CHARGE: CHromosome Assessment in R with Gene Expression data

Benjamin Mayne

October 29, 2019

Contents

1 Introduction

Chromosomal duplications, additions and deletions are important clinically as they may manifest into different diseases and disorders. For example, Trisomy 21, where individuals have three copies of chromosome 21 results in intellectual disability. In addition, in the field of cancer, chromosomal deletions can promote carcinogenesis. Detection of either chromosomal duplications or deletions from gene expression data can be done using clustering methods. CHARGE, can identify these genomic regions and whole chromosomes that have either been duplicated or deleted. Using Hartigan's Dip Test [1] and a bimodal test [2] the likelihood of there being two distinct groups for a given genomic region can be determined. This can be an informative measure to determine if a genomic region has either been duplicated or deleted in given set of samples. This vignette contains a tutorial to identify samples with and without Trisomy 21. The data set is from a publicly available data set (GSE55504) containing 16 fibroblast samples from individuals with (n=8) and without Trisomy 21 (n=8).

2 Preparing the data

CHARGE works primarily with SummarizedExperiment objects [3], a class of data objects containing the experimental, meta and genomic location data all in one. Here, in this example, the experimental data is normalised gene expression data from the RNA-seq data set. The data was mapped to the human reference genome, normalised using edgeR [4] and contains 16 samples. The data can be loaded into the R environment from the CHARGE package as shown below.

```
> library(CHARGE)
> library(GenomicRanges)
> library(SummarizedExperiment)
> library(EnsDb.Hsapiens.v86)
> data(datExprs)
> datExprs

class: RangedSummarizedExperiment
dim: 14375 16
metadata(0):
assays(1): counts
rownames(14375): ENSG00000227232 ENSG00000225972 ... ENSG00000012817
  ENSG00000198692
rowData names(0):
colnames(16): GSM1338325 GSM1338326 ... GSM1338339 GSM1338340
colData names(3): title geo_accession Group
```

3 Expression variation

The first step in using CHARGE is to remove genes with a low expression variation over the region of interest. Genes that have a low variation between samples may not be useful for clustering analysis and can be filtered out of downstream analyses. In this example, genes on chromosome 21 with a low expression variation will be removed from the analysis. The cvExpr function calculates the Coefficient of Variation (CV) of each gene over a defined region. The input is the SummarizedExperiment, datExpr and a GRanges object containing the region of interest. Here, the length of chromosome 21 will be used as a GRanges object.

```
> chr21 <- seqlengths(EnsDb.Hsapiens.v86)["21"]
> chr21Ranges <-  GRanges("21", IRanges(end = chr21, width=chr21))
> cvExpr.out <- cvExpr(se = datExprs, region = chr21Ranges)
```

The CV of the genes can be visualised using the function plotcvExpr. This function uses the output from cvExpr and produces a barplot of the CV for each gene (Figure 1). The user then has the option of removing genes below a specified quantile CV. Once a threshold has been determined clustering analysis can be performed. In this example, genes below the 25% quantile will be removed from the analysis as these are genes with low variation between samples. This will be performed is subsequent functions below in the CHARGE pipeline.

```
> plotcvExpr(cvExpr = cvExpr.out)
```

