   elif definition == 'final' and rec == 'Undef
    ':
871.                       for d in range(len(density)):
872.
873.                           crom_d,pos1_d,pos2_d,count = density
    [d]
874.
875.                           if crom == crom_d:
876.       #                       print crom_d,pos1_d,pos2_d,coun
    t
```

```
877.                               if pos1 >= int(pos1_d) and pos2
    <= int(pos2_d) and int(count) >= min_snp_mb:
878.       #                           print '/'.join(['entra1',cr
    om,str(pos1),str(pos2),'Undef'])
879.                                   xpos1=str(pos1)
880.                                   xpos2=str(pos2)
881.                                   pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Undef']))
882.       #                           done = False
883.                                   done = True
884.                               elif pos1 >= int(pos1_d) and pos
    2 <= int(pos2_d) and int(count) < min_snp_mb:
885.       #                           print 'entra2'
886.                                   xpos1=str(pos1)
887.                                   xpos2=str(pos2)
888.                                   pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Low_var|Undef']))
889.
890.       #                               pos_w_dens = FillDict(c
    rom,xpos1,xpos2,pos_w_dens)
891.                                   done = True
892.
893.                               elif int(pos1_d) <= pos1 < int(p
    os2_d) and pos2 > int(pos2_d) and int(count) >= min_snp_mb:

894.                                   xpos1=str(pos1)
895.                                   xpos2=pos2_d
896.                                   pos_w_dens.append('/'.join([
    crom,xpos1,xpos2,'Undef']))
897.                                   done = True
898.       #                           print '/'.join(['entra3',cr
    om,str(pos1),str(pos2),'Undef'])
899.       #                           done = False
900.                               elif int(pos1_d) <= pos1 < int(p
    os2_d) and pos2 > int(pos2_d) and int(count) < min_snp_mb:

901.       #                           print 'entra4'
902.                                   xpos1=str(pos1)
903.                                   xpos2=pos2_d
```

```
                                pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Low_var|Undef']))
                                done = True

                            elif pos1 < int(pos1_d) and int(pos1_d) < pos2 <= int(pos2_d) and int(count) >= min_snp_mb:

                                xpos1=pos1_d
                                xpos2=str(pos2)
                                pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Undef']))
                                done = True
        #                        done = False
        #                        done = True
                            elif pos1 < int(pos1_d) and int(pos1_d) < pos2 <= int(pos2_d) and int(count) < min_snp_mb:

        #                        print 'entra6'
                                xpos1=pos1_d
                                xpos2=str(pos2)
                                pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Low_var|Undef']))
                                done = True

                            elif pos1 < int(pos1_d) and pos2 > int(pos2_d) and int(count) >= min_snp_mb:
        #                        print '/'.join(['entra7',crom,str(pos1),str(pos2),'Undef'])
                                xpos1=pos1_d
                                xpos2=pos2_d
                                pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Undef']))
                                done = True
        #                        done = False
        #                        done = True
                            elif pos1 < int(pos1_d) and pos2 > int(pos2_d) and int(count) < min_snp_mb:
        #                        print 'entra8'
                                xpos1=pos1_d
                                xpos2=pos2_d
```

```
                                pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Low_var|Undef']))
                                done = True

                            else:###it's another region
        #                        print 'entra9'
                                continue
        #            print done
                    if done == False:###no information about the region = unvariable
                        xpos1=str(pos1)
                        xpos2=str(pos2)
        #                print '/'.join(['entra10',crom,xpos1,xpos2,'Undef'])
                        pos_w_dens.append('/'.join([crom,xpos1,xpos2,'Undef']))
                        done = True


                else:
        #            pos_w_dens = FillDict(crom,pos1,pos2,pos_w_dens)
                    pos_w_dens.append('/'.join([crom,str(pos1),str(pos2),rec]))
                    done = True

        #        print pos_w_dens
            return pos_w_dens


        def main():
            #take arguments
            param=InputPar()
            if param.fill and not param.vcf:
                sys.exit('!!!param.vcf Needed!!!')

            out1 = open('.'.join([param.out,'recomb','tab']), 'w')

```

```
        recomb,all_set = FinalRecomb(param.file,param.win,1)

        if param.only_rec:###only second recombination file
            coord = Fill(param,recomb,None,out1)
            SecondRec(out1,param.file,param,coord)

        else:
            out1.write('###E&T_recombination###\n')
            for line in recomb:
                out1.write('\t'.join(line)+'\n')

            if param.fill:###fill with E1E2,  E1E1, and T's
                out2_file,coord = Fill(param,recomb,all_set,out1)
                sec_rec = SecondRec(out2_file,param.lim,coord)

    #    print sec_rec
        density=[]
        if param.e1e2 != '-':
            with open(param.e1e2,'r') as e1e2_density:
                for l_d in e1e2_density:
                    density.append((l_d.rstrip().split('\t')))

        crom_size={'Pp01':47851209,
                   'Pp02':30405871,
                   'Pp03':27368014,
                   'Pp04':25843237,
                   'Pp05':18496697,
                   'Pp06':30767195,
                   'Pp07':22388615,
                   'Pp08':22573981}

        regions = GetRecpoint(sec_rec,crom_size)

        pos_w_dens = VarRegions(regions,density)
        pos_dens_sorted=sorted(pos_w_dens, key=lambda x: (x.split('/')[0],int(x.split('/')[1])))###sort by chrom and pos

        out1.write('###All_genos_recombination###\n')
        pos_list=[]
        prev_crom=None

        for c in range(len(pos_dens_sorted)):
            cr,pos1,pos2,rec = pos_dens_sorted[c].split('/')
            pos_list.extend((pos1,pos2))

            try:
                next_cr,next_p1,next_p2,next_rec = pos_dens_sorted[c+1].split('/')

            except:
                next_cr  = next_p1  = next_p2  = next_rec = None


            if next_cr and cr == next_cr and rec == 'Low_var' and next_rec == 'Low_var' and pos2 == next_p1:
                continue
            elif next_cr and cr == next_cr and rec == 'Low_var|Undef' and next_rec == 'Low_var|Undef' and pos2 == next_p1:
                continue
            elif next_cr and cr == next_cr and rec == 'Undef' and next_rec == 'Undef' and pos2 == next_p1:
                continue
            else:
                if cr != prev_crom:
                    pos_list[0] = '0'
                if cr != next_cr:
                    pos_list[-1] = str(crom_size[cr])

    #           print '\t'.join([cr,pos_list[0],pos_list[-1],rec])
```

```
                out1.write('\t'.join([cr,pos_list[0],pos
    _list[-1],rec])+'\n')
                prev_crom = cr
                pos_list=[]


    if __name__ == "__main__":
        main()
```

Annex 3

‘Fill_recpoints.py’ (FRP)

With all the information from the previous two scripts (‘compare_table_and_vcf.py’ and ‘find_regions_from_genotyping.py’), the CO events are already detected across the progeny. However, in order to get as close as possible to the CO breakpoint, a third script was implemented, ‘Fill_recpoints.py’, which looks for additional positions in the CO region using a second SNP core file that includes additional SNPs (of medium quality) that were not present in the first SNP core file (and represent a 40% increase in total number of SNPs available). The procedure applied is similar to what the second script does, but is only targeted to the breakpoint regions:

After comparing SNPs from the individual with the new SNP core file, in the breakpoint region:

- SNPs with the same genotype are grouped if there are at least five SNPs indicating the same genotype
- Windows with more than 10 SNPs are used as seeds. Adjacent smaller windows are joined.
- When a change in the genotype between ‘seed’ windows is observed, the breakpoint is declared and the new coordinates are annotated.

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
Created on Mon Nov  3 11:37:10 2014

@author: mtormo
"""

import argparse
import sqlite3
import sys
#import vcf

from collections import defaultdict
from collections import OrderedDict as od
from find_regions_from_genotyping import CreateBd
from find_regions_from_genotyping import FillET
from find_regions_from_genotyping import FillT
from find_regions_from_genotyping import vcf2bd

def InputPar():#input parameters
    parser = argparse.ArgumentParser(description='Fill recombination points with new filtered positions, join Ts, and start-end of the chromosome')
    parser.add_argument('--base', metavar='BASENAME' ,required=True,
                            help='Basename for input file XXX.recomb.tab and XXX.new_results.tab (Required)')
    parser.add_argument('--qual', metavar='INT', type=int,default=60,
                          help='Quality of results file to find positions in [60]')
    parser.add_argument('--qual_file', metavar='FILE' ,required=True,
                            help='New_results file with quality defined in --qual (Required)')
    parser.add_argument('--vcf', metavar='FILE' ,required=True,
                            help='Filtered vcf file from individual (Required)')
    parser.add_argument('--genos_file', metavar='FILE', required=True,
                          help='File with table of positions from 4 genotypes (Required)')
    parser.add_argument('--out', metavar='BASENAME', required=True,
                          help='Output basename file obtaining OUT.recomb.tab and OUT.new_results.tab (Required)')
    parser.add_argument('--lim', metavar='INT', type=int,default=10,
                          help='Limit of nucleotides to accept windows [10]')

    return parser.parse_args()


def GetRecpoint(param):###get recombinations points between genotypes, or empty regions
    regions=[]###crom,pos1,pos2,recomb

    with open('.'.join([param.base,'recomb.tab'])) as recomb:
        l_rec = recomb.readlines()

    l_true = False
    pos_list=[]

    for n in  range(len(l_rec)):
        rec = l_rec[n].rstrip().split('\t')


        if not l_true:
            if rec[0].find('###All')==0:
                l_true=True
        else:
            try:
                next_rec = l_rec[n+1].rstrip().split('\t')

            except:
```

```
                next_rec = None
#            print rec
            pos_list.extend((rec[1],rec[2]))

            if next_rec and rec[0] == next_rec[0]:###if it's the same chromosome
                if rec[3] == next_rec[3] or ('Low_var' in rec[3] and rec[2] == next_rec[1]) or \
                    ('Low_var' in next_rec[3] and rec[2] == next_rec[1] and not (rec[3]=='Undef' and rec[1]=='0') ):
                    pos_list=[]
                    continue
                elif rec[3]=='Undef':
                    regions.append('/'.join([rec[0],pos_list[0],next_rec[1],rec[3]+'>'+next_rec[3]]))
                elif next_rec[3]=='Undef':
                    regions.append('/'.join([rec[0],rec[2],next_rec[2],rec[3]+'>'+next_rec[3]]))
                elif 'Low_var' in rec[3] and rec[2] != next_rec[1]:
                    regions.append('/'.join([rec[0],rec[2],next_rec[1],rec[3]+'>'+next_rec[3]]))
                else:
                    regions.append('/'.join([rec[0],pos_list[-1],next_rec[1],rec[3]+'>'+next_rec[3]]))
#            print regions
            pos_list=[]

    return regions,l_rec


def NewRecomb(l_new,regions):
    pos_list=[]
    gt_count=0
    coord=defaultdict(set)
    all_set = set()###set to compare all file with new lines (later)

    check_pos=set()
    for reg in regions:
        crom,pos1,pos2,reco = reg.split('/')

        if crom+'/'+pos1+'/'+pos2 in check_pos:
            continue
        else:
            check_pos.add(crom+'/'+pos1+'/'+pos2)

            done=False
            for l in range(len(l_new)):
                filt = l_new[l].rstrip().split('\t')
                all_set.add(filt[0]+'/'+str(filt[1])+'/'+filt[2])

                gt_index = 'E'
                if filt[2].find('T')>=0:
                    gt_index = 'T'

                if filt[0] == crom and int(pos1) < int(filt[1]) < int(pos2):
                    try:
                        next_crom = l_new[l+1].rstrip().split('\t')[0]
                        next_pos = l_new[l+1].rstrip().split('\t')[1]
                        next_gt_index = 'E'
                        if l_new[l+1].rstrip().split('\t')[2].find('T')>=0:
                            next_gt_index = 'T'
                    except:
                        next_crom=None
                        next_pos=None
                        next_gt_index = None

                    if filt[0] == next_crom and gt_index == next_gt_index and next_pos and int(pos1) < int(next_pos) < int(pos2):###same window

                        pos_list.append(filt[1])
                        gt_count += 1

