SynthEx

Mengjie Chen

July 31, 2016

SynthEx version: 1.1

Please cite:
Grace O. Silva, Xiaping He, Lisle E. Mose, Joel S. Parker, Wei Sun, Charles M. Perou and Mengjie Chen
SynthEx: Capture-specific, “synthetic”-normal based whole exome/genome sequencing tool
for copy number alteration detection and tumor heterogeneity profiling.

Contents

1 Introduction of package *SynthEx* **3**

2 Installation **3**

3 Pre-processing and preparation **4**

3.1 Example Data 4

4 Quick Start **5**

4.1 Single subject analysis 5

4.2 Multiple subjects analysis 6

5 Whole exome or targeted sequencing analysis **6**

5.1 Single subject analysis 6

5.1.1 Input 6

5.1.2 Correct bias in targeted regions 7

5.1.3 Normalization 8

5.1.4 Segmentation 10

5.1.5 Purity estimation 10

5.2 Single sample analysis pipeline 12

5.3 Without variant allele frequency 12

6 Multiple sample analysis **12**

6.1 Batch effect and synthetic normal approach 12

6.2 Synthetic normal approaches 14

6.2.1 Multiple sample analysis pipeline 16

6.2.2 (Optional) Creating a synthetic normal library 17

7 Whole genome sequencing analysis **17**

1 Introduction of package *SynthEx*

SynthEx is a comprehensive suite of tools for CNA detection and tumor heterogeneity profiling. It is tailored to cater for the multiple characteristics of different next generation sequencing technologies. *SynthEx* is the first method aware of the sample-specific bias in targeted regions due to fold enrichment differences in tumor and normal samples. Fundamentally different from current methods, it applies different analytical strategies for single subject and large-scale cancer genomics studies. For large-scale cancer genomics studies, it corrects the technical bias due to capture efficiency by using a “synthetic”-normal strategy. The “synthetic”-normal samples were generated by stratifying existing normal samples (including samples outside the current study) based on capture protocols. Therefore, instead of requiring a matched pair normal sample from each tumor subject, a “synthetic” normal is used that mimics the targeted sequencing profile with the same capture efficiency level as the tumor sample. This strategy can potentially reduce both the experiment and analysis burden within a sequencing facility. In addition, since “synthetic” normals can be used for any tumor samples processed by the same protocol, they can be used to analyze mouse xenograft samples or cell line samples, paired normals of which are traditionally difficult and even impossible to obtain.

SynthEx can be used to analyze whole exome sequencing, targeted sequencing and whole genome sequencing data.

2 Installation

SynthEx relies on the following R packages: *mclust*, *flsa*, *foreach*, *DNAcopy*, *ggplot2*, *gridExtra*, *inline*, and *Rcpp*.

```r
install.packages("mclust")
install.packages("flsa")
install.packages("foreach")
install.packages("ggplot2")
install.packages("gridExtra")
install.packages("inline")
install.packages("Rcpp")
source("https://bioconductor.org/biocLite.R")
biocLite()
biocLite("DNAcopy")
```

SynthEx can be installed directly from github.

```r
install.packages("devtools")
library(devtools)
install_github("ChenMengjie/SynthEx")
```

3 Pre-processing and preparation

SynthEx takes counts data in non-overlapping genomic bins from tumor and normal samples as input. It has the option to input variant calling results from tumor sample. Thus *SynthEx* relies on other tools to generate appropriate input files. We recommend *BEDtools* for getting the bin-level count data and *freebayes* for variant calling. In addition, *SynthEx* calls the `intersectBed` function from *BEDtools* directly, so *BEDtools* should be installed. Linux or Mac users can obtain the directory of `intersectBed` by running:

```
intersectBed.dir <- system("which intersectBed", intern = TRUE)
```

If it doesn't work on your system, you need to specifiy `intersectBed.dir` manually.

When evaluating sample bias at targeted regions, it requires annotations for targets. The annotations of targeted regions for SureSelect are pre-installed *SynthEx* and will be used as the default if annotations are not specified by users. All bins will be classified into two categories: target bins, if overlapped with any selectively amplified targets, and off-target bins, otherwise. Such classification is pre-calculated for hg19 genome with 10kb, 25kb, 50kb and 100kb. If using bin size not listed here or other annotation, users need to provide such a file or generate such as file by running `createTargetBins()`.

```
library(SynthEx)
data("TargetAnnotations")
head(TargetAnnotations$bin100000)
head(TargetAnnotations$Target)
targetAnnotateBins <- createTargetBins(TargetAnnotations$Target[1:5,
    ], bin.size = 1000000)
# Input first five line to demonstrate, use
# createTargetBins(TargetAnnotations$Target, bin.size =
# 10000)
```

Each row represents a target. First three columns represent chromosome, start and end of a target and last three columns represent chromosome, start and end of its corresponding bin. Users can choose to remove centromere regions when segmenting the genome. Similarly such information is pre-calculated for hg19 genome with 10kb, 25kb, 50kb and 100kb. If using bin size not listed here or other annotation, users need to provide such a file or generate such as file by running `createCentromereBins()`.

```
centromereBins <- createCentromereBins(bin.size = 10000)
```