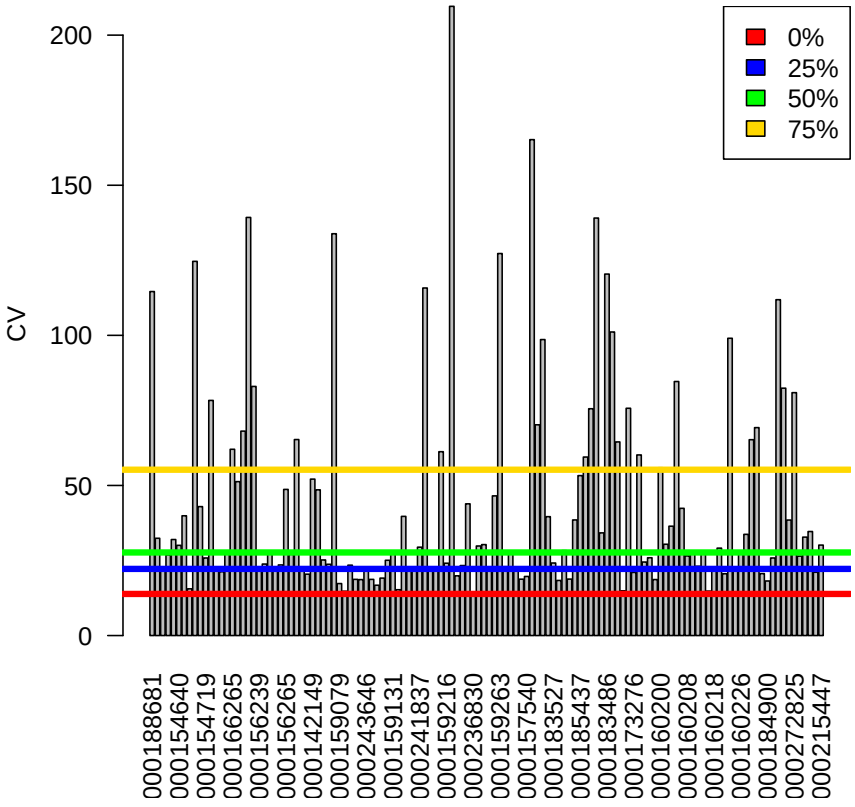

Figure 1: The expression variation of chromosome 21 genes found to be expressed in the data set.

4 Clustering analysis

The clusterExpr function requires the SummarizedExperiment and the output from cvExpr along with the user defined CV threshold. The function performs a clustering analysis using the defined genes and labels each sample as either hyperploidy or hypoploidy with respect to each group. For example, if a sample is labelled hyperploidy it means the average level of gene expression over the genomic region of interest is higher than the samples labelled hypoploidy. If the user is using a set of control samples where the number of chromosomes are known, then the labeling can be used to determine the other samples have chromosomal deletions or duplications.

```
> datExprs <- clusterExpr(se = datExprs, cvExpr = cvExpr.out,
+                         threshold = "25%")
```

The output of clusterExpr is the inputted SummarizedExperiment, but with an extra column in the meta data titled, Ploidy. This column has labelled each sample as either hyperploidy or hypoploidy. Since this data set contains control samples we would presume the Trisomy 21 samples have been labelled hyperploidy and the control samples and hypoploidy. This can be checked as shown below.

```
> data.frame(colData(datExprs))[c("Group", "Ploidy")]
```

```
     Group      Ploidy
1   Normal  Hypoploidy
2   Normal  Hypoploidy
3   Normal  Hypoploidy
4   Normal  Hypoploidy
5   Normal  Hypoploidy
6   Normal  Hypoploidy
7   Normal  Hypoploidy
8   Normal  Hypoploidy
9  Trisomy Hyperploidy
10 Trisomy Hyperploidy
11 Trisomy Hyperploidy
12 Trisomy Hyperploidy
13 Trisomy Hyperploidy
14 Trisomy Hyperploidy
15 Trisomy Hyperploidy
16 Trisomy Hyperploidy
```

As shown the control samples (Normal) have been labelled Hypoploidy with respect to the Trisomy 21 samples. Moreover, the Trisomy 21 samples have been labelled Hyperploidy with respect to the control samples. In addition, compared to the labeling that was provided with the publicly available data, the clusterExpr has correctly labelled the data. However, in other data set

the correct classification may be unknown. It is therefore important to check statically the likelihood of the genomic region being multiplied or deleted. The clustering of the samples can be visualised using a principle component analysis (PCA) where the output from clusterExpr is used as input. Here in this example, there is good separation of the samples, suggesting that there are two distinct groups. However, this can be determined statically using a bimodal test.

```
> pcaExpr(se = datExprs, cvExpr = cvExpr.out, threshold = "25%")
```