```

```
                        else:###close window
                            pos_list.append(filt[1])
                            gt_count += 1
                            if gt_count > 5:###window is bigger than 1
                                coord[crom].add('/'.join([gt_index,pos_list[0],pos_list[-1]]))

                                done=True

                            pos_list=[]
                            gt_count=0

                if done == False:###no markers for this region
                    coord[crom].add('/'.join(['NA',pos1,pos2]))
                    pos_list=[]
                    gt_count=0

            return coord,all_set

        def SortCoord(coord,regions):
            coord_sorted=od()

            for crom,c_set in sorted(coord.iteritems()):
                for cord in sorted(c_set, key=lambda x: (int(x.split('/')[1]))):###sort by chrom and pos
                    if crom in coord_sorted.keys():
                        coord_sorted[crom].append('/'.join([str(cord.split('/')[0]),str(cord.split('/')[1]),str(cord.split('/')[2])]))
                    else:
                        coord_sorted[crom]=[]
                        coord_sorted[crom].append('/'.join([str(cord.split('/')[0]),str(cord.split('/')[1]),str(cord.split('/')[2])]))

            fin_coord=od()
            check_pos=set()

            for reg in regions:
                pos_list=[]
                crom,pos1,pos2,reco = reg.split('/')
                if crom+'/'+pos1+'/'+pos2 in check_pos:
                    continue
                else:
                    check_pos.add(crom+'/'+pos1+'/'+pos2)

                if crom in coord_sorted.keys():
                    for l in range(len(coord_sorted[crom])):

                        r,p1,p2 = coord_sorted[crom][l].split('/')
                        try:
                            next_r,next_p1,next_p2 = coord_sorted[crom][l+1].split('/')
                        except:
                            next_r = next_p1 = next_p2 = None

                        pos_list.extend((p1,p2))

                        if int(pos1)<=int(p1) and int(p2)<=int(pos2):
                            if r == next_r and int(pos1)<=int(next_p1) and int(next_p2)<=int(pos2):
                                continue
                            else:
                                if crom in fin_coord.keys():

                                    fin_coord[crom].append('/'.join([r,pos_list[0],pos_list[-1]]))
                                else:
                                    fin_coord[crom]=[]
                                    fin_coord[crom].append('/'.join([r,pos_list[0],pos_list[-1]]))
                                    pos_list=[]
                        else:
                            pos_list=[]
                            continue
```

```
188.
189.                 else:
190.                     fin_coord[crom]=[]
191.                     fin_coord[crom].append('/'.join(['NA',po
    s1,pos2]))
192.             return fin_coord
193.
194.
195.         def RedefineT(n, distance,join_scafXposgtXn,all_set)
    :
196.             redef_scafXposgtXn=defaultdict(list)
197.             for cr,list_reg in sorted(join_scafXposgtXn.iter
    items()):
198.                 pos_list=[]
199.                 prev_gt=None
200.
201.                 for r in range(len(list_reg)):
202.                     gt,ori_pos1,ori_pos2 = list_reg[r].split
    ('/')
203.                     ori_pos1 = int(ori_pos1)
204.                     ori_pos2 = int(ori_pos2)
205.
206.                     try:###check next line
207.                         next_gt,next_pos1,next_pos2 = list_r
    eg[r+1].split('/')
208.
209.                     except:
210.                         next_gt = None
211.
212.                     pos_list=[]
213.                     prev_gt = gt
214.                     pos_list.extend((ori_pos1,ori_pos2))
215.
216.
217.                     if gt == 'E':
218.                         redef_scafXposgtXn[cr].append('/'.jo
    in(['E',str(pos_list[0]),str(pos_list[-1])]))
219.                     elif gt == 'T':
220.                         pos1=pos_list[0]
221.                         pos2=pos_list[-1]
```

```
222.
223.                         reg=[]
224.                         for elem in sorted(all_set, key=lamb
    da x: (x.split('/')[0],int(x.split('/')[1]))):
225.                             elem = elem.split('/')
226.                             if elem[0] == cr and int(pos1) <
    = int(elem[1]) <= int(pos2) and elem[2] == 'T':
227.                                 reg.append(int(elem[1]))
228.
229.                         whole_wind=[]
230.                         for ps in range(len(reg)):
231.                             posi = reg[ps]
232.
233.                             slid_win=[]
234.
235.                             init=posi
236.                             end=posi+distance
237.
238.                             for p in reg[ps:]:
239.                                 if init <= p <= end:
240.                                     slid_win.append(p)
241.                                 elif p > end:
242.                                     break
243.                             whole_wind.append(slid_win)
244.
245.                         pos_list=[]
246.                         cc=0
247.                         prev_check=''
248.                         for w in range(len(whole_wind)):
249.                             resu=whole_wind[w]
250.                             try:
251.                                 next_w = whole_wind[w+1]
252.                             except:
253.                                 next_w = None
254.
255.                             if len(resu) < n and next_w:
256.                                 check='>limit'
257.                             elif not next_w:
258.                                 check='last'
259.                             else:
```

```
                                    check='ok'

                                if check==prev_check or prev_che
    ck=='':
                                    cc+=1
                                    pos_list.extend([resu[0],res
    u[-1]])
                                elif not next_w:
                                    pos_list.extend([resu[0],res
    u[-1]])
                                    if pos_list[-1]-
    pos_list[0] > 4000000:
                                        redef_scafXposgtXn[cr].a
    ppend('/'.join(['E',str(pos_list[0]),str(pos_list[-1])]))

                                else:
                                    if prev_check == 'ok':
                                        redef_scafXposgtXn[cr].a
    ppend('/'.join(['T',str(pos_list[0]),str(resu[0])]))
                                    elif prev_check != 'ok' and
    pos_list[-1]-pos_list[0] > 4000000:
                                        redef_scafXposgtXn[cr].a
    ppend('/'.join(['E',str(pos_list[0]),str(pos_list[-1])]))

                                    pos_list=[]
                                    pos_list.extend([resu[0],res
    u[-1]])
                                    cc=1
                                prev_check=check

                    else:####NA
                        redef_scafXposgtXn[cr].append('/'.jo
    in(['NA',str(pos_list[0]),str(pos_list[-1])]))

            return redef_scafXposgtXn


        def Fill(param,coord,all_set):

            conn = sqlite3.connect(':memory:')
```

```

            vcf2bd(param.vcf,conn,'indv')
            CreateBd(param.genos_file,conn)

            out2_file = '.'.join([param.out,'rec_results','t
    ab'])
            out2 = open(out2_file, 'w')
            out_gt = set()

            out_gt,crompos = FillET(coord,out_gt,conn)
            out_gt = FillT(coord,all_set,out_gt,crompos)

            for newline in sorted(out_gt, key=lambda x: (x.s
    plit('/')[0],int(x.split('/')[1]))):###sort by chrom and po
    s
                out2.write('\t'.join(newline.split('/'))+'\n
    ')

            out2.close()
            return out2_file,conn


        def JoinWinGenos(scafXposgtXn,lim,cord_dict):
            join_scafXposgtXn=defaultdict(set)##dict [crom]
    [pos1-pos2 - gt] = N_of occurrencies

            for crom,dict1 in sorted(scafXposgtXn.iteritems(
    )):
                posgtXn=od()
                pos_list=[]
                gt_count=0

                for index,info in enumerate(dict1.iteritems(
    )):
                    posgt,n = info

                    try:
                        pos1, pos2, gt = posgt.split('-')
                    except:
                        posgtXn['NA'] = 0
```

```
                        join_scafXposgtXn[crom] = posgtXn
                        continue

                    try:
                        next_key = dict1.keys()[index+1]
                        next_gt = next_key.split('-')[2]
                    except:
                        next_gt = None

                    gt_count += n

                    if gt == next_gt:###same window
                        pos_list.extend((pos1,pos2))

                    else:###close window
                        pos_list.extend((pos1,pos2))
                        gt,crom_bothcord = gt.split('/')
                        cord_crom,cord1,cord2 = crom_bothcor
d.split('|')

                        posgtXn['-
'.join([pos_list[0],pos_list[-1],gt])] = gt_count
                        join_scafXposgtXn[crom] = posgtXn

                        pos_list=[]
                        gt_count=0

            return join_scafXposgtXn

        ###make windows
        def MakeWinGenos(filein,lim,coord):
            scafXposgtXn=od()##dict [crom] [pos1-
pos2 - gt]
            prev_scaf=None
            prev_gt=None
            pos_list=[]
            gt_count=0
            cord_dict={}###[crom-both_cord]=id_region

            with open(filein,'r') as infile:
                all_file = infile.readlines()

                for n in  range(len(all_file)):
                    crom,pos,gt_index = all_file[n].rstrip()
.split()

                    for each_c in coord[crom]:
                        if int(each_c.split('/')[1]) <= int(
pos) <= int(each_c.split('/')[2]):
                            region_id=each_c.split('/')[0]
                            coord_pos1 = int(each_c.split('/
')[1])
                            coord_pos2 = int(each_c.split('/
')[2])
                            both_coord='|'.join([crom,each_c
.split('/')[1],each_c.split('/')[2]])
                            cord_dict[both_coord]=region_id

                    if len(gt_index) == 2:###if it's E1 or E
2
                        gt_index = gt_index+'T'

                    try:
                        next_crom, next_pos, next_gt = all_f
ile[n+1].rstrip().split()
                        if len(next_gt) == 2:###if it's E1T
or E2T
                            next_gt = next_gt+'T'
                    except IndexError:
                        next_crom = next_pos = next_gt = Non
e

                    if crom != prev_scaf:
                        posgtXn=od()
                        gt_count=0
                        prev_gt=None

                    if gt_index=='T':
                        continue
```

```
390.
391.                     pos_list.append(pos)
392.
393.                     if crom == next_crom and gt_index == nex
    t_gt and (coord_pos1 <= int(next_pos) <= coord_pos2):###sam
    e window
394.                         gt_count += 1
395.
396.                     else:###close window
397.                         gt_count += 1
398.
399.                         if gt_index == 'E1E1' or gt_index ==
     'E1E2':
400.                             if gt_count > 5:###window is big
    ger than 10
401.                                 posgtXn['-
    '.join([pos_list[0],pos_list[-
    1],gt_index+'/'+both_coord])] = gt_count
402.                                 scafXposgtXn[crom] = posgtXn

403.                         else:
404.                                 posgtXn['-
    '.join([pos_list[0],pos_list[-
    1],gt_index+'/'+both_coord])] = gt_count
405.                                 scafXposgtXn[crom] = posgtXn

406.
407.                         pos_list=[]
408.                         gt_count=0
409.                     prev_scaf = crom
410.             join_scafXposgt = JoinWinGenos(scafXposgtXn,lim,
    cord_dict)
411.
412.             return join_scafXposgt,all_file
413.
414.         def LastRec(out2_file,lim,coord,out1):
415.
416.             join_scafXposgtXn,all_file = MakeWinGenos(out2_f
    ile,lim,coord)
417.             rec_recomb=[]
```

```
418.
419.             for crom,dict1 in sorted(join_scafXposgtXn.iteri
    tems()):
420.                 pos_list=[]
421.                 prev_gt=None
422.                 for index,info in enumerate(dict1.iteritems(
    )):
423.                     posgt,n = info
424.
425.                     if posgt=='NA':
426.                         out1.write('\t'.join([crom,'NA','-
    ','-'])+'\n')
427.                         pos_list=[]
428.                         continue
429.
430.                     pos1, pos2, gt = posgt.split('-')
431.         #             print crom,pos1,pos2,gt,n
432.                     try:###check next line
433.                         next_key = dict1.keys()[index+1]
434.                         next_pos1,next_pos2,next_gt = next_k
    ey.split('-')
435.
436.                     except:
437.                         next_key = None
438.                         next_gt = None
439.
440.                     try:
441.                         second_key = dict1.keys()[index+2]
442.                         sec_pos1,sec_pos2,sec_gt = second_ke
    y.split('-')
443.                     except:
444.                         second_key = None
445.                         sec_gt = None
446.
447.                     if prev_gt!=gt and n<lim and gt!='T':###
    below limit
448.                         continue
449.
450.                     pos_list.extend((pos1,pos2))
451.
```

```
452.
453.                     if next_key and second_key and gt != nex
    t_gt and dict1[next_key]<lim and gt == sec_gt and dict1[sec
    ond_key]>=5:
454.         #             if next_key and gt == next_gt:
455.                         prev_gt=gt
456.                         continue
457.                     else:
458.                         rec_recomb.append('/'.join([crom,pos
    _list[0],pos_list[-1],gt]))
459.         #                print '/'.join([crom,pos_list[0],po
    s_list[-1],gt])
460.                         pos_list=[]
461.                         prev_gt=None
462.             return all_file,rec_recomb
463.
464.
465.         def Ends(all_set, n,cr,ori_pos1,ori_pos2,distance,ou
    t1):
466.             reg=[]
467.         #    print cr,ori_pos1,ori_pos2
468.             reg=[]
469.             for elem in sorted(all_set, key=lambda x: (x.spl
    it('\t')[0],int(x.split('\t')[1]))):
470.                 elem = elem.split('\t')
471.                 if elem[0] == cr and int(ori_pos1) <= int(el
    em[1]) <= int(ori_pos2) and elem[2] == 'T\n':
472.                     reg.append(int(elem[1]))
473.
474.             whole_wind=[]
475.             for ps in range(len(reg)):