3.1 Example Data

`TCGAbreast` contains the count data from tumor and its paired normal of TCGA breast cancer samples with bin size of 100kb.

```
library(SynthEx)
data(TCGAbreast)
Lookup(TCGAbreast, show = 6)

## [1] "No.1 element is tumor:"
##    chr  start    end TCGA-A1-A0SM TCGA-A2-A04P TCGA-A2-A04Q
## 1 chr1      1 100000          369         3502          189
## 2 chr1 100001 200000          153          274           24
## 3 chr1 200001 300000           79           58           63
## 4 chr1 300001 400000            4            4            6
## 5 chr1 400001 500000            4            0            0
## 6 chr1 500001 600000          389          138          111
## [1] "No.2 element is normal:"
##    chr  start    end TCGA-A1-A0SM TCGA-A2-A04P TCGA-A2-A04Q
## 1 chr1      1 100000          568         3135          322
## 2 chr1 100001 200000           22          481           40
## 3 chr1 200001 300000           14          224           92
## 4 chr1 300001 400000            0            6            6
## 5 chr1 400001 500000            2            0            2
## 6 chr1 500001 600000          101          411          152
## [1] "No.3 element is samples:"
## [1] "TCGA-A1-A0SM" "TCGA-A2-A04P" "TCGA-A2-A04Q"
## [4] "TCGA-A2-A04T" "TCGA-A2-A04X" "TCGA-A2-A0CM"
```

`Lookup()` is a helper function to take a peek at all elements from a list.

4 Quick Start

First specify the directory to save intermediate and final results. If not specified, the default is the current working directory.

```
working.dir <- ".../working/directory/"
result.dir <- ".../directory/to/save/results"
```

4.1 Single subject analysis

The input of single subject analysis is names of two BED files or two matrices in BED format that store the count data for tumor sample and its matched normal. Examples can be found in folder `ExampleFiles`.

```
tumor.file <- "./ExampleFiles/TCGA-A7-A26G.tumor.bed"
normal.file <- "./ExampleFiles/TCGA-A7-A26G.normal.bed"
sample.name <- "TCGA-A7-A26G"
```

`SynthExPipeline()` is the major funtion that implements the entire *SynthEx* pipeline. Details about the steps in the pipeline can be found in following sections.

To run segmentation without genotype information, set `genotype.file=NULL`.

```
Segfrompipe <- SynthExPipeline(tumor.file, normal.file, bin.size = 100000,
    intersectBed.dir, genotype.file = NULL, result.dir, working.dir,
    prefix = sample.name, verbose = TRUE)
```

With genotype information, *SynthEx* can further estimate purity.

```
genotype.file <- "./ExampleFiles/TCGA-A7-A26G-tumor-freebayes.vcf"
Segfrompipe <- SynthExPipeline(tumor.file, normal.file, bin.size = 100000,
    intersectBed.dir, genotype.file, result.dir, working.dir,
    prefix = sample.name, verbose = TRUE)
```

The CNA calling results are saved in BED files as `'prefix'_Copynumber.bed` and `'prefix'_Event.bed`.

4.2 Multiple subjects analysis

SynthEx implements a synthetic normal approach, which untilizes normal samples sequenced by the same protocol to reduce unwanted variation due to the sequencing process from tumor samples. To use this approach, *SynthEx* requires the input of the count data of a tumor sample and multiple normal samples.

```
tumor.file <- "./ExampleFiles/TCGA-A7-A26G.tumor.bed"
multiple.normal.file <- "./ExampleFiles/multiple.normal.samples.bed"
Segfrompipe <- SynthExPipeline(tumor, multiple.normal.file, bin.size = 100000,
    intersectBed.dir, genotype.file, result.dir, working.dir,
    verbose = TRUE)
```

5 Whole exome or targeted sequencing analysis

5.1 Single subject analysis

5.1.1 Input

The input of single subject analysis is the names of two BED files or two matrices in BED format.

```
data(TCGAbreast)
tumor <- TCGAbreast$tumor[, c(1:3, 11)]
normal <- TCGAbreast$normal[, c(1:3, 11)]
sample.name <- TCGAbreast$samples[11]
head(tumor)

##    chr  start    end TCGA-A2-A0D1
## 1 chr1      1 100000         2534
## 2 chr1 100001 200000           57
## 3 chr1 200001 300000           48
## 4 chr1 300001 400000            6
## 5 chr1 400001 500000            0
## 6 chr1 500001 600000           40

head(normal)

##    chr  start    end TCGA-A2-A0D1
## 1 chr1      1 100000         1555
## 2 chr1 100001 200000          188
```

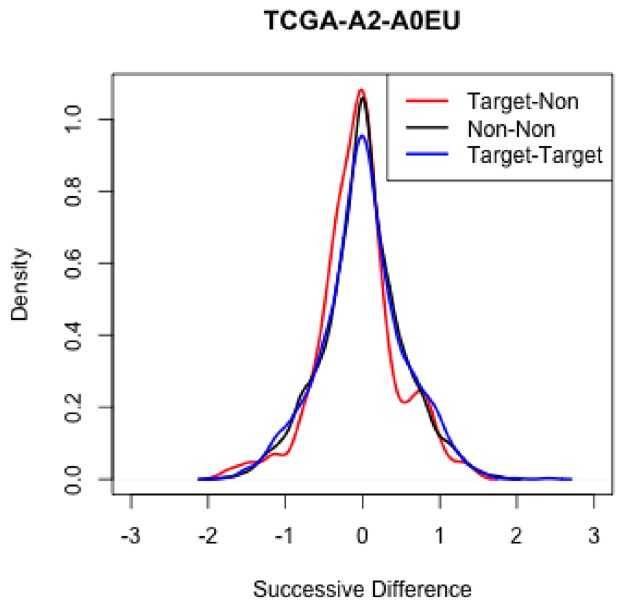

Figure 1: An example density plot of RD for three different categories.

```
## 3 chr1 200001 300000          200
## 4 chr1 300001 400000           10
## 5 chr1 400001 500000            2
## 6 chr1 500001 600000           62
```

