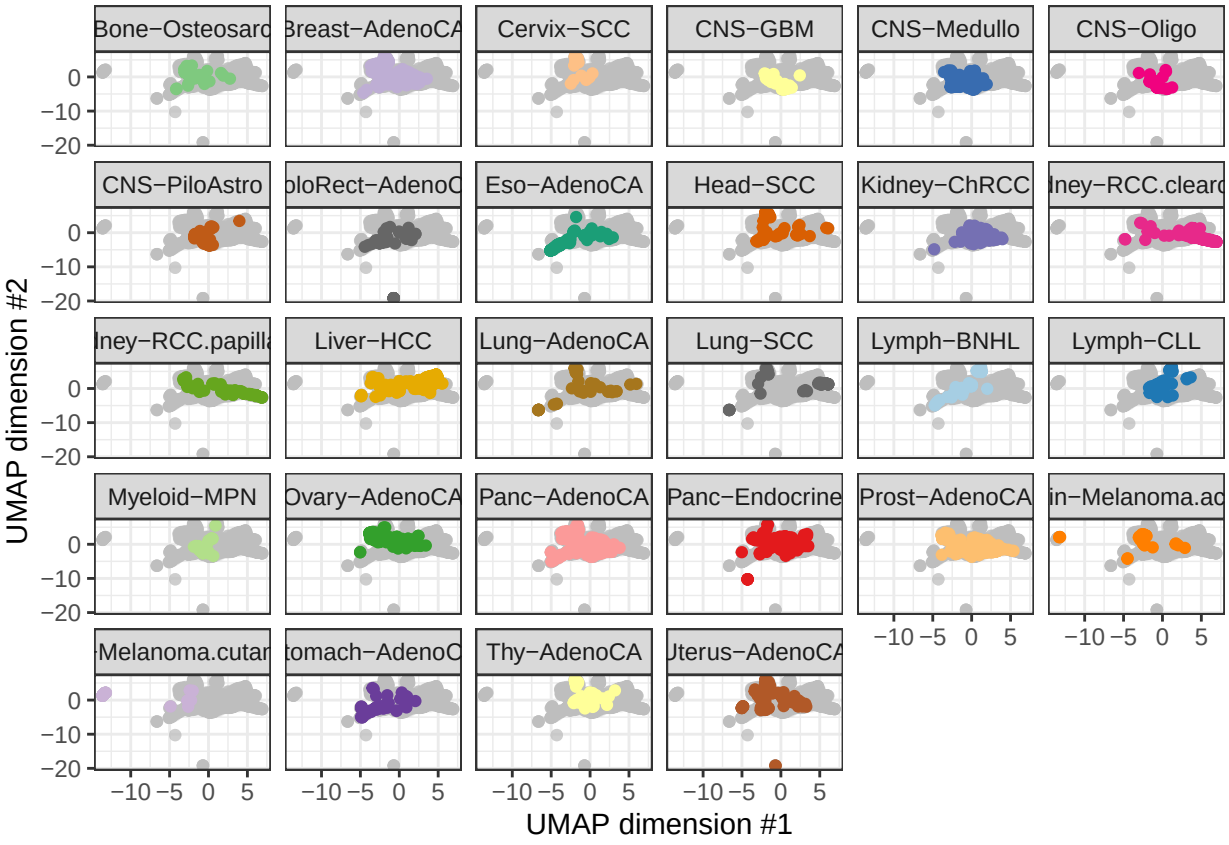
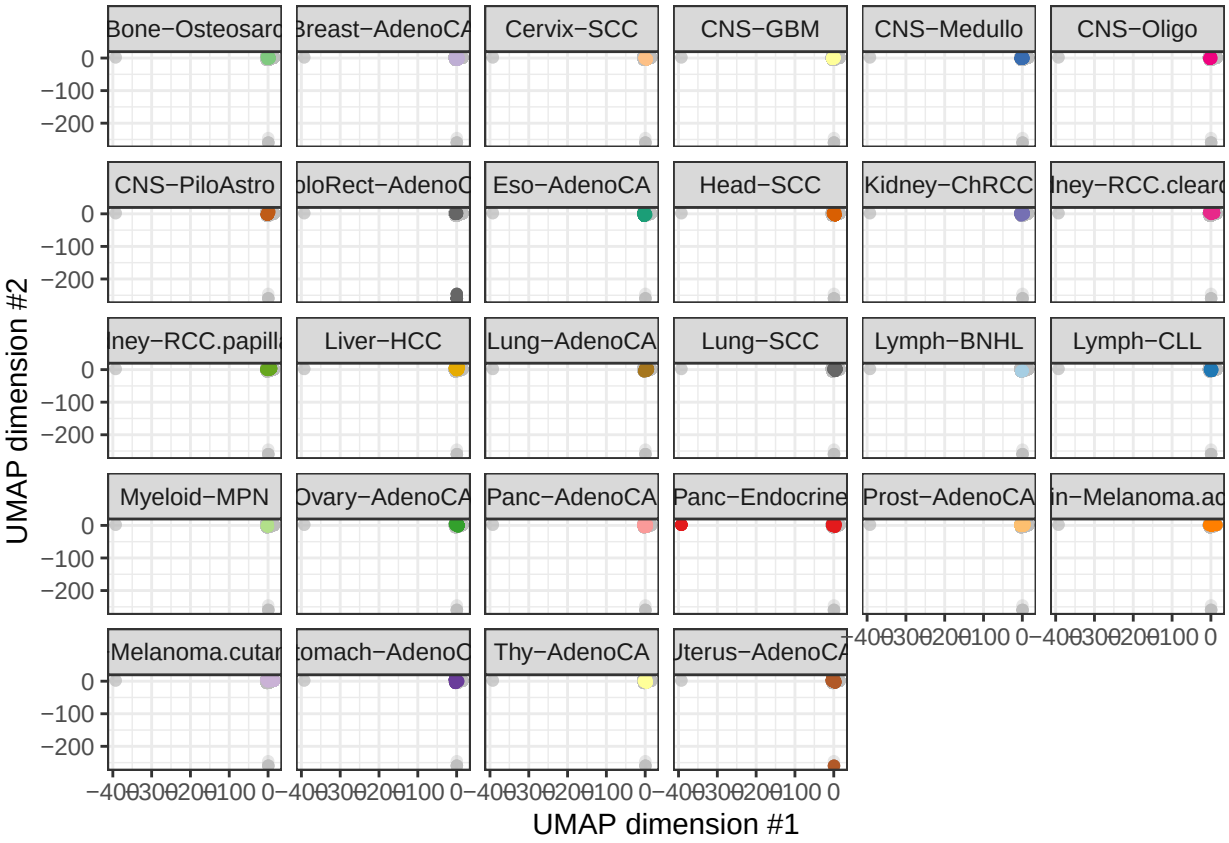
Summary of TMB runs

