peakC tutorial

Geert Geeven & Elzo de Wit

April 4, 2018

It is becoming increasingly clear the 3D organization of the genome play an important in the regulation of genes. To measure the 3D genome various 3C-based have been developed (4C/5C/Hi-C/Capture-C/Promoter-Capture-Hi-C). 4C, Capture-C and PCHiC measure the contact frequency of a single region ("viewpoint" or "bait") with the rest of the genome.

The main goal of these experiments is to identify regions that are enriched over the background ("peaks"). Because of the non-uniform distribution of the background this is not a trivial exercise. In order to model the background, we assume that the contact frequency only decreases with an increase in distance from the viewpoint (i.e. monotonically decreases). peakC achieves this by performing monotonic regression using the `isotone` package. For a more detailed discussion of the theoretical background and performance of the package we point the reader to our paper: (reference pending). The current tutorial focuses on how to perform 4C/Capture-C/PCHiC data analysis using peakC.

In order to load the package we run

```
library(peakC)
```

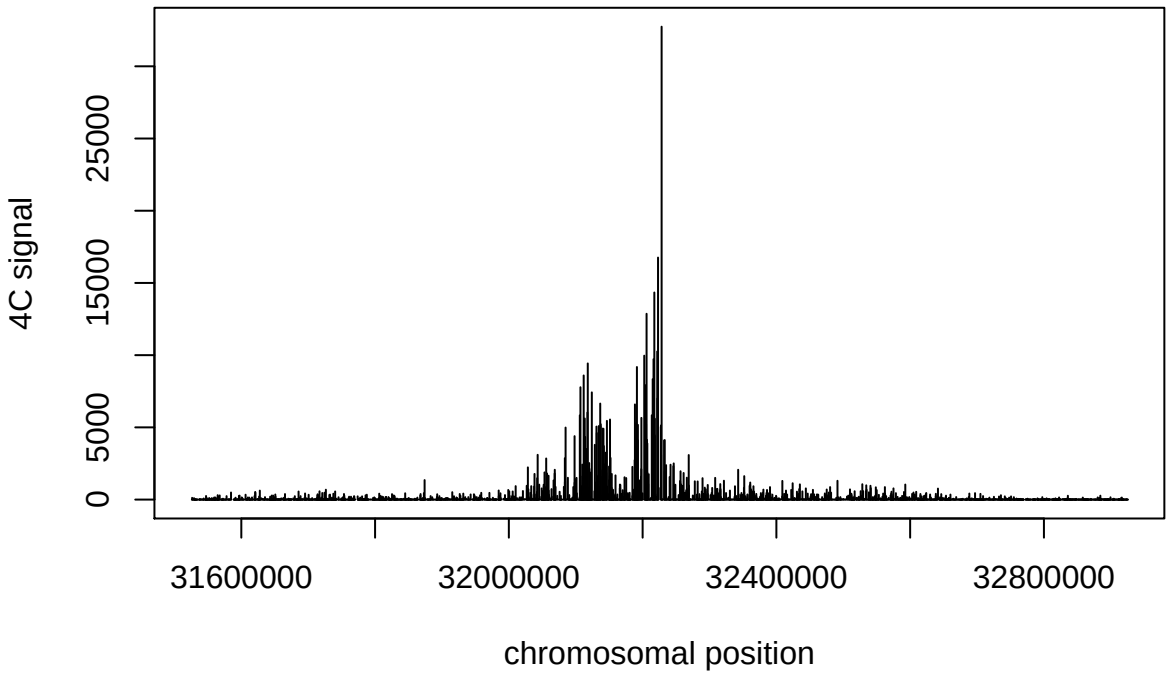
Data structure

The basic data structure for a (single) 4C/Capture-C/PCHiC experiment is a two-column matrix, with one column for the position of the fragment (end) and one column with the number of reads (coverage, 4C signal). We can use one of the peakC functions to read in the 4C/Capture-C/PCHiC data. As an example we use the `readqWig` function, which can read in wiggle files, which are compatible with the UCSC genome browser. The data we read in is a 4C experiment for one of the hemoglobin genes (alpha-globin) in fetal livers of mice.

```
data <- readqWig("data/alpha-globin_FL_1_cis.wig.gz", vp.pos=32224333, window=700e3)
head(data)
```

```
##          [,1]       [,2]
## [1,] 31526039   0.000000
## [2,] 31526481 121.623265
## [3,] 31527169   4.504565
## [4,] 31528018   0.000000
## [5,] 31529060  90.091308
## [6,] 31529432   0.000000
```