5.1.2 Correct bias in targeted regions

Given a pair of tumor and matched normal, `SynthEx` will analyze the sample-specific bias in targeted regions due to fold enrichment differences in tumor and normal samples. `SynthEx` classifies adjacent bins into either: two adjacent off-target bins, two adjacent targeted bins or an off-target bin next to a targeted bin (Figure 1a). For every pair of adjacent bins we calculate a ratio distance (RD), the difference of read ratio between target and off-target bins, and obtained three main distribution categories. When the ratios in the target and off-target bins follow the same distribution, the density of the RD from a target bin and adjacent off-target bin centralizes at 0 and has the same shape as that from two adjacent target bins or two adjacent off-target bins.

```
ratioCorrectedBias <- SynthExcorrectBias(tumor, normal, bin.size = 100000,
    rm.centromere = TRUE, targetAnnotateBins = NULL, saveplot = FALSE,
    centromereBins = NULL, chrX = FALSE, plot = TRUE, result.dir,
    prefix = sample.name, reads.threshold = 25)
```

After running `SynthExcorrectBias()`, a density plot for the distances from three categories of adjacent bins will be generated. `SynthEx` directly uses the mean of RD as an estimator of the bias and de-bias by subtracting it from the observed ratios in target bins. Naturally, the mean and variance of RD can be used as diagnostic metrics to identify ill-behaved samples. As we shown in the paper, the bias

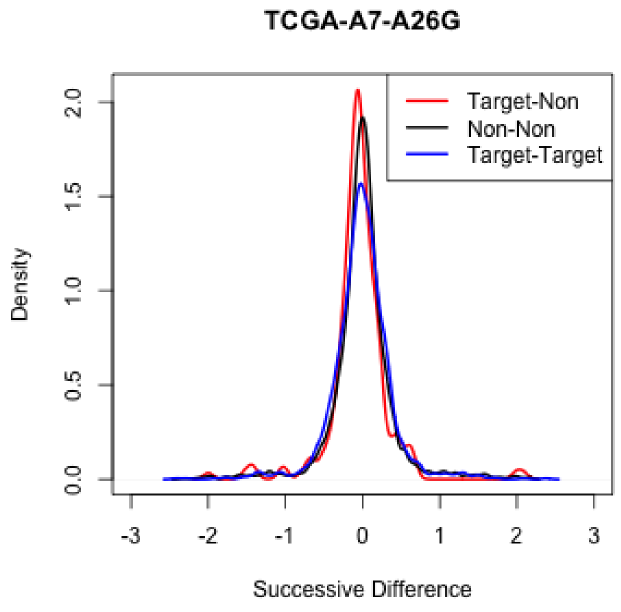

Figure 2: Another example density plot of RD for three different categories.

in targted regions is due to the differences in fold enrichment of tumor and paried normal sample. The following is an example demonstrating the existence of such bias. Correcting the bias in targeted regions is neccessary when we leverage the information in both target and non-target regions. The following shows an example with such bias in target regions.

```
tumor <- TCGAbreast$tumor[, c(1:3, 22)]
normal <- TCGAbreast$normal[, c(1:3, 22)]
sample.name <- colnames(TCGAbreast$tumor)[22]
ratioCorrectedBias <- SynthExcorrectBias(tumor, normal, bin.size = 100000,
    rm.centromere = TRUE, targetAnnotateBins = NULL, saveplot = FALSE,
    centromereBins = NULL, chrX = FALSE, plot = TRUE, result.dir,
    prefix = sample.name, reads.threshold = 25)
```

5.1.3 Normalization

After correcting bias in targeted regions, `SynthEx` will identify the baseline for copy neutral events. Most existing methods use median normalization to identify copy neutral regions. The underlying assumption for this strategy is that most regions in the genome are diploid, which does not hold for samples with aneuploidy, such as whole genome doubling. We identify the diploid genome within the tumor sample by gauging information from allele frequencies of heterozygous sites. Specifically, we calculate the median minor allele frequency (MAF) for each bin and select all bins with median MAF greater than 0.45 as candidate copy neutral events (theoretically diploid regions have mean MAF of 0.5). The possible copy number for these bins are 2, 4 and other multiples of 2. To identify diploid bins, we fit a gaussian mixture model for read ratio (RR) and use Bayesian Information Criteria (BIC) to determine the number of copy number state. Finally we assign the mixture component with the smallest mean RR as diploid. The RR

