This tutorial describes the use of Scaffremodler tools to detect structural variations with re-sequencing data sets.

Go to scaffremodler folder (Scripts can be run from any folder but the command lines in this tutorial assume you are in this folder and that you have python 2.7 version).

Available data:

data/scaffolds/ contained a multifasta file (Ref_for_SV_detection.fasta) containing 2 chromosome sequences that will be used as reference genome sequence

data/reads/ contained 2 fastq files (reads_mate1_SV.fq and reads_mate2_SV.fq) containing paired read (5 kb insert size) that have be simulated on homozygous accession diverging from the reference sequence from a duplication, an inversion, a reciprocal translocation and a deletion. The fastq files correspond to mate pair which expected orientation is reverse/forward (rf).

A – Detection of structural variations

This can be done by running the following command line:

python bin/scaffremodler_wrapper.py --ref data/scaffolds/Ref_for_SV_detection.fasta --q1 data/reads/reads_mate1_SV.fq --q2 data/reads/reads_mate2_SV.fq --step 12345678 --tool bowtie2_single --orient rf --prefix test

For complete options, see the Readme file or run "python bin/scaffremodler_wrapper.py --h"

This command line output several files. The most important files are the ***.score** files that contains for each discordant type, the corresponding discordant zones identified.

The file **test_SV_detected.tab** contains the list of structural variations detected by the pipeline. When inspecting this file, the simulated inversion, reciprocal translocation, deletion and duplication are detected by our pipeline.

```
inversion           region:   chr02     9991      100013
reciprocal_translocation  region1:chr01     2201114   2300010   region2_inv:        chr02     1601985 1746995
deletion region:    chr01     700014    800017
duplication         region_inv:         chr01     249988    353015    target:   chr02     700006    700051
```

Sometime no complete signature of the structural variation can be detected. This can be due to transposable elements, important sequence divergence or errors in the assembly of the reference sequence at the boundaries of the structural variation. In this case, a manual inspection of detected discordant zones may allow identifying these partial structural variation signatures. This can be done by inspecting the ***.score** files or by drawing the discordant zones detected by the pipeline using the following tools.

B – Preparing files for discordant zone drawing

This can be done by running the following command line:

```
python bin/conf4circos.py --cov test.cov --chr test.chrom --window 1000 --frf test_fr.score --ff test_ff.score --rr test_rr.score --ins test_ins.score --delet test_del.score --chr_rr test_chr_rr.score --chr_rf test_chr_rf.score --chr_ff test_chr_ff.score --chr_fr test_chr_fr.score --orient rf --liste_read test.list --dis_prop test.prop --ref data/scaffolds/Ref_for_SV_detection.fasta --prefix test_circos
```

Several files are generated (all beginning with the prefix **test_circos**). All these files are used in the following steps to draw circos picture of the expected elements in the expected regions.

C – Drawing circos picture

1) Drawing discordant zones detected on all chromosomes

```
python bin/draw_circos.py --config test_circos.conf --out test_circos.png --read_fr n --read_rf n --read_ff n --read_rr n --read_ins n --read_delet n --read_chr_rr n --read_chr_rf n --read_chr_fr n --read_chr_ff n --text n
```

The picture obtained is presented in Figure 1 and the inferred structure of the re-sequenced accession relative to the reference is presented in Figure 2.