This is what the data structure looks like. The `readqWig` function has selected the fragments in the 700kb downstream and upstream of the viewpoint (position set by `vp.pos`). The flanking window size can be set with `window`. The coverage score is automatically normalized to 1 million intrachromosomal reads (reads in cis). A plot of the raw 4C data looks like this:



The expression of the alpha-globin gene is extremely high in this tissue and is regulated by enhancers that flank this gene. We can already see the promoter-enhancer contacts appearing in the raw data.

Alternative functions for reading in data are:

- `readMatrix`: reads a simple two column file (wig file without the header)
- `readMultiple`: reads multiple matrix files at once, returns a list
- `readMultipleWig`: like read multiple, but for wig files
- `readMultiColumnFile`: reads a multiple column file, 1st column is fragment position, the rest of the columns are 4C/Capture-C/PCHiC data columns

---

Identifying contacted regions ("peaks")

4C analysis with replicate experiments

In order to identify peaks we *strongly* advise using replicates. For the tutorial we have done the 4C of the alpha-globin gene in triplicate. The peak calling consists of two steps. 1) Reading the data (shown above). 2) Statistical analysis using the `combined.analysis` function. A third optional step is to plot the 4C data.

Below we show the simplest way to run the peakC peak caller. Note that it runs with a default of 700kb flanking region, a running window size of 21 and an FDR threshold of 0.1. For details see the man page.

```
viewpoint <- 32224333
files <- dir(path="data/", pattern="alpha-globin_FL_[123]_cis.wig.gz", full=T)
data <- readMultipleWig(files, vp.pos = viewpoint, window = 700e3)
res <- combined.analysis(data, vp.pos = viewpoint)
```

```
## Loading required package: isotone
```

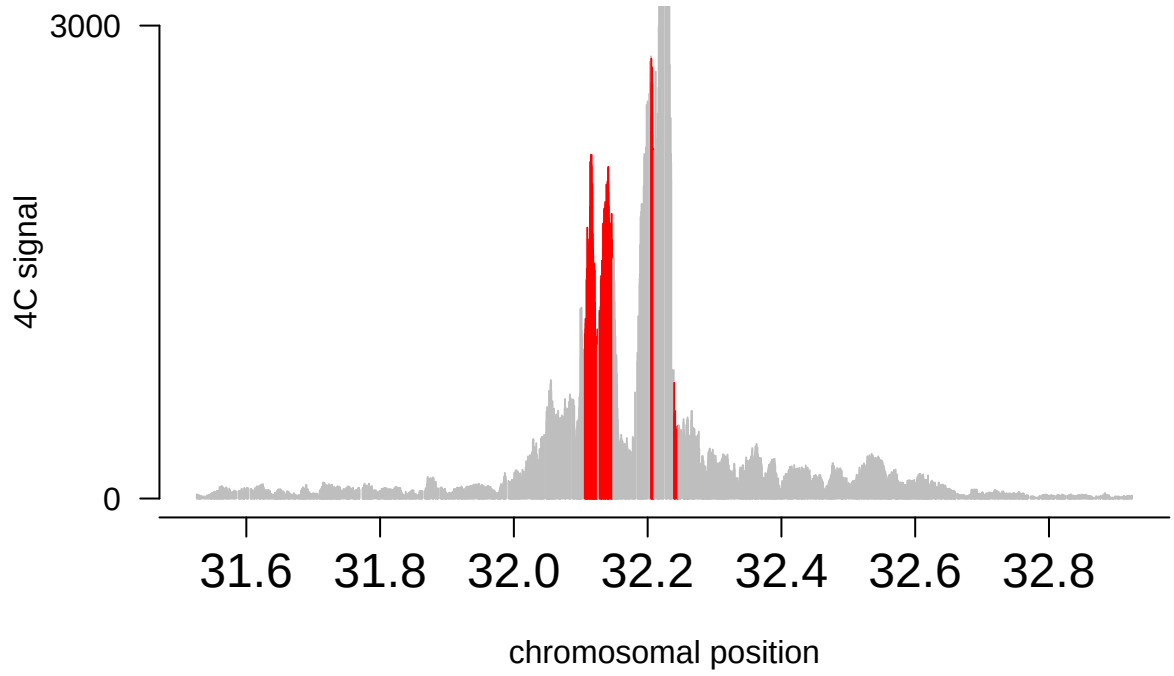
```
##     V2.x     V2.y       V2
## 4.504565 3.661837 6.089967
```

```
head(res$peak)
```

```
## [1] 32106117 32107017 32107166 32107433 32107829 32108173
```

