Cancer Genomics

July 2015

Tobias Rausch

CONTENTS

CHAPTER

ONE

STRUCTURAL VARIANT ANALYSIS USING SIMULATED DATA

1.1 Getting Started

As a quick introduction we will do a structural variant analysis of the simulated data provided by the ICGC-TCGA DREAM Mutation Calling challenge [Dream]. The data has been further down-sampled to speed up all analyses discussed hereafter. Using shell variables we first link the practical data and the picard tools.

```
mkdir sv<YourFirstName>
cd sv<YourFirstName>
export PI=/home/training/Applications/picard-tools/picard.jar
export DE=/home/training/Data/delly/
export SF=$DE/variantFiltering/somaticVariants/somaticFilter.py
export DS=/home/training/Data/ngs2015sim/
export DR=/home/training/Data/ngs2015real/
export PERL5LIB=/home/training/Applications/vcftools/perl/
```

1.2 Alignment

The simulated data has already been aligned to chr20 of hg19. The chr20 reference FASTA file is called '20.fa'. There is one bam file [Bam] for the tumor (file tumor20.bam) and one for the matched normal (file normal20.bam). Using samtools [Li2009] we can have a look at the first few records.

```
samtools view $DS/tumor20.bam | head
```

If you haven't done so yet please familiarize yourself with the bam format, the required fields present in every bam alignment record are explained below:

Col	Field	Description
1	QNAME	Query template NAME
2	FLAG	bitwise FLAG
3	RNAME	Reference sequence NAME
4	POS	1-based leftmost mapping POSition
5	MAPQ	MAPping Quality
6	CIGAR	CIGAR string
7	RNEXT	Ref. name of the mate/next read
8	PNEXT	Position of the mate/next read
9	TLEN	observed Template LENgth
10	SEQ	segment SEQuence
11	QUAL	ASCII of Phred-scaled base QUALity+33

The bitwise FLAG can be decoded using the explain flag tool from the picard distribution [ExplainFlag].

Delly [Rausch2012] uses paired-end mapping to discover structural variants and the method can handle paired-ends from different libraries as long as these have been tagged with unique read-groups (@RG tags). These read-groups are listed in the bam header.

```
samtools view -H $DS/tumor20.bam
```

1.3 Structural Variant Discovery using Delly

Since this is simulated data we will skip the alignment quality control and move on directly to the structural variant calling. Delly [Rausch2012] calls structural variants jointly on the tumor and normal genome and outputs a VCF (variant-call-format) file. The VCF specification can be found on the VCFtools website [VCFtools]. The ICGC-TCGA DREAM Mutation Calling challenge simulated only deletions, duplications and inversions so we will skip the translocation prediction. Besides the bam files, you can provide a reference genome which then triggers the split-read search of Delly. You can also provide a text file with regions to exclude from the analysis of structural variants. The default exclude map of Delly includes the telomeric and centromeric regions of all human chromosomes since these regions cannot be accurately analyzed with short-read data.

```
$DE/src/delly -t DEL -g $DS/20.fa -x $DS/hg19.ex -o del.vcf $DS/tumor20.bam $DS/normal20.bam
$DE/src/delly -t DUP -g $DS/20.fa -x $DS/hg19.ex -o dup.vcf $DS/tumor20.bam $DS/normal20.bam
$DE/src/delly -t INV -g $DS/20.fa -x $DS/hg19.ex -o inv.vcf $DS/tumor20.bam $DS/normal20.bam
```

A vcf file [VCFtools] has multiple header lines starting with the hash # sign. Below the header lines is one record for each structural variant. The record format is described in the below table:

Col	**Field**	**Description**
1	CHROM	Chromosome name
2	POS	1-based position. For an indel, this is the position preceding the indel.
3	ID	Variant identifier. Usually the dbSNP rsID.
4	REF	Reference sequence at POS involved in the variant. For a SNP, it is a single base.
5	ALT	Comma delimited list of alternative sequence(s).
6	QUAL	Phred-scaled probability of all samples being homozygous reference.
7	FILTER	Semicolon delimited list of filters that the variant fails to pass.
8	INFO	Semicolon delimited list of variant information.
9	FORMAT	Colon delimited list of the format of individual genotypes in the following fields.
10+	Samples	Individual genotype information defined by FORMAT.

You can look at the header of the vcf file using grep, '-A 1' includes the first structural variant record in the file:

```
grep "^#" -A 1 inv.vcf
```

In general, Delly uses the VCF:INFO fields for structural variant site information, such as how confident the structural variant prediction is and how accurate the breakpoints are. The genotype fields contain the actual sample genotype, its genotype quality and genotype likelihoods and various count fields for the variant and reference supporting reads and spanning pairs. If you browse through the vcf file you will notice that a subset of the Delly structural variant predictions have been refined using split-reads. These precise variants are flagged in the vcf info field with the tag 'PRECISE'. To count the number of precise and imprecise variants you can simply use grep.

```
grep -c -w "PRECISE" *.vcf
grep -c -w "IMPRECISE" *.vcf
```

Please note that this vcf file contains germline and somatic structural variants but also false positives caused by repeat-induced mis-mappings or incomplete reference sequences. As a final step we have to use the structural variant site information and the tumor and normal genotype information to filter a set of confident somatic structural variants.

