DNA Sequencing

Cavan Reilly

November 25, 2019

Table of contents

DNA-Seq data analysis

There are a number of approaches to the analysis of DNA-Seq data.

Again we will only consider approaches that have been developed for organisms with sequenced genomes.

The basic problem all of these methods must try to solve is when to flag a variant based on disagreements between an individual's pile up and the reference genome.

We will cover the basic functionality available for this in SAMtools and related tools.

BCFtools

The program bcftools can be used to identify variants.

One first downloads the latest version, unpacks it, enters the directory that gets created then copies the executable to bin.

There is also a collection of tools bundled as htslib that can be useful for whole genome sequencing data (in particular tabix, htsfile and bgzip): this can be installed in an analogous fashion.

We will continue with the example we had used when we introduced samtools.

Variant calling

There we had:

1. created an index of a fasta file
2. converted a compressed sam file to a bam file
3. created an index of the bam file
4. viewed the collection of mapped reads

However this software can do more: it can call variants and has further capabilities when combined with bcftools, such as filtering.

We can use bcftools to flag potential variants and manipulate files that hold data on variants.

Variant calling

This is accomplished with the `mpileup` function: this is used to combine information about where a collection of reads map (from a bam file) with information about a reference genome to determine where there are mismatches in a sample compared to the reference.

A capability of samtools that we didn't investigate at that time was its ability to create vcf files.

A vcf file is a text file that has information about the genotype of a sample at a collection of locations.

Variant calling

Going back to our previous example, here is the result of mpileup using its native output format (the -f means we are supplying a fasta reference file)

```
[user0014@boris examples]$ samtools mpileup -f ex1.fa \
> ex1.bam > tmp
[user0014@boris examples]$ head -n 5 tmp
seq1    36      G       1       ^~.     =
seq1    37      T       2       .^k.    =<
seq1    38      C       2       A.      =<
seq1    39      C       2       ..      ;8
seq1    40      A       2       ..      =;
```

Variant calling

In this format there is a row for each genomic position and the columns are as follows

1. chromosome
2. position on chromosome
3. reference base
4. number of reads covering the site
5. read bases
6. read qualities
7. alignment mapping qualities

Note that generally more than 1 read will cover a position, so the last 3 items are presented for every read that covers the base.

Variant calling

Interpretation of the final 3 columns is complicated: e.g. a period in the read base column indicates a match to the reference.

For example, at position 39 there were 2 reads that covered the position and they matched the reference.

An easier to interpret and more widely used format is the vcf format: here is the syntax for that (-uv means provide vcf formatted output):

```
[user0014@boris bcftools-1.9]$ bcftools mpileup -f \
> ~/bin/samtools-1.9/examples/ex1.fa \
> ~/bin/samtools-1.9/examples/ex1.bam | \
> bcftools call -mv -Ob -o calls.bcf
```

Variant calling

Then we can examine that file and see that there is a large header that explains the content then a row for each genomic position.

We can also get that output in a binary format

```
[user0014@boris bcftools-1.9]$ bcftools view calls.bcf
```

Variant calling

One can also filter the variant calls using methods from bcftools using syntax like the following (here the option about QUAL filters for quality scores).

```
[user0014@boris examples]$ bcftools view -i '%QUAL>=60' \
> calls.bcf
```

Reading vcf files into R

A number of packages have been developed with the goal of providing a means of reading vcf files into R for further analysis. These include

1. VPA
2. vcfR
3. VariantAnnotation

The package vcfR is a stable way to examine vcf files in R.

These files can be quite large and processing them with R may not be the best approach.

Reading vcf files into R with vcfR

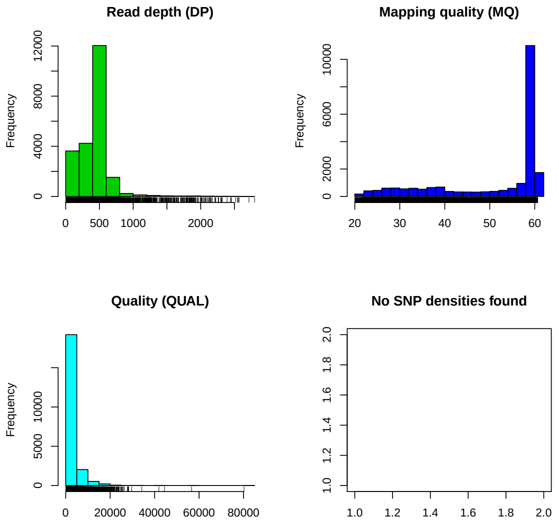
```
install.packages('vcfR')

pkg <- "pinfsc50"
vcf_file <- system.file("extdata",
+ "pinf_sc50.vcf.gz", package = pkg)
vcf <- read.vcfR( vcf_file, verbose = FALSE )
```