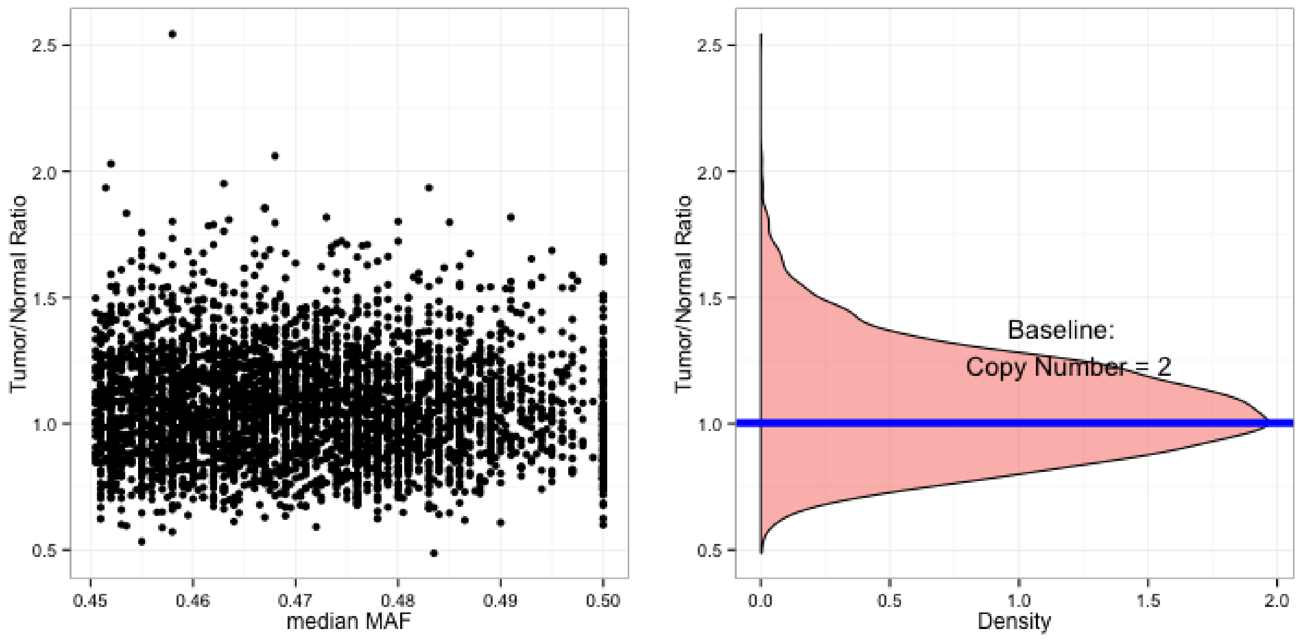

Figure 3: Estimating baseline for copy neutral events using genotype information.

in all bins will be adjusted so that diploid regions have expected RR equal to 1.

We use `freebayes` to call the variants from tumor samples. A python script within *`SynthEx`* can scan all the variants in a VCF file output by `freebayes` and select SNPs with MAF greater than 0.05 and supporting reads greater than a threshold (specified by option `cutoff`, default is 20). An example VCF file can be found in folder `ExampleFiles`. Due to the space limit, we only show the first 200,000 rows of the `freebayes` output.

```
genotype.file <- "TCGA-A7-A26G-tumor-freebayes.vcf"
```

```
ratioNormalized <- normalization(ratioCorrectedBias, intersectBed.dir,
    genotype.file, vcf = TRUE, working.dir, result.dir, cutoff = 20,
    plot = TRUE, saveplot = FALSE, prefix = sample.name)
```

VCF files generated by other tools can also been processed by the provided python script if they report genotype, read depth and depth of one allele for each variant through ”GT”, ”DP” and ”RO”, respectively. As an alternative, the author can input non-vcf file with preprocessed SNPs when setting `vcf = FALSE`. This require the input in BED format. Example BED files can be found in folder `ExampleFiles` (as `example.genotype.bed`).

`normalization()` will create a density plot for bins with median MAF greater than 0.45 and indicate the estimated baseline of diploid genome. Normalization step is optional. If genotype data is not provided, median normalization will be used. We strongly recommend to include normalization step if the purpose is to estimate ploidy or call integer copy number.

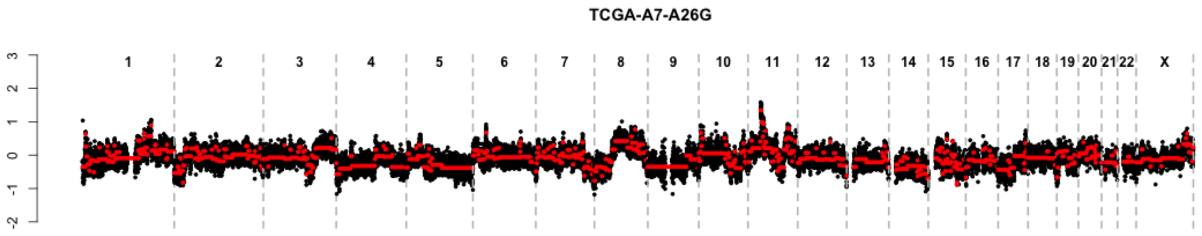

Figure 4: An example for segmentation

5.1.4 Segmentation

The default segmentation method in `SynthEx` is Circular Binary Segmentation (CBS) with default parameters. In this method, CNA signals are modeled as normal random variables with shifts in mean. However, this assumption may not hold due to sequencing biases. A nonparametric model without assuming normality may be more precise. Thus we implement an alternative trending filtering procedure. SynthEx also implements a third segmentation method, SomatiCA, which adds denoising and model selection procedure to CBS. We recommend users to test all three methods on their datasets. Based on our own experience, trending filtering works best on WGS data and SomatiCA works best on WXS data.

```
Seg <- createSegments(ratioNormalized, "CBS")
## createSegments(ratioNormalized, 'TrendFiltering')
## createSegments(ratioNormalized, 'SomaticaEx', smoothk = 10)
Segments <- singleCNreport(Seg, report = FALSE, result.dir, saveplot = TRUE,
    prefix = sample.name, plotNormalized = TRUE, WGD = 1.35,
    pos.prop.threhold = 0.6, pos.log2ratio.threhold = 0.75)
```

`singleCNreport()` can generate a visulization of the segmentation result. When `report = TRUE`, `singleCNreport()` will write the result into a file named `paste0(prefix, "-Segment-", segmentMethod, "-", bin.size, ".txt")`. Users can apply option `saveplot = TRUE` to save the plot into a jpg file instead of shown on the screen. In addition, `SynthEx` will report whole genome doubling event. The criteria for a whole genome doubling call is either the ploidy greater than `WGD` $\times$ 2 or the proportion of positive segments greater than `pos.prop.threhold`) and the mean log2 ratio greater than `pos.log2ratio.threhold`.