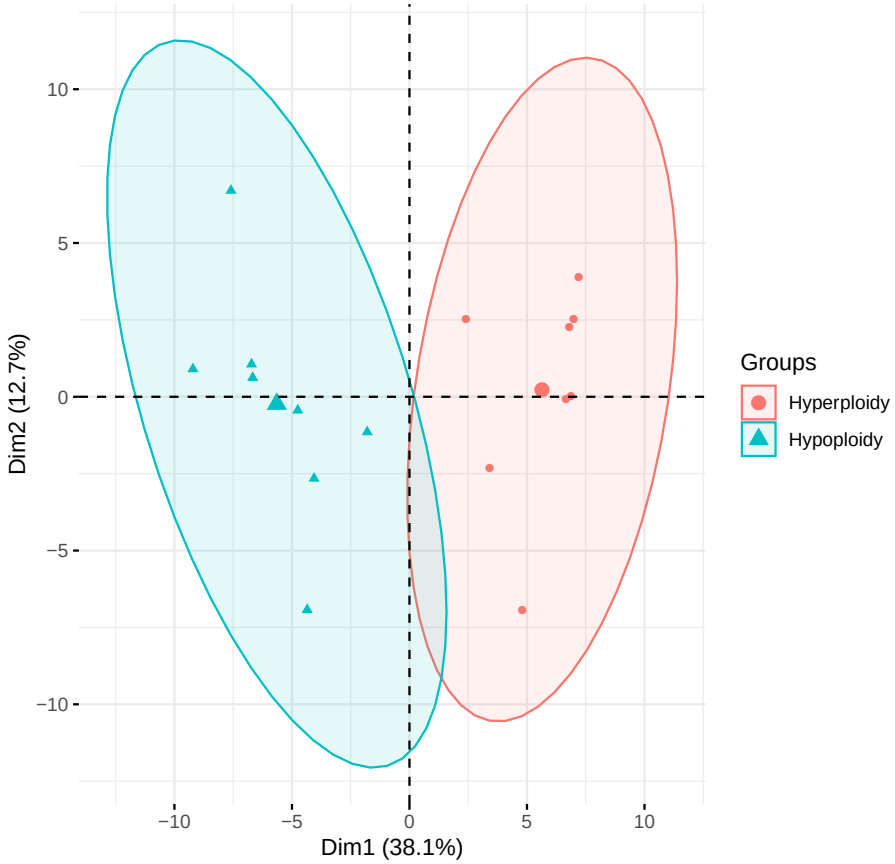

Figure 2: The expression variation of chromosome 21 genes found to be expressed in the data set.

5 Bimodal Test

The CHARGE package contains a wrapper function titled bimodalTest, which utilizes the diptest [1] and modes [2] R packages. This function tests for the likelihood of being two distinct groups using gene expression within the region of interest. The function requires the same inputs as the other functions and return a list of 2.

```
> bimodalTest.out <- bimodalTest(se = datExprs, cvExpr = cvExpr.out,
+                                threshold = "25%")
> bimodalTest.out[[1]]

  Bimodality.Coefficient Bimodality.Ratio Dip.Statistic Dip.pvalue
1              0.7657733        0.9128046     0.1198899 0.03669467
```

The first part of the list is a data frame containing the statistics from the bimodality test, which contains four values. The Bimodality coefficient, ranges from 0 to 1, where a value greater than 5/9 suggest bimodality [2]. The amount of bimodality can be interpreted from the bimodality ratio. In this example, since there is an even split of samples we expect the ratio to be close to 1. The dip statistic and p-value test for unimodality [1] and can be used to determine if the region is statically significant. Here, the p-value is less than 0.05 and therefore we can conclude that there are two distinct groups. The density of the samples can also be plotted as shown below.

```
> plot(bimodalTest.out[[2]])
```