`combined.analysis` outputs a list of with the results of the peak calling (stored here in `res`). The contents of the results list will be discussed in detail below. What is important for now is the `peak` vector in the list contains the fragments that represent significant interactions with the viewpoint. We can use the output of `combined.analysis` to create a plot by calling the `plot_C` function. The significant fragment windows are shown in red.

```
plot_C(res)
```



Single experiment 4C analysis

Below an explanation will be given for performing single experiment analysis.

First read in the data:

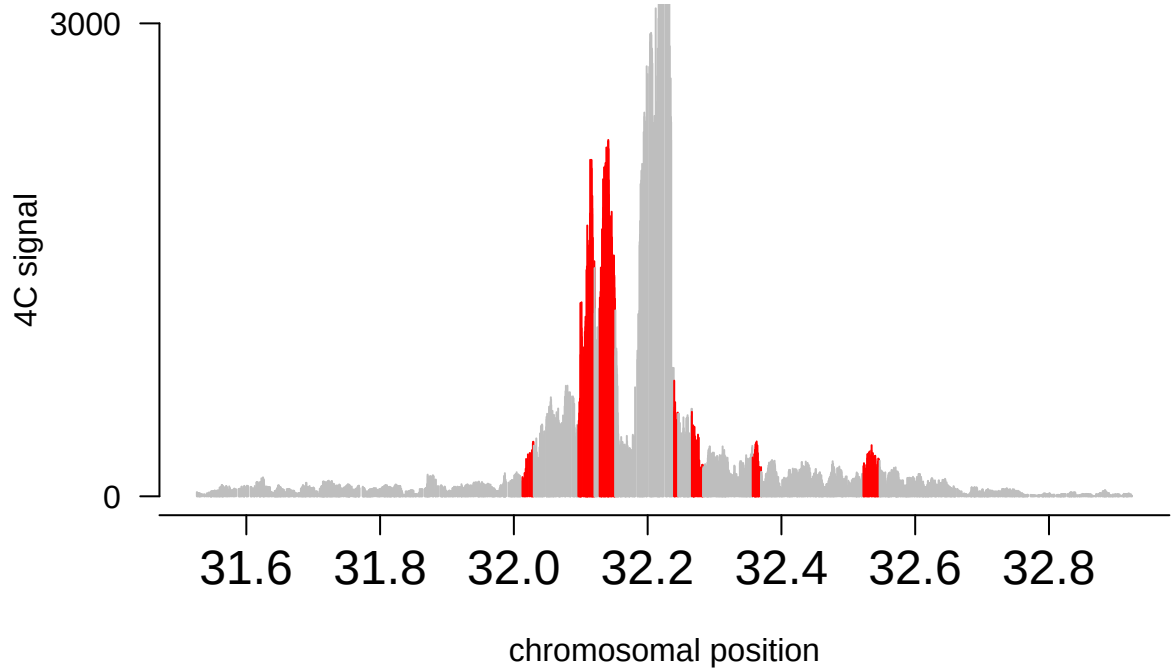
```
data <- readqWig("data/alpha-globin_FL_1_cis.wig.gz", vp.pos=32224333, window=700e3)
```

and the run the command for identifying the peaks

```
data <- list(data) #turn data into a list
res <- single.analysis(data, vp.pos=32224333, qWd = 2.5)
```

The `qWd` can be varied to increase or lower the stringency of the peak calling. Note that we need make the matrix output of `readqWig` compatible with the `single.analysis` function, by turning it into a list. To plot the significant fragments, we again use the `plot_C` function.

```
plot_C(res)
```



The single experiment peak caller calls most of the peaks that were called by the single experiment peak caller. However, an additional set of peaks are called, some of which are likely false positive peaks.

