Introduction to *MutationalPatterns*

Francis Blokzijl Roel Janssen Ruben Van Boxtel Edwin Cuppen

October 17, 2016

Contents

1 Introduction

Mutational processes leave characteristic footprints in genomic DNA. This package provides an easy-to-use toolset for the characterization and visualization of mutational patterns in base substitution catalogues of e.g. tumour samples or DNA-repair deficient cells. The package covers a wide range of patterns including: mutational signatures, transcriptional strand bias, genomic distribution and association with genomic features, which are collectively meaningful for studying the activity of mutational processes. The package provides functionalities for both extracting mutational signatures de novo and inferring the contribution of previously identified mutational signatures in a given sample. MutationalPatterns integrates with common R genomic analysis workflows and allows easy association with (publicly available) annotation data.

This package provides a comprehensive set of flexible functions for easy finding and plotting of such mutational patterns in base substitution catalogues.

2 Related packages

2.1 Comparison to SomaticSignatures

2.1.1 Similar functionality

SomaticSignatures provides functions for genomic context determination and de novo signature extraction using NMF decomposition of 96 trinucleotide count matrices. MutationalPatterns provides this functionality too but the plotting is different, because MutationalPatterns offers the functionality to extract signatures *de novo* from a 192 feature matrix, with 96 trinucleotide X 2 transcriptional strands. This allows assessment of transcriptional strand bias of mutational signatures, which is important to characterize the underlying mutational mechanism.

2.1.2 Unique functionality

The following functions can be found in MutationalPatterns, but not in SomaticSignatures.

- `plot_contribution`: A function to determine the optimal contribution of previously identified signatures, e.g. cosmic cancer signatures, to reconstruct the mutational profile of just a single sample. This is useful for two reasons:
  1. for NMF you need many samples with distinct mutational profiles, as it relies on dimensionality reduction.

2. In order to further characterize "known" mutational signatures and find the underlying mutational mechanisms, this function can be used to determine the contribution of these signatures in different samples, e.g. normal cells, or cells with defective DNA repair mechanisms etc.

- `plot_enrichment_depletion`: A plotting function and statistical test for enrichment or depletion of mutations in any (publicly available) annotated genomic region such as transcription factor binding site or "open chromatin". This is useful for the characterization of mutational mechanisms.
- `strand_bias_test`, `plot_strand_bias`: A statistical test and a plotting function for transcriptional strand bias in mutation catalogs.
- `plot_96_profile`: A plotting function to visualize the difference between two 96 mutational profiles and calculate RSS.

3 Data for use with this package

One of the first thing you will want to do is load a reference genome. After this you will probably want to load your VCF datasets. We provided an example data set with the package.

3.1 List all available reference genomes (BSgenome)

```
> library(BSgenome)
> # Let's display the first five entries only to save space.
> available.genomes()[1:5]
```

```
[1] "BSgenome.Alyrata.JGI.v1"
[2] "BSgenome.Amellifera.BeeBase.assembly4"
[3] "BSgenome.Amellifera.UCSC.apiMel2"
[4] "BSgenome.Amellifera.UCSC.apiMel2.masked"
[5] "BSgenome.Athaliana.TAIR.04232008"
```

3.2 Load your data

```
> ref_genome <- "BSgenome.Hsapiens.UCSC.hg19"
> library(ref_genome, character.only = TRUE)
```

3.3 Load example data

```
> library(MutationalPatterns)
```

3.4 Load data

Small data samples are included in the package. You can download larger samples from the MutationalPatterns-data repository at https://github.com/CuppenResearch/MutationalPatterns-data/.

The data in that repository consists of somatic mutation catalogues of nine normal human adult stem cells from three tissues (Blokzijl et al., 2016).

To load data, we need to locate it:

```
> vcf_files <- list.files(system.file("extdata", package="MutationalPatterns"),
+                         pattern = ".vcf", full.names = TRUE)
```

And define corresponding names for the datasets:

```
> sample_names <- c(
+     "colon1", "colon2", "colon3",
+     "intestine1", "intestine2", "intestine3",
+     "liver1", "liver2", "liver3")
```

Now we can load it:

```
> vcfs <- read_vcfs_as_granges(vcf_files, sample_names, genome = "hg19")
> summary(vcfs)
```

```
     Length       Class        Mode
          9 GRangesList          S4
```

4 Take a subset of the chromosomes

If you want to use a subset of the chromosomes, like all chromosomes except the X, Y or MT, you can use the following snippet:

```
> auto <- extractSeqlevelsByGroup(species="Homo_sapiens",
+                                 style="UCSC",
+                                 group="auto")
> vcfs <- lapply(vcfs, function(x) keepSeqlevels(x, auto))
```

5 Mutation characteristics

Now that we have loaded the data, unified its naming, and filtered the datasets to what we want to analyze, we can start our search for characteristic footprints.

5.1 Base substitution types

We can retrieve base substitutions from vcf object as "REF>ALT" using `mutations_from_muts`.

```
> mutations_from_vcf(vcfs[[1]])
```

```
  [1] "A>T" "C>T" "G>T" "T>C" "T>C" "G>A" "C>T" "A>G" "C>T" "G>A" "C>A"
 [12] "A>C" "C>T" "G>T" "C>T" "C>T" "C>T" "A>T" "T>C" "C>A" "G>T" "C>T"
 [23] "C>T" "C>T" "G>A" "T>G" "G>A" "C>A" "C>T" "T>A" "C>T" "T>A" "T>C"
 [34] "G>A" "C>A" "A>G" "C>G" "T>G" "C>A" "T>C" "G>A" "G>A" "A>T" "T>G"
 [45] "T>C" "G>A" "C>A" "C>T" "C>A" "C>T" "C>T" "C>T" "G>C" "G>A" "G>A"
 [56] "T>C" "C>T" "G>A" "C>A" "G>A" "C>T" "C>A" "A>G" "T>C" "G>C" "C>T"
 [67] "G>A" "A>G" "G>T" "G>T" "G>A" "G>A" "G>A" "C>T" "C>T" "G>A" "C>A"
 [78] "C>A" "G>T" "G>A" "T>C" "T>A" "G>A" "C>T" "C>T" "G>A" "C>T" "T>C"
 [89] "G>T" "T>C" "C>T" "C>A" "G>A" "G>A" "T>A" "C>T" "C>G" "C>T" "C>G"
[100] "C>T" "C>T" "C>A" "C>T" "C>T" "G>A" "C>A" "G>A" "A>G" "A>C" "T>C"
[111] "T>C" "T>C" "T>C" "A>C" "G>A" "T>C" "G>C" "A>C" "G>A" "G>A" "G>A"
[122] "G>A" "T>C" "C>T" "G>T" "T>C" "C>A" "G>A" "G>C" "T>C" "T>C" "C>T"
[133] "T>C" "G>T" "T>A" "C>A" "C>A" "G>A" "G>T" "G>A" "A>T" "C>T" "G>A"
[144] "C>T" "A>G" "G>A" "G>C" "G>A" "G>T" "C>T" "G>A" "G>A" "G>C" "A>G"
[155] "C>T" "A>G" "C>T" "C>T" "C>A" "G>A" "T>C" "G>A" "A>G" "C>T" "C>T"
[166] "C>T" "G>A" "T>G" "C>T" "C>G" "G>A" "G>T" "C>A" "G>T" "A>C" "A>T"
[177] "A>G" "G>A" "G>A" "C>A" "C>T" "C>T" "T>G" "G>A" "A>C" "T>A" "C>T"
[188] "A>G" "C>A" "G>A" "C>G" "C>T" "G>A" "A>G" "G>A" "G>A" "A>T" "C>G"
[199] "C>T" "G>A" "G>T" "G>A" "A>G" "T>A" "T>C" "G>T" "T>C" "G>T" "C>T"
```

```
[210] "G>A" "C>T" "G>A" "G>T" "C>T" "C>T" "C>T" "G>A" "T>A" "G>A" "G>A"
[221] "T>C" "G>A" "A>T" "T>C" "T>C" "C>T" "C>T" "C>T" "C>T" "G>A" "T>C"
[232] "G>A" "C>T" "C>G" "C>T" "C>T" "G>A" "G>A" "C>T" "C>T" "G>A" "G>T"
[243] "C>G" "G>A" "C>T" "C>T" "C>T" "G>T" "C>T" "C>A" "G>A" "C>A" "T>C"
[254] "C>T" "G>A" "T>A" "A>G" "G>A" "G>A" "C>T" "G>T" "C>T" "G>A" "C>T"
[265] "A>C" "T>C" "G>T" "C>T" "A>T" "A>G" "C>A" "G>A" "G>T" "C>T" "C>T"
[276] "G>A" "T>A" "G>A" "T>C" "T>C" "C>A" "G>A" "A>G" "G>A" "C>T" "G>A"
[287] "C>T" "A>G" "A>G" "A>C" "C>A" "A>C" "T>G" "G>A" "C>T" "C>A" "C>T"
[298] "C>T" "G>A" "G>A" "C>A" "C>T" "G>A" "G>T" "G>C" "G>A" "G>A" "C>A"
[309] "C>T" "A>G" "C>A" "C>T" "G>A" "G>A" "C>T" "G>C" "G>A" "C>T" "C>G"
[320] "T>G" "G>T" "A>G" "G>T" "C>T" "G>T" "G>A" "T>A" "T>C" "G>A" "G>A"
[331] "C>T" "G>A" "G>T" "C>T" "G>A" "T>A" "A>T" "C>G" "C>T" "G>A" "G>T"
[342] "C>T" "T>A" "C>A" "A>T" "G>A" "C>T" "G>T" "A>C" "A>G" "G>A" "G>A"
[353] "C>T" "T>G" "G>A" "A>G" "C>A" "G>A" "T>C" "A>G" "C>T" "T>C" "C>T"
[364] "C>A" "A>G" "G>C" "C>T" "A>C" "C>T" "C>A" "C>T" "C>A" "A>T" "C>T"
[375] "G>A" "G>T" "C>A" "C>G" "C>A" "G>T" "C>A" "G>A" "C>T" "G>A" "C>T"
[386] "A>G" "T>G" "G>A" "C>T" "T>C" "C>T" "C>T" "C>T" "G>A" "C>T" "G>A"
[397] "G>T" "G>A" "C>T" "C>T" "T>C" "G>A" "T>C" "G>A" "C>T" "T>C" "G>T"
[408] "T>G" "C>T" "T>C" "G>A" "C>T" "G>A" "T>C" "A>G" "C>T" "C>T" "G>A"
[419] "G>A" "G>A" "C>T" "G>A" "G>A" "G>A" "C>T" "C>T" "T>C" "G>A" "C>T"
[430] "G>A" "C>T" "G>A" "C>T" "G>A" "A>G" "G>A" "A>C" "G>A" "A>G" "C>T"
[441] "C>T" "A>T" "G>A" "C>T" "C>T" "G>A" "C>T" "A>C" "C>T" "G>A" "T>G"
[452] "C>T" "G>C" "G>T" "A>G" "G>T" "C>T" "C>T" "C>T" "G>A" "G>T" "C>T"
[463] "T>C" "T>C" "T>C" "G>A" "C>T" "T>C" "G>A" "C>G" "A>C" "A>T" "C>T"
[474] "G>T" "A>G" "C>T" "C>T" "G>A" "G>T" "G>C" "C>T" "G>A" "C>T" "C>T"
[485] "G>A" "A>G" "C>T" "G>A" "C>T" "C>T" "C>A"
```

