VCF_creator : module of DiscoSnp++

Mapping and VCF Creation features

Table of contents

User part

VCF_creator : one module in DiscoSnp++ Mapping and VCF Creation

VCF_creator is designed to map on a genome the output of DiscoSnp++ the Single Nucleotide Polymorphism (SNP) and small indels. This module create a Variant Calling Format (VCF) from an alignment obtained with the software BWA **(Li H. and Durbin R. (2010) Fast and accurate long-read alignment with Burrows-Wheeler Transform. Bioinformatics, Epub. [PMID: 20080505])**.

Quick starting

- Download and uncompress the DiscoSnp++
- Install BWA (you can add it to your PATH)

Running VCF_creator

- The main script ***run_VCF_creator.sh*** has three modes. You can use all modes according to your needs :
  - **MODE 1** : You havn't the reference genome but you want to create a vcf (it will summarize the DiscoSnp++ informations and will give the position of the variant on the upper path of DiscoSnp++ variant). The module will create a vcf from the output of DiscoSnp++ :
    - *./run_VCF_creator.sh -p <disco_file> -o <output> [-l <seed_size>] [-n <mismatch_number>] [-s <bwa_errors_in_seed>] [-w]"*
  - **MODE 2** : You want to aligne your variants against a reference genome. The module will run BWA to make an alignment between your reference genome and the output of DiscoSnp++ :
    - *./run_VCF_creator.sh -G <ref> -p <disco_file> -o <output> [-B <path_bwa>] [-l <seed_size>] [-n <mismatch_number>] [-s <bwa_errors_in_seed>] [-w]*
  - **MODE 3** : You already have an alignment (.sam file) and you want to create a vcf file :
    - *./run_VCF_creator.sh -f <sam_file> -n <mismatch_number> -o <output> [-l <seed_size>] [-s <bwa_errors_in_seed>] [-w]*

Options

General options :

- -p : DiscoSnp++ output file (<file>.fasta) (**Mandatory** unless MODE 3)
- -o : ouput (<file>.vcf) (**Mandatory)**
- -G : reference genome (<file.fasta>) (Only in MODE 2)
- -B : bwa path (*/home/me/my_programs/bwa-0.7.12/)* (note that bwa must be pre-compiled) (Only in MODE 2 ; optionnal if BWA is in the PATH)
- -f : alignment already done (<file>.sam) (Only in MODE 3)
- -h : show help
- -w : remove waste tmp files (index files)
- -I : creation of an output (VCF format) specific to IGV ((Integrative Genomics Viewer) : Sorting VCF by mapping positions and removing unmapped variants

BWA specifics options (- warning, bwa mapping running time highly depends on this parameter) :

- -s : *bwa option* : seed distance (Optionnal *default value 0*)
- -l : *bwa option* : length of the seed for alignment (Optionnal *default value 10*)
- -n : *bwa option* : maximal bwa mapping distance
  - Optionnal in MODE 1 and MODE 3 (*default value 3)*
  - **Mandatory** in MODE 3

