convert a genetic map to recombination map

2022-07-29

chr_phy	phy_pos	chr_gen	gen_pos
physical chr	physical position (in base)	genetic chr	genetic position

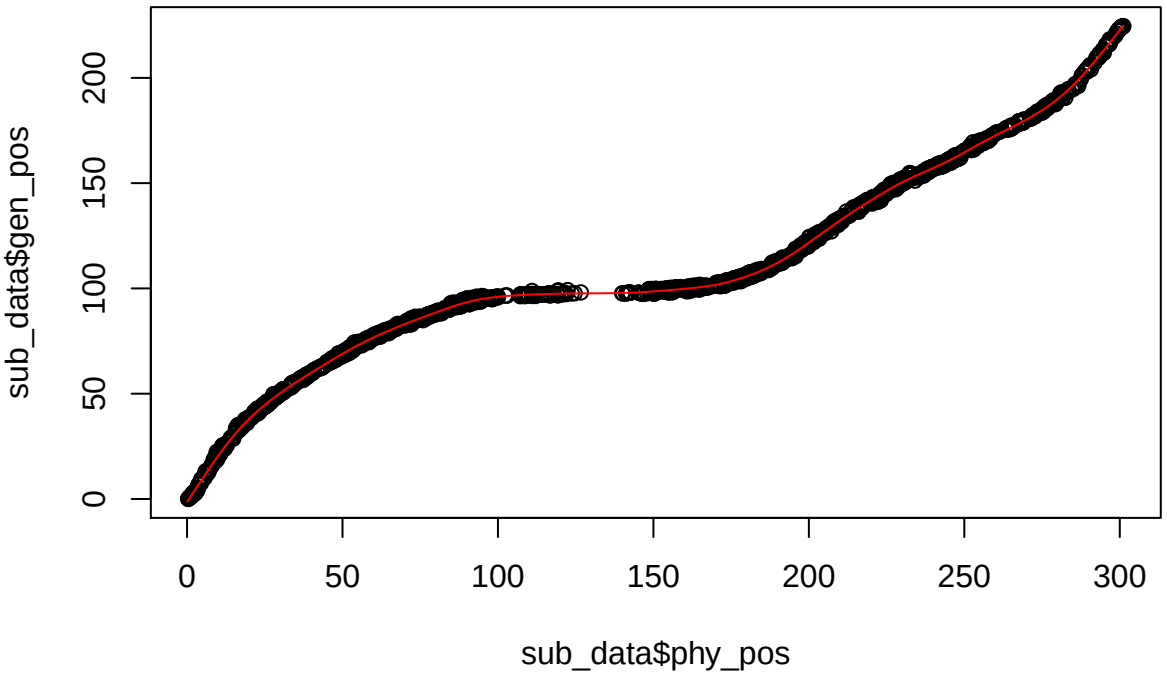
1. Load data (to edit if rerun)

```
data = read.table('~/Domisol_thesis/rawdata_system/maize/carte_maize.txt',h=T)
data$phy_pos = data$coordinate # i just rename this column to match the column syntax
head(data)
```

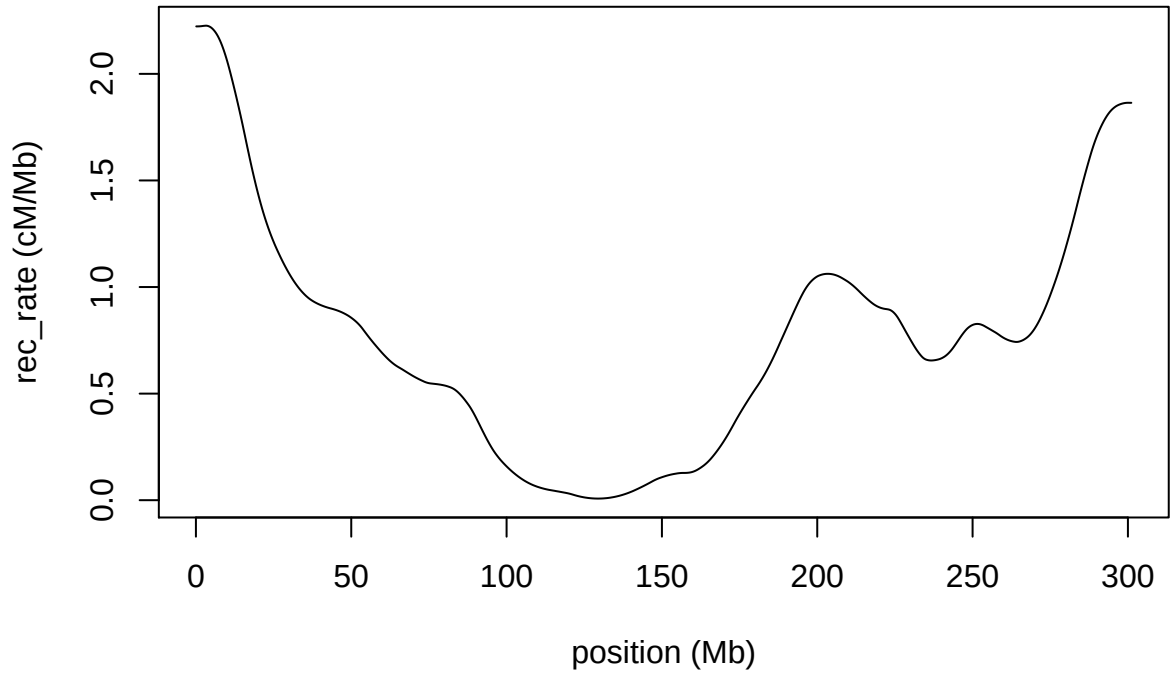
```
##   map chr_gen      SNP_name chr_gen.1 gen_pos    status chr_phy coordinate
## 1 All       1 PZE_101000169         1     0.0 composite       1     379844
## 2 All       1 PZE_101000209         1     0.1 composite       1     395380
## 3 All       1 PZE_101000256         1     0.2 composite       1     485953
## 4 All       1 PZE_101000301         1     0.4 composite       1     613257
## 5 All       1 PZE_101000349         1     0.5 composite       1     681704
## 6 All       1 PZE_101000348         1     0.5 composite       1     681011
##   phy_pos
## 1  379844
## 2  395380
## 3  485953
## 4  613257
## 5  681704
## 6  681011
```