Reading vcf files into R with vcfR

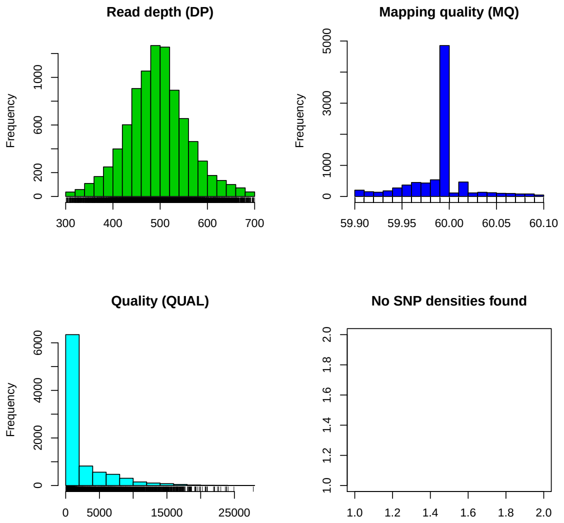
One can load a vcf file as a chromosome level object: here this example is using a supercontig followed by some quality filtering and replotting.

```
chrom <- create.chromR(name='Supercontig', vcf=vcf)

plot(chrom)

chrom <- masker(chrom, min_QUAL = 1, min_DP = 300,
+  max_DP = 700, min_MQ = 59.9,  max_MQ = 60.1)
plot(chrom)
```

Reading vcf files into R with vcfR



Reading vcf files into R with vcfR

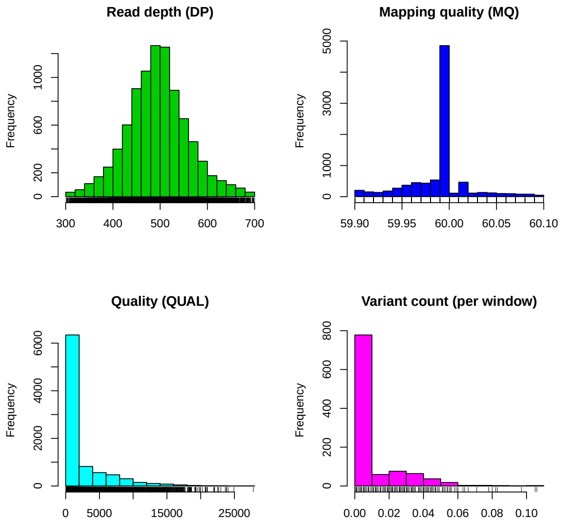


Reading vcf files into R with vcfR

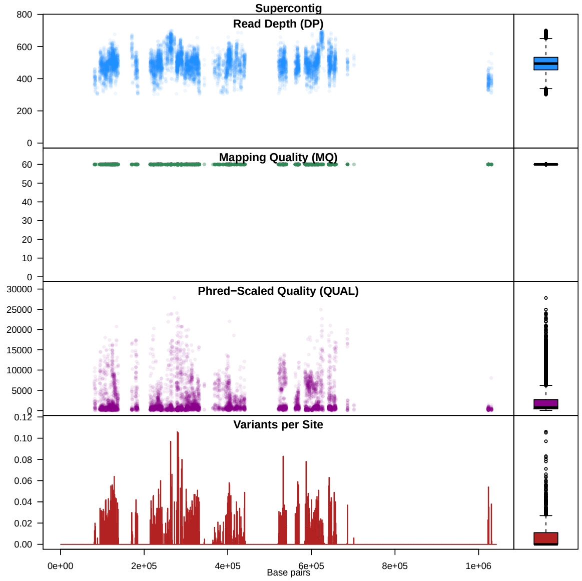
Now we can process the resulting dataset and examine variants:

```
chrom <- proc.chromR(chrom, verbose=TRUE)

plot(chrom)

chromoqc(chrom, dp.alpha=20)
```

Reading vcf files into R with vcfR



Reading vcf files into R with vcfR



GATK

For SNP identification the best practice currently seems to be based on using the genome analysis toolkit (GATK).

This toolkit consists of a collection of java programs that implement a multistep analysis.

There are 3 phases to the analysis process.