476.                 posi = reg[ps]
477.
478.                 slid_win=[]
479.
480.                 init=posi
481.                 end=posi+distance
482.
483.                 for p in reg[ps:]:
484.                     if init <= p <= end:
485.                         slid_win.append(p)
486.                     elif p > end:
487.                         break
488.                 whole_wind.append(slid_win)
489.
490.             pos_list=[]
491.             refine=[]
492.             cc=0
493.             prev_check=''
494.             for w in range(len(whole_wind)):
495.                 resu=whole_wind[w]
496.                 try:
497.                     next_w = whole_wind[w+1]
498.                 except:
499.                     next_w = None
500.
501.                 if len(resu) < n and next_w:
502.                     check='>limit'
503.                 elif not next_w:
504.                     check='last'
505.                 else:
506.                     check='ok'
507.
508.                 if check==prev_check or prev_check=='':
509.                     cc+=1
510.                     pos_list.extend([resu[0],resu[-1]])
511.                 elif not next_w:
512.                     pos_list.extend([resu[0],resu[-1]])
513.                     if pos_list[-1]-pos_list[0] > 4000000:
514.                         refine.append([cr,str(pos_list[0]),s
    tr(pos_list[-1]),'E'])
515.
516.                 else:
517.                     if prev_check == 'ok':
518.                         refine.append([cr,str(pos_list[0]),s
    tr(resu[0]),'T'])
519.                     elif prev_check != 'ok' and pos_list[-
    1]-pos_list[0] > 4000000:
520.                         refine.append([cr ,str(pos_list[0]),
    str(pos_list[-1]),'E'])
```

```
521.
522.                     pos_list=[]
523.                     pos_list.extend([resu[0],resu[-1]])
524.                     cc=1
525.                 prev_check=check
526.
527.             return refine
528.
529.
530.
531.         def CountPos(rec_results,cr,ori_pos1,ori_pos2,out1):

532.             gt_dict={}
533.             for res in rec_results:
534.                 res = res.rstrip().split()
535.                 if cr == res[0] and int(ori_pos1) < int(res[
    1]) < int(ori_pos2):
536.                     if res[2] in gt_dict.keys():
537.                         gt_dict[res[2]] += 1
538.                     else:
539.                         gt_dict[res[2]] = 1
540.
541.
542.             if len(gt_dict)>0:
543.                 p_dict=''
544.                 for k,v in gt_dict.iteritems():
545.                     p_dict=p_dict+'/'+k+':'+str(v)
546.                 out1.write('\t'.join([cr,str(ori_pos1),str(o
    ri_pos2),p_dict])+'\n')
547.         #        print 'count1',[cr,str(ori_pos1),str(ori_po
    s2),p_dict]
548.             else:
549.         #        print 'count2',[cr,str(ori_pos1),str(ori_po
    s2),'0']
550.                 out1.write('\t'.join([cr,str(ori_pos1),str(o
    ri_pos2),'0'])+'\n')
551.
552.
553.         def PrintRecomb(rec_results,rec_recomb,original_rec,
    out1,n,distance,conn):
```

```
554.             l_true = False
555.             done = False
556.             toprint=[]
557.             t_reg=[]
558.
559.             for g in range(len(original_rec)):###print old r
    ecomb and save positions to use with recpoints
560.
561.                 rec = original_rec[g].rstrip().split('\t')
562.
563.                 if rec[0].find('#')==-
    1 and rec[1].find('T')>=0:###define T regions from E-
    T recombination
564.                     if rec[1] == 'only_T' or rec[1] == 'T>E'
    :
565.                         if not [rec[0],rec[2],rec[3]] in t_r
    eg:
566.                             t_reg.append([rec[0],rec[2],rec[
    3]])
567.                     elif rec[1] == 'E>T':
568.                         if not [rec[0],rec[4],rec[5]] in t_r
    eg:
569.                             t_reg.append([rec[0],rec[4],rec[
    5]])
570.
571.                 if not l_true:
572.                     if rec[0].find('###All')==0:
573.                         l_true=True
574.                 else:
575.                     try:
576.                         next_rec = original_rec[g+1].rstrip(
    ).split('\t')
577.                     except:
578.                         next_rec = None
579.                     if rec[3] == 'Undef' or (next_rec and re
    c[0]==next_rec[0] and rec[2]!=next_rec[1]):
580.                         if rec[3] == 'Undef':
581.                             ori_pos1 = int(rec[1])
582.                             ori_pos2 = int(rec[2])
583.                         else:
```

```
584.                             ori_pos1 = int(rec[2])
585.                             ori_pos2 = int(next_rec[1])
586.
587.                             toprint.append(rec)
588.       #                      print 'entra1'
589.
590.                         for recpoint in rec_recomb:
591.                             r_crom,r_pos1,r_pos2,r_gt = recp
    oint.split('/')
592.       #                      print '\t',recpoint
593.
594.                             if r_crom == rec[0] and (ori_pos
    1 <= int(r_pos1) <= ori_pos2 or ori_pos1 <= int(r_pos2) <=
    ori_pos2):###overlapping regions and recpoints
595.                                 done=True
596.                                 if ori_pos1 <= int(r_pos1) <
    = ori_pos2 and int(r_pos2) > ori_pos2:###r_pos1 inside, r_p
    os2 outside
597.                                     r_pos2=str(ori_pos2)
598.                                 if ori_pos1 > int(r_pos1) an
    d ori_pos1 <= int(r_pos2) <= ori_pos2:###r_pos1 outside, r_
    pos1 inside
599.                                     r_pos1=str(ori_pos1)
600.
601.                                 if ori_pos1 <= int(r_pos1) <
    = ori_pos2 and ori_pos1 <= int(r_pos2) <= ori_pos2:###recom
    bination inside region
602.       #                             print 'entra2'
603.                                     if ori_pos1 == int(r_pos
    1):
604.       #                                 print [r_crom,r_pos
    1,r_pos2,r_gt]
605.       #                                 print [r_crom,r_pos
    2,str(ori_pos2),'Undef']
606.                                         toprint.append([r_cr
    om,r_pos1,r_pos2,r_gt])
607.       #                                 toprint.append(recp
    oint.split('/'))
608.                                         toprint.append([r_cr
    om,r_pos2,str(ori_pos2),'Undef'])
609.                                         ori_pos1=int(r_pos2)
610.                                     elif ori_pos2 == int(r_p
    os2):
611.                                         if len(toprint) > 0
    and str(ori_pos1) == toprint[-1][1] and rec != toprint[-
    1]:
612.                                             del toprint[-
    1]
613.       #                                 print [r_crom,str(o
    ri_pos1),r_pos1,'Undef']
614.       #                                 print [r_crom,r_pos
    1,r_pos2,r_gt]
615.                                         toprint.append([r_cr
    om,str(ori_pos1),r_pos1,'Undef'])
616.                                         toprint.append([r_cr
    om,r_pos1,r_pos2,r_gt])
617.       #                                 toprint.append(recp
    oint.split('/'))
618.                                     else:
619.                                         if len(toprint) > 0
    and str(ori_pos1) == toprint[-1][1] and rec != toprint[-
    1]:
620.                                             del toprint[-
    1]
621.       #                                 print [r_crom,str(o
    ri_pos1),r_pos1,'Undef']
622.       #                                 print [r_crom,r_pos
    1,r_pos2,r_gt]
623.       #                                 print [r_crom,r_pos
    2,str(ori_pos2),'Undef']
624.                                         toprint.append([r_cr
    om,str(ori_pos1),r_pos1,'Undef'])
625.                                         toprint.append([r_cr
    om,r_pos1,r_pos2,r_gt])
626.       #                                 toprint.append(recp
    oint.split('/'))
627.                                         toprint.append([r_cr
    om,r_pos2,str(ori_pos2),'Undef'])
```

```
                                        ori_pos1=int(r_pos2)

                        if done==False:###Undef
        #                           print 'entra5',[rec[0],str(ori_pos1),str(ori_pos2),'Undef']
                                    toprint.append([rec[0],str(ori_pos1),str(ori_pos2),'Undef'])
                    else:
        #                print 'entra6',rec
                        toprint.append(rec)

                    done = False


            out1.write('###Last_recombination###\n')
            for p in range(len(toprint)):
                cr,p1,p2,re = toprint[p]
        #        print cr,p1,p2,re
                done=False
                try:
                    prev_cr,prev_p1,prev_p2,prev_re = toprint[p-1]
                except:
                    prev_cr = prev_re = None
                try:
                    next_cr,next_p1,next_p2,next_re = toprint[p+1]
                except:
                    next_cr = next_re = None


                if 'Low_var' in re:

                    if prev_cr and prev_cr == cr and cr != next_cr:###last part of the chromosome
                        for reg in t_reg:
                            if reg[0] == cr and int(reg[2]) > int(p1) and int(reg[2]) < int(p2):
                                done=True
                                out1.write('\t'.join([cr,p1,reg[2],'T-Low_var'])+'\n')
                                out1.write('\t'.join([cr,reg[2],p2,re])+'\n')

                    elif next_cr and next_cr == cr and cr != prev_cr:###first part of the chromosome
                        for reg in t_reg:
                            if reg[0] == cr and int(reg[1]) > int(p1) and int(reg[1]) < int(p2):
                                done=True
                                out1.write('\t'.join([cr,p1,reg[1],re])+'\n')
                                out1.write('\t'.join([cr,reg[1],p2,'T-Low_var'])+'\n')

                elif 'Undef' in re:
                    if prev_cr != cr or next_cr != cr:

                        refine = Ends(rec_results, n,cr,p1,p2,distance,out1)
                        for ref in range(len(refine)):
                            each_ref = refine[ref]

                            try:
                                next_ref=refine[ref+1]
                            except:
                                next_ref=None

                            if each_ref[0] == cr and int(each_ref[2]) > int(p1) and int(each_ref[2]) < int(p2):
                                done=True

                                if int(p1) == 0 and int(each_ref[1])-
                int(p1) > 1000000:###first range of the cromosome
                                    ecoord={}
                                    ecoord[cr]=['/'.join(['E',p1,each_ref[1]])]
```

```
                                    out_gt_e=set()
                                    out_gt_e,crompos = FillET(ecoord,out_gt_e,conn)
                                    for newline in sorted(out_gt_e, key=lambda x: (x.split('/')[0],int(x.split('/')[1])
                                    ):###sort by chrom and pos
                                        newline = newline.split('/')
                                        out1.write('\t'.join([newline[0],newline[1],newline[1],newline[2]])+'\n')
                                else:
                                    CountPos(rec_results,cr,p1,each_ref[1],out1)

                                out1.write('\t'.join([cr,each_ref[1],each_ref[2],each_ref[3]])+'\n')

                                if not next_ref:###last range

                                    if int(p2)-int(each_ref[2]) > 1000000:
                                        ecoord={}
                                        ecoord[cr]=['/'.join(['E',each_ref[2],p2])]
                                        out_gt_e=set()
                                        out_gt_e,crompos = FillET(ecoord,out_gt_e,conn)
                                        for newline in sorted(out_gt_e, key=lambda x: (x.split('/')[0],int(x.split('/')[1]))):###sort by chrom and pos
                                            newline = newline.split('/')
                                            out1.write('\t'.join([newline[0],newline[1],newline[1],newline[2]])+'\n')
                                    else:
                                        CountPos(rec_results,cr,each_ref[2],p2,out1)

                                else:
                                    p1=each_ref[2]


                        if done==False:###not in T region
                            done=True
                            CountPos(rec_results,cr,p1,p2,out1)


                    else:
                        for reg in t_reg:
                            if reg[0] == cr and int(reg[2]) > int(p1) and int(reg[2]) < int(p2):
                                done=True
                                out1.write('\t'.join([cr,p1,reg[2],'T'])+'\n')
                        #                        print 'entra1',[cr,p1,reg[2],'T']
                                CountPos(rec_results,cr,reg[2],p2,out1)

                            elif reg[0] == cr and int(reg[1]) > int(p1) and int(reg[1]) < int(p2):
                                done=True
                                CountPos(rec_results,cr,p1,reg[1],out1)
                                out1.write('\t'.join([cr,reg[1],p2,'T'])+'\n')
                        #                        print 'entra2',[cr,reg[1],p2,'T']

                            elif reg[0] == cr and int(reg[1]) < int(p1) and int(reg[2]) > int(p2):
                        #                        print 'entra3',[cr,p1,p2,'T']
                                done=True
                                out1.write('\t'.join([cr,p1,p2,'T'])+'\n')

                        if done==False:###not in T region
```

```
742.       #                    print 'entra5'
743.                           done=True
744.                           CountPos(rec_results,cr,p1,p2,ou
    t1)
745.
746.               if done==False:
747.       #            print 'entra_fals',toprint[p]
748.                   out1.write('\t'.join(toprint[p])+'\n')
749.
750.
751.       def main():
752.           param=InputPar()
753.
754.           regions,original_rec = GetRecpoint(param)
755.
756.           with open('.'.join([param.qual_file])) as new:
757.               l_new = new.readlines()
758.
759.           coord,all_set = NewRecomb(l_new,regions)
760.
761.           fin_coord = SortCoord(coord,regions)
762.
763.           n=10
764.           distance=100000
765.           redef_coord = RedefineT(n, distance,fin_coord,al
    l_set)
766.
767.
768.           out2_file,conn = Fill(param,redef_coord,all_set)

769.
770.
771.           out1 = open('.'.join([param.out,'recomb','tab'])
    , 'w')
772.
773.
774.           rec_results,rec_recomb = LastRec(out2_file,param
    .lim,redef_coord,out1)
775.
776.           PrintRecomb(rec_results,rec_recomb,original_rec,
    out1,n, distance,conn)
777.
778.
779.
780.       if __name__ == "__main__":
781.           main()
```