We can retrieve base substitutions from vcf and convert to the 6 types of base substitution types that are distinguished by convention: C>A, C>G, C>T, T>A, T>C, T>G. For example, when the reference allele is G and the alternative allele is T (G>T), this functions returns the G:C>T:A mutation as a C>A mutation:

```
> mutation_types(vcfs[[1]])
```

```
  [1] "T>A" "C>T" "C>A" "T>C" "T>C" "C>T" "C>T" "T>C" "C>T" "C>T" "C>A"
 [12] "T>G" "C>T" "C>A" "C>T" "C>T" "C>T" "T>A" "T>C" "C>A" "C>A" "C>T"
 [23] "C>T" "C>T" "C>T" "T>G" "C>T" "C>A" "C>T" "T>A" "C>T" "T>A" "T>C"
 [34] "C>T" "C>A" "T>C" "C>G" "T>G" "C>A" "T>C" "C>T" "C>T" "T>A" "T>G"
 [45] "T>C" "C>T" "C>A" "C>T" "C>A" "C>T" "C>T" "C>T" "C>G" "C>T" "C>T"
 [56] "T>C" "C>T" "C>T" "C>A" "C>T" "C>T" "C>A" "T>C" "T>C" "C>G" "C>T"
 [67] "C>T" "T>C" "C>A" "C>A" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>A"
 [78] "C>A" "C>A" "C>T" "T>C" "T>A" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C"
 [89] "C>A" "T>C" "C>T" "C>A" "C>T" "C>T" "T>A" "C>T" "C>G" "C>T" "C>G"
[100] "C>T" "C>T" "C>A" "C>T" "C>T" "C>T" "C>A" "C>T" "T>C" "T>G" "T>C"
[111] "T>C" "T>C" "T>C" "T>G" "C>T" "T>C" "C>G" "T>G" "C>T" "C>T" "C>T"
[122] "C>T" "T>C" "C>T" "C>A" "T>C" "C>A" "C>T" "C>G" "T>C" "T>C" "C>T"
[133] "T>C" "C>A" "T>A" "C>A" "C>A" "C>T" "C>A" "C>T" "T>A" "C>T" "C>T"
[144] "C>T" "T>C" "C>T" "C>G" "C>T" "C>A" "C>T" "C>T" "C>T" "C>G" "T>C"
[155] "C>T" "T>C" "C>T" "C>T" "C>A" "C>T" "T>C" "C>T" "T>C" "C>T" "C>T"
[166] "C>T" "C>T" "T>G" "C>T" "C>G" "C>T" "C>A" "C>A" "C>A" "T>G" "T>A"
[177] "T>C" "C>T" "C>T" "C>A" "C>T" "C>T" "T>G" "C>T" "T>G" "T>A" "C>T"
[188] "T>C" "C>A" "C>T" "C>G" "C>T" "C>T" "T>C" "C>T" "C>T" "T>A" "C>G"
[199] "C>T" "C>T" "C>A" "C>T" "T>C" "T>A" "T>C" "C>A" "T>C" "C>A" "C>T"
[210] "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>T" "C>T" "T>A" "C>T" "C>T"
[221] "T>C" "C>T" "T>A" "T>C" "T>C" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C"
[232] "C>T" "C>T" "C>G" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>A"
[243] "C>G" "C>T" "C>T" "C>T" "C>T" "C>A" "C>T" "C>A" "C>T" "C>A" "T>C"
```

```
[254] "C>T" "C>T" "T>A" "T>C" "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>T"
[265] "T>G" "T>C" "C>A" "C>T" "T>A" "T>C" "C>A" "C>T" "C>A" "C>T" "C>T"
[276] "C>T" "T>A" "C>T" "T>C" "T>C" "C>A" "C>T" "T>C" "C>T" "C>T" "C>T"
[287] "C>T" "T>C" "T>C" "T>G" "C>A" "T>G" "T>G" "C>T" "C>T" "C>A" "C>T"
[298] "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>A" "C>G" "C>T" "C>T" "C>A"
[309] "C>T" "T>C" "C>A" "C>T" "C>T" "C>T" "C>T" "C>G" "C>T" "C>T" "C>G"
[320] "T>G" "C>A" "T>C" "C>A" "C>T" "C>A" "C>T" "T>A" "T>C" "C>T" "C>T"
[331] "C>T" "C>T" "C>A" "C>T" "C>T" "T>A" "T>A" "C>G" "C>T" "C>T" "C>A"
[342] "C>T" "T>A" "C>A" "T>A" "C>T" "C>T" "C>A" "T>G" "T>C" "C>T" "C>T"
[353] "C>T" "T>G" "C>T" "T>C" "C>A" "C>T" "T>C" "T>C" "C>T" "T>C" "C>T"
[364] "C>A" "T>C" "C>G" "C>T" "T>G" "C>T" "C>A" "C>T" "C>A" "T>A" "C>T"
[375] "C>T" "C>A" "C>A" "C>G" "C>A" "C>A" "C>A" "C>T" "C>T" "C>T" "C>T"
[386] "T>C" "T>G" "C>T" "C>T" "T>C" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T"
[397] "C>A" "C>T" "C>T" "C>T" "T>C" "C>T" "T>C" "C>T" "C>T" "T>C" "C>A"
[408] "T>G" "C>T" "T>C" "C>T" "C>T" "C>T" "T>C" "T>C" "C>T" "C>T" "C>T"
[419] "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C" "C>T" "C>T"
[430] "C>T" "C>T" "C>T" "C>T" "C>T" "T>C" "C>T" "T>G" "C>T" "T>C" "C>T"
[441] "C>T" "T>A" "C>T" "C>T" "C>T" "C>T" "C>T" "T>G" "C>T" "C>T" "T>G"
[452] "C>T" "C>G" "C>A" "T>C" "C>A" "C>T" "C>T" "C>T" "C>T" "C>A" "C>T"
[463] "T>C" "T>C" "T>C" "C>T" "C>T" "T>C" "C>T" "C>G" "T>G" "T>A" "C>T"
[474] "C>A" "T>C" "C>T" "C>T" "C>T" "C>A" "C>G" "C>T" "C>T" "C>T" "C>T"
[485] "C>T" "T>C" "C>T" "C>T" "C>T" "C>T" "C>A"
```