Multi-column Capture-C experiment

In addition to 4C experiments peakC can also analyse Capture-C data. Below an example is shown how to identify significant fragments from a Capture-C experiment with 4 replicates. This data was taken from (Davies et al. 2015). A file with multiple columns can be read as follows and return a list of data.frames.

```
data <- readMultiColumnFile("data/Hba_cis.txt", vp.pos = 32189804, window=300e3)
head(data[[1]])
```

```
##       frag_pos frag_score
## 23826 31891309  32.730928
## 23827 31891573  28.055081
## 23828 31892305   0.000000
## 23829 31892486  18.703388
## 23830 31893036  23.379235
## 23831 31893231   4.675847
```

```
head(data[[2]])
```

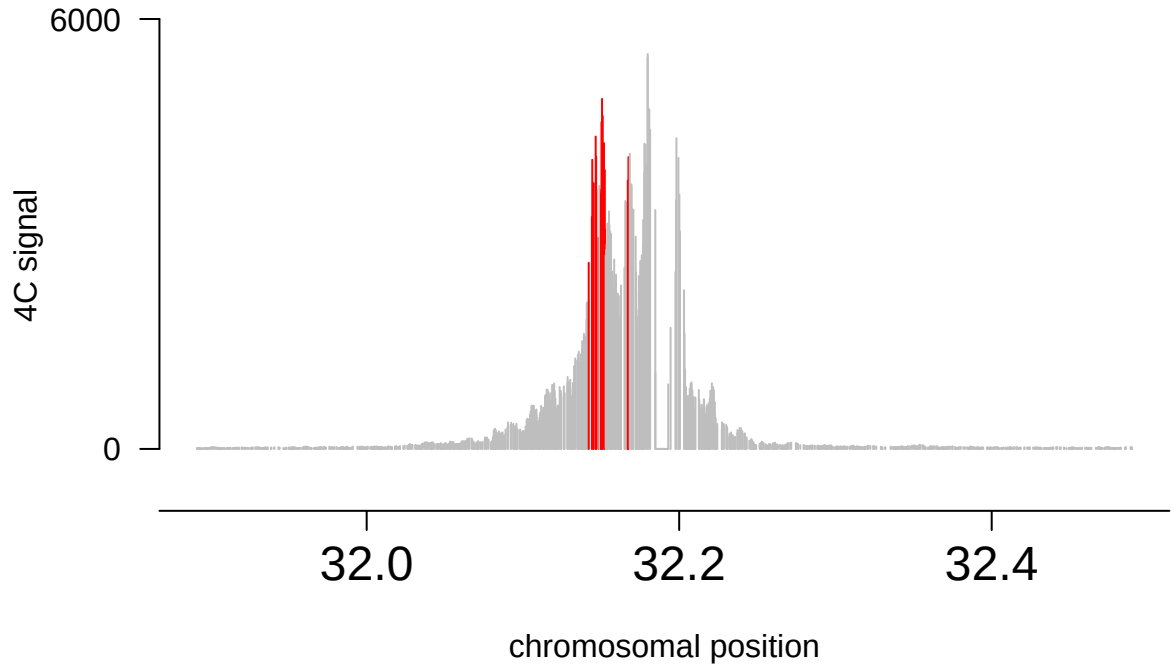
```
##       frag_pos frag_score
## 23826 31891309   0.000000
## 23827 31891573   0.000000
## 23828 31892305   5.412749
## 23829 31892486  10.825498
## 23830 31893036  43.301994
## 23831 31893231  10.825498
```

The list can be passed to the `combined.analysis` function as follows:

```
vp.pos <- c(32182969,32196638)
res <- combined.analysis(data=data, num.exp = 2, vp.pos = vp.pos, wSize = 5)
```

```
## frag_score.x frag_score.y
##     4.675847     5.412749
```

```
plot_C(res, y.max=6000, y.min=-600)
```



Note that in the these Capture-C experiments two baits were used. We would like to exclude the sequence in between the baits, which is why we use two values for the position of the viewpoint. The fragments between the baits are excluded from the

Due to the higher resolution of the Capture-C experiment we have changed the window size, i.e. the number fragments over which the averaging is performed, `wSize` to 5. Even though we read in 4 experiments, when we set `num.exp` to 2 only the first two experiments are used for the peak calling.