Output

- **Final results are in your <file>.vcf.** It's a Variant Call Format that will summarize all the mapping informations and the header of DiscoSnp++ informations. Example :

```
#CHROM POS     ID      REF     ALT     QUAL    FILTER INFO    FORMAT G1      G2      G3      G4      G5
Pseudomonas   49949  950     A       C       .       PASS

Ty=SNP;Rk=0.02807;MULTI=.;DT=0;UL=.;UR=.;CL=.;CR=.;C1=1195,6;C2=1240,11;C3=891,8;C4=1315,13;C5=1028,
14;Genome=A;Sd=1     GT:DP:PL      0/0:1201:20,3463,23748      0/0:1251:14,3506,24540
       0/0:899:13,2520,17635      0/0:1328:14,3694,25997      0/0:1042:16,2828,20252
Pseudomonas   167569 10388_1       G       C       .       PASS

Ty=SNP;Rk=0.05257;MULTI=.;DT=1;UL=.;UR=.;CL=.;CR=.;C1=71,76;C2=80,78;C3=50,66;C4=71,73;C5=71,69;Geno
me=G;Sd=1     GT:DP:PL      0|1:147:997,16,1097 0|1:158:1144,16,1104       0|1:116:674,20,993  0|
1:144:1006,16,1046  0|1:140:1018,16,978
Pseudomonas   167575 10388_2       G       T       .       PASS

Ty=SNP;Rk=0.05257;MULTI=.;DT=1;UL=.;UR=.;CL=.;CR=.;C1=71,76;C2=80,78;C3=50,66;C4=71,73;C5=71,69;Geno
me=T;Sd=1     GT:DP:PL      0|1:147:997,16,1097 0|1:158:1144,16,1104       0|1:116:674,20,993  0|
1:144:1006,16,1046  0|1:140:1018,16,978
Pseudomonas   163823 16254  G       GCTGCAAGGCG   .       PASS

Ty=DEL;Rk=0.01243;MULTI=.;DT=0;UL=.;UR=.;CL=.;CR=.;C1=1170,104;C2=1341,115;C3=938,87;C4=1261,118;C5=
1027,85;Genome=.;Sd=-1     GT:DP:PL      0/0:1274:585,2289,21858    0/0:1456:630,2654,25096
       0/0:1025:505,1810,17488    0/0:1379:684,2420,23494    0/0:1112:459,2061,19258
```

- In this example there are the three types of DiscoSnp++ variants . In red : simple SNP ; In green close SNPs ; In black INDEL.
- The VCF file has the following fields :
  - **CHROM** : chromosome id where the prediction is mapped, or variant id of the upper

path if the variant is unmapped or if no reference genome is provided.