To retrieve the context (one base upstream and one base downstream) of the positions in the VCF object from the reference genome, you can use the `get_mut_context` function:

```
> mutation_context(vcfs[[1]], ref_genome)
```

```
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
"AAA" "ACG" "TGC" "GTC" "ATT" "GGG" "ACG" "CAT" "GCG" "AGG" "ACG" "TAA"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
"ACG" "TGA" "TCT" "CCC" "CCG" "CAT" "ATT" "TCT" "TGA" "ACG" "ACG" "ACC"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
"CGA" "GTA" "CGT" "ACC" "CCG" "TTG" "ACA" "ATC" "ATA" "GGT" "TCT" "TAC"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr2  chr2  chr2
"TCT" "ATG" "ACC" "TTT" "CGT" "TGC" "TAT" "TTT" "TTT" "GGT" "GCA" "ACG"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"ACA" "ACG" "ACG" "ACG" "AGT" "TGA" "CGT" "TTT" "ACG" "CGC" "ACC" "GGA"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"ACG" "ACT" "TAT" "ATC" "AGA" "ACA" "AGG" "TAT" "GGA" "TGG" "GGG" "AGA"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"CGT" "ACG" "CCT" "CGT" "ACA" "CCC" "TGG" "TGT" "CTA" "ATT" "CGA" "TCA"
 chr2  chr2  chr2  chr2  chr2  chr3  chr3  chr3  chr3  chr3  chr3  chr3
"GCG" "AGA" "GCG" "ATA" "AGA" "TTA" "ACT" "TCT" "CGC" "AGG" "ATG" "TCG"
 chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3
"CCA" "ACA" "ACA" "GCG" "CCG" "GCG" "TCG" "CCG" "CGT" "ACA" "GGG" "AAG"
 chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr4  chr4  chr4  chr4
"AAG" "ATT" "ATA" "ATT" "ATA" "AAT" "CGC" "GTT" "TGT" "GAC" "GGA" "AGA"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4
"TGA" "AGT" "CTA" "CCT" "TGG" "TTG" "TCC" "GGC" "GGG" "TTC" "TTT" "GCG"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4
"TTC" "GGT" "ATA" "ACC" "TCT" "CGG" "AGT" "CGG" "GAT" "CCC" "CGG" "CCG"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr5
"GAG" "CGC" "AGA" "TGC" "TGA" "GCC" "GGA" "GGA" "GGT" "AAT" "ACG" "GAC"
 chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5
"GCG" "GCC" "GCT" "TGG" "ATT" "CGA" "AAT" "GCT" "GCG" "TCA" "AGC" "ATG"
```

```
 chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5
"ACA" "CCT" "TGC" "CGT" "TCC" "TGA" "CAT" "AAT" "CAA" "GGT" "CGC" "ACA"
 chr5  chr5  chr5  chr5  chr5  chr5  chr6  chr6  chr6  chr6  chr6  chr6
"ACA" "ACG" "CTC" "CGT" "AAA" "CTG" "TCG" "AAT" "GCC" "GGC" "ACT" "GCG"
 chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6
"CGC" "AAT" "AGC" "CGC" "CAT" "TCC" "TCG" "CGT" "GGC" "CGC" "GAG" "ATC"
 chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6
"ATA" "AGA" "ATT" "TGG" "ACT" "CGG" "ACG" "GGA" "AGA" "GCG" "ACG" "CCA"
 chr6  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"GGC" "CTT" "TGG" "TGT" "ATT" "TGT" "AAA" "ATG" "TTT" "CCT" "TCG" "GCC"
 chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"CCC" "CGT" "CTC" "CGT" "CCG" "ACC" "ACG" "GCG" "CGT" "CGG" "ACG" "TCG"
 chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"CGT" "AGG" "TCA" "CGT" "GCA" "TCT" "TCC" "AGT" "CCC" "GCT" "AGG" "GCC"
 chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8
"ATA" "ACG" "CGT" "ATT" "TAA" "AGG" "GGT" "ACG" "CGG" "ACG" "CGC" "CCG"
 chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8
"AAA" "GTA" "TGC" "GCG" "AAT" "AAT" "TCG" "CGA" "TGC" "CCG" "CCT" "CGG"
 chr8  chr8  chr8  chr8  chr9  chr9  chr9  chr9  chr9  chr9  chr9  chr9
"CTT" "AGA" "GTT" "ATA" "GCA" "CGT" "AAA" "AGA" "ACA" "GGT" "TCT" "CAC"
 chr9  chr9  chr9  chr9  chr9  chr9  chr9  chr9 chr10 chr10 chr10 chr10
"TAT" "TAA" "ACA" "AAG" "TTT" "CGA" "TCT" "ACC" "CCA" "ACG" "CGT" "TGA"
chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10
"CCG" "ACG" "AGT" "AGG" "TGC" "AGG" "AGG" "ACA" "GCG" "TAT" "TCC" "GCC"
chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr11 chr11 chr11 chr11
"GGC" "TGG" "TCA" "AGT" "TGA" "CCG" "TCA" "ATT" "AGA" "AAC" "AGA" "GCA"
chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11
"TGC" "TGT" "TTT" "CTT" "CGC" "CGC" "ACT" "CGA" "TGG" "ACG" "TGT" "TTG"
chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr12
"CAG" "ACA" "TCT" "CGC" "GGG" "ACA" "ATC" "GCA" "AAA" "CGT" "ACG" "TGG"
chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12
"CAA" "AAT" "CGG" "AGT" "CCG" "ATG" "CGT" "AAA" "GCA" "CGT" "TTA" "AAA"
chr12 chr12 chr12 chr12 chr12 chr12 chr13 chr13 chr13 chr13 chr13 chr13
"ACG" "CTT" "ACT" "ACG" "AAT" "TGA" "GCG" "AAT" "CCG" "TCG" "ACG" "ACT"
chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr14
"CAG" "TCA" "AGG" "AGG" "CCA" "ACC" "TCA" "AGA" "TCA" "CGA" "ACG" "CGC"
chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14
"CCG" "AAA" "CTA" "CGA" "GCG" "ATA" "GCG" "TCT" "ACT" "TGC" "ACG" "CGG"
chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr15 chr15
"GGA" "CGT" "CCT" "ACG" "TTG" "CGT" "ATC" "GGC" "ACG" "CTG" "GGG" "GTT"
chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15
"ACC" "ATT" "CGC" "CCG" "CGG" "GTA" "AAG" "CCC" "GCG" "TGC" "GGG" "CGT"
chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16
"ACG" "CGC" "CGG" "TGA" "GCG" "ACA" "TTC" "TGC" "CCG" "CGC" "ACG" "GGC"
chr17 chr17 chr17 chr17 chr17 chr17 chr17 chr17 chr18 chr18 chr18 chr18
"GCC" "CGT" "AAT" "CGT" "GAG" "CGG" "CAC" "CCG" "CCC" "TAA" "GGG" "GCC"
chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18
"GCG" "AGA" "TCG" "AAA" "TCA" "AGT" "GTC" "CCG" "AGT" "AGA" "CAT" "TGC"
chr18 chr18 chr19 chr19 chr19 chr19 chr19 chr19 chr19 chr19 chr20 chr20
"GCG" "ACG" "ACG" "CGT" "TGG" "GCG" "CTT" "ATA" "TTA" "TGC" "ACG" "GTA"
chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr21 chr21
"CGG" "ACA" "GAA" "CAT" "ACT" "CGT" "AAG" "ACA" "GCG" "CGT" "TGA" "GGG"
chr21 chr21 chr21 chr21 chr21 chr21 chr21 chr22 chr22 chr22 chr22
"CCT" "CGT" "TCG" "GCG" "TGC" "CAG" "CCC" "CGG" "ACG" "CCT" "ACC"
```