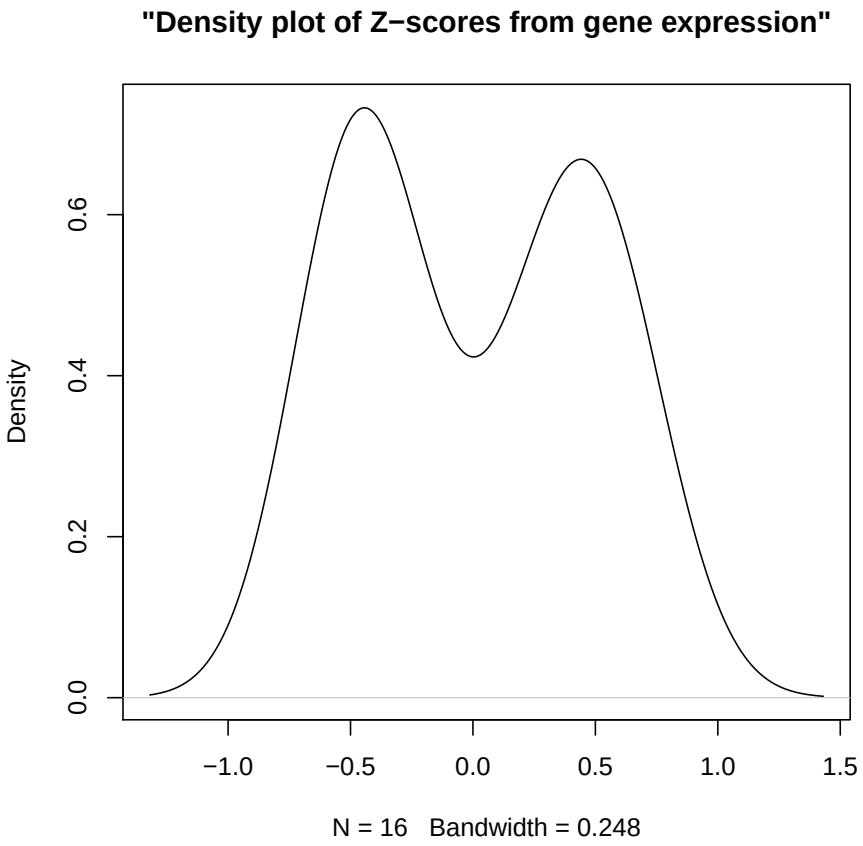

Figure 3: Densiy plot of mean Z scores.

6 Expression Finder

This tutorial has focused on a data set where the region of interest was known. However, in other data sets the region may be unknown. CHARGE contains a function that uses a sliding window approach to scan regions and tests for bimodality. The user can adjust the size of the window, defined by the binWidth and how far the bin will slide along, known as the binStep. This function can be run using multiple threads, which may be desirable when using a small binWidth and binStep. In this tutorial, we'll use a large binWidth to cover the largest chromosome to shorten computing time.

```
> chrLengths <- GRanges(seqinfo(EnsDb.Hsapiens.v86)[c(1:22, "X", "Y")])
> exprFinder.out <- exprFinder(se = datExprs, ranges = chrLengths,
+                              binWidth = 1e+9, binStep = 1e+9, threshold = "25%")
> exprFinder.out[1:3 ,c(1,6:10)]

  seqnames strand Bimodality.Coefficient Bimodality.Ratio Dip.Statistic
1        Y      *              0.7931294        0.7597249    0.14312546
2       21      *              0.7657733        0.9128046    0.11988986
3       14      *              0.7056185               NA    0.08775777
   Dip.pvalue
1 0.003219665
2 0.036694669
3 0.356822561
```

The function returns a data frame, which can be turned into a Granges object using the GRanges function. The function has tested every chromosome and only chromosome Y and 21 have returned as being significant. The Y chromosome was returned as it is most likely detecting sex differences. This function can also be used with a smaller binWidth to identify regions that either been duplicated or deleted.