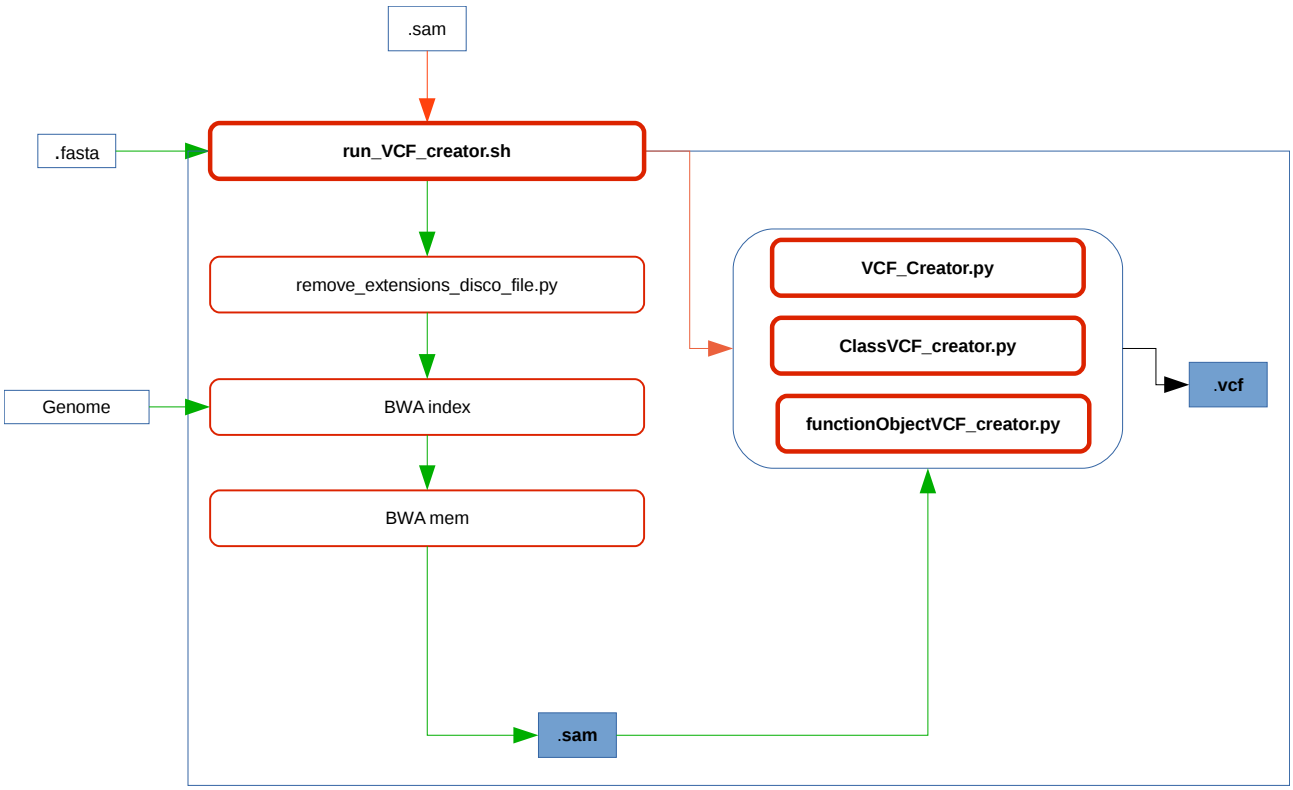
- **POS** (1-based):
  - If a reference genome is provided and if the couple is mapped on a unique position : the mapping position of the variant
  - If a reference genome is provided and if the couple is not uniquely mapped : one of the positions of the variant (1-based lefmost position)
  - Else (no reference genome provided or unmapped couple) : position of the variant on the upper path of the discoSnp++ prediction (including the left extension)
- **ID** : identification of the variant. For the close SNPs, the SNP number is added to the ID. Example : *10388_2*
- **REF** :
  - If one of the two predictions maps this position : the corresponding variant
  - Else, or if no reference genome provided : the lexicographically smallest of the two variants
  - In case of close SNPs : the first is defined as previously described. The following SNPs are those located on the same path
- **ALT** : The variant non reported as the “REF” variant
- **QUAL** : “.” (unused)
- **FILTER** :
  - PASS if the variant is mapped at unique position
  - MULTIPLE if the variant is mapped on multiple positions
  - “.” : if the variant is unmapped or if no reference genome is provided
- **INFO** :
  - **Ty** : Type of variant
    - SNP : If the variant is a simple SNP or close SNPs
    - INS : If the variant mapped corresponds to the path carrying the insertion
    - DEL : If the variant mapped corresponds to the path carrying the deletion
  - **Rk** : Rank of the prediction computed by DiscoSnp++
  - **UL** : Length of the left unitig (“.” if not computed)
  - **UR** : Length of the right unitig (“.” if not computed)
  - **CL** : Length of the left contig (“.” if not computed)
  - **CR** : Length of the right contig (“.” if not computed)
  - **Genome** : Applies only for SNPs when a reference genome is provided (“.” for INDELs and when no reference genome provide or if the variant is unmapped). Reference nucleotide. **Important Remark** : If one of the two predictions matches the reference : equal to the “REF” field, else equal to the nucleotide of the reference genome.

        - **Sd** : Applies only when a reference genome is provided (“.” if no reference genome provided or if the variant is unmapped). Strand of the prediction mapping. “1” : Forward ; “-1” : Reverse. **Important Remark** : Fields “REF” , “ALT” and “Genome” are based on the mapped predictions. If Sd is 1 then these fields correspond to the DiscoSnp++ prediction, else if the Sd is “-1”, then they correspond to the reverse complement of the DiscoSnp++ predictions.
    - **FORMAT** : Description of the genotype fields (G1, G2, G3 …)
        - **GT** : genotype, encodes as allele values. The genotype is determined by DiscoSnp++ (which considered the upper path as reference). If the “REF” corresponds the upper path, the genotype in the VCF is identical to the genotype in DiscoSnp++, else it's the opposite (1/1 becomes 0/0 and so on). About genotypes :
            - If the separator is a “/” the genotype are unphased (INDEL, Simple SNP)
            - If the separator is a “|” the genotype are phased (Close SNPs with the same ID)
        - **DP** : Combined depth across samples (sum)
        - **PL** : Phred-scaled Genotype Likelihood (given by DiscoSnp++)
        - **HQ** : Haplotype Quality (Q in DiscoSnp++ Header)
    - **Genotype :** G1, G2 , G3 and so on. Each fields contains all the informations of the format fields for each samples.

Developper part

Introduction

VCF_creator is designed to create a .vcf (Variant Calling Format) from a file .sam or .fasta.
Important Remark : All the function names will be writen in red italic. All the variable names will be written in blue.