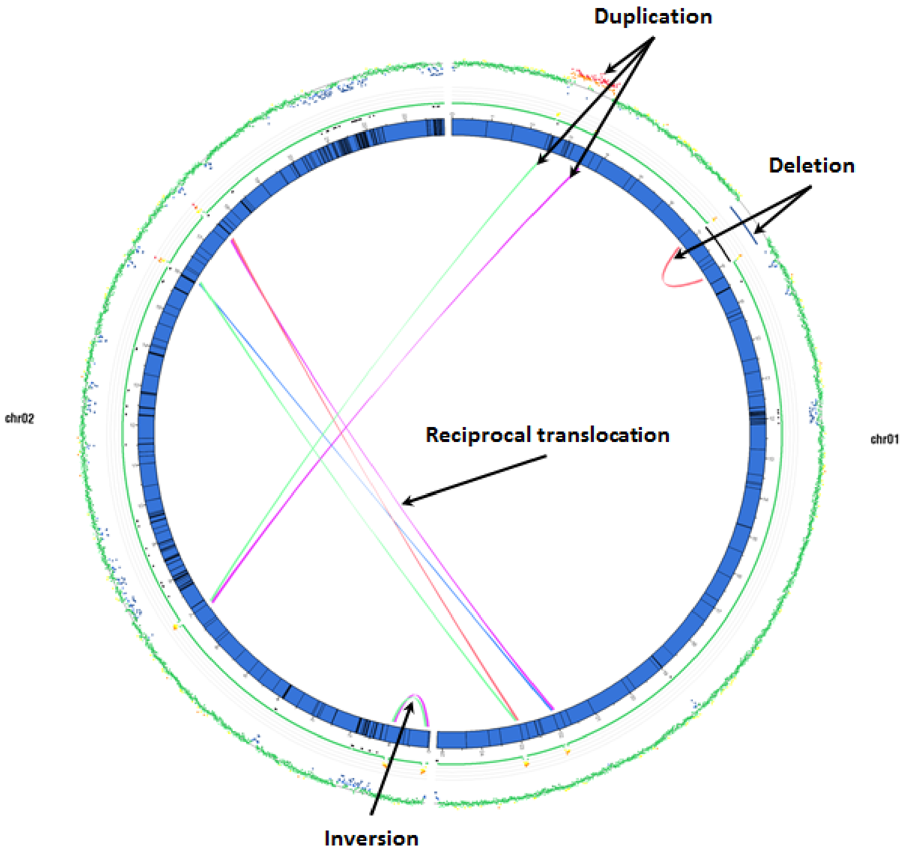

Figure 1 : Circos picture representing discordant regions identified by scaffremodler tools. For discordant regions color code: please refers to Annexe1 at the end of this document. The blue circle represents the reference sequence and black boxes symbolize N regions in the reference. The second circle represents discordant read proportion calculated on window size of 1kb. The third circle (outer circle), represent the read mean coverage along the genome calculated on window size of 1kb.

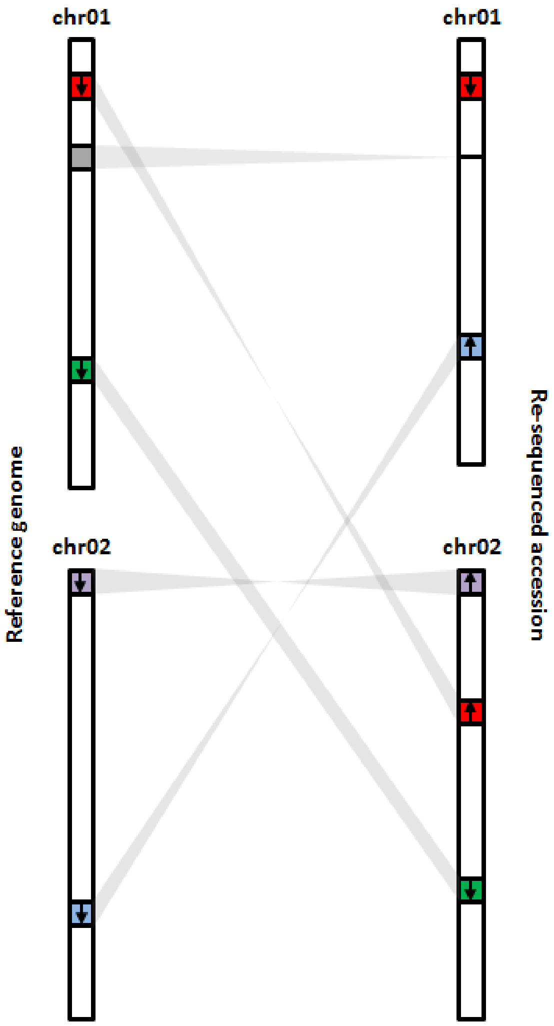

Figure 2 : Structure comparison between the reference genome used and the re-sequenced accessions. The duplication event is in red, the deletion event is in grey, the reciprocal translocation is symbolized by green and blue boxes and the inversion event is in purple.

2) Mapped reads inspection in a genomic region

Scaffremodler tools can be used to inspect read mapping configuration in specific regions. For example, read mapping configuration at boundaries of the inversion on chromosome 2 can be inspected running the following command line:

```
python bin/draw_circos.py --config test_circos.conf --out test_zoom_on_inversion.png --rr n --ff n --text n --draw chr02:0:30000-chr02:90000:110000
```

The obtained representation is presented in Figure 3. In this picture, two cluster of discordant read can be observed (purple cluster and green cluster). The absence of overlap and the contiguity of the green and purple clusters suggest that this is really a SV and not a TE movement. The absence of concordant reads overlapping the SV boundaries suggest that the inversion is present in both haplotypes of the re-sequenced accession.