5.1.5 Purity estimation

Accurate estimation of purity is a prerequisite for the quantization of integer level copy number and the clonality analysis. We implement an approach with the following core idea: the proportion of intermixed normal cells can be estimated from the shift of copy ratios of clonal CNAs from their expectations in the pure homogeneous tumor sample. In this approach, we first used spectral clustering (option `tau` specifies the free parameter in Gaussian kernel) to cluster all the segment into k group $(k = 1, .., 7)$. Then we used BIC to select the best model with the data likelihood modeled by a mixture of normal distributions (the standard deviation of which is specified by `sigma`). Since purity es-

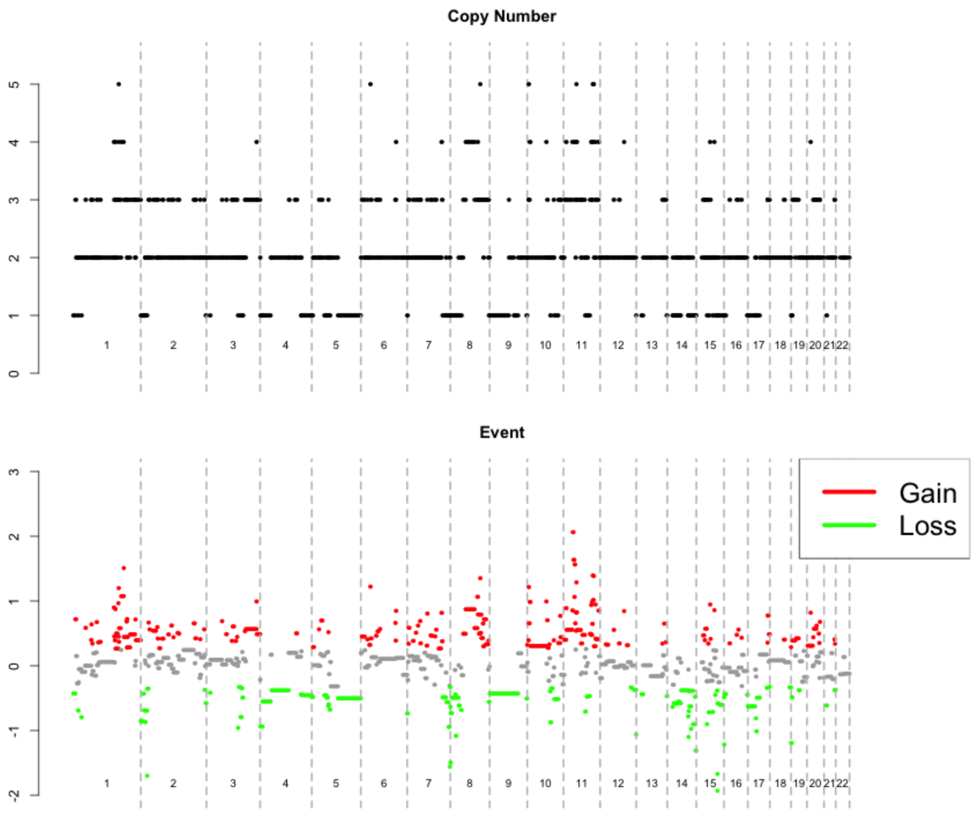

Figure 5: Correcting for sample purity and making loss/gain calls for segments.

timation is known to be confounded with ploidy and clonality, which makes it suffer from identifiability problem. Thus we introduced several heuristic based filters to assist the assignment of integer copy numbers. These filters including thresholds on minimum number of segments in each integer copy level (option `group.length.threshold`), minimum distance of MAF between two integer levels (option `maf.control`), minimum distance of ratio between two integer levels (option `delta`) and minimum length of the segments used in the estimation (as a proportion of total genome length, option `prop.threshold`). Purity is estimation based the resulted integer copy assignment. Genotype data is required for purity estimation. Option `gain.threshold` and `loss.threshold` specify the cut-off for copy number gain and copy number loss events, respectively. For TCGA breast cancer and head neck cancer WXS data, we found for gain and for loss leading to highly consistent results with SNP array data. Segmentation results after adjusting for purity can be visualized by function `chromosomeView()`.

```
PurityCorrected <- purityEstimate(Segments, working.dir, result.dir,
    intersectBed.dir, prefix = sample.name, report = TRUE, prop.threshold = 0.0005,
    delta = 0.1, maf.control = 0.01, tau = 2, sigma = 0.1, len.threshold.K = 10,
    group.length.threshold = 2, gain.threshold = log2(1.2), loss.threshold = log2(0.8
chromosomeView(PurityCorrected, saveplot = FALSE)
```

`SynthEx` reports copy number gain/loss event as well as integer copy number for each segment. The copy number gain/loss calls are estimated from unnormalized data, i.e., the gain/loss event is respect to the median of genome. Thus gain/loss events can be compared among different samples regardless of the ploidy of each sample. Integer copy number calls are estimated from normalized data where the baseline is diploid. For example, in a sample with whole genome doubling, a segment with copy number 2 may correspond to a copy loss event.

5.2 Single sample analysis pipeline

The above steps are integrated in function `SynthExPipeline()`. When working on multiple samples, we suggest first to run each step and choose desired model parameters and segmentation methods, and then take advantage of `SynthExPipeline()` to perform repetitiva analysis. The default `SynthExPipeline()` will report all results under `result.dir`.

```
Segfrompipe <- SynthExPipeline(tumor, normal, bin.size = 100000,
    intersectBed.dir, genotype.file, result.dir, working.dir,
    verbose = TRUE)
```

5.3 Without variant allele frequency

Without variant allele frequency, *SynthEx* will only perform bias correction and segmentation.

```
Segfrompipe <- SynthExPipeline(tumor, normal, bin.size = 100000,
    intersectBed.dir, genotype.file = NULL, result.dir, working.dir,
    verbose = TRUE)

## [1] "Bias correction finished."
## [1] "Segmentation finished."
## [1] "Report can be found at: ../results ."
```

