RAPIDR

Kitty Lo

November 20, 2014

Contents

1 Intended use of RAPIDR

RAPIDR was written as part of the RAPID project in the UK for the evaluation of non-invasive prenatal diagnosis (NIPD). We have made our code available in the public domain so that other groups around the world who would like to perform NIPD in their own laboratories can use our analysis pipeline.

2 Create binned counts file from BAMs

RAPIDR fits into the NIPD workflow after the sequencing and alignment of reads, as illustrated in Figure 1. To create BAM files from fastq files outputted from the sequencer, a short-read aligner such as Bowtie or BWA can be used. It is important to only keep alignments with unique mappings. Once you have the BAM files, the first step is to create a binned counts file. This is done using the function `makeBinnedCountsFile` in RAPIDR, which in turn calls functions in the `GenomicRanges` R-package. RAPIDR requires `GenomicRanges` version 1.14.4, and will not work with the newest version as the function `summarizeOverlaps` has moved to another package.

An example of how to create the binned file:

```
> makeBinnedCountsFile(bam.file.list = c("file1.bam", "file2.bam"),
+                       sampleIDs = c("sample1", "sample2"),
+                          binned.counts.fname = output.fname,
+                       k = 20000)
```

- `bam.file.list` is a list of indexed BAM files
- `sampleIDSs` is a list of sample ids in the same order as the BAM file list
- `binned.counts.fname` is file name of the output binned counts file

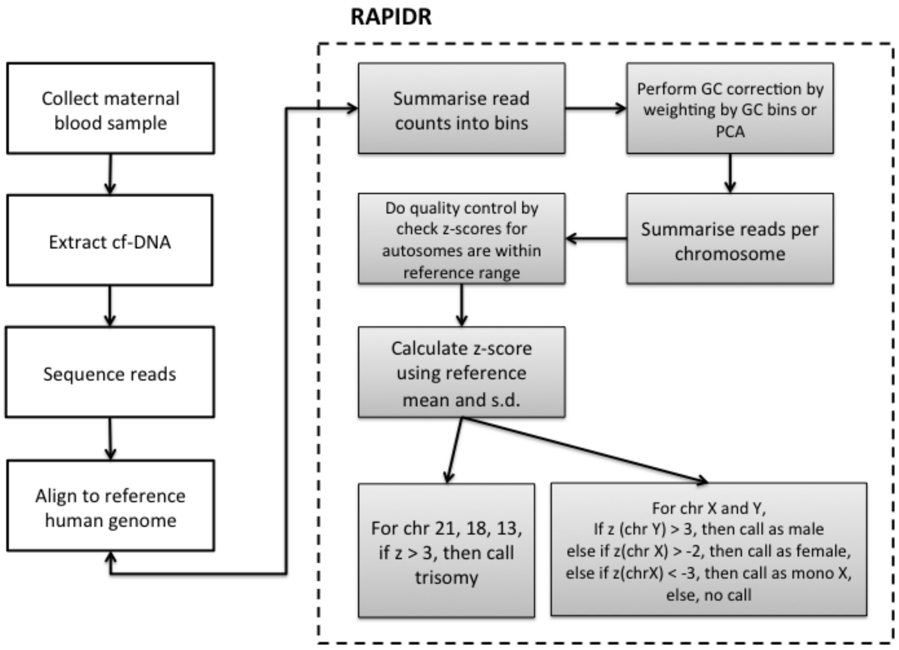

RAPIDR
Collect maternal blood sample
Extract cf-DNA
Sequence reads
Align to reference human genome
Summarise read counts into bins
Perform GC correction by weighting by GC bins or PCA
Summarise reads per chromosome
Do quality control by check z-scores for autosomes are within reference range
Calculate z-score using reference mean and s.d.
For chr 21, 18, 13, if z > 3, then call trisomy
For chr X and Y, If z (chr Y) > 3, then call as male else if z(chr X) > -2, then call as female, else if z(chrX) < -3, then call as mono X, else, no call

Figure 1: NIPD workflow

- `k` is the bin size, default is 20000kb

The resulting binned counts file is a comma delimited file. The header contains the chromosome names (e.g. chr1, chr2, etc.) and the first column are the sample IDs. This is the most computationally intensive part of the pipeline. We found that it takes around 2 minutes to bin each BAM file. The size of the binned counts file will depend on the bin size and the number of samples.

2.1 Masking

The human genome is full of low complexity and repetitive regions which makes aligning short single-end reads difficult. We recommend only keeping reads that can be uniquely mapped. However, even with this condition imposed, we found that there are still bins with abnormally high number of mapped reads. Rapidr offers two solutions to this problem: the first is to identify all the bins with abnormally high read counts and exclude them from the analysis, and the second is to mask out parts of genome known to be of low complexity, e.g. the regions flagged by repeatmasker or regions with low mappability.