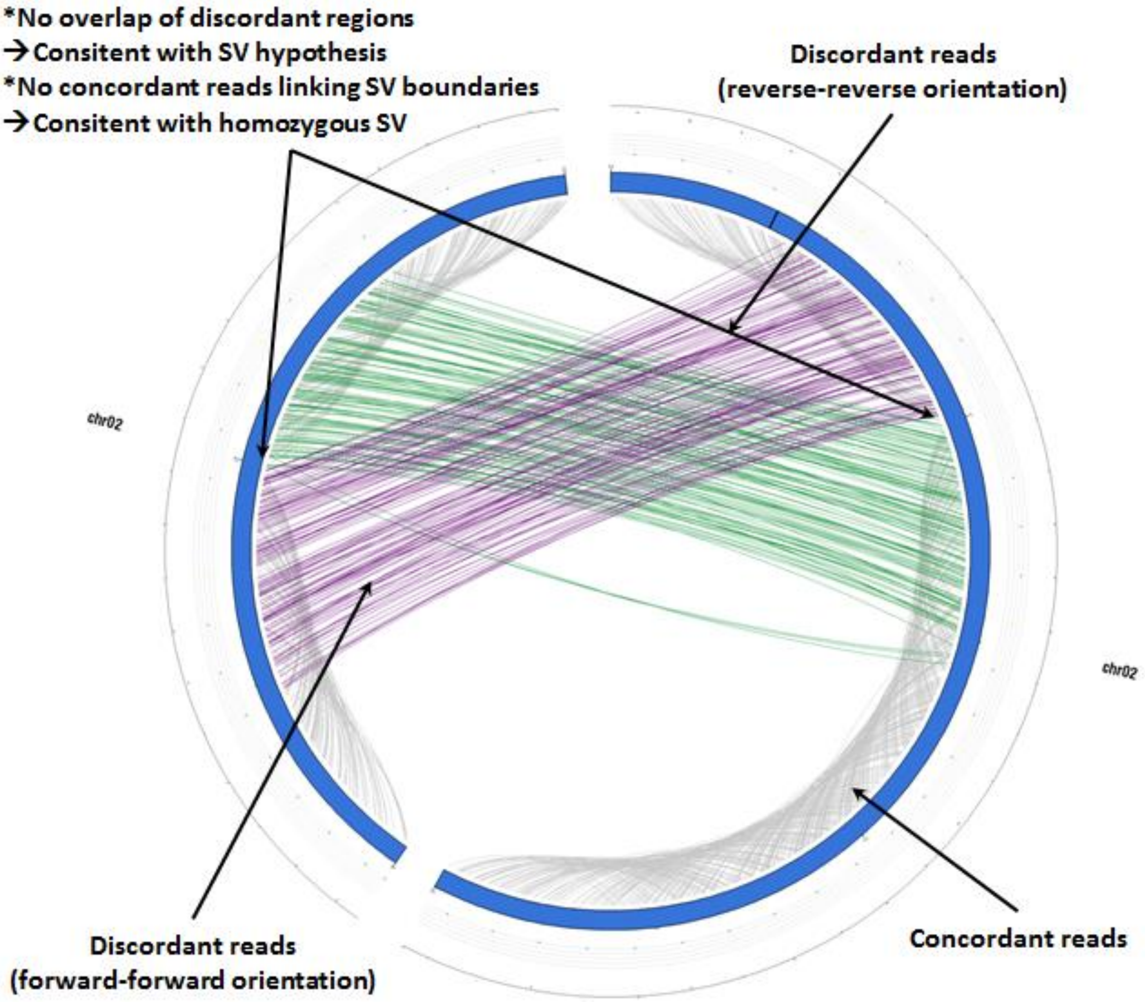
*No overlap of discordant regions
→Consitent with SV hypothesis
*No concordant reads linking SV boundaries
→Consitent with homozygous SV
Discordant reads
(reverse-reverse orientation)
chr02
chr02
Discordant reads
(forward-forward orientation)
Concordant reads

Figure 3 : Circos picture representing mapped read at the boundary of the inversion identified by scaffremodler tools.

Annex 1: Circos color coding.

All drawn links are color coded based on discordant type:

- Concordant read : grey
- Inter chromosomal discordance:

Reverse-forward : red
Forward-reverse : blue
Forward-forward : green
Reverse-reverse : purple

- Intra chromosomal discordance:

Insertion : red
Deletion : orange
Reverse-reverse : purple
Forward-forward : green

If expected orientation is reverse-forward :
    Forward-reverse : blue
If expected orientation is forward-reverse:
    Reverse-forward : blue