Bi188 2013

Computational exercise 1. Answers

1 Questions

1.1 Coverage estimation

How good of a coverage of the exome did you get? Explain your algorithm/calculation.

This question has multiple answers at varying levels of accuracy.

We begin by recalling the definition of the coverage C over some set of sequence features:

$$C = \frac{\sum_{i \in 1,..,N} L_i}{|B|} \tag{1}$$

Where N is the number of reads, L_i is the length of each individual read, and $|B|$ is the size of the base pair space over which we calculate the coverage (this would be the length of the genome for a whole genome, we will discuss in a second what it is for an exome).

In this case, all reads have the same length, so the equation becomes:

$$C = \frac{N * L}{|B_E|} \tag{2}$$

Where $|B_E|$ is the length of the exome. There are a few things to note, however:

1. We define the exome as the set of exonic bases in the genome. A lot of genes have alternative splice isoforms, which means that many exons are annotated more than once. Also, alternative splice site selection also happens, which results in overlapping but non-identical exons.

2. We are not sequencing only the exome - we are doing a capture of the exome using 50bp-long probes on genomic fragments with an average length of 150-200bp. This means we could get the last 50bp of an exon to capture those 50bp plus an additional 100-150bp extending in the intron. This is a bit underdefined, and if anyone had asked about it, I would have told you to assume equal coverage extending into the 100bp on each side of the exon, but nobody did.

Anyway, assuming equal coverage of the exome and the flanking 100bp, we can get a quick estimate of our coverage as follows:

$$C = \frac{N * L}{|B_{exome+flanking100bp}|} \tag{3}$$

In this case, the read length L is 75bp. The number of reads can be easily obtained with the `samtools idxstats` program, which outputs the following:

```
chr      length      number alignments on chromosome      number unaligned reads
```

As we have retained only the unique alignments, the sum of the values in the third column gives the total number of aligned reads (remember that we are calculating this on the BAM file we are left with after removing duplicate reads). On the data you were given, the number of reads is 92,581,905.

The number of exonic bases can be easily calculated with a simple python script, an example of which can be found here:

```
/woldlab/bostau/data00/pub/georgi/Bi188/Exercise1/GTFExonicBasePairs.py

python GTFExonicBasePairs.py hg19-refSeq.withNames.gtf hg19-refSeq.withNames.number_bases
                                                                                -extend 100
Number exonic basepairs = 108260906
```

The number of extended exonic base pairs the script produces is 108,260,906.
The coverage is then:

$$C = \frac{N * L}{|B_{exome+flanking100bp}|} = \frac{92581905 * 75}{108260906} \approx 64X.$$

If you did not extend the exons, you should have gotten the following number of exonic bases:

```
python GTFExonicBasePairs.py hg19-refSeq.withNames.gtf hg19-refSeq.withNames.number_bases
                                                                                  -extend 100
Number exonic basepairs = 66994534
```

And correspondingly:

$$C = \frac{N * L}{|B_{exome+flanking100bp}|} = \frac{92581905 * 75}{66994534} \approx 103X.$$

Finally, you might have noticed that not all of your reads were aligned with the alginments for those being suppressed due to the read aligning more than once to the genome.

```
13806421 (10.11%) aligned >1 times
```

If you were to do a really sophisticated coverage calculation, you would have taken this into account by creating a mappability track for the genome, then counting read coverage only over the uniquely mappable exonic bases. This would have taken you quite some time to do though, and we by no means expected you to do it.

Scoring: 3 points total, 1 point partial credit for the incorrect answer discussed below, 2 points partial credit for the $\sim 103X$ answer, 3 points for the $\sim 64X$ answer

Note that what most of you did was to take the left and right coordinates of the exons, subtract the left from the right one, then sum the resulting numbers. This gives an answer close to the correct 64X (68X) but for the wrong reasons. As I explained above, due to alternative splicing, a lot of basespairs are annotated as belonging to multiple transcripts (and, accordingly, multiple exons); if you do this, you will count them multiple times. This overcounting happens to almost match the additional basepairs that should be added for the flanking intronic sequence that is also captured, but this is only true for the refSeq annotation you happen to be using here. If you were using GENCODE, you would have been off by a wide margin. This is why I am giving only 1 point partial credit here.

1.2 Variant detection

How many SNVs do you detect in the sample? How many of them are novel? How would you classify them according to their expected effect on protein sequence and how many in each class do you see? Ignore indels for the purpose of this exercise.