We have included a mask file based on repeatmasker in the package. The mask file needs to be in the bed format. The following code snippet produces a binned counts file with masking:

```
> mask <- "mask.bed"
> makeBinnedCountsFile(bam.file.list = c("file1.bam", "file2.bam"),
+                      sampleIDs = c("sample1", "sample2"),
+                         binned.counts.fname = output.fname,
+                      k = 20000, mask)
```

When a mask file is supplied, two output binned counts files are created.

3 Build the reference

Before samples with unknown outcomes can be analysed, we need to build a reference using a set of samples with no known aneuploidy. It is very important that the laboratory preparation method used for this reference set is exactly the same as the method that will be used for the later test samples. Whilst it is not essential for the reference set to contain aneuploidy samples, if aneuploidy samples are provided, RAPIDR will provide estimates of sensitivity and specificity (along with the respective confidence intervals).

3.1 GC correction

It is well known that raw read counts do not follow ideal counting statistics due to GC bias introduced by the PCR process. Various methods have been proposed to normalise the counts. RAPIDR implements two methods from the literature: the Normalised Chromosome Value (NCV) method and weighting of counts by GC content bins. NCV is the simpler of the two methods and entails taking the ratio between the counts mapped to the chromosome of interest and some other chromosome. The chromosome used in the denominator is chosen such that the variance of the NCV ratios is minimised. In the GC bin weighting method, we first determine the read count for bins of some size (the default in RAPIDR is 20kb), and then average the counts in each 0.5% of GC content bin. Counts in each GC content bin is weighted by $W_i = \bar{M}/M_i$, where $\bar{M}$ is the average counts across all GC content bins, and M_i is the average count in bin i.

We also implemented a third method for GC correction based on principal component analysis (PCA), which as far we know, has not been used in the context of NIPD. In the PCA method, we first use the normals in the reference set to find the principal components by treating the counts ratios in each bin as a variable. Then we regress the count ratios in each bin with the first 10 principal components, assuming that they represent systematic noise. There is also the option to set the number of principal components to exclude by setting the parameter `numPC`.

Here is an example of how to create a reference set:

```
> library(RAPIDR)
> rapidr.dir <- system.file(package = "RAPIDR")
> data(outcomes)
> data(gcContent)
> # Make some binned data
> T21.pos <- which(outcomes$Dx == "T21")
> chr.lens <- sapply(gcContent, length)
> chr.names <- names(chr.lens)
> # Make the header
> header <- c("SampleID")
> for (i in 1:length(chr.lens)) {
+    header <- c(header, rep(chr.names[i], chr.lens[i]))
+ }
> nbins <- sum(chr.lens)
> ncols <- nbins + 1
> binned.counts <- matrix(nrow = nrow(outcomes), ncol = ncols)
> for (i in 1:nrow(binned.counts)) {
+    binned.counts[i,] <- rpois(ncols, lambda = 100)
+    if (i %in% T21.pos) {
+       binned.counts[i, 139087:141493] <- rpois(chr.lens[21], lambda = 115)
+    }
+ }
> binned.counts[,1] <- outcomes$SampleID
> colnames(binned.counts) <- header
> t <- tempfile()
> write.table(binned.counts, file = t, col.names = TRUE, row.names = FALSE, quote = FALSE, sep = ",")
> counts.fname <- t
> message(t)
> gcContent.fname <- paste(rapidr.dir, "/data/gcContent.RData", sep = "")
> head(outcomes)

  SampleID     Dx Gender
1     1000 Normal Female
2     1001 Normal Female
3     1002    T21 Female
```

```
4      1003 Normal   Male
5      1004 Normal   Male
6      1005 Normal   Male

> ref.set <- createReferenceSetFromCounts(counts.fname,
+                               outcomes,
+                                      gcCorrect = FALSE,
+                                      PCA = FALSE,
+                                   filterBin = FALSE,
+                                      gcContentFile = gcContent.fname)

 [1] 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009
Levels: 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009
```

- `counts.fname` is the file name of the binned counts file created using `makeBinnedCountsFile` function described above
- `outcomes` is a data frame containing sample IDs, Gender and Diagnosis (Dx) of each sample in the reference set. It is important that the column names of the data.frame is exactly as described
- `gcCorrect` whether to do gc correction, default is FALSE
- `PCA` whether to do PCA correction, default is FALSE

In order to perform GC correction, we need information about the GC content along the genome in the choice of bin size. The gcContent file is an R data object that needs to be supplied to the function. We have included one such GC content object, binned to the default size of 20k bp. If you wish use a bin size that is not the default, you will need to create your own gcContent RData object using the function `makeGCContentData`.