With `get_type_context`, you can retrieve the types and context of the base substitution types for all positions in the VCF object. For the base substitutions that are converted to the conventional base substitution types, the reverse complement of the context is returned.

```
> type_context(vcfs[[1]], ref_genome)
```

```
$types
  [1] "T>A" "C>T" "C>A" "T>C" "T>C" "C>T" "C>T" "T>C" "C>T" "C>T" "C>A"
 [12] "T>G" "C>T" "C>A" "C>T" "C>T" "C>T" "T>A" "T>C" "C>A" "C>A" "C>T"
 [23] "C>T" "C>T" "C>T" "T>G" "C>T" "C>A" "C>T" "T>A" "C>T" "T>A" "T>C"
 [34] "C>T" "C>A" "T>C" "C>G" "T>G" "C>A" "T>C" "C>T" "C>T" "T>A" "T>G"
 [45] "T>C" "C>T" "C>A" "C>T" "C>A" "C>T" "C>T" "C>T" "C>G" "C>T" "C>T"
 [56] "T>C" "C>T" "C>T" "C>A" "C>T" "C>T" "C>A" "T>C" "T>C" "C>G" "C>T"
 [67] "C>T" "T>C" "C>A" "C>A" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>A"
 [78] "C>A" "C>A" "C>T" "T>C" "T>A" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C"
 [89] "C>A" "T>C" "C>T" "C>A" "C>T" "C>T" "T>A" "C>T" "C>G" "C>T" "C>G"
[100] "C>T" "C>T" "C>A" "C>T" "C>T" "C>T" "C>A" "C>T" "T>C" "T>G" "T>C"
[111] "T>C" "T>C" "T>C" "T>G" "C>T" "T>C" "C>G" "T>G" "C>T" "C>T" "C>T"
[122] "C>T" "T>C" "C>T" "C>A" "T>C" "C>A" "C>T" "C>G" "T>C" "T>C" "C>T"
[133] "T>C" "C>A" "T>A" "C>A" "C>A" "C>T" "C>A" "C>T" "T>A" "C>T" "C>T"
[144] "C>T" "T>C" "C>T" "C>G" "C>T" "C>A" "C>T" "C>T" "C>T" "C>G" "T>C"
[155] "C>T" "T>C" "C>T" "C>T" "C>A" "C>T" "T>C" "C>T" "T>C" "C>T" "C>T"
[166] "C>T" "C>T" "T>G" "C>T" "C>G" "C>T" "C>A" "C>A" "C>A" "T>G" "T>A"
[177] "T>C" "C>T" "C>T" "C>A" "C>T" "C>T" "T>G" "C>T" "T>G" "T>A" "C>T"
[188] "T>C" "C>A" "C>T" "C>G" "C>T" "C>T" "T>C" "C>T" "C>T" "T>A" "C>G"
[199] "C>T" "C>T" "C>A" "C>T" "T>C" "T>A" "T>C" "C>A" "T>C" "C>A" "C>T"
[210] "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>T" "C>T" "T>A" "C>T" "C>T"
[221] "T>C" "C>T" "T>A" "T>C" "T>C" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C"
[232] "C>T" "C>T" "C>G" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>A"
[243] "C>G" "C>T" "C>T" "C>T" "C>T" "C>A" "C>T" "C>A" "C>T" "C>A" "T>C"
[254] "C>T" "C>T" "T>A" "T>C" "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>T"
[265] "T>G" "T>C" "C>A" "C>T" "T>A" "T>C" "C>A" "C>T" "C>A" "C>T" "C>T"
[276] "C>T" "T>A" "C>T" "T>C" "T>C" "C>A" "C>T" "T>C" "C>T" "C>T" "C>T"
[287] "C>T" "T>C" "T>C" "T>G" "C>A" "T>G" "T>G" "C>T" "C>T" "C>A" "C>T"
[298] "C>T" "C>T" "C>T" "C>A" "C>T" "C>T" "C>A" "C>G" "C>T" "C>T" "C>A"
[309] "C>T" "T>C" "C>A" "C>T" "C>T" "C>T" "C>T" "C>G" "C>T" "C>T" "C>G"
[320] "T>G" "C>A" "T>C" "C>A" "C>T" "C>A" "C>T" "T>A" "T>C" "C>T" "C>T"
[331] "C>T" "C>T" "C>A" "C>T" "C>T" "T>A" "T>A" "C>G" "C>T" "C>T" "C>A"
[342] "C>T" "T>A" "C>A" "T>A" "C>T" "C>T" "C>A" "T>G" "T>C" "C>T" "C>T"
[353] "C>T" "T>G" "C>T" "T>C" "C>A" "C>T" "T>C" "T>C" "C>T" "T>C" "C>T"
[364] "C>A" "T>C" "C>G" "C>T" "T>G" "C>T" "C>A" "C>T" "C>A" "T>A" "C>T"
[375] "C>T" "C>A" "C>A" "C>G" "C>A" "C>A" "C>A" "C>T" "C>T" "C>T" "C>T"
[386] "T>C" "T>G" "C>T" "C>T" "T>C" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T"
[397] "C>A" "C>T" "C>T" "C>T" "T>C" "C>T" "T>C" "C>T" "C>T" "T>C" "C>A"
[408] "T>G" "C>T" "T>C" "C>T" "C>T" "C>T" "T>C" "T>C" "C>T" "C>T" "C>T"
[419] "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "C>T" "T>C" "C>T" "C>T"
[430] "C>T" "C>T" "C>T" "C>T" "C>T" "T>C" "C>T" "T>G" "C>T" "T>C" "C>T"
[441] "C>T" "T>A" "C>T" "C>T" "C>T" "C>T" "C>T" "T>G" "C>T" "C>T" "T>G"
[452] "C>T" "C>G" "C>A" "T>C" "C>A" "C>T" "C>T" "C>T" "C>T" "C>A" "C>T"
[463] "T>C" "T>C" "T>C" "C>T" "C>T" "T>C" "C>T" "C>G" "T>G" "T>A" "C>T"
[474] "C>A" "T>C" "C>T" "C>T" "C>T" "C>A" "C>G" "C>T" "C>T" "C>T" "C>T"
[485] "C>T" "T>C" "C>T" "C>T" "C>T" "C>T" "C>A"

$context
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
```

```
"TTT" "ACG" "GCA" "GTC" "ATT" "CCC" "ACG" "ATG" "GCG" "CCT" "ACG" "TTA"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
"ACG" "TCA" "TCT" "CCC" "CCG" "ATG" "ATT" "TCT" "TCA" "ACG" "ACG" "ACC"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1
"TCG" "GTA" "ACG" "ACC" "CCG" "TTG" "ACA" "ATC" "ATA" "ACC" "TCT" "GTA"
 chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr1  chr2  chr2  chr2
"TCT" "ATG" "ACC" "TTT" "ACG" "GCA" "ATA" "TTT" "TTT" "ACC" "GCA" "ACG"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"ACA" "ACG" "ACG" "ACG" "ACT" "TCA" "ACG" "TTT" "ACG" "GCG" "ACC" "TCC"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"ACG" "ACT" "ATA" "ATC" "TCT" "ACA" "CCT" "ATA" "TCC" "CCA" "CCC" "TCT"
 chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2  chr2
"ACG" "ACG" "CCT" "ACG" "ACA" "CCC" "CCA" "ACA" "CTA" "ATT" "TCG" "TCA"
 chr2  chr2  chr2  chr2  chr2  chr3  chr3  chr3  chr3  chr3  chr3  chr3
"GCG" "TCT" "GCG" "ATA" "TCT" "TTA" "ACT" "TCT" "GCG" "CCT" "ATG" "TCG"
 chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3
"CCA" "ACA" "ACA" "GCG" "CCG" "GCG" "TCG" "CCG" "ACG" "ACA" "CCC" "CTT"
 chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr3  chr4  chr4  chr4  chr4
"CTT" "ATT" "ATA" "ATT" "ATA" "ATT" "GCG" "GTT" "ACA" "GTC" "TCC" "TCT"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4
"TCA" "ACT" "CTA" "CCT" "CCA" "TTG" "TCC" "GCC" "CCC" "TTC" "TTT" "GCG"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4
"TTC" "ACC" "ATA" "ACC" "TCT" "CCG" "ACT" "CCG" "ATC" "CCC" "CCG" "CCG"
 chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr4  chr5
"CTC" "GCG" "TCT" "GCA" "TCA" "GCC" "TCC" "TCC" "ACC" "ATT" "ACG" "GTC"
 chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5
"GCG" "GCC" "GCT" "CCA" "ATT" "TCG" "ATT" "GCT" "GCG" "TCA" "GCT" "ATG"
 chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5  chr5
"ACA" "CCT" "GCA" "ACG" "TCC" "TCA" "ATG" "ATT" "TTG" "ACC" "GCG" "ACA"
 chr5  chr5  chr5  chr5  chr5  chr5  chr6  chr6  chr6  chr6  chr6  chr6
"ACA" "ACG" "CTC" "ACG" "TTT" "CTG" "TCG" "ATT" "GCC" "GCC" "ACT" "GCG"
 chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6
"GCG" "ATT" "GCT" "GCG" "ATG" "TCC" "TCG" "ACG" "GCC" "GCG" "CTC" "ATC"
 chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6  chr6
"ATA" "TCT" "ATT" "CCA" "ACT" "CCG" "ACG" "TCC" "TCT" "GCG" "ACG" "CCA"
 chr6  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"GCC" "CTT" "CCA" "ACA" "ATT" "ACA" "TTT" "ATG" "TTT" "CCT" "TCG" "GCC"
 chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"CCC" "ACG" "CTC" "ACG" "CCG" "ACC" "ACG" "GCG" "ACG" "CCG" "ACG" "TCG"
 chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7  chr7
"ACG" "CCT" "TCA" "ACG" "GCA" "TCT" "TCC" "ACT" "CCC" "GCT" "CCT" "GCC"
 chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8
"ATA" "ACG" "ACG" "ATT" "TTA" "CCT" "ACC" "ACG" "CCG" "ACG" "GCG" "CCG"
 chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8  chr8
"TTT" "GTA" "GCA" "GCG" "ATT" "ATT" "TCG" "TCG" "GCA" "CCG" "CCT" "CCG"
 chr8  chr8  chr8  chr8  chr9  chr9  chr9  chr9  chr9  chr9  chr9  chr9
"CTT" "TCT" "GTT" "ATA" "GCA" "ACG" "TTT" "TCT" "ACA" "ACC" "TCT" "GTG"
 chr9  chr9  chr9  chr9  chr9  chr9  chr9  chr9 chr10 chr10 chr10 chr10
"ATA" "TTA" "ACA" "CTT" "TTT" "TCG" "TCT" "ACC" "CCA" "ACG" "ACG" "TCA"
chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10
"CCG" "ACG" "ACT" "CCT" "GCA" "CCT" "CCT" "ACA" "GCG" "ATA" "TCC" "GCC"
chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr10 chr11 chr11 chr11 chr11
"GCC" "CCA" "TCA" "ACT" "TCA" "CCG" "TCA" "ATT" "TCT" "GTT" "TCT" "GCA"
chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11
```

```
"GCA" "ACA" "TTT" "CTT" "GCG" "GCG" "ACT" "TCG" "CCA" "ACG" "ACA" "TTG"
chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr11 chr12
"CTG" "ACA" "TCT" "GCG" "CCC" "ACA" "ATC" "GCA" "TTT" "ACG" "ACG" "CCA"
chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12 chr12
"TTG" "ATT" "CCG" "ACT" "CCG" "ATG" "ACG" "TTT" "GCA" "ACG" "TTA" "TTT"
chr12 chr12 chr12 chr12 chr12 chr12 chr13 chr13 chr13 chr13 chr13 chr13
"ACG" "CTT" "ACT" "ACG" "ATT" "TCA" "GCG" "ATT" "CCG" "TCG" "ACG" "ACT"
chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr13 chr14
"CTG" "TCA" "CCT" "CCT" "CCA" "ACC" "TCA" "TCT" "TCA" "TCG" "ACG" "GCG"
chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14
"CCG" "TTT" "CTA" "TCG" "GCG" "ATA" "GCG" "TCT" "ACT" "GCA" "ACG" "CCG"
chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr14 chr15 chr15
"TCC" "ACG" "CCT" "ACG" "TTG" "ACG" "ATC" "GCC" "ACG" "CTG" "CCC" "GTT"
chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15 chr15
"ACC" "ATT" "GCG" "CCG" "CCG" "GTA" "CTT" "CCC" "GCG" "GCA" "CCC" "ACG"
chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16 chr16
"ACG" "GCG" "CCG" "TCA" "GCG" "ACA" "TTC" "GCA" "CCG" "GCG" "ACG" "GCC"
chr17 chr17 chr17 chr17 chr17 chr17 chr17 chr17 chr18 chr18 chr18 chr18
"GCC" "ACG" "ATT" "ACG" "CTC" "CCG" "GTG" "CCG" "CCC" "TTA" "CCC" "GCC"
chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18 chr18
"GCG" "TCT" "TCG" "TTT" "TCA" "ACT" "GTC" "CCG" "ACT" "TCT" "ATG" "GCA"
chr18 chr18 chr19 chr19 chr19 chr19 chr19 chr19 chr19 chr19 chr20 chr20
"GCG" "ACG" "ACG" "ACG" "CCA" "GCG" "CTT" "ATA" "TTA" "GCA" "ACG" "GTA"
chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr20 chr21 chr21
"CCG" "ACA" "TTC" "ATG" "ACT" "ACG" "CTT" "ACA" "GCG" "ACG" "TCA" "CCC"
chr21 chr21 chr21 chr21 chr21 chr21 chr21 chr22 chr22 chr22 chr22
"CCT" "ACG" "TCG" "GCG" "GCA" "CTG" "CCC" "CCG" "ACG" "CCT" "ACC"
```

Using `mut_type_occurrences`, you can count mutation type occurrences for the loaded samples in a list of VCF objects:

```
> # We will use the output of this function later to make plots a little
> # bit later.
> type_occurrences <- mut_type_occurrences(vcfs, ref_genome)
> type_occurrences
```

	C>A	C>G	C>T	T>A	T>C	T>G	C>T at CpG	C>T other
colon1	77	23	263	26	77	25	147	116
colon2	68	17	259	38	84	22	144	115
colon3	68	22	261	44	73	19	156	105
intestine1	79	25	251	28	87	18	147	104
intestine2	82	24	256	34	70	18	139	117
intestine3	72	28	252	35	77	22	153	99
liver1	85	32	251	32	71	17	132	119
liver2	63	18	269	34	82	23	146	123
liver3	74	27	260	30	80	19	155	105

5.2 Mutation spectrum

A mutation spectrum shows the rate of mutation occurrences at different sites. The following functions provide insight into the mutation spectrum of your samples using plotting functions and mutation matrices.