Lena Morrill

24/05/2021

```
## Loading required package: viridisLite

## Loading required package: coda

## Loading required package: MASS

## Warning in .recacheSubclasses(def@className, def, env): undefined subclass
## "numericVector" of class "Mnumeric"; definition not updated

## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)

## ## Copyright (C) 2003-2021 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park

## ##
## ## Support provided by the U.S. National Science Foundation

## ## (Grants SES-0350646 and SES-0350613)
## ##

## Error in slot(i, "count_matrices_all") :
##   cannot get a slot ("count_matrices_all") from an object of type "logical"

## Error in slot(i, "count_matrices_all") :
##   cannot get a slot ("count_matrices_all") from an object of type "logical"
```


Bone−Osteosarc
Breast−AdenoCA
Cervix−SCC
CNS−GBM
CNS−Medullo
CNS−Oligo
CNS−PiloAstro
oloRect−AdenoC
Eso−AdenoCA
Head−SCC
Kidney−ChRCC
ney−RCC.clearc
ney−RCC.papill
Liver−HCC
Lung−AdenoCA
Lung−SCC
Lymph−BNHL
Lymph−CLL
Myeloid−MPN
Ovary−AdenoCA
Panc−AdenoCA
Panc−Endocrine
Prost−AdenoCA
in−Melanoma.ac
Melanoma.cutar
tomach−AdenoC
Thy−AdenoCA
Uterus−AdenoCA
UMAP dimension #2
UMAP dimension #1
0
−10
−20
−10 −5 0 5