Annex 4

‘Genotype_by_absence.py’ (GBA)

```
# -*- coding: utf-8 -*-
import argparse
from operator import itemgetter
from bisect import bisect_left

parser = argparse.ArgumentParser(description='Genotypes any individual from T1E population. Takes into acount the absence of T or E2 alleles to determine a recomb (E1<->T or E1<->E2). Takes as input the table of position for the 4 genotypes and the vcf file of the individual to genotype.')

parser.add_argument('--table_positions', metavar='FILE', help='Table with SNP information for the 4 genotypes')
parser.add_argument('--vcf_filt', metavar='FILE', help='VCF file filtered for a specific individual')
parser.add_argument('--outfile', metavar='FILE', help='Name of the output')
parser.add_argument('--results', metavar='FILE', help='Creates a results file with all the SNP for the individual')

args = parser.parse_args()

IND_ID = args.vcf_filt.split('filtered/')[1].split('_filt')[0]
#IND_ID = args.vcf_filt.split('_filt')[0]

E1_SNPs = {}
E2_SNPs = {}
T_SNPs = {}


def find_E1(E1E2, E1T, E2T):
    E1_allele = ''

    if (E1E2 != '-') and (E2T == '-') and (E1T != '-'):
        E1_allele = E1E2[0]

    elif (E1E2 != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        E1_allele = E1E2[0]

    else:
        E1_allele = '-'

    return E1_allele

##################################################


def find_E2(E1E2, E1T, E2T):
    E2_allele = ''

    if (E1E2 != '-') and (E1T == '-') and (E2T != '-'):
        E2_allele = E1E2[1]
    elif (E1E2 != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        E2_allele = E1E2[1]
    else:
        E2_allele = '-'

    return E2_allele


##################################################

def find_T(E1E2, E1T, E2T):
    T_allele = ''

    if (E1E2 == '-') and (E1T != '-') and (E2T != '-'):
        T_allele = E1T[1]
    elif (E2T != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        T_allele = E2T[1]
    else:
        T_allele = '-'