GC bin weighting and PCA correction can be applied by setting options in the `createReferenceSetFromCounts` function. They can both be set, in which case, the bin weighting is performed first and subsequently PCA is applied.

The NCV method is accessed by setting the method option to NCV. When the NCV method is used, RAPIDR uses the ratio of counts between the chromosome of interest and one other chromosome instead of the ratio of counts to all autosomes. There are two other method options: zscore (default) and median absolute deviation (MAD). The MAD method calculates a MAD score instead of z-score and is more robust to outliers.

3.2 The rapidr.ref object

The output to the `createReferenceFromCounts` is a rapidr.ref object.

```
> print(ref.set)

Do PCA:  FALSE
Do gcCorrect:  FALSE
Method used:  zscore
Baselines:
T21: Mean  0.017  Std. dev.  2.5e-05
T18: Mean  0.027  Std. dev.  2.3e-05
T13: Mean  0.04  Std. dev.  4.7e-05
            T21 T13 T18 Turner Female Male
tp            1   0   0    0.0      5    0
tn            9  10  10    9.0      0    5
fp            0   0   0    1.0      4    0
fn            0   0   0    0.0      0    4
Sensitivity   1 NaN NaN    NaN      1    0
Specificity   1   1   1    0.9      0    1
```

There are five slots in the rapidr.ref object:

- `baselines` contains the mean and standard deviation of the counts ratios in the reference set

- `baseline.perf` contains the performance (sensitivity and specificity for T21, T18, T13 and X0) results of the reference set
- `do.PCA` logical value to indicate whether PCA correction is applied
- `do.gcCorrect` logical value to indicate whether GC correction is applied
- `excl.bins` list of bins that has been excluded from further analysis. Reasons for exclusion can be because an excess number of reads were mapped to these bins in chrY from females, or because the variance of the bin counts were higher than expected

4 Test unknown sample

Once a rapidr.ref object is made, it can be used to call aneuploidy for unknown test samples. We assume again that BAM files have been made using an alignment software. To create the binned counts file from BAM files, the `makeBinnedCountsFile` function can be used. Once the binned counts file has been made for the test sample, it can be tested using:

```
> test.results <- testUnknowns(ref.set,
+                              counts.fname,
+                              gcContentFile = gcContent.fname)
> print(test.results)
```

```
Total test samples:  10
Trisomy calls:
T21:        1
T18:        0
T13:        0
Sex chromosome calls:
Females:        9
Males:          0
Turners:        1
No calls:        0
```

ref.set is a rapidr.ref object that contains the baseline reference made with the `createReferenceFromCounts` function. The corrections that were made to create the baseline is stored in the rapidr.ref object, and the same corrections will be applied to the test sample counts. The output of the `testUnknowns` function is a `rapidr.test` object and the slots in it are:

- `results` data.frame of the calls and the related z-scores for each unknown sample
- `qc` data.frame of the qc information, see Section 5
- `baselines` mean and standard deviations of the samples in the reference set

4.1 Trisomy calls

Depending on the method selected, RAPIDR either calculates a z-score, an NCV score or a MAD score. For all three methods, trisomy is called when the absolute value of the score is greater than 3.

4.2 Sex chromosome calls

To determine fetal sex and to call Turners syndrome (currently the only sex chromosome aneuploidy supported by RAPIDR), we used a slightly more complex rule than the one used to call trisomies. The reference mean and standard deviation of the counts ratios of chromosomes X and Y are constructed using only females with normal outcomes. Given an unknown sample, the z-score is calculated for chr X and chr Y (z_X and z_Y) and the following rules are applied sequentially - if $z_Y > 3$, the sample is called male; if $z_X > -2$, the sample is called female; if $z_X < -3$, the sample is called as monosomy X; finally, if z_X is between -2 and -3, then the sample sex is called as unknown. To improve the accuracy of fetal sex calling, we also excluded bins in chr Y with a high number of counts mapped from females.

4.3 Fetal fraction

For euploid males, RAPIDR also provides estimates of the percentage of fetal DNA in the cfDNA, known as the fetal fraction. The fetal fraction is in the column chrY_FF of the `results` slot in the `rapidr.test` object and is calculated using the equation:

$$FF = 2 \times \left| \frac{z_{ij} \times \sigma_i}{\mu_i} \right| \tag{1}$$

Fetal fraction is an important determinant of NIPD sensitivity. However, the simple z-score test does not account for the fetal fraction. In the clinical setting, it may be advisable to set a lower limit for fetal fraction below which the NIPD test would return an inconclusive result.