Librairies python :

To run VCF_creator, you will need the following python librairies : os, sys, subprocess, re, time, getopt.

Structure of the input file

VCF_creator use a specific structure for its input files :

- Example for the <file>. sam (SEE doc of the samfile for more informations) :

```
SNP_higher_path_2028|P_1:30_T/G,P_2:50_G/C|high|nb_pol_2|C1_213|C2_215|C3_193|C4_295|C5_214|
G1_0/1:3468,22,2929|G2_0/1:3860,29,2902|G3_0/1:2708,18,2728|G4_0/1:3433,26,4291|G5_0/1:2918,19,3037|Q1_36|
Q2_37|Q3_37|Q4_37|Q5_38|rank_0.06490  16      Pseudomonas     650609  23      81M     *       0       0
        GGGGGCGAGCAGCAGATGCTCGCGATCGCCCGGGCGCTCATGAGCCGGCCAAAGCTGTTGCTGTTGGACGAGCCGAGCCTC *       XT:A:U  NM:i:1
        X0:i:1  X1:i:1  XM:i:1  XO:i:0  XG:i:0  MD:Z:50C30      XA:Z:Pseudomonas,-650610,1S80M,3;
SNP_lower_path_2028|P_1:30_T/G,P_2:50_G/C|high|nb_pol_2|C1_240|C2_263|C3_192|C4_252|C5_208|G1_0/1:3468,22,2929|
G2_0/1:3860,29,2902|G3_0/1:2708,18,2728|G4_0/1:3433,26,4291|G5_0/1:2918,19,3037|Q1_32|Q2_32|Q3_33|Q4_32|Q5_32|
rank_0.06490    16      Pseudomonas     650609  23      81M     *       0       0
        GGGGGCGAGCAGCAGATGCTCGCGATCGCCGGGGCGCTCATGAGCCGGCCCAAGCTGTTGCTGTTGGACGAGCCGAGCCTC *       XT:A:U  NM:i:1
        X0:i:1  X1:i:1  XM:i:1  XO:i:0  XG:i:0  MD:Z:30C50      XA:Z:Pseudomonas,-650610,1S80M,3;
```

Mandatory :

- Tab-delimited file
- The rows work in pairs (upper path following by the lower path) (we do not considerate supplementary alignment in samfile)
- The header of the variant must contain (discoSnp++ format) : the position of the variant (`P_1:30_T/G,P_2:50_G/C`) , the coverage (`C1_213|C2_215|C3_193|C4_295|C5_214`)
- The line of the samfile has to contain (in case of mapped variant): the cigarcode, the MD tag, the NM tag

- Example for the <file>.fasta :

```
>SNP_higher_path_13736|P_1:30_A/G|high|nb_pol_1|C1_53425|C2_1|C3_30|C4_3|C5_19|G1_0/0:1947,160066,1067676|
G2_0/1:40,9,20|G3_1/1:1895477,284398,3575|G4_0/1:34,9,54|G5_1/1:1306661,196101,2493|Q1_66|Q2_61|Q3_36|Q4_55|
Q5_33|rank_0.99909
AAGAGCACACGTCTGAACTCCAGTCACATAAGGATCTCGTATGCCGTCTTCTGCTTGAAAA
>SNP_lower_path_13736|P_1:30_A/G|high|nb_pol_1|C1_22|C2_2|C3_94832|C4_2|C5_65370|G1_0/0:1947,160066,1067676|
G2_0/1:40,9,20|G3_1/1:1895477,284398,3575|G4_0/1:34,9,54|G5_1/1:1306661,196101,2493|Q1_45|Q2_62|Q3_66|Q4_66|
Q5_66|rank_0.99909
AAGAGCACACGTCTGAACTCCAGTCACATAGGGATCTCGTATGCCGTCTTCTGCTTGAAAA
>SNP_higher_path_15103|P_1:30_C/G|high|nb_pol_1|C1_4|C2_1|C3_1|C4_1|C5_7|G1_0/0:4,16,84|G2_0/0:4,7,24|
G3_0/0:4,7,24|G4_0/1:97,15,17|G5_0/1:44,14,124|Q1_42|Q2_40|Q3_56|Q4_66|Q5_46|rank_0.65101
GAGTGCCGCGGCCAGAACGGTCGCCGCTTCCCCCTCAGCGATGATCCACGCCTGCTCGACC
```