6 Multiple sample analysis

6.1 Batch effect and synthetic normal approach

As shown in *SynthEx* paper, we observed strong “batch effect” in multiple sample genomic studies. Here we demonstrate the batch effect using TCGA breast cancer dataset in *SynthEx*. We selected top 1000 bins with largest vairance across 97 samples. Figure shows the heat map of selected bins. Each row represents a bin with mean centered at 0. Each colomn represents a sample. The corresponding dendrogram was obtained from hierarchical clustering analysis using Euclidean distance with average linkage. Several distinct patterns have been observed as if the samples are generated by different sequencing protocols. Based on our experience, such batch effect even exisits for sequencing samples from the same plate processed by the same protocols.

```
normal.counts <- TCGAbreast$normal[, -c(1:3)]
log.normal.counts <- apply(normal.counts, 2, function(x) {
    log10(x + 0.01)
})
normal.var <- apply(log.normal.counts, 1, var)
cutoff <- sort(normal.var, decreasing = TRUE)[1000]
```

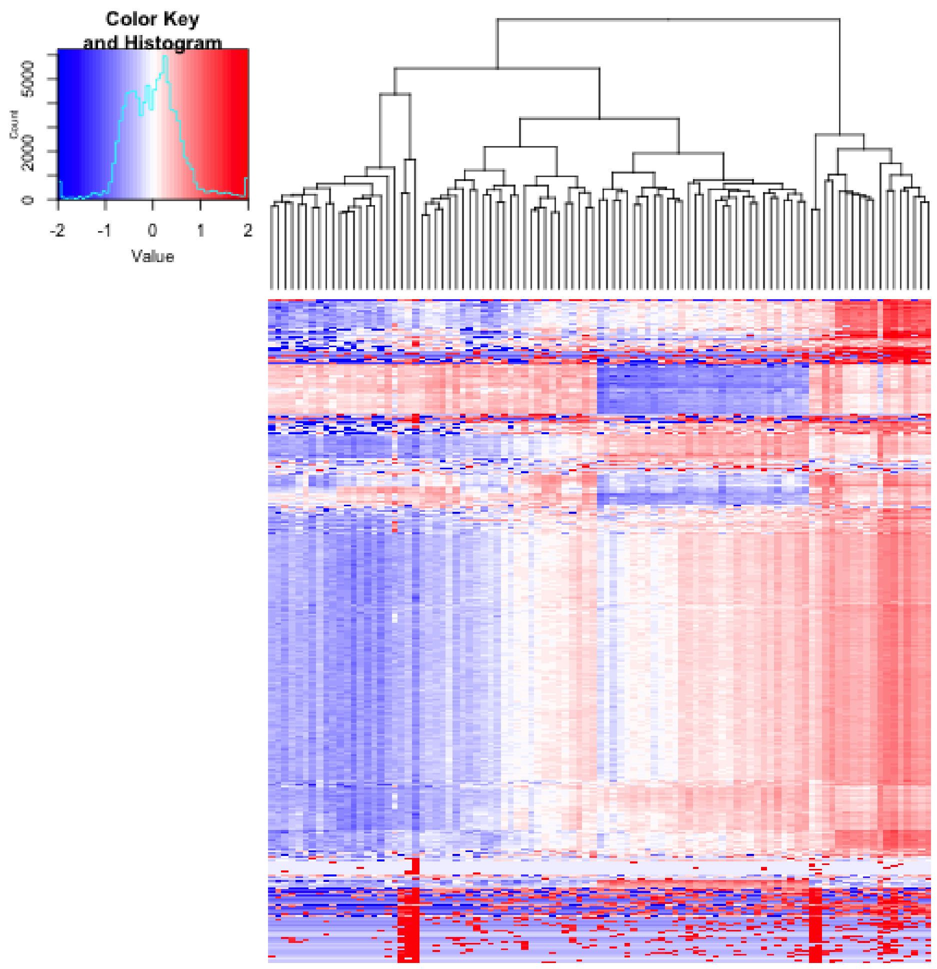

Figure 6: Heat map of log counts of top 1000 bins with largest vairance across 97 normal samples. Each bin is centered at 0.

```
selected <- log.normal.counts[normal.var >= cutoff, ]
truncated.data <- apply(selected, 1, function(x) {
    y <- x - mean(x)
    y[y > 2] <- 2
    y[y < -2] <- -2
    return(y)
})  # each row centered at 0; data are truncated for visualization purpose
library(gplots)
palette.gr.marray <- colorRampPalette(c("blue", "white", "red"))(56)
heatmap.2(as.matrix(t(truncated.data)), trace = "none", col = palette.gr.marray,
    key = T, symbreaks = T, labRow = NA, labCol = NA, dendrogram = "column")
```

```
mm <- mean(unlist(selected))
truncated.data2 <- apply(selected, 1, function(x) {
```