GATK

In the first phase, raw reads are mapped to generate SAM files and an accurate base error model is used to improve on the vendor's base quality calls (which are not very accurate).

In phase 2, the SAM/BAM files are analyzed to discover SNPs, short indels and copy number variants.

Finally, in phase 3 additional information, such as pedigree structure, known sites of variation and linkage disequilibrium, is brought to bear on the detected variants.

Detailed instructions can be found in the following publication (on the reading list)

Van der Auwera GA, Carneiro M, Hartl C, Poplin R, del Angel G, et al. (2013), "From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline", *CURRENT PROTOCOLS IN BIOINFORMATICS* 43:11.10.1-11.10.33.

Genome STRiP and copy number variation

There is substantial evidence from multiple sources that there is extensive variation in terms of deletions and insertions.

Such deletions and insertions imply that certain individuals are missing (or have extra copies of) certain portions of the genome.

This leads to copy number variation (CNV).

With next generation sequencing we can detect this from paired end reads that span too large of a genomic segment and from local variation in read depth.

Genome STRiP and copy number variation

The GATK has a workflow for detecting copy number variations: genome STRiP (for genome structure in populations).

To use this set of tools you need a FASTA file containing the reference sequence used for alignment and the BAM files that are the result of the alignment.

The FASTA file must be indexed using the `faidx` tool in SAMtools.

Genome STRiP and copy number variation

You also need to install the alignment program BWA and a set of programs called Picard.

The result is a file in the vcf format that has variant calls for each subject.

You also need data from at least 20 to 30 subjects, however if your sample size is too small you can use data from the 1000 Genomes project as your background population.

Microbiomics

One application of DNA-Seq technology is to determine the proportion of microbes in a sample.

We can do this as it has been discovered that all bacterial species have a component of the ribosome (the 16S ribosomal RNA) that is similar across species but differs by a sufficient amount at certain hypervariable locations to allow identification of at least the genus of a bacterial species.

We note that there are array based tools that allow one to do this too, e.g. the human oral microbe identification microarray (HOMIM).

For example we may want to determine what bacterial species are present in some sample, for example, in someone's mouth.

Microbiomics

The primary limitation of this method is that some common species have very similar sequences, for example here is part of an alignment of 2 *Streptococcal* species (*S. anginosus* and *S. intermedius*) that shows 97% sequence similarity:

```
Length=1558

                                       Sort alignments for this subject sequence by:
                                                    E value   Score   Percent identity
                                         Query start position Subject start position
 Score = 2571 bits (2850),  Expect = 0.0
 Identities = 1502/1555 (97%), Gaps = 0/1555 (0%)
 Strand=Plus/Plus
Query  19    TTTGATCCTGGTTCAGGACGAACGCTGGCGGCGTGCCTAATACATGCAAGTAGGACGCAC  78
             ||||||||||||||||||||||||||||||||||||||||||||||||||||| ||||||
Sbjct  1     TTTGATCCTGGTTCAGGACGAACGCTGGCGGCGTGCCTAATACATGCAAGTAGAACGCAC  60
Query  79    AGTTTATACCGTAGCTTGCTACACCATAGACTGTGAGTTGCGAACGGGTGAGTAACGCGT  138
             ||  |  ||||||| || ||||||| ||  ||||||||||||||||||||||||||||||
Sbjct  61    AGGATGCACCGTAGTTTACTACACCGTATTCTGTGAGTTGCGAACGGGTGAGTAACGCGT  120

...
```

Microbiomics

With the HOMIM system one obtains a value between 0 and 5 that indicates the quantity of a combination of species.

So a simple application would be to test for differences in microbe levels between 2 patient groups.

Given the semi-quantitative nature of this data I have used permutation tests to test for differences-this isn't too computationally intensive as there are only a several hundred bacteria represented on the microarray.

What is tricky is that any given probe typically interrogates multiple species, so going from a set of differentially expressed probes to a collection of differentially expressed species is difficult.

Microbiomics

A far more popular way to address the question of differential microbe levels is to use next generation sequencing.

So, one obtains a sample, sequences short reads then aligns these reads to the 16S rRNA DNA sequence of many bacteria to determine which it matches the best.

Rather than using Illumina's sequencing technology, pyrosequencing (or 454 sequencing) is typically used in these investigations.

This technology gives longer reads than Illumina's method: 800 bases are typical currently (and this appears to be increasing over time, as does Illumina's).

Microbiomics

Many early publications took used the ribosomal database project for alignment of the sequences they generate.