Mandatory :

- Header : discoSnp++ format
- The rows work in pairs (upper path following by the lower path)

Program Steps : VCF_creator.py

Operates in General

- Prints the header of the VCF
- Counts the number of Genotype
- Goes through the input file
  - Initialyzes Variant object by checking the type of the prediction
    - SNP
    - INDEL
    - SNPSCLOSE
  - Checks the mapping to fill the VCF file
  - Fills the VCF

FunctionObjectVCF_creator.py

- *InitVariant* : Initializing all usefull objects to VCF creation.

- Starts with finding the variant type => initialyzes variant_object
- Initialyzes the VCF object => vcf_field_object
- Gets the information from discosnp++ header => variant_object.*FillInformationFromHeader*

- *MappingTreatement* :
  - Gets from the dicoAllele all the positions (of each variant on the path), and the nucleotides for each variant : variant_object.*GetPolymorphismFromHeader*
  - Gets the coverage of each path : variant_object.upper_path.*RetrieveCoverage*
  - Gets the quality of each path : variant_object.upper_path.*RetrieveQuality*
  - Checks the position of the variant on the reference (because of a possible shift during the mapping), and determines if the variant is identical to the reference genome (same nucleotide at the same position) : variant_object.upper_path.*CheckPosVariantFromRef*
  - Determines the path that will be used as reference (by checking the resuly of the previous step) variant_object.*WhichPathIsTheRef*
  - Defines the mapping position for the couple of path : variant_object.*GetMappingPositionCouple*
  - Fills the filter fields with *CheckAtDistanceXBestHits*
  - Gets the genotypes : variant_object.GetGenotypes

- *UnmappedTreatement* :
  - Gets the sequence for each path variant_object.upper_path.*GetSeq*
  - Gets from the dicoAllele all the positions (of each variant on the path), and the nucleotides for each variant : variant_object.*GetPolymorphismFromHeader*
  - Gets the coverage of each path : variant_object.upper_path.*GetCoverage*
  - Gets the quality of each path : variant_object.upper_path.*RetrieveQuality*
  - Fills the attribut of the path object and get the position of each variant on the path variant_object.upper_path.*CheckPosVariantFromRef*
  - Determines the path that will be used as reference (by checking the resuly of the previous step) variant_object.*WhichPathIsTheRef*
  - Defines the mapping position for the couple of path : variant_object.*GetMappingPositionCouple*
  - Fills the filter fields with *CheckAtDistanceXBestHits*
  - Gets the genotypes : variant_object.GetGenotypes

- *CounterGenotype*:
    - Count the number of Genotypes in the input
- *CheckAtDistanceXBestHits*:
    - Prediction validation : checks if the couple is validated with only one mapping position
        - In case of unmapped path, the filter field will be "."
        - Finds the smallest distance of mapping for all the positions of the upper path
        - Finds the smallest distance of mapping for all the positions of the lower path
        - For each path, we find all positions mapped with the smallest distance
        - The variant will be considered as PASS if there is one and only one mapping position with the smallest mapping distance for both path (else more than one position, the variant will be considered as MULTIPLE)

ClassVCF_creator.py

- Class VARIANT:

  Attributs:
    - upper_path : line in the file corresponding to the upper path
    - lower_path : line in the file corresponding to the lower path
    - variantID: ID of discoSnp++
    - discoName: name of the variant : SNP_higher_path_99
    - unitigLeft: length of the unitig left
    - unitigRight: length of the unitig right
    - contigLeft: length of the contig left
    - contigRight: length of the contig right
    - rank: rank calculated by discosnp++
    - nb_pol: number of polymorphisme in the disco path
    - dicoGeno: dictionnary with the information of the genotype dicoGeno[listgeno[0]]=[listgeno[1],listlikelihood]
    - dicoAllele: dictionnary of all the information from the header of discosnp++ : depending on the variant
    - mappingPositionCouple : mapping position of the bubble after correction
- Class PATH:

  Attributs:

- listCoverage : list of all the coverage by sample for the path
- dicoMappingPos : dictionnary with all the mapping positions associated with their number of mismatches with the reference
- listNucleotideReverse : list of all the variant (snp) of the path on the reverse strand
- listNucleotideForward : list of all the variant (snp) of the path on the forward strand
- boolReverse : Boolean to know if the strand is reverse(-1) or forward(1)
- posMut : MD tag of the samfile "MD:Z:5A10A0A25G17" => 5A10A0A25G17
- cigarcode : cigarcode of the samfile "61M"
- boolRef : Boolean to know if the path is identical to the reference
- nucleoRef : Nucleotide corresponding to the variant on the reference
- nucleo : nucleotide corresponding of the variant on the path
- listPosVariantOnPathToKeep : list of the positions of all the variant on the in case of Reverse or Forward mapped path
- listPosReverse : mapping position(s) of the variant on the path (if it is mapped on the reverse strand)
- listPosForward : mapping position(s) of the variant on the path (if it is mapped on the forward strand)
- listSam : line of the samfile
- discoName : name given by discoSnp++
- seq : sequence
- mappingPosition : mapping position of the path

- Class SNP:

- Class INDEL:

  Attributs:

  - insert : sequence of the insertion
  - nucleotideStart: nucleotide just before the insertion
  - smallestSequence : smallest sequence of the two paths
  - longestSequence : longest sequence of the two paths
  - ambiguityPos : possible shift of the ambiguity

- Class SNPSCLOSE:

Attributs:

- dicoCloseSNPUp : dictionnary with all the informations for close snps : boolean to know if the allele is identical to the reference,position on the variant, reverse nucleotide for the allele, if the path is reverse or not, mapping position
- dicoCloseSNPLow

- Class VCFFIELD:

  Attributs:

  - filterField : PASS, MULTIPLE or .. Fills by CheckAtDistanceXBestHit
  - variantType:
  - phased : phased genotype
  - formatField : INFO with contig, rank,...
  - genotype : genotype field
  - chrom :
  - ref :
  - alt :
  - qual :
  - nucleoRef : nucleotide of the reference
  - reverse : Strand of mapping

How to test VCF_creator :

./test_VCF_creator.sh /my/home/to/bwa-?.?.?/ >logTest.txt

If the modification in VCF creator do not change the output, logTest.txt will be :

Test VCF_Creator will take :

#HMMM_with_indels_k_31_c_4_D_500_P_5_b_1_withlow_coherent.fa

#GenomePseudomonasNC_007492.fasta

#data_test_creator.sam

****************Running Test 1****************

##Will create a vcf file for IGV : Sorting VCF by mapping positions and removing unmapped variants

...Indexation : Using the existing index...

... Creation of the vcf file : done ...==> afac.vcf

... Creation of the vcf file for IGV: done ...==> afac_for_IGV.vcf

2c2

< ##filedate=2016324

> ##filedate=2016322

!!! Test 1 : Difference between the two files !!!

*****************Running Test 2*****************

...Skipping alignment phase...

... Creation of the vcf file : done ...==> VCF_data_test.vcf

2c2

< ##filedate=2016322

> ##filedate=2016324

!!! Test 2 : Difference between the two files!!!

*****************Running Test 3*****************

...Ghost mode...

...Creation of a vcf without alignment...

... Creation of the vcf file : done ...==> afac.vcf

2c2

< ##filedate=2016324

> ##filedate=2016322

!!! Test 1 : Difference between the two files !!!

To recreate the gold standard :

./run_VCF_creator.sh -G GenomePseudomonasNC_007492.fasta -p HMMM_with_indels_k_31_c_4_D_500_P_5_b_1_withlow_coherent.fa -o GOLDstandardHMMPseudomonas.vcf -B $pathBWA -I

./run_VCF_creator.sh -f data_test_creator.sam -o GoldVCF_data_test.vcf

./run_VCF_creator.sh -p HMMM_with_indels_k_31_c_4_D_500_P_5_b_1_withlow_coherent.fa -o GOLDstandardHMMPseudomonasWithoutRef.vcf