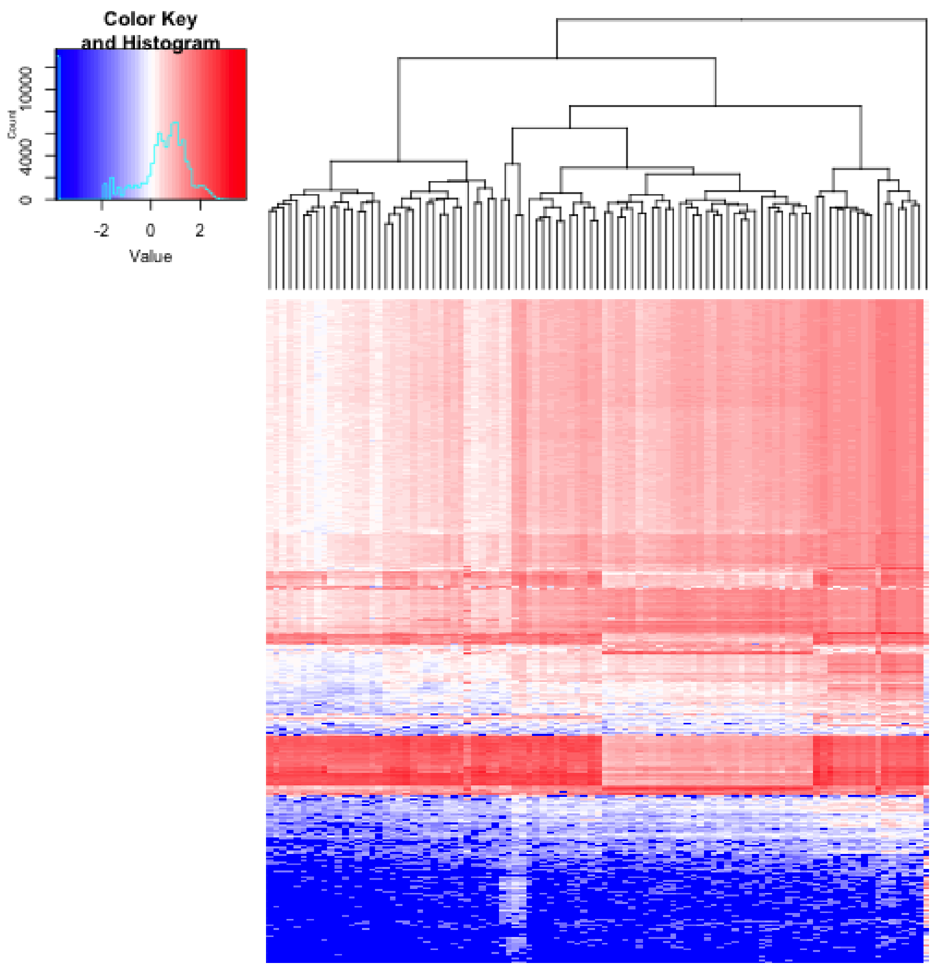

Figure 7: Heat map of log counts of top 1000 bins with largest vairance across 97 normal samples. The data is centered by the mean of all elements in the log count matrix.

```
    y <- x - mm
    y[y > 4] <- 4
    y[y < -4] <- -4
    return(y)
})  # centered by the mean of all entries; data are truncated for visualization purpo
heatmap.2(as.matrix(t(truncated.data2)), trace = "none", col = palette.gr.marray,
    key = T, symbreaks = T, labRow = NA, labCol = NA, dendrogram = "column")
```

6.2 Synthetic normal approaches

We find using a normal smple with similar enrichment pattern (termed as synthetic normal) will lead to better result than matched normal. *SynthEx* implements different ways to generate “synthetic normal”.

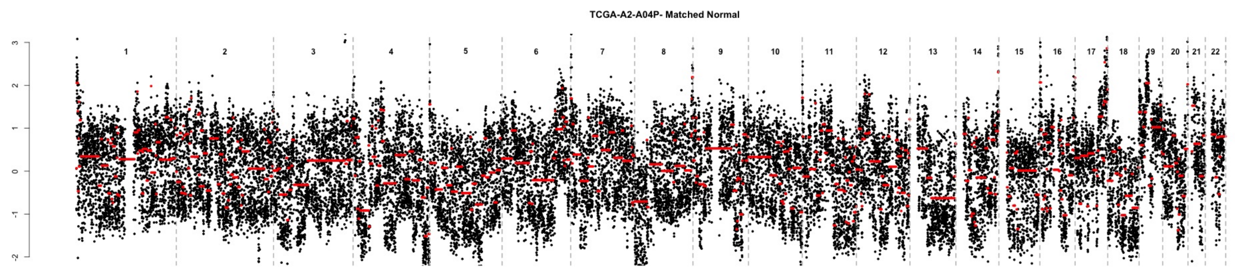

Figure 8: Segmentation of TCGA-A2-A04P using matched normal.

The most efficient way requires the input of the count of a to-be-analyzed tumor sample and a count matrix for normal samples either from the same study or from the same sequencing facility. *SynthEx* will scan through all normal samples, calcualte mean square successive variance and select K normals with smallest variance. Then the synthetic normal will be generated by taking median across K selected normals for each bin. The default K is 1.

Here shows an example of which the segmentation gets greatly improved if using synthetic normal approach.

```
tumor <- TCGAbreast$tumor[, c(1:3, 5)]
normal <- TCGAbreast$normal[, c(1:3, 5)]
sample.name <- colnames(TCGAbreast$tumor)[5]
Ratio <- SynthExcorrectBias(tumor, normal, bin.size = 100000,
    rm.centromere = TRUE, targetAnnotateBins = NULL, saveplot = FALSE,
    centromereBins = NULL, chrX = FALSE, plot = FALSE, result.dir,
    prefix = sample.name, reads.threshold = 25)
# Ratio <- normalization(Ratio, intersectBed.dir,
# genotype.file, vcf = TRUE, working.dir, result.dir, cutoff
# = 20, plot = FALSE, saveplot = FALSE, prefix = sample.name)
# #Due to the space limit, genotype file for this sample is
# not provided.
Seg <- createSegments(Ratio, "CBS")
Segments <- singleCNreport(Seg, report = FALSE, result.dir, saveplot = TRUE,
    prefix = paste0(sample.name, "- Matched Normal"), plotNormalized = TRUE,
    WGD = 1.35, pos.prop.threhold = 0.6, pos.log2ratio.threhold = 0.75)
```