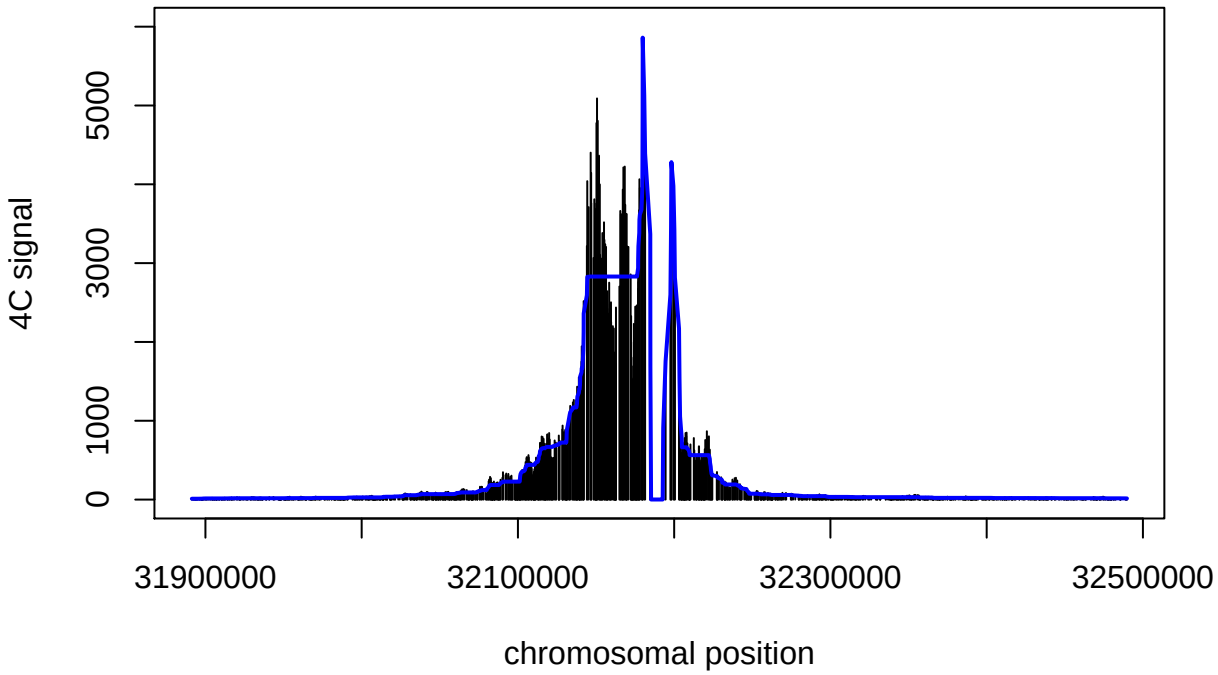
In the function `plot_C` we can play with the limits of the vertical axis by varying `y.min` and `y.max`. By setting the y.min to a negative value it is possible to insert annotations such as gene tracks or ChIPseq data.

Note that there is a difference between the 4C and the CapC data because there is a 43kb difference in the position of the bait or viewpoint.

Output details

Let's look into the resulting data structure in bit more detail. For the two replicate Capture-C experiment the result is a list with 6 elements. Let's go over them: `res$dbR` contains the original data (1st column fragment position, columns 2 until n+1 contain the n experiments). Columns n+2 until 2n+1 contain the estimated background model.

```
plot(res$dbR[,1], res$dbR[,2], type='h', ylim=c(0,6000), xlab="chromosomal position", ylab="4C signal")
lines(res$dbR[,1], res$dbR[,4], lwd=2, col='blue')
```



First we plot the raw data in the second column, next we add the background model shown by the blue line.

`res$peak` contains the fragments that have been called as significant peaks.

`res$num.exp` stores the number of experiments that are stored in `res`.

`res$p.value` contains for every fragment a rank product probability score whether the fragment is a peak fragment.

`res$sel` contains the fragments that are used in the analysis; fragments too close to the viewpoint are not used in the analysis. This distance can be set using the `minDist` parameter in `combined.analysis`.

References

Davies, James O J, Jelena M Telenius, Simon J McGowan, Nigel A Roberts, Stephen Taylor, Douglas R Higgs, and Jim R Hughes. 2015. "Multiplexed analysis of chromosome conformation at vastly improved sensitivity." *Nature Methods* 13 (1). Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved.: 74–80. doi:10.1038/nmeth.3664.