Using the `plot_spectrum` function, you can plot the mutation spectrum over all samples. This function plots the mean relative contribution of each of the 6 base substitution types. Error bars indicate standard deviation over all samples. The n indicates the total number of mutations in the set.

```
> p1 <- plot_spectrum(type_occurrences)
```

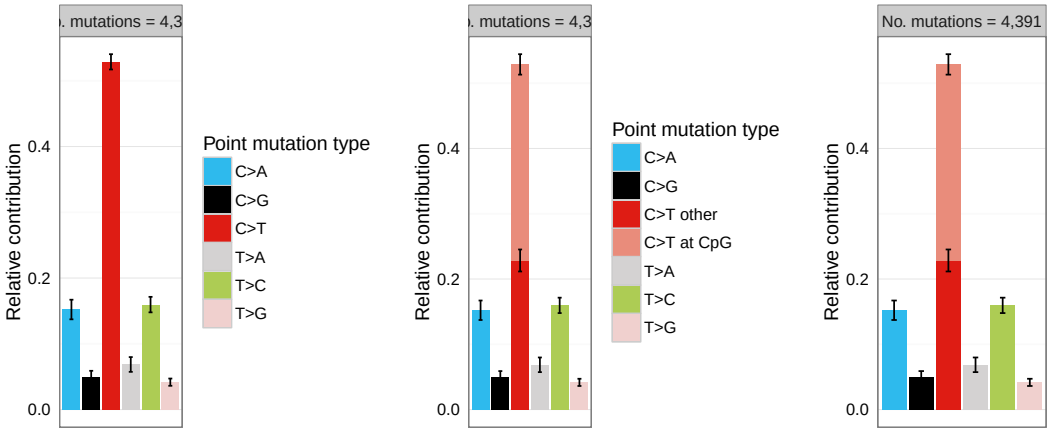
Using the same function, you can plot the mutation spectrum with distinction between C>T at CpG sites:

```
> p2 <- plot_spectrum(type_occurrences, CT = TRUE)
```

Plot spectrum without legend:

```
> p3 <- plot_spectrum(type_occurrences, CT = TRUE, legend = FALSE)
```

```
> library("gridExtra")
> grid.arrange(p1, p2, p3, ncol=3, widths=c(3,3,1.8))
```



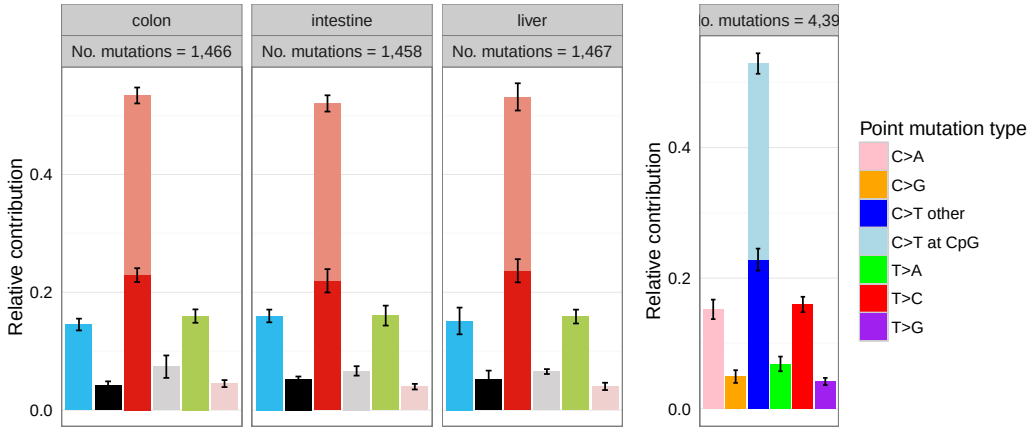
Plot spectrum for each tissue separately:

```
> tissue <- c(rep("colon", 3), rep("intestine", 3), rep("liver", 3))
> p4 <- plot_spectrum(type_occurrences, by = tissue, CT = TRUE, legend = FALSE)
```

Specify 7 colors for spectrum plotting:

```
> my_colors <- c("pink", "orange", "blue", "lightblue", "green", "red", "purple")
> p5 <- plot_spectrum(type_occurrences, CT = TRUE, legend = TRUE,
+                     colors = my_colors)
```

```
> grid.arrange(p4, p5, ncol=2, widths=c(3,2))
```



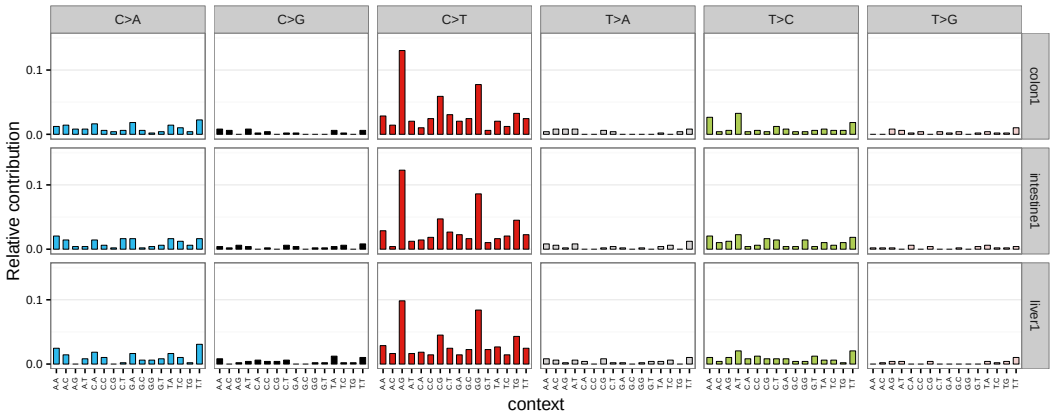
5.3 96 Mutation profile

Make 96 trinucleodide mutation count matrix:

```
> mut_mat <- mut_matrix(vcf_list = vcfs, ref_genome = ref_genome)
```

Plot 96 profile of three samples:

```
> plot_96_profile(mut_mat[,c(1,4,7)])
```



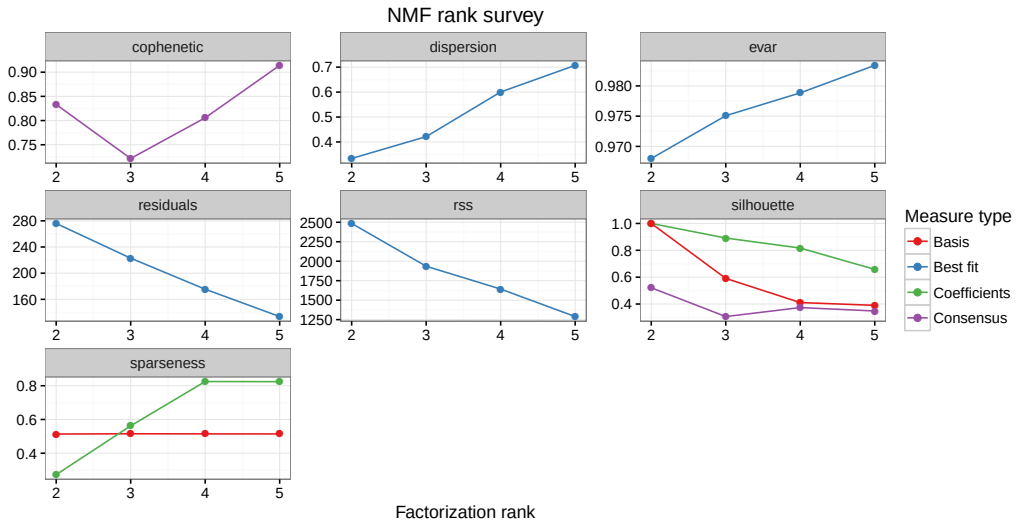
6 Mutational signatures

6.1 De novo mutational signature extraction using NMF

A critical parameter in NMF is the factorization rank, which is the number of mutational signatures. Determine the optimal factorization rank using the NMF package (Gaujoux and Seoighe, 2010). As described in their paper: "...a common way of deciding on the rank is to try different values, compute some quality measure of the results, and choose the best value according to this quality criteria. The most common approach is to choose the smallest rank for which cophenetic correlation coefficient starts decreasing. Another approach is to choose the rank for which the plot of the residual sum of squares (RSS) between the input matrix and its estimate shows an inflection point."

We can use the NMF package to determine which rank we should use to extract signatures using `extract_signatures`:

```
> # Add a tiny psuedocount to avoid a 0 in the matrix.
> mut_mat = mut_mat + 0.0001
> # Use the NMF package to generate an estimate plot.
> library("NMF")
> estimate = nmf(mut_mat, rank=2:5, method="brunet", nrun=100, seed=123456)
> plot(estimate)
```



From the estimate plot we can attempt to determine a proper rank, which we then use the extract the signatures.

```
> nmf_res <- extract_signatures(mut_mat, rank = 2)
```

Provide column names for the signatures.