We can see the read ratio data using matched normal are very noisy. This can be fixed using the synthetic normal approach. Besides the count data of to-be-analyzed sample, *SynthEx* requires the input of the count data of a collection of normal samples (in `count.matrix`). These normal samples are generated using the same sequencing protocol as the to-be-analyzed tumor sample.

```
tumor <- TCGAbreast$tumor[, c(1:3, 5)]
count.matrix <- TCGAbreast$normal
head(count.matrix[, 1:6])
```

```
##    chr  start    end TCGA-A1-A0SM TCGA-A2-A04P TCGA-A2-A04Q
## 1 chr1      1 100000          568         3135          322
## 2 chr1 100001 200000           22          481           40
```

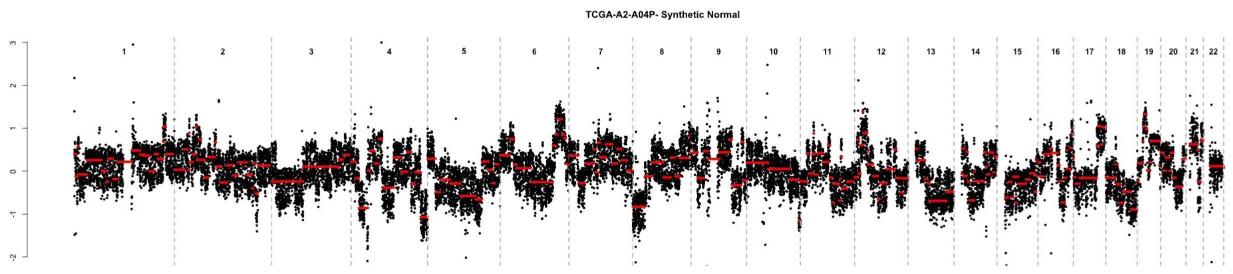

Figure 9: Segmentation of TCGA-A2-A04P using synthetic normal.

```
## 3 chr1 200001 300000            14          224           92
## 4 chr1 300001 400000             0            6            6
## 5 chr1 400001 500000             2            0            2
## 6 chr1 500001 600000           101          411          152
```

```
Ratio <- SynthExcorrectBias(tumor, count.matrix, bin.size = 100000,
    rm.centromere = TRUE, targetAnnotateBins = NULL, saveplot = FALSE,
    centromereBins = NULL, chrX = FALSE, plot = FALSE, result.dir,
    prefix = sample.name, reads.threshold = 25)
# Ratio <- normalization(Ratio, intersectBed.dir,
# genotype.file, vcf = TRUE, working.dir, result.dir, cutoff
# = 20, plot = FALSE, saveplot = FALSE, prefix = sample.name)
# #Due to the space limit, genotype file for this sample is
# not provided.
Seg <- createSegments(Ratio, "CBS")
Segments <- singleCNreport(Seg, report = FALSE, result.dir, saveplot = TRUE,
    prefix = paste0(sample.name, "- Synthetic Normal"), plotNormalized = TRUE,
    WGD = 1.35, pos.prop.threhold = 0.6, pos.log2ratio.threhold = 0.75)
```

The default number of samples used to generate the synthetic normal is 1. We can change that by setting K. The following sample will average 10 samples with smallest variance to obtain the synthetic normal.

```
sample.name <- colnames(TCGAbreast$tumor)[5]
Ratio <- SynthExcorrectBias(tumor, count.matrix, K = 10, bin.size = 100000,
    rm.centromere = TRUE, targetAnnotateBins = NULL, saveplot = FALSE,
    centromereBins = NULL, chrX = FALSE, plot = FALSE, result.dir,
    prefix = sample.name, reads.threshold = 25)
```

6.2.1 Multiple sample analysis pipeline

The above steps are integrated in function `SynthExPipeline()`.

```
Segfrompipe <- SynthExPipeline(tumor, count.matrix, bin.size = 100000,
    intersectBed.dir, genotype.file, result.dir, working.dir,
    verbose = TRUE)
```

6.2.2 (Optional) Creating a synthetic normal library

The most sophisticated way to generate synthetic normal involves the assessment of quality metrics using `picard`. We recommend using this approach for large scale genomic studies with a sample size greater than 500 or for a sequencing facility. The construction of synthetic library requires pre-calculated fold enrichment levels of all normal samples and the groupID of each sample. Each group represents an enrichment pattern. Such grouping struture can be learned by unsupervised approach such as heatmap shown in Figure6. Within each group, the normal samples will be stratified based on the library sizes and fold enrichment levels. The synthetic normal for a given library size and fold enrichment level is calculated as the median of all normal samples fall into that category.

```
SyntheticLibrary <- create_synthetic_library(counts, foldEnrichment,
    groupID, interval = 50000000)
ratioCorrectedBias <- SynthExcorrectBias(tumor, SyntheticLibrary,
    bin.size = 100000, rm.centromere = TRUE, targetAnnotateBins = NULL,
    saveplot = FALSE, centromereBins = NULL, chrX = FALSE, plot = TRUE,
    result.dir, prefix = sample.name, reads.threshold = 25)
```

7 Whole genome sequencing analysis

The synthetic normal approach can be applied to whole genome sequencing(WGS) as well. Function `calratioWGS` can take both a matched normal sample or a count matrix of multiple normal samples. For WGS data, the step to correct for the bias in target region will be skipped. The normalization, segmentation and purity estimation step will be the same as described previously. As for segmentation, we found trend filtering outperformed others on our WGS samples. Thus we suggest to test this method on your data.

```
WGSratio <- calratioWGS(tumor, normal, bin.size = 100000, rm.centromere = TRUE,
    K = 1, centromereBins = NULL, chrX = FALSE, reads.threshold = 25)
Seg <- createSegments(WGSratio, "TrendFiltering")
```