```

```
    return T_allele


#########################################################

def genotype_SNPs(vcf):

    EG_allele = []
    MB_allele = []
    UNSURE = []

    for line in vcf:

        if not line.startswith("Pp"):
            continue
        else:

            CHR = (line.strip().split("\t")[0])
            POS = int(line.strip().split("\t")[1])
            REF = line.strip().split("\t")[3]
            ALT = line.strip().split("\t")[4]
            GENO = (line.strip().split("\t")[9])[0:3]

            ID = CHR+":"+str(POS)


            if GENO == "1/1":
                IND = ALT+ALT
            elif GENO == "0/1":
                IND = REF+ALT

            if E1_SNPs.get(ID) != None:

                E1_E2_T = [E1_SNPs.get(ID), E2_SNPs.get(ID), T_SNPs.get(ID)]


                    if (IND[0] in E1_E2_T) or (IND[1] in E1_E2_T):

                    if (IND[0] == E1_E2_T[0]) or (IND[1] == E1_E2_T[0]):
                    UNSURE.append((CHR, POS, "-"+":"+"-"+":"+"E1"))

                    elif (IND[0] == E1_E2_T[1]) or (IND[1] == E1_E2_T[1]):
                                EG_allele.append((CHR, POS, "E2"+":"+"-"+":"+"-"))

                    else:
                    MB_allele.append((CHR, POS, "-"+":"+"T"+":"+"-"))


            return MB_allele+EG_allele+UNSURE


        ##########################################################################

        def cluster(data, maxgap): #Arrange data into groups where successive elements differ by no more than *maxgap*

            data.sort()
            groups = [[data[0]]]
            for x in data[1:]:
                if abs(x - groups[-1][-1]) <= maxgap:
                    groups[-1].append(x)
                else:
                    groups.append([x])
            return groups
```

```
133.       ##################################################
   ###################################
134.       def filter_clusters(group, min_SNPs_per_cluster):
135.
136.           sizes = []
137.
138.           for i in range(len(group)):
139.               sizes.append(len(group[i]))
140.
141.           good_group = []
142.           for a in sizes:
143.               if a > min_SNPs_per_cluster:
144.                   good_group.append(group[sizes.index(a)])
    # delete all groups of close positions with less than x ma
   rkers
145.           return good_group
146.
147.
148.       ##################################################
   ###################################
149.
150.
151.       def get_SNP_ind(FLsorted, x, y):
152.           chromosomes = ['Pp01','Pp02','Pp03','Pp04','Pp05
   ','Pp06','Pp07','Pp08']
153.           if x == "T":
154.               a = 1
155.           elif x == "E2":
156.               a = 0
157.           elif x == "E1":
158.               a = 2
159.           POS = []
160.           SNP = 0
161.           dict_of_POS = {}
162.           for CHR in chromosomes:
163.               for i in FLsorted:
164.
165.                   if CHR == i[0]:
166.                       if i[2].split(":")[a] == x:
167.                           SNP += 1
168.                           POS.append(i[1])
169.               if SNP > y:
170.                   dict_of_POS[CHR] = POS
171.                   POS = []
172.                   SNP = 0
173.               else:
174.                   dict_of_POS[CHR] = 0
175.                   POS = []
176.                   SNP = 0
177.
178.           return dict_of_POS
179.
180.
181.
182.       ##################################################
   ####
183.       def sum_up_filt_clusters_T(key, filt_cluster, dist_f
   ilt_clusters):
184.           regions = []
185.           how_many_groups = len(filt_cluster)
186.           index_max = filt_cluster.index(max(filt_cluster)
   )
187.           remaining_top_groups = how_many_groups - 1 - ind
   ex_max
188.           remaining_bot_groups = index_max
189.           starting_group = []
190.
191.
192.           if remaining_top_groups > 0:
193.               starting_group = filt_cluster[index_max]
194.               next_group = filt_cluster[index_max+1]
195.
196.
197.               for i in range(remaining_top_groups):
198.
199.                   last_snp = starting_group[-1]
200.                   closest_snp = next_group[0]
201.
202.                   if (closest_snp - last_snp) < dist_filt_
   clusters:
```

```
                        starting_group = starting_group + next_group
                        starting_group.sort()

                        if remaining_top_groups == 1:
                            break
                        else:
                            next_group = filt_cluster[index_max+2+i]
                    else:
                        regions.append(starting_group)
                        starting_group = next_group
                        starting_group.sort()

                        if remaining_top_groups == 1:
                            break
                        else:
                            next_group = filt_cluster[index_max+2+i]


                if remaining_bot_groups > 0:
                    previous_group = filt_cluster[index_max-1]


                    for i in range(remaining_bot_groups):

                        last_snp = starting_group[0]
                        closest_snp = previous_group[-1]

                        if (last_snp - closest_snp) < dist_filt_clusters:
                            starting_group = starting_group + previous_group
                            starting_group.sort()

                            if remaining_bot_groups == 1:
                                break
                            else:
                                previous_group = filt_cluster[index_max-2-i]

                        else:
                            regions.append(starting_group)
                            starting_group = previous_group

                            starting_group.sort()

                            if remaining_bot_groups == 1:
                                break
                            else:
                                previous_group = filt_cluster[index_max-2-i]


                regions.append(starting_group)


            else:
                if remaining_bot_groups > 0:
                    starting_group = filt_cluster[index_max]

                    previous_group = filt_cluster[index_max-1]


                    for i in range(remaining_bot_groups):

                        last_snp = starting_group[0]
                        closest_snp = previous_group[-1]

                        if (last_snp - closest_snp) < dist_filt_clusters:
                            starting_group = starting_group + previous_group
                            starting_group.sort()

                            if remaining_bot_groups == 1:
```

```
                                    break
                                else:
                                    previous_group = filt_cluste
    r[index_max-2-i]

                            else:
                                regions.append(starting_group)
                                starting_group = previous_group

                                starting_group.sort()

                                if remaining_bot_groups - i == 1
    :
                                    break
                                else:
                                    previous_group = filt_cluste
    r[index_max-2-i]

                    regions.append(starting_group)

                return regions

            ##################################################
    ####
            def sum_up_filt_clusters_E1(key, filt_cluster, dist_
    filt_clusters):
                EBD = {'Pp01': ['2500000-7400000', '17500000-
    20000000', '23000000-30000000', '30500000-
    35000000', '36000000-47851208'],
                       'Pp02': ['0-1100000', '1600000-
    5500000', '13600000-16700000', '18500000-
    21700000', '21900000-29400000'],
                       'Pp03': ['1800000-6600000', '8000000-
    11000000', '19600000-22000000'],
                       'Pp04': ['0-1000000', '9700000-
    25843236'],
                       'Pp05': ['0-9400000', '10600000-
    12800000', '13100000-14400000', '15600000-18496696'],
                       'Pp06': ['0-10500000', '11500000-
    13500000', '13800000-23000000', '24500000-
    26200000', '27800000-30767194'],
                       'Pp07': ['20000000-22388614'],
                       'Pp08': ['0-4100000', '19600000-
    22573980']}

                regions = []

                how_many_groups = len(filt_cluster)
                index_max = filt_cluster.index(max(filt_cluster)
    )
                remaining_top_groups = how_many_groups - 1 - ind
    ex_max
                remaining_bot_groups = index_max
                starting_group = []
                next_group = []
                previous_group = []


                if remaining_top_groups > 0:
                    starting_group = filt_cluster[index_max]
                    next_group = filt_cluster[index_max+1]


                    for i in range(remaining_top_groups):

                        last_snp = starting_group[-1]
                        closest_snp = next_group[0]

                        if (closest_snp - last_snp) < dist_filt_
    clusters:
                            starting_group = starting_group + ne
    xt_group
                            starting_group.sort()

                            if remaining_top_groups - i == 1:
                                break
                            else:
```

```
                            next_group = filt_cluster[index_max+2+i]
                    else:
                        solved = 0
                        for Pp, dist in EBD.iteritems():
                            if key == Pp:
                                for coord in dist:

                                    start_EBD = int(coord.split("-")[0])
                                    end_EBD = int(coord.split("-")[1])

                                    midpoint = closest_snp-((closest_snp-last_snp)/2)

                                    if (midpoint > start_EBD) and (midpoint < end_EBD):
                                        starting_group = starting_group + next_group
                                        starting_group.sort()
                                        solved = 1
                        if solved == 0:
                            regions.append(starting_group)
                            starting_group = next_group
                            if remaining_top_groups - i == 1:
                                regions.append(starting_group)
                                break
                            else:
                                next_group = filt_cluster[index_max+2+i]


                        if remaining_top_groups - i == 1:
                            break
                        else:
```

```
                            next_group = filt_cluster[index_max+2+i]


                if len(regions) == 0:

                    if remaining_bot_groups > 0:
                        previous_group = filt_cluster[index_max-1]


                    for i in range(remaining_bot_groups):

                        last_snp = starting_group[0]
                        closest_snp = previous_group[-1]

                        if (last_snp - closest_snp) < dist_filt_clusters:
                            starting_group = starting_group + previous_group
                            starting_group.sort()

                            if remaining_bot_groups - i == 1:
                                break
                            else:
                                previous_group = filt_cluster[index_max-2-i]
                        else:
                            solved = 0
                            for Pp, dist in EBD.iteritems():
                                if key == Pp:
                                    for coord in dist:
                                        start_EBD = int(coord.split("-")[0])
                                        end_EBD = int(coord.split("-")[1])
```

```
                                                midpoint = last_snp-((last_snp-closest_snp)/2)

                                                if (midpoint > start_EBD) and (midpoint < end_EBD):
                                                    starting_group = starting_group + previous_group
                                                    starting_group.sort()
                                                    solved = 1
                                        if solved == 0:
                                            regions.append(starting_group)
                                            starting_group = previous_group

                                            if remaining_bot_groups - i == 1:
                                                regions.append(starting_group)
                                            else:
                                                previous_group = filt_cluster[index_max-2-i]

                                        if remaining_bot_groups - i == 1:
                                            break
                                        else:
                                            previous_group = filt_cluster[index_max-2-i]


                            regions.append(starting_group)

                        else:
                            if remaining_bot_groups > 0:
                                starting_group = filt_cluster[index_max]
                                previous_group = filt_cluster[index_max-1]

                                for i in range(remaining_bot_groups):

                                    last_snp = starting_group[0]
                                    closest_snp = previous_group[-1]

                                    if (last_snp - closest_snp) < dist_filt_clusters:
                                        starting_group = starting_group + previous_group
                                        starting_group.sort()

                                        if remaining_bot_groups - i == 1:
                                            break
                                        else:
                                            previous_group = filt_cluster[index_max-2-i]
                                    else:
                                        solved = 0
                                        for Pp, dist in EBD.iteritems():
                                            if key == Pp:
                                                for coord in dist:
                                                    start_EBD = int(coord.split("-")[0])
                                                    end_EBD = int(coord.split("-")[1])

                                                    midpoint = last_snp-((last_snp-closest_snp)/2)

                                                    if (midpoint > start_EBD) and (midpoint < end_EBD):
                                                        starting_group = starting_group + previous_group
```

```
                                                starting_gro
    up.sort()
                                                solved = 1

                                    if solved == 0:
                                        regions.append(starting_
    group)
                                        starting_group = previou
    s_group

                                        if remaining_bot_groups
    - i == 1:
                                            regions.append(start
    ing_group)
                                        else:
                                            previous_group = fil
    t_cluster[index_max-2-i]

                                    if remaining_bot_groups - i
    == 1:
                                        break
                                    else:
                                        previous_group = filt_cl
    uster[index_max-2-i]


                else:
                    if remaining_bot_groups > 0:
                        starting_group = filt_cluster[index_max]

                        previous_group = filt_cluster[index_max-
    1]


                        for i in range(remaining_bot_groups):

                            last_snp = starting_group[0]
                            closest_snp = previous_group[-1]

                            if (last_snp - closest_snp) < dist_f
    ilt_clusters:
                                starting_group = starting_group
    + previous_group
                                starting_group.sort()

                                if remaining_bot_groups == 1:
                                    break
                                else:
                                    previous_group = filt_cluste
    r[index_max-2-i]
                            else:
                                solved = 0
                                for Pp, dist in EBD.iteritems():

                                    if key == Pp:
                                        for coord in dist:
                                            start_EBD = int(coor
    d.split("-")[0])
                                            end_EBD = int(coord.
    split("-")[1])

                                            midpoint = last_snp-
    ((last_snp-closest_snp)/2)

                                            if (midpoint > start
    _EBD) and (midpoint < end_EBD):
                                                starting_group =
     starting_group + previous_group
                                                starting_group.s
    ort()
                                                solved = 1


                                if solved == 0:
                                        regions.append(starting_
    group)
                                        starting_group = previou
    s_group
```

```
                                    previous_group = filt_cl
uster[index_max-2-i]

                            if remaining_bot_groups == 1:
                                break
                            else:
                                previous_group = filt_cluste
r[index_max-2-i]


                regions.append(starting_group)

            return regions

        ##################################################
####

        def sum_up_filt_clusters_E2(key, filt_cluster, dist_
filt_clusters):
            EBD = {'Pp01': ['2500000-7400000', '17500000-
20000000', '23000000-30000000', '30500000-
35000000', '36000000-47851208'],
                   'Pp02': ['0-1100000', '1600000-
5500000', '13600000-16700000', '18500000-
21700000', '21900000-29400000'],
                   'Pp03': ['1800000-6600000', '8000000-
11000000', '19600000-22000000', '26800000-27368013'],
                   'Pp04': ['0-1000000', '9700000-
25843236'],
                   'Pp05': ['0-9400000', '10600000-
12800000', '13100000-14400000', '15600000-18496696'],
                   'Pp06': ['0-10500000', '11500000-
13500000', '13800000-23000000', '24500000-
26200000', '27800000-30767194'],
                   'Pp07': ['20000000-22388614'],
                   'Pp08': ['0-4100000', '19600000-
22573980']}

            regions = []
```

```

            how_many_groups = len(filt_cluster)
            index_max = filt_cluster.index(max(filt_cluster)
)
            remaining_top_groups = how_many_groups - 1 - ind
ex_max
            remaining_bot_groups = index_max
            starting_group = []
            next_group = []
            previous_group = []


            if remaining_top_groups > 0:
                starting_group = filt_cluster[index_max]
                next_group = filt_cluster[index_max+1]


                for i in range(remaining_top_groups):

                    last_snp = starting_group[-1]
                    closest_snp = next_group[0]

                    if (closest_snp - last_snp) < dist_filt_
clusters:
                        starting_group = starting_group + ne
xt_group
                        starting_group.sort()

                        if remaining_top_groups - i == 1:
                            break
                        else:
                            next_group = filt_cluster[index_
max+2+i]
                    else:
                        solved = 0
                        for Pp, dist in EBD.iteritems():
                            if key == Pp:
                                for coord in dist:

```

```
                                    start_EBD = int(coord.split("-")[0])
                                    end_EBD = int(coord.split("-")[1])

                                    midpoint = closest_snp-((closest_snp-last_snp)/2)

                                    if (midpoint > start_EBD) and (midpoint < end_EBD) and (closest_snp - end_EBD < 500000):
                                        starting_group = starting_group + next_group
                                        starting_group.sort()
                                        solved = 1
                            if solved == 0:
                                regions.append(starting_group)
                                starting_group = next_group
                                if remaining_top_groups - i == 1:
                                    regions.append(starting_group)
                                    break
                                else:
                                    next_group = filt_cluster[index_max+2+i]


                            if remaining_top_groups - i == 1:
                                break
                            else:
                                next_group = filt_cluster[index_max+2+i]


                    if len(regions) == 0:

                        if remaining_bot_groups > 0:
```

```
                        previous_group = filt_cluster[index_max-1]


                        for i in range(remaining_bot_groups):

                            last_snp = starting_group[0]
                            closest_snp = previous_group[-1]

                            if (last_snp - closest_snp) < dist_filt_clusters:
                                starting_group = starting_group + previous_group
                                starting_group.sort()

                                if remaining_bot_groups - i == 1:
                                    break
                                else:
                                    previous_group = filt_cluster[index_max-2-i]
                            else:
                                solved = 0
                                for Pp, dist in EBD.iteritems():
                                    if key == Pp:
                                        for coord in dist:
                                            start_EBD = int(coord.split("-")[0])
                                            end_EBD = int(coord.split("-")[1])

                                            midpoint = last_snp-((last_snp-closest_snp)/2)

                                            if (midpoint > start_EBD) and (midpoint < end_EBD) and (start_EBD - closest_snp < 500000):
```

```
                                                starting_group = starting_group + previous_group
                                                starting_group.sort()
                                                solved = 1
                                    if solved == 0:
                                        regions.append(starting_group)
                                        starting_group = previous_group

                                        if remaining_bot_groups - i == 1:
                                            regions.append(starting_group)
                                        else:
                                            previous_group = filt_cluster[index_max-2-i]

                                    if remaining_bot_groups - i == 1:
                                        break
                                    else:
                                        previous_group = filt_cluster[index_max-2-i]


                        regions.append(starting_group)

                    else:
                        if remaining_bot_groups > 0:
                            starting_group = filt_cluster[index_max]
                            previous_group = filt_cluster[index_max-1]

                            for i in range(remaining_bot_groups):

                                last_snp = starting_group[0]
                                closest_snp = previous_group[-1]

                                if (last_snp - closest_snp) < dist_filt_clusters:
                                    starting_group = starting_group + previous_group
                                    starting_group.sort()

                                    if remaining_bot_groups - i == 1:
                                        break
                                    else:
                                        previous_group = filt_cluster[index_max-2-i]
                                else:
                                    solved = 0
                                    for Pp, dist in EBD.iteritems():
                                        if key == Pp:
                                            for coord in dist:
                                                start_EBD = int(coord.split("-")[0])
                                                end_EBD = int(coord.split("-")[1])

                                                midpoint = last_snp-((last_snp-closest_snp)/2)

                                                if (midpoint > start_EBD) and (midpoint < end_EBD) and (start_EBD - closest_snp < 500000):
                                                    starting_group = starting_group + previous_group
                                                    starting_group.sort()
                                                    solved = 1

                                    if solved == 0:
```

```
                                    regions.append(starting_
    group)
                                    starting_group = previou
    s_group
                                    if remaining_bot_groups
    - i == 1:
                                        regions.append(start
    ing_group)
                                    else:
                                        previous_group = fil
    t_cluster[index_max-2-i]
                                if remaining_bot_groups - i
    == 1:
                                    break
                                else:
                                    previous_group = filt_cl
    uster[index_max-2-i]


            else:
                if remaining_bot_groups > 0:
                    starting_group = filt_cluster[index_max]

                    previous_group = filt_cluster[index_max-
    1]


                    for i in range(remaining_bot_groups):

                        last_snp = starting_group[0]
                        closest_snp = previous_group[-1]

                        if (last_snp - closest_snp) < dist_f
    ilt_clusters:
                            starting_group = starting_group
    + previous_group
                            starting_group.sort()
```

```

                            if remaining_bot_groups == 1:
                                break
                            else:
                                previous_group = filt_cluste
    r[index_max-2-i]
                        else:
                            solved = 0
                            for Pp, dist in EBD.iteritems():

                                if key == Pp:
                                    for coord in dist:
                                        start_EBD = int(coor
    d.split("-")[0])
                                        end_EBD = int(coord.
    split("-")[1])

                                        midpoint = last_snp-
    ((last_snp-closest_snp)/2)

                                        if (midpoint > start
    _EBD) and (midpoint < end_EBD) and (start_EBD - closest_snp
     < 500000):
                                            starting_group =
     starting_group + previous_group
                                            starting_group.s
    ort()
                                            solved = 1


                            if solved == 0:
                                    regions.append(starting_
    group)
                                    starting_group = previou
    s_group
                                    previous_group = filt_cl
    uster[index_max-2-i]

                            if remaining_bot_groups == 1:
                                break
```

```
                            else:
                                previous_group = filt_cluste
    r[index_max-2-i]


                regions.append(starting_group)

            return regions

        ##################################################
    ####

        def define_T_Windows(IND_SNPs, dist_clusters, min_SN
    P_per_cluster, dist_filt_clusters):
            CHR_SIZE = [47851208, 30405870, 27368013, 258432
    36, 18496696, 30767194, 22388614, 22573980]

            dict_of_POS = get_SNP_ind(IND_SNPs, 'T', 500)

            T_windows = {}

            for key in dict_of_POS:
                T_region = []

                if dict_of_POS[key] == 0:
                    T_windows[key] = 0

                else:

                    group = cluster(dict_of_POS[key], dist_c
    lusters) # get clusters of closer positions

                    filt_cluster = filter_clusters(group, mi
    n_SNP_per_cluster)
                    filt_cluster.sort()


                    if len(filt_cluster) == 1:

                        T_region.append(filt_cluster[0])

                        T_windows[key] = T_region

                    else:
                        T_windows[key] = sum_up_filt_cluster
    s_T(key, filt_cluster, dist_filt_clusters)

            return T_windows



        ##################################################
    ################################

        def define_E2_Windows(IND_SNPs, dist_clusters, min_S
    NP_per_cluster, dist_filt_clusters):

            dict_of_POS = get_SNP_ind(IND_SNPs, 'E2', 50)


            E2_windows = {}

            for key in dict_of_POS:

                E2_region = []


                if dict_of_POS[key] == 0:
                    E2_windows[key] = 0

                else:

                    group = cluster(dict_of_POS[key], dist_c
    lusters) # get clusters of closer positions

                    filt_cluster = filter_clusters(group, mi
    n_SNP_per_cluster)
                    filt_cluster.sort()
```

```
793.
794.                 if len(filt_cluster) == 0:
795.
796.                 E2_windows[key] = 0
797.
798.                     elif len(filt_cluster) == 1:
799.
800.                         E2_region.append(filt_cluster[0])
801.
802.                         E2_windows[key] = E2_region
803.
804.                     else:
805.                         E2_windows[key] = sum_up_filt_cluste
    rs_E2(key, filt_cluster, dist_filt_clusters)
806.
807.
808.             return E2_windows
809.
810.         ##################################################
    ##################################
811.         def define_E1_Windows(IND_SNPs, dist_clusters, min_S
    NP_per_cluster, dist_filt_clusters):
812.
813.             dict_of_POS = get_SNP_ind(IND_SNPs, 'E1', 50)
814.
815.
816.             E1_windows = {}
817.
818.             for key in dict_of_POS:
819.
820.                 E1_region = []
821.
822.
823.                 if dict_of_POS[key] == 0:
824.                     E1_windows[key] = 0
825.
826.                 else:
827.
828.                     group = cluster(dict_of_POS[key], dist_c
    lusters) # get clusters of closer positions
```

```
829.
830.                     filt_cluster = filter_clusters(group, mi
    n_SNP_per_cluster)
831.                     filt_cluster.sort()
832.
833.                 if len(filt_cluster) == 0:
834.
835.                 E1_windows[key] = 0
836.
837.                     elif len(filt_cluster) == 1:
838.
839.                         E1_region.append(filt_cluster[0])
840.
841.                         E1_windows[key] = E1_region
842.
843.                     else:
844.                         E1_windows[key] = sum_up_filt_cluste
    rs_E1(key, filt_cluster, dist_filt_clusters)
845.
846.
847.                 if E1_windows[key] == 0:
848.                 a = 1
849.         #             print key, "no E1 region"
850.                 elif len(E1_windows[key]) == 1:
851.                 a = 1
852.         #             print key, len(E1_windows[key]), E1_win
    dows[key][0][0], E1_windows[key][0][-1]
853.                 else:
854.
855.                     new_dict = E1_windows.copy()
856.                     del E1_windows[key]
857.                     good_E1 = []
858.                     for i in new_dict[key]:
859.                         if i[-1]-i[0] > 1000000:
860.                             good_E1.append(i)
861.
862.                     E1_windows[key] = good_E1
863.         #             print key, len(E1_windows[key])
864.         #             for i in E1_windows[key]:
865.         #                 print i[0], i[-1]
```

```
866.
867.             return E1_windows
868.
869.         ########################################################
    ##################################
870.
871.         def check_next_absent(list_of_table_POS, POS, x):
872.             list_of_table_POS.sort()
873.             n = list_of_table_POS.index(POS)
874.
875.             if x == 'higher':
876.                 possible_POS = len(list_of_table_POS)-n
877.
878.                 next_absent = list_of_table_POS[n+1]
879.
880.             elif x == 'lower':
881.                 possible_POS = n
882.                 next_absent = list_of_table_POS[n-1]
883.
884.             print "there are "+str(possible_POS)+" more posi
    tions after the boundary. Closest is "+str(abs(POS-
    next_absent))+"bp away. Selecting this one..."
885.
886.             return next_absent
887.
888.
889.         ########################################################
    ##################################
890.
891.         def Establish_T_map(Regions_T_allele, T_SNPs):
892.             chromosomes = ['Pp01','Pp02','Pp03','Pp04','Pp05
    ','Pp06','Pp07','Pp08']
893.             CHR_SIZE = [47851208, 30405870, 27368013, 258432
    36, 18496696, 30767194, 22388614, 22573980]
894.
895.             T_map = {}
896.
897.
898.             for CHR in chromosomes:
899.
900.                 CHR_N = chromosomes.index(CHR)
901.
902.                 list_of_table_T = []
903.
904.                 for key, value in T_SNPs.iteritems():
905.                     CHR2 = key.split(":")[0]
906.                     POS = key.split(":")[1]
907.                     if (CHR == CHR2) and value != '-':
908.                         list_of_table_T.append(int(POS))
909.
910.                 list_of_table_T.sort()
911.
912.                 T_map[CHR] = []
913.
914.                 if Regions_T_allele[CHR] == 0:
915.
916.                     T_map[CHR].append([str('0'+'\t'+str(CHR_
    SIZE[CHR_N])+'\t'+'E1')])
917.
918.                 elif len(Regions_T_allele[CHR]) == 1:
919.                     list_of_IND_T = sorted(Regions_T_allele[
    CHR][0])
920.                     first_T = list_of_IND_T[0]
921.                     last_T = list_of_IND_T[-1]
922.
923.                     if (first_T < 500000) and ((CHR_SIZE[CHR
    _N] - last_T) < 500000):
924.
925.                         T_map[CHR].append([str('0'+'\t'+str(
    CHR_SIZE[CHR_N])+'\t'+'T')])
926.
927.                     elif (first_T > 500000) and ((CHR_SIZE[C
    HR_N] - last_T) > 500000):
928.
929.                         next_absent_left = check_next_absent
    (list_of_table_T, first_T, 'lower')
930.
931.                         next_absent_right = check_next_absen
    t(list_of_table_T, last_T, 'higher')
932.
```

```
                        T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])
                        T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])


                elif first_T > 500000:

                    next_absent_left = check_next_absent(list_of_table_T, first_T, 'lower')


                    T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])


                elif (CHR_SIZE[CHR_N] - last_T) > 500000:

                    next_absent_right = check_next_absent(list_of_table_T, last_T, 'higher')

                    T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])
            else:
                windows = Regions_T_allele[CHR]
                windows.sort()

                for i in windows:

                    if i == windows[0]:

                        list_of_IND_T = windows[0]
                        first_T = list_of_IND_T[0]
                        last_T = list_of_IND_T[-1]
                        next_absent_left = 0

                        if first_T > 500000:
                            next_absent_left = check_next_absent(list_of_table_T, first_T, 'lower')

                            next_absent_right = check_next_absent(list_of_table_T, last_T, 'higher')

                            if next_absent_left == 0:


                                T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])
                                print "A", next_absent_right, last_T
                            else:

                                T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])
                                T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])
                                print "B", next_absent_right, last_T

                        elif i == windows[-1]:
                            list_of_IND_T = windows[-1]
                            first_T = list_of_IND_T[0]
                            last_T = list_of_IND_T[-1]
                            next_absent_right = 0

                            if CHR_SIZE[CHR_N] - last_T > 500000:

                                next_absent_right = check_next_absent(list_of_table_T, last_T, 'higher')


                            next_absent_left = check_next_absent(list_of_table_T, first_T, 'lower')

                            if next_absent_right == 0:

```

```
                                T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])

                            else:

                                T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])
                                T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])
                                print "B", next_absent_right, last_T

                        else:
                            list_of_IND_T = i
                            first_T = list_of_IND_T[0]
                            last_T = list_of_IND_T[-1]

                            next_absent_right = check_next_absent(list_of_table_T, last_T, 'higher')
                            next_absent_left = check_next_absent(list_of_table_T, first_T, 'lower')


                            T_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_T)+'\t'+'E1>T')])
                            T_map[CHR].append([str(str(last_T)+'\t'+str(next_absent_right)+'\t'+'T>E1')])
                            print "B", next_absent_right, last_T

            return T_map
        ##########################################################################################
        def Establish_E2_map(Regions_E2_allele, E2_SNPs):
            chromosomes = ['Pp01','Pp02','Pp03','Pp04','Pp05','Pp06','Pp07','Pp08']
            CHR_SIZE = [47851208, 30405870, 27368013, 25843236, 18496696, 30767194, 22388614, 22573980]
            EBD = {'Pp01': [0, 36000000],
                   'Pp02': [1100000, 30405870],
                   'Pp03': [0, 26800000],
                   'Pp04': [1000000, 9700000],
                   'Pp05': [9400000, 15600000],
                   'Pp06': [10500000, 27800000],
                   'Pp07': [0, 20000000],
                   'Pp08': [4100000, 20000000]}

            E2_map = {}


            for CHR in chromosomes:

                CHR_N = chromosomes.index(CHR)

                list_of_table_E2 = []

                for key, value in E2_SNPs.iteritems():
                    CHR2 = key.split(":")[0]
                    POS = key.split(":")[1]
                    if (CHR == CHR2) and value != '-':
                        list_of_table_E2.append(int(POS))

                list_of_table_E2.sort()

                CHR_start = EBD[CHR][0]
                CHR_end = EBD[CHR][1]

                E2_map[CHR]=[]

                if Regions_E2_allele[CHR] == 0:

                    E2_map[CHR].append([str('0'+'\t'+str(CHR_SIZE[CHR_N])+'\t'+'E1')])

                elif len(Regions_E2_allele[CHR]) == 1:
                    list_of_IND_E2 = sorted(Regions_E2_allele[CHR][0])
                    first_E2 = list_of_IND_E2[0]
                    last_E2 = list_of_IND_E2[-1]

```

```
                    if (first_E2 - CHR_start < 1000000) and ((CHR_end - last_E2) < 1000000):

                        E2_map[CHR].append([str('0'+'\t'+str(CHR_SIZE[CHR_N])+'\t'+'E2')])

                    elif (first_E2 - CHR_start > 1000000) and ((CHR_end - last_E2) > 1000000):

                        next_absent_left = check_next_absent(list_of_table_E2, first_E2, 'lower')

                        next_absent_right = check_next_absent(list_of_table_E2, last_E2, 'higher')


                        E2_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])
                        E2_map[CHR].append([str(str(last_E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])

                    elif first_E2 - CHR_start > 1000000:

                        next_absent_left = check_next_absent(list_of_table_E2, first_E2, 'lower')


                        E2_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])

                    elif CHR_end - last_E2 > 1000000:

                        next_absent_right = check_next_absent(list_of_table_E2, last_E2, 'higher')

                        E2_map[CHR].append([str(str(last_E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])
                else:
                    windows = Regions_E2_allele[CHR]
                    windows.sort()

                    for i in windows:

                        if i == windows[0]:

                            list_of_IND_E2 = windows[0]
                            first_E2 = list_of_IND_E2[0]
                            last_E2 = list_of_IND_E2[-1]
                            next_absent_left = 0

                            if first_E2 - CHR_start > 1000000:
                                next_absent_left = check_next_absent(list_of_table_E2, first_E2, 'lower')

                            next_absent_right = check_next_absent(list_of_table_E2, last_E2, 'higher')

                            if next_absent_left == 0:


                                E2_map[CHR].append([str(str(last_E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])
                            else:

                                E2_map[CHR].append([str(str(next_absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])
                                E2_map[CHR].append([str(str(last_E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])

                        elif i == windows[-1]:
                            list_of_IND_E2 = windows[-1]
                            first_E2 = list_of_IND_E2[0]
                            last_E2 = list_of_IND_E2[-1]
                            next_absent_right = 0

                            if CHR_end - last_E2 > 1000000:
```

```
1123.
1124.                                 next_absent_right = check_ne
    xt_absent(list_of_table_E2, last_E2, 'higher')
1125.
1126.
1127.                             next_absent_left = check_next_ab
    sent(list_of_table_E2, first_E2, 'lower')
1128.
1129.                             if next_absent_right == 0:
1130.
1131.                                 E2_map[CHR].append([str(str(
    next_absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])
1132.
1133.                             else:
1134.
1135.                                 E2_map[CHR].append([str(str(
    next_absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])
1136.                                 E2_map[CHR].append([str(str(
    last_E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])
1137.
1138.                         else:
1139.                             list_of_IND_E2 = i
1140.                             first_E2 = list_of_IND_E2[0]
1141.                             last_E2 = list_of_IND_E2[-1]
1142.
1143.                             next_absent_right = check_next_a
    bsent(list_of_table_E2, last_E2, 'higher')
1144.                             next_absent_left = check_next_ab
    sent(list_of_table_E2, first_E2, 'lower')
1145.
1146.
1147.                             E2_map[CHR].append([str(str(next
    _absent_left)+'\t'+str(first_E2)+'\t'+'E1>E2')])
1148.                             E2_map[CHR].append([str(str(last
    _E2)+'\t'+str(next_absent_right)+'\t'+'E2>E1')])
1149.
1150.          return E2_map
1151.
1152.      ##################################################
    ###################################
```

```
1153.
1154.      with open(args.table_positions, 'r') as table:
1155.
1156.          next(table)
1157.          n = 0
1158.          for line in table:
1159.              CHR = (line.strip().split("\t")[0])
1160.              POS = int(line.strip().split("\t")[1])
1161.              E1E1 = line.strip().split("\t")[2]
1162.              E1E2 = line.strip().split("\t")[3]
1163.              E1T = line.strip().split("\t")[4]
1164.              E2T = line.strip().split("\t")[5]
1165.
1166.
1167.              E1_SNPs[CHR+":"+str(POS)] = find_E1(E1E2, E1
    T, E2T)
1168.
1169.
1170.              E2_SNPs[CHR+":"+str(POS)] = find_E2(E1E2, E1
    T, E2T)
1171.
1172.
1173.
1174.              T_SNPs[CHR+":"+str(POS)] = find_T(E1E2, E1T,
     E2T)
1175.
1176.
1177.
1178.      with open(args.vcf_filt, 'r') as vcf:
1179.
1180.
1181.          Attribute_allele_to_SNPs = genotype_SNPs(vcf)
1182.
1183.          IND_SNPs = sorted(Attribute_allele_to_SNPs, key=
    itemgetter(0,1))
1184.
1185.
1186.          Regions_T_allele = define_T_Windows(IND_SNPs, 10
    000, 100, 2000000)
1187.
```

```
1188.
1189.          Regions_E2_allele = define_E2_Windows(IND_SNPs,
    100000, 100, 2000000)
1190.
1191.
1192.
1193.          Regions_E1_allele = define_E1_Windows(IND_SNPs,
    40000, 100, 1000000)
1194.
1195.
1196.
1197.
1198.      with open(args.outfile, 'a') as out:
1199.
1200.
1201.          final_T_map = Establish_T_map(Regions_T_allele,
    T_SNPs)
1202.
1203.          final_E2_map = Establish_E2_map(Regions_E2_allel
    e, E2_SNPs)
1204.
1205.
1206.      #    out.write("IND"+"\t"+"CHR"+"\t"+"POS1"+"\t"+"PO
    S2"+"\t"+"GENO"+"\n")
1207.
1208.          chromosomes = ['Pp01','Pp02','Pp03','Pp04','Pp05
    ','Pp06','Pp07','Pp08']
1209.
1210.          for i in chromosomes:
1211.              if len(final_T_map[i]) == 1:
1212.
1213.                  out.write(IND_ID+'\t'+i+'\t'+final_T_map
    [i][0][0]+'\n')
1214.              else:
1215.                  for a in final_T_map[i]:
1216.
1217.                      out.write(IND_ID+'\t'+i+'\t'+a[0]+'\
    n')
1218.
1219.              if len(final_E2_map[i]) == 1:
```

```
1220.
1221.                  out.write(IND_ID+'\t'+i+'\t'+final_E2_ma
    p[i][0][0]+'\n')
1222.              else:
1223.                  for b in final_E2_map[i]:
1224.
1225.                      out.write(IND_ID+'\t'+i+'\t'+b[0]+'\
    n')
1226.
1227.
1228.      with open(args.results, 'w') as results:
1229.          results.write("CHR"+"\t"+"POS"+"\t"+"EG"+"\t"+"M
    B"+"\t"+"E1"+"\n")
1230.
1231.          for i in IND_SNPs:
1232.              CHR = i[0]
1233.              POS = str(i[1])
1234.              EG = i[2].split(":")[0]
1235.              MB = i[2].split(":")[1]
1236.              E1 = i[2].split(":")[2]
1237.
1238.              results.write(CHR+"\t"+POS+"\t"+EG+"\t"+MB+"
    \t"+E1+"\n")
```