1.4 Somatic Structural Variant Filtering

Delly ships with a somatic filtering python script. For a set of confident somatic calls one could exclude all structural variants <400bp, require a minimum variant allele frequency of 10%, no support in the matched normal and an overall confident structural variant site prediction with the VCF filter field being equal to PASS.

```
python $SF -t DEL -v del.vcf -o del.filt.vcf -a 0.1 -m 400 -f
python $SF -t DUP -v dup.vcf -o dup.filt.vcf -a 0.1 -m 400 -f
python $SF -t INV -v inv.vcf -o inv.filt.vcf -a 0.1 -m 400 -f
```

Using VCFtools [VCFtools] we can merge all somatic structural variants together in a single vcf file.

```
vcf-concat del.filt.vcf dup.filt.vcf inv.filt.vcf  | vcf-sort > somatic.sv.vcf
vcf-validator somatic.sv.vcf
```

For large VCF files you should also zip and index them using bgzip and tabix.

```
bgzip somatic.sv.vcf
tabix somatic.sv.vcf.gz
```

The final step will be to browse some of these somatic structural variants in IGV [IGV]. To make that easier we will also create a bed file with the start and end coordinates of each structural variant. This vcf2bed conversion is a bit ugly because VCF was designed for single-nucleotide variants, so I wrote a small shell script to do this conversion, called ‘vcf2Bed.sh’.

```
$DE/converter/dellyVcf2Bed.sh somatic.sv.vcf.gz > somatic.sv.bed
head somatic.sv.bed
```

We then start IGV using the chr20 reference.

```
igv.sh -g $DS/20.fa
```

Once IGV has started use ‘File’ and ‘Load from File’ to load the tumor and normal bam alignment file. Then import ‘somatic.sv.bed’ from your working directory using ‘Regions’ and ‘Import Regions’. The somatic structural variants can then be browsed easily using ‘Regions’ and ‘Region Navigator’. Select a structural variant in the Region Navigator and click ‘View’, which will center the IGV alignment view on the selected structural variant. It’s usually best to zoom out once then by clicking on the ‘-‘ sign in the toolbar at the top, so you can view all supporting abnormal paired-ends. To highlight the abnormal paired-ends please right click in IGV on tumor20.bam and activate ‘View as pairs’. In the same menu, please open ‘Color alignments by’ and then switch to “pair orientation’, afterwards repeat these steps for normal20.bam. For deletions, you can also color the alignments by “insert size”. You can obviously play around with all the different visualization options but the above two are what I found most useful for a discordant paired-end analysis. Using the Region Navigator you can easily iterate now through the predicted, somatic SVs.

CHAPTER

TWO

STRUCTURAL VARIANT ANALYSIS USING CANCER DATA

2.1 Alignment Quality Control

For structural variant calling several alignment quality control metrics are important. All paired-end mapping methods heavily rely on the insert size distribution. Read-depth based callers can be affected by GC-content biases. The percentage of mapped reads, singletons, duplicates and properly paired reads are additional metrics you should evaluate prior to any structural variant discovery. The picard [Picard] and samtools [Samtools] libraries provide some commands to compute these simple alignment statistics. Since this is a real data set we will use some of these below:

```
samtools flagstat $DR/normal2.bam
samtools flagstat $DR/tumor2.bam
java -jar $PI CollectInsertSizeMetrics I=$DR/normal2.bam O=isN.txt R=$DR/2.fa H=isN.pdf
java -jar $PI CollectInsertSizeMetrics I=$DR/tumor2.bam O=isT.txt R=$DR/2.fa H=isT.pdf M=0.5
evince isN.pdf
evince isT.pdf
java -jar ${PI} CollectGcBiasMetrics I=$DR/normal2.bam O=gcN.txt R=$DR/2.fa CHART=gcN.pdf
java -jar ${PI} CollectGcBiasMetrics I=$DR/tumor2.bam O=gcT.txt R=$DR/2.fa CHART=gcT.pdf
evince gcN.pdf
evince gcT.pdf
```

Please read the on-line documentation for these tools to get an explanation of the different metrics [Picard]. Obviously there are some general rules how a good alignment should look like, e.g. a heavy GC-bias, mapping percentages below 70% or multiple peaks in the insert size distribution are clearly warning signals. However, these statistics vary largely by protocol and hence, it's usually best to compare multiple different sequencing runs using the same protocol (DNA-seq, RNA-seq, ChIP-seq, paired-end, single-end or mate-pair) against each other, which then highlights the outliers.