2. Exemple for one chromosome

```
data$phy_pos = data$phy_pos / 1e6 # transform base position in Mb position

smooth_strengh = 0.7 # choose a value between 0 and +infinite.
#The higher is the value, the smoother the map will be

sub_data = subset(data,subset = chr_phy == 1)
sm = smooth.spline(x=sub_data$phy_pos,y=sub_data$gen_pos,spar=smooth_strengh)

### check if the smooth is enough
predx = seq(0.1,max(sub_data$phy_pos),by=0.1)
predy = predict(sm,x=predx,deriv=0)$y
plot(sub_data$phy_pos,sub_data$gen_pos,type='p')
lines(predx,predy,col='red')
```




```
#### you must obtain a growing monotonous dot line , and the red line must be in the dot line.

pred_deriv = predict(sm,x=predx,deriv=1)$y
plot(predx,pred_deriv,type='l',xlab='position (Mb)',ylab='rec_rate (cM/Mb)')
```




```
# here the map seems smooth enough to remove all the noise

sub_rec_map = data.frame('chr'=1,'start' = predx - (0.1/2 - 1e-6) ,
                         'end' = predx + 0.1/2 , 'rec_rate' = pred_deriv)
head(sub_rec_map)
```

```
##   chr    start  end rec_rate
## 1   1 0.050001 0.15 2.222786
## 2   1 0.150001 0.25 2.222786
## 3   1 0.250001 0.35 2.222786
## 4   1 0.350001 0.45 2.222784
```

```
## 5   1 0.450001 0.55 2.222781
## 6   1 0.550001 0.65 2.222790
```

The map generated express the recombination rate in cM/Mb. The recombination rate is express in probability of recombination per base, per generation, per individual. Furthermore, the physical position (start and end) are express in Mb too, which doesn't match the file format use in RIDGE (which is in pb)

chr	start (bp)	end (bp)	rec_rate
chr name	start position	end position of window	recombination rate

3. Transform cM/Mb to recombination rate using haldane transformation

$$d_{AB} = -50.ln(1 - 2.r_{AB})$$

with d_{AB} the genetic distance between A and B in cM. Our map give us a result in cM, so

$$\frac{cM}{Mb}.Mb = d_{AB}$$

And r_{AB} is the recombination rate between A and B. So r_{AB} is equal to

$$r_{AB} = \frac{1 - e^{d_{AB}/-50}}{2}$$

```
sub_rec_map$rec_rate = sub_rec_map$rec_rate * 0.1 / 1e5 #to transform into d_AB / locus
#size to set the distance between d_AB at 1 base
sub_rec_map$rec_rate = (1-exp(sub_rec_map$rec_rate/-50))/2
sub_rec_map$start = sub_rec_map$start * 1e6
sub_rec_map$end = sub_rec_map$end * 1e6
head(sub_rec_map)
```

```
##   chr  start    end     rec_rate
## 1   1  50001 150000 2.222786e-08
## 2   1 150001 250000 2.222786e-08
## 3   1 250001 350000 2.222786e-08
## 4   1 350001 450000 2.222784e-08
## 5   1 450001 550000 2.222781e-08
## 6   1 550001 650000 2.222790e-08
```

4. Apply on all chr

```
sp_data = split.data.frame(data,f=data$chr_phy)
rc_maps = do.call(rbind,lapply(sp_data,function(sub_data){
    sm = smooth.spline(x=sub_data$phy_pos,y=sub_data$gen_pos,spar=smooth_strengh)
    predx = seq(0.1,max(sub_data$phy_pos),by=0.1)
    pred_deriv = predict(sm,x=predx,deriv=1)$y
    sub_rec_map = data.frame('chr'=sub_data$chr_phy[1],'start' = predx - (0.1/2 - 1e-6) ,
                             'end' = predx + 0.1/2 , 'rec_rate' = pred_deriv)
    return(sub_rec_map)
}))
rc_maps$rec_rate = rc_maps$rec_rate * 0.1 / 1e5
rc_maps$rec_rate = (1-exp(rc_maps$rec_rate/-50))/2
rc_maps$start = rc_maps$start * 1e6
rc_maps$end = rc_maps$end * 1e6
head(rc_maps)
```

```
##     chr  start    end     rec_rate
## 1.1   1  50001 150000 2.222786e-08
## 1.2   1 150001 250000 2.222786e-08
```

```
## 1.3   1 250001 350000 2.222786e-08
## 1.4   1 350001 450000 2.222784e-08
## 1.5   1 450001 550000 2.222781e-08
## 1.6   1 550001 650000 2.222790e-08
```

```
write.table(rc_maps,file='full_map.txt',row.names=F)
```