Package **CompSign**

Lena Morrill

2022

CompSign is a toolkit for differential abundance analysis of mutational signatures using a mixed effects Dirichlet-multinominal model (or simpler variations). The compositional nature of mutational signature exposures has often been overlooked but has important implications, as the analyses must be done in relative terms.

Contents

1 Installation

`CompSign` can be installed as usual from github:

```
library(devtools)
devtools::install_github("lm687/CompSign")
```

```
library(CompSign)

## Loading required package:  TMB
## Warning in checkMatrixPackageVersion():  Package version inconsistency detected.
## TMB was built with Matrix version 1.4.1
## Current Matrix version is 1.5.1
## Please re-install 'TMB' from source using install.packages('TMB', type = 'source') or ask
CRAN for a binary version of 'TMB' matching CRAN's 'Matrix' package
## Loading required package:  RcppEigen

library(gridExtra)
library(TMB)
# setwd(dirname(rstudioapi::getSourceEditorContext()£path))
```

2 Datasets

Two datasets can be loaded as follows using the ‘data()‘ function:

```
data(PancEndocrine_signaturesMSE) ## it doesn't work
data(ProstAdenoCA_chrom) ## it works
```

The package contains the following datasets of exposures of mutational signatures and metadata of the corresponding samples. These datasets are:

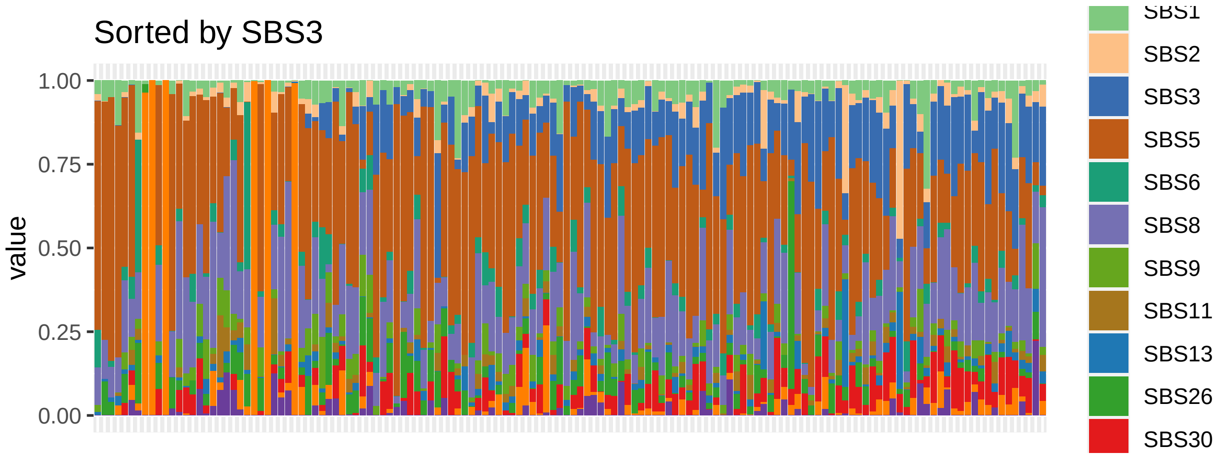
- `PancEndocrine_signaturesMSE`: Signature exposures for early and late mutations, in the PCAWG Panc-Endocrine cohort
- `ProstAdenoCA_chrom`: Signature exposures for each chromosome, in the PCAWG Prost-AdenoCA cohort

`PancEndocrine_signaturesMSE` is an object of class `sign`

```
PancEndocrine_signaturesMSE = load_PCAWG("../inst/extdata/roo/Panc-Endocrine_signaturesMSE_ROO.RDS",
                                        read_directly = T,
                                        typedata = "signaturesMSE", override_warning_X_Z = T)
PancEndocrine_signaturesMSE_v2 = load_PCAWG(ct = "Panc-Endocrine", typedata = "signaturesMSE", path_to_da
```