Ignoring indels, I got XXXX SNVs, of which XXX were and XXX were not present in dbSNP. I did this as follows:

```
/woldlab/bostau/data00/pub/georgi/Bi188/Exercise1/VCF_general_comparison.py

python VCF_general_comparison.py vcf1 vcf2 outfile_prefix [-noVCF1-only] [-noVCF2-only]
                                                                    [-ignoreIndels]

python VCF_general_comparison.py SNVs dbSNP.bz2 SNVs-vs-dnSNP -noVCF2-only -ignoreIndels
```

Which will produce two files - one with the common SNPs, and one with the ones only found in the first file, i.e. novel SNVs from the de novo calls.

We then have to annotate the variants. I would have picked the following categories of variants with respect to their functional impact on the genes that contain them:

- synonymous
- missense
- nonsense
- splice site
- 3'UTR
- 5'UTR

Where all of these should be self-explanatory except for one needed clarification that I would consider a splice-site disrupting SNV one that changes the two basepairs on each end of the intron. This is my take on the problem of classifying them:

```
/woldlab/bostau/data00/pub/georgi/Bi188/Exercise1/VariantAnnotation.py

python VariantAnnotation.py hg19.fa SNVs-vs-dnSNP.vcf1-only.vcf hg19-refSeq.withNames.gtf
                                                        SNVs-vs-dnSNP.vcf1-only.annotation
```

I got 1209 non-indel non-dnSBP SNPs, of which the following were within exons or potentially affecting splice sites (there are none of the latter):

homo/heterozygous	Effect	Number
0/1	3'UTR	81
0/1	5'UTR	126
0/1	missense	100
0/1	nonsense	6
0/1	synonymous	52
1/1	3'UTR	36
1/1	5'UTR	39
1/1	missense	31
1/1	nonsense	2
1/1	synonymous	18

Here is something you should know, because it actually illustrates the reality of dealing with high-throughput sequencing data. The data the we gave you was not real data, it was generated by mixing a certain number of dbSNP SNVs with a few handpicked ones, then simulating reads from the resulting diploid genome. I did not handpick 1209 novel SNPs, I picked 10,000 random dbSNPs and 20 handpicked ones. The difference is false positives, due to mapping errors and to `samtools` not doing its job as well as it should. There are also false negatives - not all of the dbSNP SNVs showed up, which is because some reads are not alignable because they fell in repetetive regions of the genome. And as scary as this might sound, it is even scarier when you consider this is **simulated** data, which has less complex error structure than real-life data. This is the reason in real life people do extensive SNP filtering, and programs like GATK spend so much time recalibrating quality scores and realigning reads. For the purpose of what you did, there was no need to throw you in the deep end of the pool and have you deal with all that complexity, in fact, it would have been harmful as it would have confused you, but now that the exercise is over, it is good for you to be aware of the issue.

Scoring: 4 points total

1.3 Identification of causal variants

Now you want to find the most likely causal variant(s) for the condition. Explain how you would identify them and list the variant(s) and gene(s) you think are most likely to be mutated in this patient. Can you figure out which disease the patient has based on what you can find in the literature about you top candidate gene? Again, ignore indels for the purpose of this exercise. Use the following (tab-delimited) format for submitting your putative causal variants (use whatever categories you find appropriate to classify variants). Submit both answers to the questions (preferably by e-mail in pdf format) and a link to the code you wrote and your output files on the cluster.

```
#CHROM  POS        REF  ALT  GT   GeneName  GeneID     TranscriptName  TranscriptID  Effect
chr1    247007112  T    C,A  1/1  AHCTF1    NM_015446  AHCTF1          NM_015446     missense
chr1    110603213  G    A    1/1  ALX3      NM_006492  ALX3            NM_006492     UTR3
....
```

The most obvious candidates for the causal mutation are nonsense mutations, and those are naturally the first place to look. Splice mutations would also have such an effect, and incidentally, the original ataxia telangiectasia mutation is in fact affecting splicing. There should be no such mutations in your output though. You should, however, have found several nonsense SNPs, some of which homozygous. The correct answer is none of those because on close examination you would have noticed that they are very close to the stop codon in the ORF or have some other property that makes them unlikely to be

what you are looking for. You might have noticed, however, that there are two nonsense mutations in the ATM gene. Those need not be on different chromosomes, but it is reasonable to assume that this is indeed a case of a compound null heterozygosity given that the gene itself is known to be involved in precisely the kind of syndrome described in the beginning.

Scoring: 3 points total

Note that some of you gave the correct answer in terms of the gene involved but without actually completing the exercise. This is not good enough, and I have given 0.5 points partial credit for this. The reason it is not good enough is that mutations in other DNA repair pathway genes result in overlapping with the description of the case symptoms, i.e. I did not given you enough information for you to tell me the answer without analyzing the data.