2.2 Complex Structural Variants

The simulated data contained only simple inversions, tandem duplications and deletions. This cancer data set harbors many complex rearrangements instead. As a first step we will again compute all somatic structural variants using Delly [Rausch].

```
$DE/src/delly -t DEL -g $DR/2.fa -x $DR/hg19.ex -s 15 -o del.vcf $DR/tumor2.bam $DR/normal2.bam
$DE/src/delly -t DUP -g $DR/2.fa -x $DR/hg19.ex -o dup.vcf $DR/tumor2.bam $DR/normal2.bam
$DE/src/delly -t INV -g $DR/2.fa -x $DR/hg19.ex -o inv.vcf $DR/tumor2.bam $DR/normal2.bam
python $SF -t DEL -v del.vcf -o del.filt.vcf -a 0.3 -m 400 -f
python $SF -t DEL -v dup.vcf -o dup.filt.vcf -a 0.3 -m 400 -f
python $SF -t DEL -v inv.vcf -o inv.filt.vcf -a 0.3 -m 400 -f
vcf-concat del.filt.vcf dup.filt.vcf inv.filt.vcf  | vcf-sort > somatic.sv.vcf
vcf-validator somatic.sv.vcf
bgzip somatic.sv.vcf
```

```
tabix somatic.sv.vcf.gz
$DE/converter/dellyVcf2Bed.sh somatic.sv.vcf.gz > somatic.sv.bed
head somatic.sv.bed
```

We can now again use IGV [IGV] to look at some of the somatic structural variants in detail and if time permits you are welcome to do so. However, IGV is less good for complex rearrangements so we will very quickly create our own read-depth ratio plot using R statistics [R]. To create the coverage track we use an auxilliary tool called ‘cov’ included in the Delly package [Rausch].

```
$DE/src/cov -f tn.cov.gz $DR/tumor2.bam $DR/normal2.bam
R
```

In R [R] we first load the coverage track and then overlay the somatic structural variants.

```
library(ggplot2)
library(reshape2)
library(scales)
cov = read.table("tn.cov.gz", comment.char="$", header=T)
sv = read.table("somatic.sv.bed")
colnames(sv) = c("chr", "start", "end", "id")
sv = sv[(sv$end - sv$start)>10000000,]
sv$middle = as.integer( (sv$start + sv$end) / 2 )
sv$yv = 4
sv = melt(sv, id.vars=c("chr", "id", "yv"))
sv$svtype = substr(sv$id, 0,3)
sv[sv$variable=="middle",]$yv = sv[sv$variable=="middle",]$yv - 0.5
sv = sv[order(sv$value),]
cov=cov[cov$normal2!=0,]
covNorm=median(cov$normal2)/median(cov$tumor2)
cov$tumor2=log2(covNorm*cov$tumor2/cov$normal2)
p=ggplot(data=cov, aes(x=cov$start, y=cov$tumor2)) + geom_point(alpha=3/4, size=2)
p=p + xlab("chr2") + ylab("Log2 read-depth ratio") + scale_x_continuous(labels = comma)
p=p + geom_line(data=sv, aes(x=value, y=yv, group=id, colour=svtype)) + labs(colour="SV")
p
quit()
```

The R plot is still not interactive. Delly [Rausch] includes a first beta version of a browsable read-depth plot with structural variant connections. We first need to convert the bam files to hdf5 coverage tracks:

```
python $DE/vis/suave/suave_bam_to_h5.py -s tumor2 -o tumor2.hdf5 $DR/tumor2.bam
python $DE/vis/suave/suave_bam_to_h5.py -s normal2 -o normal2.hdf5 $DR/normal2.bam
```

Then we start a local webserver with python using the hdf5 files and the somatic structural variant calls:

```
python $DE/vis/suave/suave_server.py -f tumor2.hdf5 -f normal2.hdf5 -v somatic.sv.vcf.gz
```

As you can see on the command line the server is running on the localhost on port 5000. So we open a new terminal and start firefox with that URL:

```
firefox http://127.0.0.1:5000/
```

In the browser, go to setup and load the tumor and normal bam file, select chr2, and click “Visualize”. If you move the mouse into the read-depth window, you can zoom, pan and select subregions in the lower overview panel. You can also click on the SV arcs to center the view on the selected structural variant. To go back to the full chromosome view just double-click in the overview panel at the bottom. Once you are done you can close the server with CTRL-C.

This concludes my short practical on structural variant calling. I hope you enjoyed it!

BIBLIOGRAPHY

[References]

[IGV] Thorvaldsdóttir H, Robinson JT, Mesirov JP. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. Brief Bioinform. 2013 Mar;14(2):178-92.

[Li2009] Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R; 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009 Aug 15;25(16):2078-9.

[Rausch2012] Rausch T, Zichner T, Schlattl A, Stütz AM, Benes V, Korbel JO. DELLY: structural variant discovery by integrated paired-end and split-read analysis. Bioinformatics. 2012 Sep 15;28(18):i333-i339.

[URLs]

[Bam] http://samtools.github.io/hts-specs/

[Delly] https://github.com/tobiasrausch/delly

[Dream] https://www.synapse.org/#!Synapse:syn312572

[ExplainFlag] http://broadinstitute.github.io/picard/explain-flags.html

[Picard] http://broadinstitute.github.io/picard/

[R] http://www.r-project.org/

[Rausch] http://tobiasrausch.github.io

[Samtools] http://www.htslib.org

[VCFtools] http://vcftools.sourceforge.net/