Cole JR, Wang Q, Cardenas E et al. (2008), "The ribosomal database project: improved alignments and new tools for rRNA analysis", *Nucleic Acids Research*, 37, D141-D145.

Currently DADA2 is a popular R package for conducting this analysis (although it uses results from this project).

This resource can then be used to generate a table of counts for each sample and each bacterial species detected.

We can then use edgeR or DESeq to test for differences between groups (and control for covariates).

Microbiomics

As an example, we will consider a data set I came across where the researchers were interested in bacterial populations in the lungs of COPD patients (thanks to Alexa Pragman, and the Isaacson and Wendt labs).

COPD is chronic obstructive pulmonary disease and is very common medical problem throughout the world (its main cause is smoking).

As frequent infections are a common symptom, the researchers were interested in comparing the lung microbiome of COPD subjects (classified as either moderate or severe) to healthy controls.

They used the resources of the ribosomal database project to generate a table of counts.

Microbiomics

First read in the count data and set up the group indicators.

```
> bactCnt <- read.table("http://www.biostat.umn.edu/ cavanr/bovineCounts.txt")
```

```
> dis <- factor(substring(colnames(bactCnt),1,3))
> dim(bactCnt)
[1] 142 32
```

Then initialize using the DGElist command.

```
> bactDEL<- DGEList(counts=bactCnt, group=dis)
Calculating library sizes from column totals.
```

Microbiomics

Then set up the design matrix as we've seen before, and estimate the dispersions.

```
> design <- model.matrix(~dis)
> bactDEL <- estimateGLMTrendedDisp(bactDEL, design)
```

So now proceed to get the genus level dispersions.

```
> bactDEL <- estimateGLMTagwiseDisp(bactDEL, design)
> fit <- glmFit(bactDEL, design)
> lrt <- glmLRT(fit, coef=2)
```

Microbiomics

Then we can examine the top hits in terms of differences between the moderate COPD patients and the controls.

```
Coefficient:  disMOD
                    logFC    logCPM        LR       PValue         FDR
Brevibacillus   -13.722702 14.810695 15.104481 0.0001017215 0.008070975
Oribacterium      9.701643 12.662380 14.464858 0.0001427989 0.008070975
Selenomonas      13.285029 12.824495 13.872045 0.0001956876 0.008070975
Atopobium        12.660858 11.703525 13.563390 0.0002306402 0.008070975
Prevotella        9.233926 13.455132 13.022587 0.0003077563 0.008070975
Actinomyces       7.734458 17.424870 12.830431 0.0003410271 0.008070975
Campylobacter     9.830081  9.116819 11.820702 0.0005857576 0.011882510
Centipeda        12.895376 12.722930 11.205179 0.0008156937 0.014478564
Bifidobacterium  11.624636 10.634536  9.787951 0.0017565909 0.025533975

Solobacterium     9.633208  9.067843  9.744943 0.0017981673 0.025533975
```

Microbiomics

Since some genera are represented by a single count in a single subject, we may want to consider doing some filtering.

Here is a summary of the total counts for all genera in this data set.

```
> apply(bactCnt,1,sum)
 Saccharomonospora Ornithinimicrobium         Micrococcus    Cryptosporangium
                 1                  1                   1                   1
       Kineococcus      Dolosigranulum     Syntrophomonas      Fastidiosipila
                 1                  1                   1                   1
...
      Enterococcus    Pseudoramibacter         Clostridium        Modestobacter
                 9                  9                   9                  10
   Cryptobacterium          Tannerella        Dysgonomonas        Sphingomonas
                10                 10                  10                  10
...
     Streptococcus     Corynebacterium   Propionibacterium         Actinomyces
             26394               30727               75877               82189
```

Microbiomics

So let's require that there are at least 10 counts total.

```
> bactCntF <- bactCnt[apply(bactCnt,1,sum) >= 10,]
> bactDEL1 <- DGEList(counts=bactCntF, group=dis)
> bactDEL1 <- estimateGLMTrendedDisp(bactDEL1,
+ design)
```

So proceeding as before.

```
bactDEL1 <- estimateGLMTagwiseDisp(bactDEL1, design)
```

Now we'll look at design matrix to determine what the parameters represent.

Microbiomics

```
> design
   (Intercept) disMOD disSVR
 1           1      0      0
 2           1      0      0
 3           1      0      0
...
12           1      1      0
13           1      1      0
14           1      1      0
...
30           1      0      1
31           1      0      1
32           1      0      1
```

so coefficient 2 in the model corresponds to a difference between the moderates and the controls while coefficient 3 tests for a difference between severe and controls.