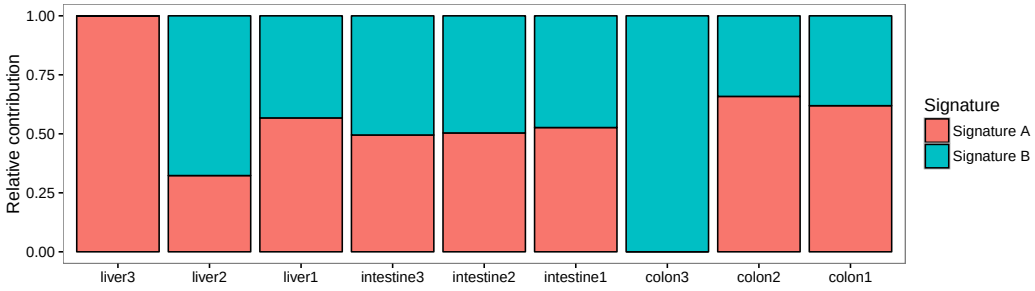
```
> colnames(nmf_res$signatures) <- c("Signature A", "Signature B")
```

Now we can plot 96-profile of the signatures:

```
> plot_96_profile(nmf_res$signatures)
```

Plot signature contribution:

```
> rownames(nmf_res$contribution) <- c("Signature A", "Signature B")
> plot_contribution(nmf_res$contribution, nmf_res$signature, mode = "relative")
```



Other contribution plot examples:

```
> # Plot absolute signature contribution
> plot_contribution(nmf_res$contribution, nmf_res$signature, mode = "absolute")
> # Plot contribution of signatures for subset of samples with index parameter
> plot_contribution(nmf_res$contribution, nmf_res$signature, mode = "absolute",
```

```
+                     index = c(1,2))
> # flip X and Y coordinates
> plot_contribution(nmf_res$contribution, nmf_res$signature, mode = "absolute",
+                     coord_flip = TRUE)
```

Compare reconstructed mutation profile with original mutation profile:

```
> plot_compare_profiles(mut_mat[,1],
+                         nmf_res$reconstructed[,1],
+                         profile_names = c("Original", "Reconstructed"))
```

6.2 Fit 96 mutation profiles to known signatures

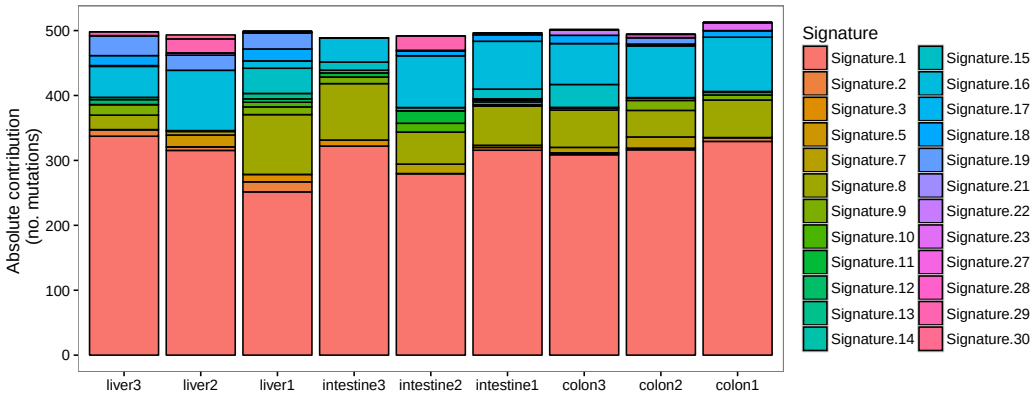
Download signatures from pan-cancer study (Alexandrov et al. 2013)

```
> sp_url <- paste("http://cancer.sanger.ac.uk/cancergenome/assets/",
+                 "signatures_probabilities.txt", sep = "")
> cancer_signatures = read.table(sp_url, sep = "\t", header = TRUE)
> # Reorder (to make the order of the trinucleotide changes the same)
> cancer_signatures = cancer_signatures[order(cancer_signatures[,1]),]
> # Only signatures in matrix
> cancer_signatures = as.matrix(cancer_signatures[,4:33])
```

Fit mutation matrix to cancer signatures. This function finds the optimal linear combination of mutation signatures that most closely reconstructs the mutation matrix by solving non-negative least-squares constraints problem.

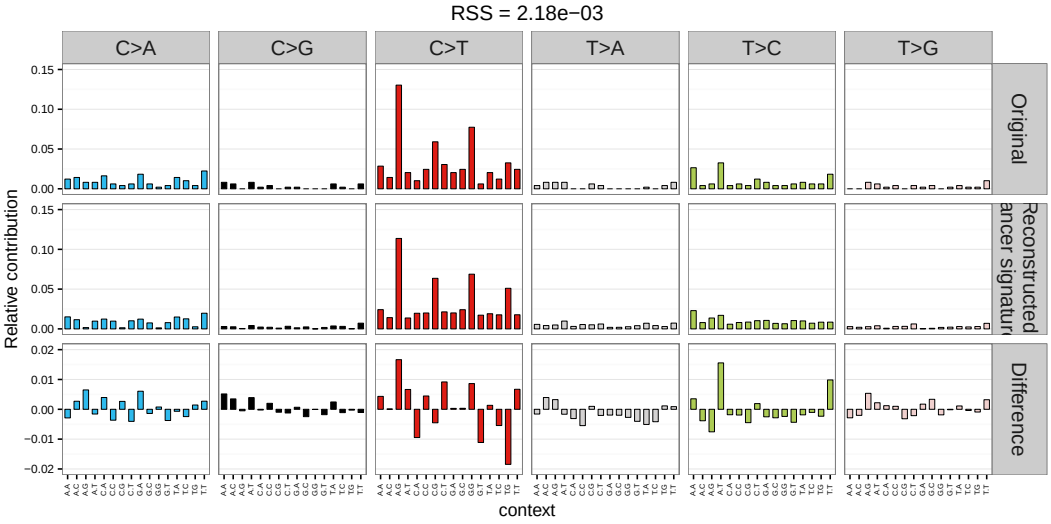
```
> fit_res <- fit_to_signatures(mut_mat, cancer_signatures)
```

```
> # select signatures with some contribution
> select <- which(rowSums(fit_res$contribution) > 0)
> # plot contribution
> plot_contribution(fit_res$contribution[select,],
+                     cancer_signatures[,select],
+                     coord_flip = FALSE,
+                     mode = "absolute")
```



Compare reconstructed mutation profile of sample 1 using cancer signatures with original profile

```
> plot_compare_profiles(mut_mat[,1], fit_res$reconstructed[,1],
+                       profile_names = c("Original",
+                           "Reconstructed \n cancer signatures"))
```



7 Transcriptional strand bias

7.1 Strand bias analysis

For the mutations within genes it can be determined whether the mutation is on the transcribed or non-transcribed strand, which is interesting to study involvement of transcription-coupled repair. To this end, it is determined whether the "C" or "T" base (since by convention we regard base substitutions as C>X or T>X) are on the same strand as the gene definition. Base substitions on the same strand as the gene definitions are considered "untranscribed", and on the opposite strand of gene bodies as transcribed, since the gene definitions report the coding or sense strand, which is untranscribed. No strand information is reported for base substitution that overlap with more than one gene body.

Find gene definitions for your reference genome.

```
> # Get ``known genes'' table from UCSC for hg19
> # biocLite("TxDb.Hsapiens.UCSC.hg19.knownGene")
> library("TxDb.Hsapiens.UCSC.hg19.knownGene")
> genes_hg19 <- genes(TxDb.Hsapiens.UCSC.hg19.knownGene)
```

Get transcriptional strand information for all positions in the first VCF object. Base substitions on the same strand as the gene definitions are considered. "-" for positions outside gene bodies, "U" for untranscribed/sense/coding strand, "T" for transcribed/anti-sense/non-coding strand.

```
> strand_from_vcf(vcfs[[1]], genes_hg19)
```

```
  [1] "-" "-" "T" "-" "-" "-" "-" "T" "-" "U" "-" "T" "-" "-" "-" "T"
 [17] "-" "T" "-" "T" "-" "U" "U" "U" "T" "U" "U" "-" "-" "-" "T" "T"
 [33] "T" "U" "T" "-" "U" "-" "-" "-" "-" "U" "T" "-" "T" "U" "-" "-"
 [49] "T" "-" "-" "U" "T" "U" "U" "T" "-" "-" "U" "U" "-" "T" "U" "-"
 [65] "-" "-" "-" "U" "-" "U" "T" "T" "-" "-" "T" "-" "T" "-" "-" "-"
 [81] "-" "-" "-" "-" "-" "-" "U" "-" "-" "-" "-" "-" "-" "-" "-" "-"
 [97] "U" "-" "U" "U" "U" "-" "-" "-" "T" "-" "U" "T" "U" "T" "T" "U"
[113] "-" "-" "-" "-" "T" "U" "-" "-" "-" "-" "U" "-" "-" "-" "-" "-"
[129] "T" "-" "-" "-" "-" "-" "-" "-" "-" "T" "-" "-" "-" "T" "T" "-"
[145] "-" "-" "-" "-" "-" "-" "-" "-" "T" "-" "-" "-" "-" "U" "-" "-"
```

```
[161] "-" "-" "U" "-" "U" "-" "-" "-" "-" "-" "-" "-" "T" "-" "-" "-"
[177] "U" "U" "-" "-" "U" "-" "T" "-" "-" "-" "-" "-" "-" "-" "T" "U"
[193] "-" "-" "-" "-" "U" "-" "-" "-" "-" "-" "T" "-" "-" "U" "-" "-"
[209] "T" "-" "U" "-" "-" "-" "T" "T" "-" "-" "U" "-" "-" "-" "-" "U"
[225] "-" "-" "-" "-" "-" "-" "-" "U" "U" "T" "T" "-" "T" "-" "U" "-"
[241] "-" "-" "-" "U" "-" "T" "-" "-" "-" "-" "T" "-" "-" "T" "-" "-"
[257] "-" "-" "U" "U" "-" "T" "U" "T" "-" "U" "-" "-" "-" "-" "T" "-"
[273] "-" "-" "T" "-" "-" "-" "-" "-" "-" "-" "-" "U" "-" "-" "T" "U"
[289] "-" "-" "-" "U" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
[305] "-" "T" "U" "-" "T" "U" "U" "T" "U" "-" "U" "-" "-" "-" "-" "-"
[321] "-" "U" "-" "-" "U" "U" "-" "-" "-" "-" "-" "-" "T" "-" "U" "T"
[337] "-" "-" "U" "-" "-" "-" "-" "-" "-" "-" "T" "T" "T" "-" "U" "U"
[353] "T" "-" "U" "U" "-" "U" "-" "T" "-" "-" "-" "U" "-" "U" "T" "U"
[369] "U" "-" "U" "-" "-" "-" "-" "-" "-" "T" "-" "T" "-" "T" "-" "-"
[385] "-" "-" "-" "-" "-" "-" "-" "-" "T" "-" "U" "-" "T" "-" "-" "-"
[401] "-" "T" "T" "T" "-" "-" "-" "-" "-" "-" "-" "-" "T" "T" "-" "U"
[417] "-" "-" "-" "-" "U" "-" "-" "-" "-" "-" "-" "U" "T" "U" "-" "-"
[433] "-" "-" "T" "U" "-" "T" "-" "U" "-" "-" "-" "-" "T" "U" "U" "U"
[449] "-" "T" "T" "U" "T" "T" "T" "-" "-" "U" "U" "U" "-" "-" "T" "T"
[465] "T" "-" "U" "-" "-" "-" "U" "-" "-" "-" "T" "T" "-" "-" "-" "-"
[481] "-" "-" "-" "T" "-" "-" "-" "U" "-" "-" "T"
```