5 Quality controls

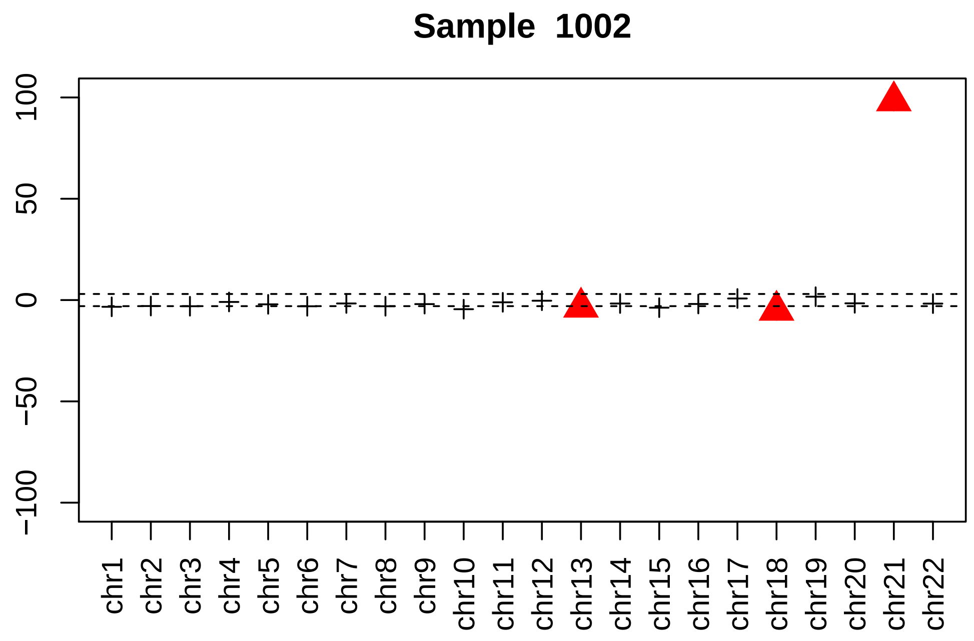
Quality control is an important part of NIPD when the test is used in the clinical setting. RAPIDR provides quality control for two issues that may impact the test results and the corresponding column name is in brackets:

- Total number of reads (QC.reads)
- Whether it is a good match to the reference set (QC.refmatch)

There are two columns in the `qc` slot of the `rapidr.test` object that provides quality control information. QC.reads is set to FALSE if there are less than 2 million total reads. To test whether each test sample is a good match to the reference set, RAPIDR calculates the z-scores for all autosomes, excluding chr 21, chr 18 and chr 13 (which may contain aneuploidy). If any of the z-scores is above 3 or below -3, QC.refmatch is set to FALSE.

The z-scores for all autosomes can be visualised using:

```
> plotTestSample(test.results, "1002")
```



6 Technical information about R session

```
> sessionInfo()

R version 3.1.1 (2014-07-10)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.iso885915       LC_NUMERIC=C
 [3] LC_TIME=en_US.iso885915        LC_COLLATE=C
 [5] LC_MONETARY=en_US.iso885915    LC_MESSAGES=en_US.iso885915
 [7] LC_PAPER=en_US.iso885915       LC_NAME=C
 [9] LC_ADDRESS=C                   LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.iso885915 LC_IDENTIFICATION=C

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

other attached packages:
[1] RAPIDR_0.1.1

loaded via a namespace (and not attached):
 [1] BBmisc_1.8              BatchJobs_1.5           BiocGenerics_0.12.0
 [4] BiocParallel_1.0.0      Biostrings_2.34.0       DBI_0.3.1
 [7] GenomeInfoDb_1.2.0      GenomicAlignments_1.2.0 GenomicRanges_1.18.1
[10] IRanges_2.0.0           PropCIs_0.2-5           RSQLite_1.0.0
[13] Rcpp_0.11.3             Rsamtools_1.18.1        S4Vectors_0.4.0
[16] XVector_0.6.0           base64enc_0.1-2         bitops_1.0-6
[19] brew_1.0-6              checkmate_1.5.0         chron_2.3-45
[22] codetools_0.2-9         data.table_1.9.4        digest_0.6.4
[25] fail_1.2                foreach_1.4.2           iterators_1.0.7
[28] parallel_3.1.1          plyr_1.8.1              reshape2_1.4
[31] sendmailR_1.2-1         stats4_3.1.1            stringr_0.6.2
[34] tools_3.1.1             zlibbioc_1.12.0
```