Microbiomics

So now we fit the GLM and conduct tests for the 2 coefficients.

By specifying the coefficient we can get genera that differ between the controls and the moderates and the controls and the severe patients.

```
> fit1 <- glmFit(bactDEL1, design)
> lrt1.2 <- glmLRT(fit1, coef=2)
> lrt1.3 <- glmLRT(fit1, coef=3)
```

and we can examine the results sorted by *p*-values, first the moderates versus the controls

Microbiomics

```
> topTags(lrt1.2)
Coefficient:  disMOD
                    logFC    logCPM        LR       PValue         FDR
Brevibacillus   -13.721455 14.810756 14.872653 0.0001150184 0.004840491
Oribacterium      9.701546 12.662820 14.420692 0.0001461872 0.004840491
Selenomonas      13.285275 12.825141 13.862729 0.0001966601 0.004840491
Atopobium        12.661042 11.704238 13.548598 0.0002324650 0.004840491
Prevotella        9.233108 13.455509 12.943439 0.0003210447 0.004840491
Actinomyces       7.734822 17.425438 12.870362 0.0003338270 0.004840491
Campylobacter     9.830233  9.117536 11.790320 0.0005953950 0.007399910
Centipeda        12.895653 12.723502 11.201568 0.0008172823 0.008887945
Bifidobacterium  11.624857 10.635275  9.779017 0.0017651462 0.016029713

Solobacterium     9.633091  9.068224  9.700184 0.0018424958 0.016029713
```

Microbiomics

and here are the genera that differ between the severe cases and controls

```
> topTags(lrt1.3)
Coefficient:  disSVR
                 logFC    logCPM        LR       PValue         FDR
Neisseria    15.393096 13.579971 17.929674 2.292192e-05 0.001810056
Oribacterium 11.144734 12.662820 16.796477 4.161048e-05 0.001810056
Selenomonas  13.032916 12.825141 13.203214 2.794694e-04 0.008104614
Centipeda    13.393287 12.723502 11.483554 7.021475e-04 0.015271707
Prevotella    8.419435 13.455509 11.056891 8.835820e-04 0.015374326
Veillonella   9.046095 11.290540 10.626291 1.114911e-03 0.016166215
Rothia        7.853258 15.426206  9.626303 1.918103e-03 0.019610812
Eubacterium  10.450272  9.359486  9.541963 2.008269e-03 0.019610812
Parvimonas   12.576958 10.787986  9.471517 2.086867e-03 0.019610812

Paenibacillus 16.763945 14.931613  9.295272 2.297460e-03 0.019610812
```

Note that *Prevotella* is in both lists: this genus has been found in samples from patients with respiratory infections and is found in the mouths of patients with periodontal disease.

Some of the others (e.g. *Neisseria*) are known as very bad bacterial strains.

Microbiomics

And we can further examine the results by accessing this table, for example we may want to control the FDR while accounting for dependence among the tests rather than just using Benjamini Hochberg (which is what is reported in the table):

```
> padj1=p.adjust(lrt1.2$table$PValue, "BY")
> summary(padj1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02444 0.22060 0.60170 0.61180 1.00000 1.00000
> sum(padj1<.1)
[1] 10
```

Microbiomics

We can use a similar approach to test for differences between the control group and the severe group.

```
> padj2=p.adjust(lrt1.3$table$PValue, "BY")
> summary(padj2)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.009139 0.232100 0.546100 0.604400 1.000000 1.000000
> sum(padj2<.1)
[1] 12
```

Microbiomics

We can then examine the genera we are identifying with this very conservative approach.

```
> rownames(bactCntF)[padj1<.1]
 [1] "Solobacterium"   "Campylobacter"   "Bifidobacterium" "Atopobium"
 [5] "Oribacterium"    "Centipeda"       "Selenomonas"     "Prevotella"
 [9] "Brevibacillus"   "Actinomyces"
> rownames(bactCntF)[padj2<.1]
 [1] "Eubacterium"   "Campylobacter" "Fusobacterium" "Parvimonas"
 [5] "Veillonella"   "Oribacterium"  "Centipeda"     "Selenomonas"
 [9] "Neisseria"     "Prevotella"    "Paenibacillus" "Rothia"
> intersect(rownames(bactCntF)[padj1<.1], rownames(bactCntF)[padj2<.1])
[1] "Campylobacter" "Oribacterium"  "Centipeda"     "Selenomonas"

[5] "Prevotella"
```