Make mutation count matrix with transcriptional strand information (96 trinucleotides * 2 strands = 192 features). NB: only those mutations that are located within gene bodies are counted.

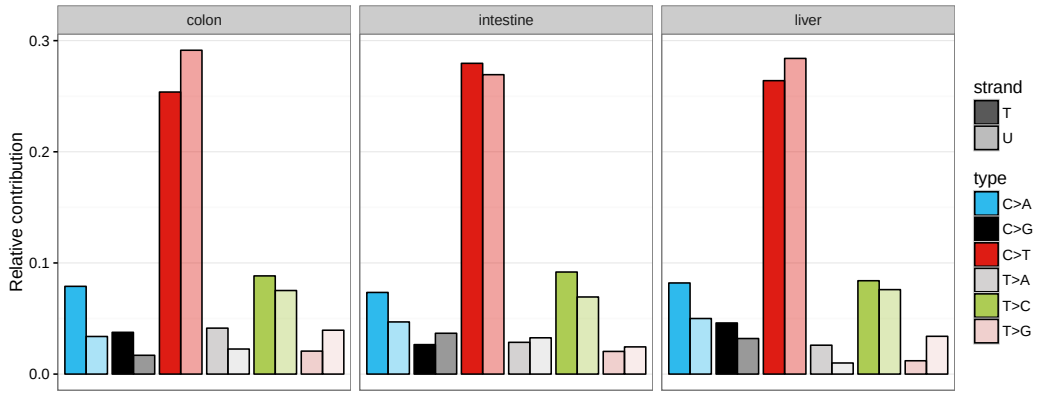
```
> mut_mat_s <- mut_matrix_stranded(vcfs, ref_genome, genes_hg19)
> head(mut_mat_s, 10)
```

```
      colon1 colon2 colon3 intestine1 intestine2 intestine3 liver1
ACA-u      0      1      1          1          1          0      3
ACA-t      2      0      2          1          3          2      2
ACC-u      1      1      1          1          1          1      0
ACC-t      1      1      1          2          0          2      0
ACG-u      1      0      0          0          0          0      0
ACG-t      0      1      0          1          0          0      0
ACT-u      0      0      0          0          1          0      0
ACT-t      1      1      0          1          1          0      0
CCA-u      1      0      1          0          0          0      1
CCA-t      2      1      4          3          2          1      1
      liver2 liver3
ACA-u      1      0
ACA-t      2      3
ACC-u      1      1
ACC-t      2      1
ACG-u      0      0
ACG-t      1      0
ACT-u      0      0
ACT-t      0      2
CCA-u      0      0
CCA-t      0      1
```

Perform strand bias analysis:

```
> strand_counts <- strand_occurrences(mut_mat_s, by=tissue)
```

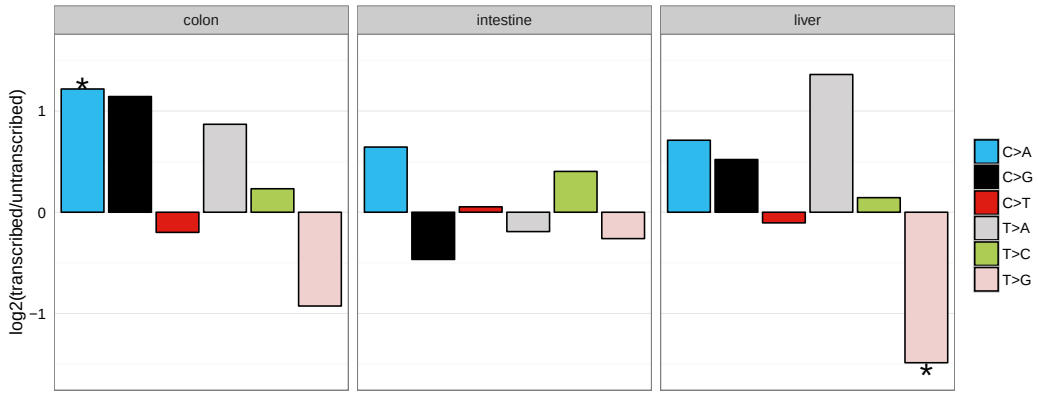
```
> plot_strand(strand_counts, mode = "relative")
```



Perform poisson test for strand asymmetry significance testing:

```
> strand_bias <- strand_bias_test(strand_counts)
```

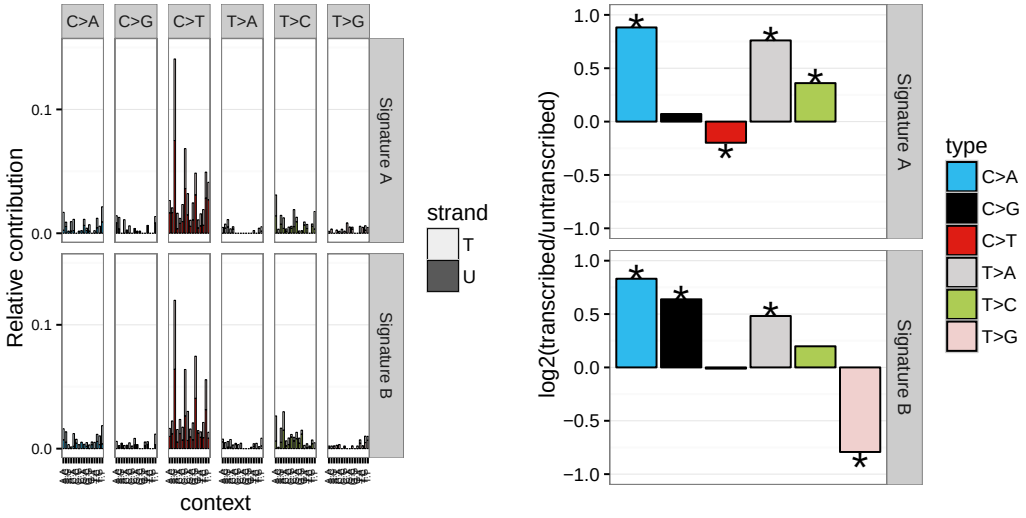
```
> plot_strand_bias(strand_bias)
```



7.2 Extract signatures with strand bias

```
> # Extract 2 signatures
> nmf_res_strand <- extract_signatures(mut_mat_s, rank = 2)
> # Provide signature names
> colnames(nmf_res_strand$signatures) <- c("Signature A", "Signature B")
```

```
> # Plot signatures with 192 features
> a <- plot_192_profile(nmf_res_strand$signatures)
> # Plot strand bias per mutation type for each signature with significance test
> b <- plot_signature_strand_bias(nmf_res_strand$signatures)
> # Load the gridExtra library for combining plots.
> grid.arrange(a, b, ncol=2, widths=c(1,1))
```



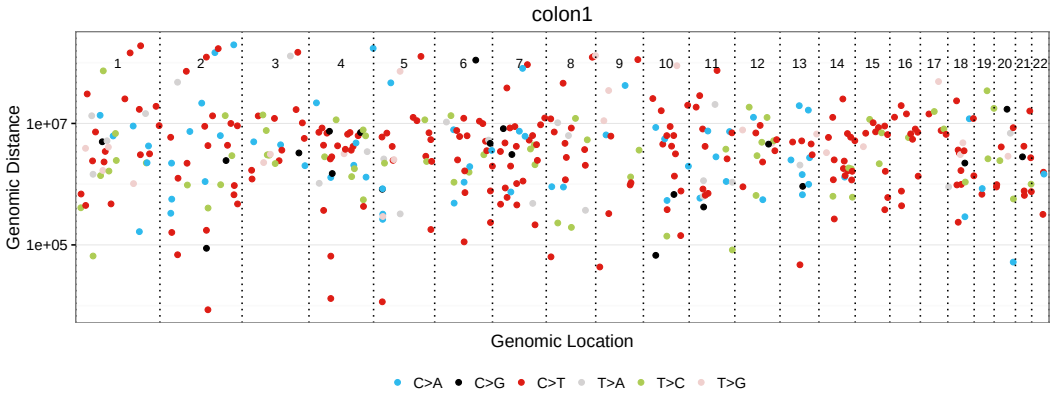
8 Genomic distribution

8.1 Rainfall plot

A rainfall plot visualizes mutation types and intermutation distance. Rainfall plots can be used to visualize the distribution of mutations along the genome or a subset of chromosomes. The y-axis corresponds to the distance of a mutation with the previous mutation and is log10 transformed. Drop-downs from the plots indicate clusters or "hotspots" of mutations.