Annex 5

‘get_gene_conversions.py’

```
# -*- coding: utf-8 -*-
"""
Created on Sun Jan 29 16:23:10 2017

@author: octavio
"""

import argparse
from operator import itemgetter

parser = argparse.ArgumentParser(description='Genotypes any individual from T1E population. Takes into acount the absence of T or E2 alleles to determine a recomb (E1<->T or E1<->E2). Takes as input the table of position for the 4 genotypes and the vcf file of the individual to genotype.')

parser.add_argument('--table_positions', metavar='FILE', help='Table with SNP information for the 4 genotypes')
parser.add_argument('--vcf_filt', metavar='FILE', help='VCF file filtered for a specific individual')
parser.add_argument('--regions_file', metavar='FILE', help='File "regions.recomb" from a T1E individual. Is the output file from script find_regions_from_genotyping.py')
parser.add_argument('--trf', metavar='FILE', help='The tandem repeat finder file with all TRFs found for E1 genome')
#parser.add_argument('--lower_bound_GC', metavar='FILE', help='Name of output file for conservative  GCs')
parser.add_argument('--final_GCs', metavar='FILE', help='Name of output file for permissive GCs')


args = parser.parse_args()

IND_ID = args.vcf_filt.split('filtered/')[1].split('_filt')[0]
#IND_ID = args.vcf_filt.split('_filt')[0]

E1_SNPs = {}
E2_SNPs = {}
T_SNPs = {}


def define_E_regions(list_of_lines):
    E_regions = []
    for i in list_of_lines:

        chromosome = i[0]
        recomb = i[1]
        start1 = i[2]
        end1 = i[3]
        start2 = i[4]
        end2 = i[5]

        if recomb == 'E>T':
            E_regions.append(str(chromosome)+':'+str(start1)+'-'+str(end1))
        elif recomb == 'T>E':
            E_regions.append(str(chromosome)+':'+str(start2)+'-'+str(end2))
        elif recomb == 'only_E':
            E_regions.append(str(chromosome)+':'+str(start1)+'-'+str(end1))
        else:
            continue
    return E_regions




def find_E1(E1E2, E1T, E2T):
    E1_allele = ''

    if (E1E2 != '-') and (E2T == '-') and (E1T != '-'):
```

```
        E1_allele = E1E2[0]

    elif (E1E2 != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        E1_allele = E1E2[0]

    else:
        E1_allele = '-'

    return E1_allele

#####################################################

def find_E2(E1E2, E1T, E2T):
    E2_allele = ''

    if (E1E2 != '-') and (E1T == '-') and (E2T != '-'):
        E2_allele = E1E2[1]
    elif (E1E2 != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        E2_allele = E1E2[1]
    else:
        E2_allele = '-'

    return E2_allele


#####################################################

def find_T(E1E2, E1T, E2T):
    T_allele = ''

    if (E1E2 == '-') and (E1T != '-') and (E2T != '-'):
        T_allele = E1T[1]
    elif (E2T != '-') and (E1T != '-') and (E2T != '-') and (E1T != E2T):
        T_allele = E2T[1]
    else:
        T_allele = '-'


    return T_allele


    ###################################################
    #
    def genotype_SNPs(vcf):

        EG_allele = []
        MB_allele = []
        UNSURE = []

        for line in vcf:

            if not line.startswith("Pp"):
                continue
            else:

                CHR = (line.strip().split("\t")[0])
                POS = int(line.strip().split("\t")[1])
                REF = line.strip().split("\t")[3]
                ALT = line.strip().split("\t")[4]
                GENO = (line.strip().split("\t")[9])[0:3]

                ID = CHR+":"+str(POS)


                if E1_SNPs.get(ID) != None:

                    E1_E2_T = [E1_SNPs.get(ID), E2_SNPs.get(ID), T_SNPs.get(ID)]


                    if GENO == "1/1":
                        IND = ALT+ALT
                        if (IND[0] in E1_E2_T):

                            if (IND[0] == E1_E2_T[0]):
```

```
                                    UNSURE.append((CHR, POS, "-"+":"+"-"+":"+"E1"+":"+"1/1"))

                                elif (IND[0] == E1_E2_T[1]):

                                    EG_allele.append((CHR, POS, "E2"+":"+"-"+":"+"-"+":"+"1/1"))

                                else:
                                    MB_allele.append((CHR, POS, "-"+":"+"T"+":"+"-"+":"+"1/1"))


                        elif GENO == "0/1":
                            IND = REF+ALT

                            if (IND[0] in E1_E2_T) or (IND[1] in E1_E2_T):

                                if (IND[0] == E1_E2_T[0]) or (IND[1] == E1_E2_T[0]):
                                    UNSURE.append((CHR, POS, "-"+":"+"-"+":"+"E1"+":"+"0/1"))

                                elif (IND[0] == E1_E2_T[1]) or (IND[1] == E1_E2_T[1]):
                                    EG_allele.append((CHR, POS, "E2"+":"+"-"+":"+"-"+":"+"0/1"))

                                else:
                                    MB_allele.append((CHR, POS, "-"+":"+"T"+":"+"-"+":"+"0/1"))

            return MB_allele+EG_allele+UNSURE


        head_lines = []

        with open(args.regions_file, 'r') as regions:
            regions_lines = regions.readlines()

            for i in regions_lines:
                line_field = i.strip().split('\t')

                if len(line_field) > 4:
                    head_lines.append(line_field)
                else:
                    continue

            E_regions = define_E_regions(head_lines)

        with open(args.table_positions, 'r') as table:

            next(table)
            n = 0
            for line in table:
                CHR = (line.strip().split("\t")[0])
                POS = int(line.strip().split("\t")[1])
                E1E1 = line.strip().split("\t")[2]
                E1E2 = line.strip().split("\t")[3]
                E1T = line.strip().split("\t")[4]
                E2T = line.strip().split("\t")[5]


                E1_SNPs[CHR+":"+str(POS)] = find_E1(E1E2, E1T, E2T)


                E2_SNPs[CHR+":"+str(POS)] = find_E2(E1E2, E1T, E2T)


                T_SNPs[CHR+":"+str(POS)] = find_T(E1E2, E1T, E2T)


        with open(args.vcf_filt, 'r') as vcf:
            with open(args.trf, 'r') as trf:
```

```
200.
201.                 TRF = trf.readlines()
202.                 TRF_dict = {}
203.                 CHR2 = 'PP'
204.
205.                 for tandem in TRF:
206.                     if 'Sequence' in tandem:
207.                         CHR2 = tandem.strip().split(': ')[1]
208.
209.
210.                     else:
211.                         if CHR2 in TRF_dict:
212.
213.                             TRF_dict[CHR2].append(tandem)
214.
215.                         else:
216.                             TRF_dict.setdefault(CHR2, [])
217.
218.
219.                 Attribute_allele_to_SNPs = genotype_SNPs(vcf
    )
220.
221.                 IND_SNPs = sorted(Attribute_allele_to_SNPs,
    key=itemgetter(0,1))
222.
223.
224.
225.                 GC_NO_TANDEM = []
226.                 GC_TEMP = []
227.                 GC_start = 0
228.                 T_SNPs = 0
229.                 total_match = 0
230.
231.
232.                 for a in E_regions:
233.                     E_chromosome = a.strip().split(':')[0]
234.                     E_start = int((a.strip().split(':')[1]).
    split('-')[0])
```

```
235.                     E_end = int((a.strip().split(':')[1]).sp
    lit('-')[1])
236.
237.                     for i in IND_SNPs:
238.                         CHR = i[0]
239.                         POS = i[1]
240.                         EG = i[2].split(":")[0]
241.                         MB = i[2].split(":")[1]
242.                         E1 = i[2].split(":")[2]
243.                         GENO = i[2].split(":")[3]
244.
245.                         if (E_chromosome == CHR) and (POS >
    E_start) and (POS < E_end):
246.
247.                             if (MB != '-
    ') and (GENO != '1/1'):
248.
249.                                 T_SNPs += 1
250.
251.                                 match = 0
252.
253.
254.                                 for A in TRF_dict[CHR]:
255.
256.                                     if A.split(' ')[0].isdig
    it():
257.                                         trf_start = int(A.st
    rip().split(' ')[0])-50
258.                                         trf_end = int(A.stri
    p().split(' ')[1])+50
259.
260.                                         if (POS > trf_start)
     and (POS < trf_end):
261.                         #                   print CHR, POS,
     '//', CHR, trf_start, trf_end
262.
263.                                             match = 1
264.                                             total_match += 1
265.                                             break
```

```
                                if match == 0:

                                    if len(GC_TEMP) == 0:

                                        GC_TEMP.append((CHR, POS))

                                    elif len(GC_TEMP) == 1:

                                        GC_start = GC_TEMP[0][1]

                                        if (POS-GC_start) > 1000:
                                            GC_NO_TANDEM.append((CHR, GC_start, GC_start, '1'))
                                            GC_TEMP = []
                                            GC_TEMP.append((CHR, POS))
                                        else:
                                            GC_TEMP.append((CHR, POS))
                                    else:
                                        n = len(GC_TEMP)
                                        GC_start = GC_TEMP[0][1]
                                        GC_end = GC_TEMP[-1][1]

                                        if (POS-GC_end) > 1000:
                                            GC_NO_TANDEM.append((CHR, GC_start, GC_end, n))
                                            GC_TEMP = []
                                            GC_TEMP.append((CHR, POS))
                                        else:
                                            GC_TEMP.append((CHR, POS))
```

```
                            else:
                                if len(GC_TEMP) == 0:
                                    continue

                                elif len(GC_TEMP) == 1:
                                    GC_start = GC_TEMP[0][1]

                                    GC_NO_TANDEM.append((CHR, GC_start, GC_start, '1'))

                                    GC_TEMP = []


                                else:
                                    n = len(GC_TEMP)
                                    GC_start = GC_TEMP[0][1]

                                    GC_end = GC_TEMP[-1][1]
                                    GC_NO_TANDEM.append((CHR, GC_start, GC_end, n))

                                    GC_TEMP = []
                        else:
                            if len(GC_TEMP) == 0:
                                continue

                            elif len(GC_TEMP) == 1:
                                GC_start = GC_TEMP[0][1]
                                GC_NO_TANDEM.append((CHR, GC_start, GC_start, '1'))

                                GC_TEMP = []


                            else:
                                n = len(GC_TEMP)
```

```
                                GC_start = GC_TEMP[0][1]
                                GC_end = GC_TEMP[-1][1]
                                GC_NO_TANDEM.append((CHR, GC
    _start, GC_end, n))

                                GC_TEMP = []

                GC_NO_TANDEM2 = list(set(GC_NO_TANDEM))
                GC_NO_TANDEM = []
                GC_NO_TANDEM = sorted(GC_NO_TANDEM2, key=ite
    mgetter(0,1))


        with open(args.final_GCs, 'a') as out_upper:

            print "From "+str(T_SNPs)+" T SNPs in individual
     "+IND_ID+", "+str(total_match)+" of them overlap with tand
    em repeats +-50 bp. Excluding these overlaps..."

            for GC in GC_NO_TANDEM:
                num_SNPS = int(GC[3])
                SIZE = str(GC[2]-GC[1])

                out_upper.writelines(IND_ID+'\t'+GC[0]+'\t'+s
    tr(GC[1])+'\t'+str(GC[2])+'\t'+str(GC[3])+'\t'+SIZE+'\n')
```