All samples - clonal and subclonal - sorted by increasing SBS3 exposure:

```
createBarplot(normalise_rw(non_duplicated_rows(PancEndocrine_signaturesMSE$Y)),
              order_labels = names(sort(non_duplicated_rows(PancEndocrine_signaturesMSE$Y)[,'SBS3'],
                                        decreasing = F)), remove_labels=T)+ggtitle('Sorted by SBS3')
```



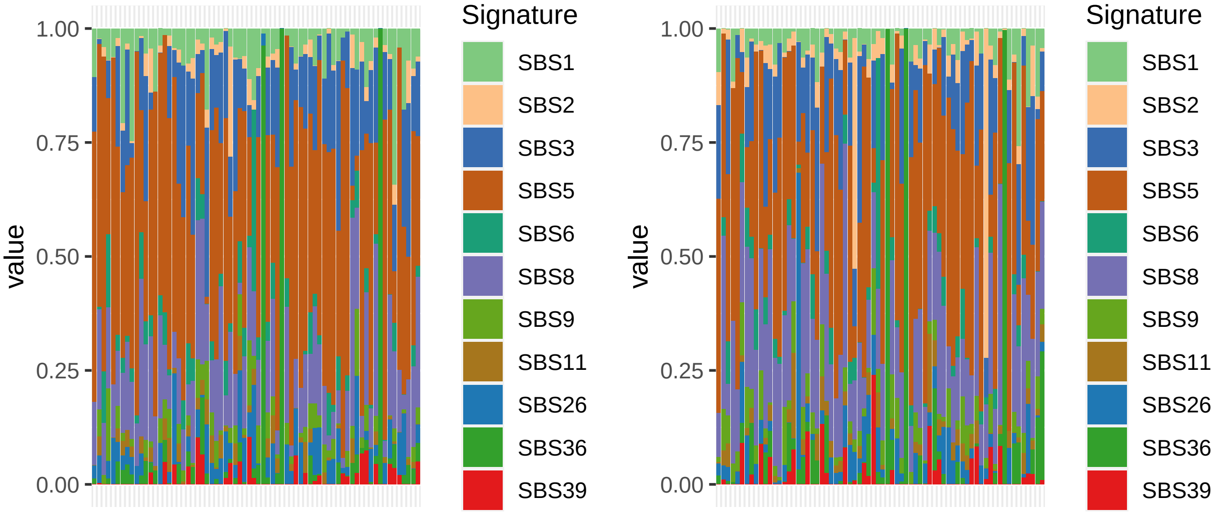
We create a simplified object containing exposures of fewer signatures (i.e. a subcomposition of the original signature vectors):

```
simplified_object <- give_subset_sigs_TMBobj(PancEndocrine_signaturesMSE,
                                            sigs_to_remove = c('SBS13', 'SBS17a', 'SBS17b', 'SBS30'))
```

The clonal and subclonal exposures are, respectively, the two barplots below:

```
do.call('grid.arrange', list(grobs=lapply(split_matrix_in_half(simplified_object$Y), function(i) createBa

## Creating plot... it might take some time if the data are large. Number of samples: 70
## Creating plot... it might take some time if the data are large. Number of samples: 70
```



3 Running the model for differential abundance

Running the model `diagREDMsinglelambda` with the dataset `simplified_object`.

```
diagDM_no_small_sigs <- wrapper_run_TMB(object = simplified_object,
                                        model = "diagREDMsinglelambda", use_nlminb=T, smart_init_vals=F)
```