7 Session Information

This analysis was conducted on:

```
> sessionInfo()

R version 3.6.1 (2019-07-05)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: OS X El Capitan 10.11.6

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib

locale:
```

```
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] parallel  stats4    stats     graphics  grDevices utils     datasets
[8] methods   base

other attached packages:
 [1] EnsDb.Hsapiens.v86_2.99.0   ensembldb_2.10.0
 [3] AnnotationFilter_1.10.0     GenomicFeatures_1.38.0
 [5] AnnotationDbi_1.48.0        SummarizedExperiment_1.16.0
 [7] DelayedArray_0.12.0         BiocParallel_1.20.0
 [9] matrixStats_0.55.0          Biobase_2.46.0
[11] CHARGE_1.6.0                GenomicRanges_1.38.0
[13] GenomeInfoDb_1.22.0         IRanges_2.20.0
[15] S4Vectors_0.24.0            BiocGenerics_0.32.0

loaded via a namespace (and not attached):
 [1] httr_1.4.1               bit64_0.9-7              assertthat_0.2.1
 [4] askpass_1.1              BiocFileCache_1.10.0     blob_1.2.0
 [7] GenomeInfoDbData_1.2.2   Rsamtools_2.2.0          progress_1.2.2
[10] ggrepel_0.8.1            factoextra_1.0.5         pillar_1.4.2
[13] RSQLite_2.1.2            backports_1.1.5          lattice_0.20-38
[16] glue_1.3.1               digest_0.6.22            ggsignif_0.6.0
[19] XVector_0.26.0           colorspace_1.4-1         Matrix_1.2-17
[22] plyr_1.8.4               FactoMineR_1.42          XML_3.98-1.20
[25] pkgconfig_2.0.3          biomaRt_2.42.0           zlibbioc_1.32.0
[28] purrr_0.3.3              scales_1.0.0             tibble_2.1.3
[31] openssl_1.4.1            ggplot2_3.2.1            ggpubr_0.2.3
[34] lazyeval_0.2.2           magrittr_1.5             crayon_1.3.4
[37] memoise_1.1.0            MASS_7.3-51.4            tools_3.6.1
[40] prettyunits_1.0.2        hms_0.5.1                stringr_1.4.0
[43] munsell_0.5.0            cluster_2.1.0            Biostrings_2.54.0
[46] flashClust_1.01-2        compiler_3.6.1           rlang_0.4.1
[49] grid_3.6.1               RCurl_1.95-4.12          rappdirs_0.3.1
[52] leaps_3.0                labeling_0.3             bitops_1.0-6
[55] gtable_0.3.0             DBI_1.0.0                curl_4.2
[58] R6_2.4.0                 GenomicAlignments_1.22.0 modes_0.7.0
[61] dplyr_0.8.3              rtracklayer_1.46.0       bit_1.1-14
[64] zeallot_0.1.0            ProtGenerics_1.18.0      stringi_1.4.3
[67] Rcpp_1.0.2               vctrs_0.2.0              scatterplot3d_0.3-41
[70] dbplyr_1.4.2             tidyselect_0.2.5         diptest_0.75-7
```

8 References

[1] Hartigan JA, Hartigan PM. The Dip Test of Unimodality. The Annals of Statistics. 1985;13(1):70-84.

[2] Deevi S. modes: Find the Modes and Assess the Modality of Complex and Mixture Distributions, Especially with Big Datasets. 2016.

[3] Morgan M, Obenchain V, Hester J and Pag?s H (2017). SummarizedExperiment: SummarizedExperiment container. R package version 1.8.0.

[4] Zhou X, Lindsay H, Robinson MD (2014). Robustly detecting differential expression in RNA sequencing data using observation weights. Nucleic Acids Research, 42(11), e91.