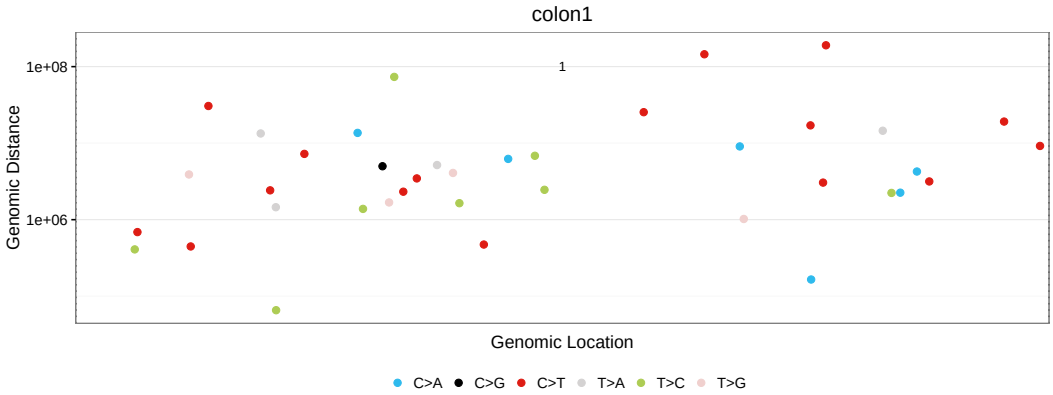
Make rainfall plot of sample 1 over all autosomal chromosomes

```
> # Define autosomal chromosomes
> chromosomes <- seqnames(get(ref_genome))[1:22]
> # Make a rainfall plot
> plot_rainfall(vcfs[[1]], title = names(vcfs[1]),
+                   chromosomes = chromosomes, cex = 1.5)
```



Make rainfall plot of the first sample over chromosome 1:

```
> chromosomes <- seqnames(get(ref_genome))[1]
> plot_rainfall(vcfs[[1]], title = names(vcfs[1]),
+                   chromosomes = chromosomes[1], cex = 2)
```



8.2 Enrichment or depletion of mutations in genomic regions

Test for enrichment or depletion of mutations in certain genomic regions, such as promoters, CTCF binding sites and transcription factor binding sites. To use your own genomic region definitions (based on e.g. ChipSeq experiments) specify your genomic regions in a named list of GRanges objects. Alternatively, use publically available genomic annotation data, like in the example below.

8.2.1 Example: regulation annotation data from Ensembl using biomaRt

The following example displays how to download regulation annocation data for genome build hg19. For other datasets, see the *biomaRt* documentation (Durinck et al. 2005).

Load biomaRt package

```
> ## In case you hadn't installed biomaRt yet, you should comment out
> ## the following lines:
> # source("https://bioconductor.org/biocLite.R")
> # biocLite("biomaRt")
>
> # Load the biomaRt package.
> library(biomaRt)
```

Download data from Ensembl using biomaRt

```
> ## We can query the BioMart database, but this may take a long time
> ## though, so we take some shortcuts by loading the results from our
> ## examples.  The corresponding code for downloading data can be
> ## found above the command we run.
>
> # regulatory <- useEnsembl(biomart="regulation",
> #                          dataset="hsapiens_regulatory_feature",
> #                          GRCh = 37)
>
> ## Download the regulatory CTCF binding sites and convert them to
> ## a GRanges object.
```

```
> # CTCF <- getBM(attributes = c('chromosome_name',
> #                              'chromosome_start',
> #                              'chromosome_end',
> #                              'feature_type_name',
> #                              'cell_type_name'),
> #               filters = "regulatory_feature_type_name",
> #               values = "CTCF Binding Site",
> #               mart = regulatory)
> #
> # CTCF_g <- reduce(GRanges(CTCF$chromosome_name,
> #                  IRanges(CTCF$chromosome_start,
> #                  CTCF$chromosome_end)))
> CTCF_g <- readRDS(system.file("states/CTCF_g_data.rds",
+                     package="MutationalPatterns"))
> ## Download the promoter regions and conver them to a GRanges object.
> # promoter = getBM(attributes = c('chromosome_name', 'chromosome_start',
> #                                 'chromosome_end', 'feature_type_name'),
> #                  filters = "regulatory_feature_type_name",
> #                  values = "Promoter",
> #                  mart = regulatory)
> # promoter_g = reduce(GRanges(promoter$chromosome_name,
> #                     IRanges(promoter$chromosome_start,
> #                             promoter$chromosome_end)))
> promoter_g <- readRDS(system.file("states/promoter_g_data.rds",
+                       package="MutationalPatterns"))
> # flanking = getBM(attributes = c('chromosome_name',
> #                                 'chromosome_start',
> #                                 'chromosome_end',
> #                                 'feature_type_name'),
> #                  filters = "regulatory_feature_type_name",
> #                  values = "Promoter Flanking Region",
> #                  mart = regulatory)
> # flanking_g = reduce(GRanges(
> #                       flanking$chromosome_name,
> #                       IRanges(flanking$chromosome_start,
> #                       flanking$chromosome_end)))
>
> flanking_g <- readRDS(system.file("states/promoter_flanking_g_data.rds",
+                                   package="MutationalPatterns"))
>
```

Combine all genomic regions (GRanges objects) in a named list.

```
> regions <- GRangesList(promoter_g, flanking_g, CTCF_g)
> names(regions) <- c("Promoter", "Promoter flanking", "CTCF")
```

Don't forget to use the same naming standard consistently:

```
> seqlevelsStyle(regions) <- "UCSC"
```

8.3 Test for significant depletion or enrichment in genomic regions

It is necessary to include a list with Granges of regions that were surveyed in your analysis for each sample, that is: positions in the genome at which you have enough high quality reads to call a mutation. This can for example be determined using CallableLoci tool by GATK. If you would not include the surveyed area in your analysis, you might for

example see a depletion of mutations in a certain genomic region that is solely a result from a low coverage in that region, and therefore does not represent an actual depletion of mutations.

```
> ## Get the filename with surveyed/callable regions
> surveyed_file <- list.files(system.file("extdata",
+                             package = "MutationalPatterns"),
+                             pattern = ".bed",
+                             full.names = TRUE)
> ## Import the file using rtracklayer and use the UCSC naming standard
> library(rtracklayer)
> surveyed <- import(surveyed_file)
> seqlevelsStyle(surveyed) <- "UCSC"
> ## For this example we use the same surveyed file for each sample.
> surveyed_list <- rep(list(surveyed), 9)
```

Test for enrichment or depletion of mutations in your defined genomic regions using a binomial test. For this test, the chance of observing a mutation is calculated as the total number of mutations, divided by the total number of surveyed bases.

```
> ## Calculate the number of observed and expected number of mutations in
> ## each genomic regions for each sample.
> distr <- genomic_distribution(vcfs, surveyed_list, regions)
```

```
> ## Perform the enrichment/depletion test by tissue type.
> distr_test <- enrichment_depletion_test(distr, by = tissue)
> ## Or without specifying the 'by' parameter.
> distr_test2 <- enrichment_depletion_test(distr)
```

```
> plot_enrichment_depletion(distr_test)
TableGrob (2 x 1) "arrange": 2 grobs
  z     cells    name           grob
1 1 (1-1,1-1) arrange gtable[layout]
2 2 (2-2,1-1) arrange gtable[layout]
```

9 Session Information

```
R version 3.3.1 (2016-06-21)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.9.5 (Mavericks)

locale:
```

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

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

other attached packages:
 [1] biomaRt_2.30.0
 [2] TxDb.Hsapiens.UCSC.hg19.knownGene_3.2.2
 [3] GenomicFeatures_1.26.0
 [4] AnnotationDbi_1.36.0
 [5] doParallel_1.0.10
 [6] iterators_1.0.8
 [7] foreach_1.4.3
 [8] gridExtra_2.2.1
 [9] MutationalPatterns_1.0.0
[10] NMF_0.20.6
[11] bigmemory_4.5.19
[12] bigmemory.sri_0.1.3
[13] Biobase_2.34.0
[14] cluster_2.0.5
[15] rngtools_1.2.4
[16] pkgmaker_0.22
[17] registry_0.3
[18] BSgenome.Hsapiens.UCSC.hg19_1.4.0
[19] BSgenome_1.42.0
[20] rtracklayer_1.34.0
[21] Biostrings_2.42.0
[22] XVector_0.14.0
[23] GenomicRanges_1.26.0
[24] GenomeInfoDb_1.10.0
[25] IRanges_2.8.0
[26] S4Vectors_0.12.0
[27] BiocGenerics_0.20.0

loaded via a namespace (and not attached):
 [1] SummarizedExperiment_1.4.0 VariantAnnotation_1.20.0
 [3] reshape2_1.4.1             lattice_0.20-34
 [5] colorspace_1.2-7           XML_3.98-1.4
 [7] pracma_1.9.5               DBI_0.5-1
 [9] BiocParallel_1.8.0         RColorBrewer_1.1-2
[11] plyr_1.8.4                 stringr_1.1.0
[13] zlibbioc_1.20.0            munsell_0.4.3
[15] gtable_0.2.0               codetools_0.2-15
[17] labeling_0.3               BiocInstaller_1.24.0
[19] Rcpp_0.12.7                xtable_1.8-2
[21] scales_0.4.0               Rsamtools_1.26.0
[23] BiocStyle_2.2.0            ggplot2_2.1.0
[25] digest_0.6.10              stringi_1.1.2
[27] grid_3.3.1                 quadprog_1.5-5
[29] tools_3.3.1                bitops_1.0-6
[31] magrittr_1.5               RCurl_1.95-4.8
[33] RSQLite_1.0.0              Matrix_1.2-7.1
[35] gridBase_0.4-7             compiler_3.3.1
```

```
[37] GenomicAlignments_1.10.0
```