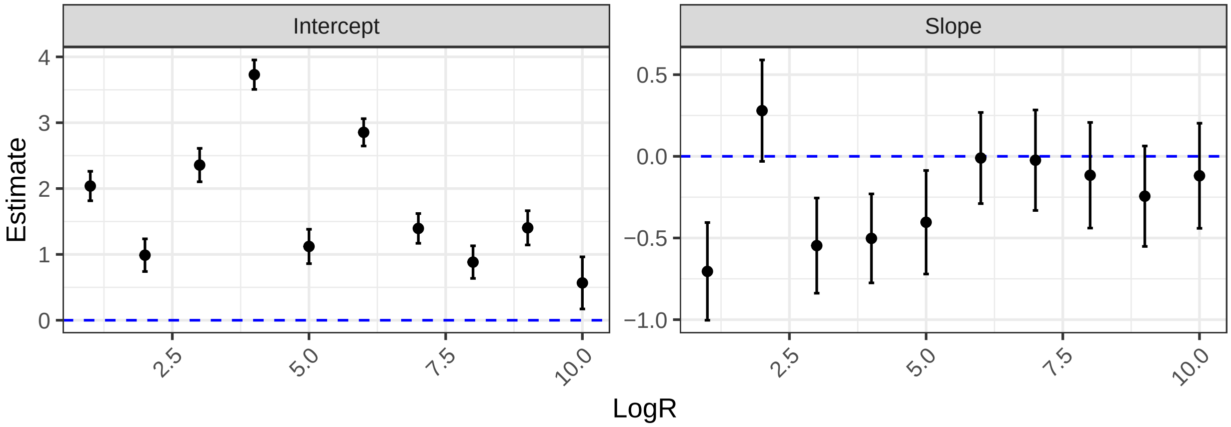
These are the resulting object with the estimates and their standard deviations

```
diagDM_no_small_sigs

## sdreport(.) result
##                Estimate   Std. Error
## beta         2.03816989   0.22364979
## beta        -0.70462291   0.29906778
## beta         0.98802508   0.24807864
## beta         0.27961170   0.31041508
## beta         2.35678443   0.25394114
## beta        -0.54670864   0.29139794
## beta         3.72930980   0.22354545
## beta        -0.50256634   0.27244238
## beta         1.12061131   0.26120736
## beta        -0.40374986   0.31695130
## beta         2.85345624   0.20725587
## beta        -0.01040204   0.27904077
## beta         1.39443550   0.22601419
## beta        -0.02376817   0.30764840
## beta          0.88271641   0.24739078
## beta         -0.11595268   0.32338625
## beta          1.40319938   0.26010658
## beta         -0.24423079   0.30749239
## beta          0.56678869   0.39570330
## beta         -0.11912460   0.32177469
## logs_sd_RE  -1.47024544   0.45923552
## logs_sd_RE  -0.88130213   0.33984452
## logs_sd_RE   0.16819836   0.25484909
## logs_sd_RE  -0.49517372   0.23244385
## logs_sd_RE  -0.19676508   0.30654989
## logs_sd_RE -11.11528206 295.68420500
## logs_sd_RE -11.28655610 211.05505047
## logs_sd_RE  -1.26141845   0.53837408
## logs_sd_RE   0.03024120   0.26234776
## logs_sd_RE   1.78115732   0.22930526
## log_lambda   2.85825168   0.05956044
## Maximum gradient component: 0.01517549
```

These are the betas for this model

```
plot_betas(diagDM_no_small_sigs)
```



and the p-value indicating differential abundance

```
wald_TMB_wrapper(diagDM_no_small_sigs)

##              [,1]
## [1,] 2.842257e-07
```

3.1 Other models

Other models can be run as follows:

```
res <- wrapper_run_TMB(object = simplified_object,
                       model = "diagREDMsinglelambda", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "diagRE_DM", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "diagRE_M", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "FEDMsinglelambda", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "FE_DM", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "fullREDMsinglelambda", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "fullRE_DMonefixedlambda", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "fullRE_DM", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "fullRE_M", use_nlminb=T, smart_init_vals=F)
res <- wrapper_run_TMB(object = simplified_object,
                       model = "singleRE_DM", use_nlminb=T, smart_init_vals=F)
```