Bone−Osteosarc
Breast−AdenoCA
Cervix−SCC
CNS−GBM
CNS−Medullo
CNS−Oligo
CNS−PiloAstro
loRect−AdenoC
Eso−AdenoCA
Head−SCC
Kidney−ChRCC
ney−RCC.clear
ney−RCC.papill
Liver−HCC
Lung−AdenoCA
Lung−SCC
Lymph−BNHL
Lymph−CLL
Myeloid−MPN
Ovary−AdenoCA
Panc−AdenoCA
Panc−Endocrine
Prost−AdenoCA
in−Melanoma.ac
Melanoma.cutan
tomach−AdenoC
Thy−AdenoCA
Uterus−AdenoCA
UMAP dimension #2
UMAP dimension #1

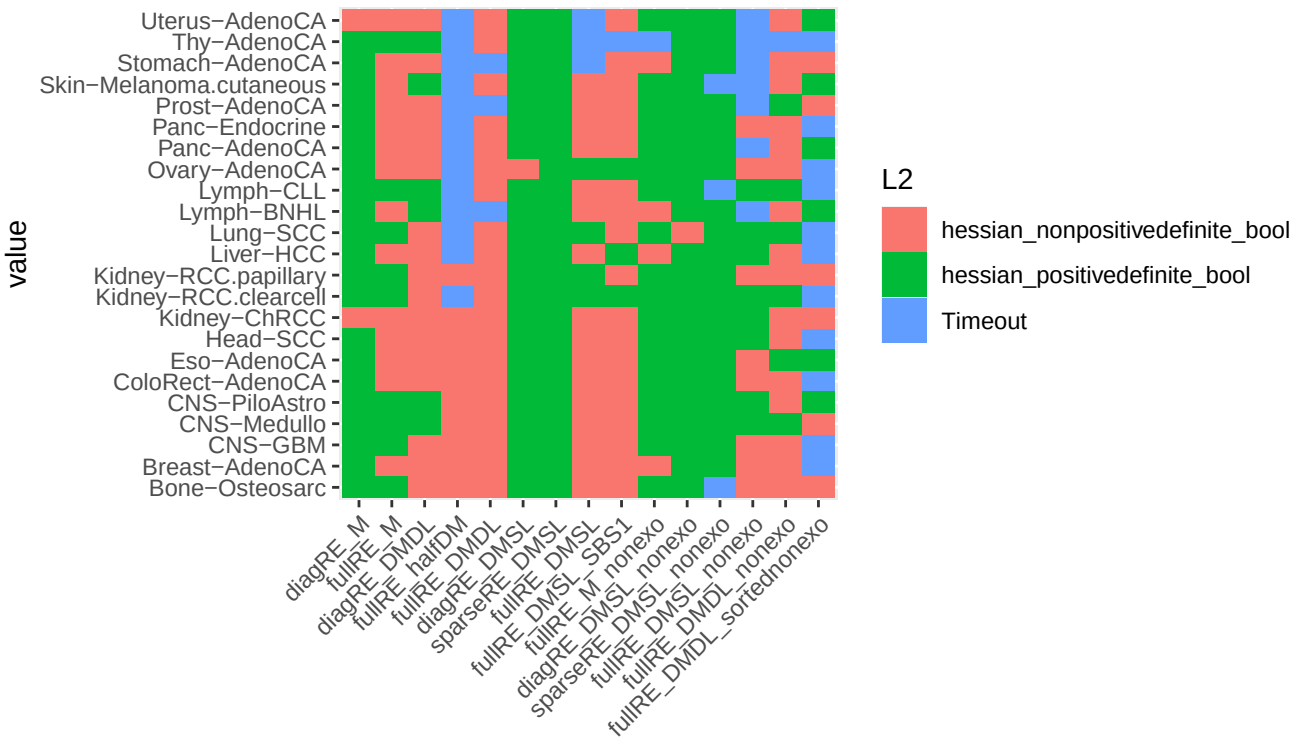
Information about models

Default order of categories for each model

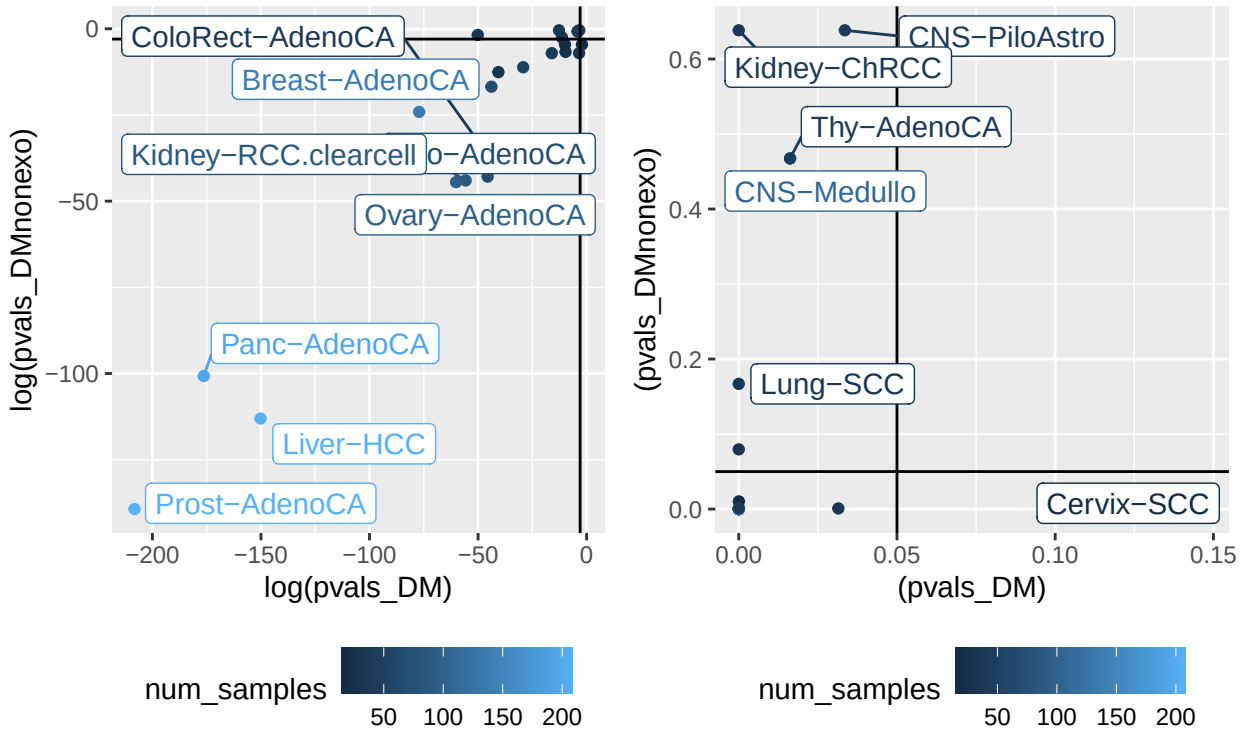
Name model	Extension	Sorted	File in which th
fullREDMsinglelambda	fullRE_DMSL_	Not sorted	run_TMB_PCAWG
fullREDMsinglelambda2	fullRE_DMSL2_	Sorted	run_TMB_PCAWG
diagREDMsinglelambda	diagRE_DMSL_	Unknown	run_TMB_PCAWG
fullRE_M	fullRE_M_	Sorted in previous version of wrapper_run_TMB	run_TMB_PCAWG
diagRE_DM	diagRE_DM_	Sorted in previous version of wrapper_run_TMB	run_TMB_PCAWG
fullRE_DM	fullRE_DM_	Sorted in previous version of wrapper_run_TMB	run_TMB_PCAWG
sparseRE_DMSL2	sparseRE_nonexo_DMSL_	Sorted	find_subset_s
fullREDMsinglelambda	fullRE_nonexo_DMSL_	Not sorted	find_subset_s
fullRE_M	fullRE_nonexo_M_	Not sorted	find_subset_s
diagREDMsinglelambda	diagRE_nonexo_DMSL_	Not sorted	find_subset_s
fullRE_DM	fulLRE_nonexo_DM_	Not sorted	find_subset_s
diagREDMsinglelambda	diagRE_DMSL_	Not sorted	find_subset_s
fullREDMsinglelambda for SP	fullREDMsinglelambda_..._signaturesPCAWG	Not sorted (it was commented out)	run_TMB_PCAWG
fullREM for SP	fullREM_..._signaturesPCAWG	Not sorted (it was commented out)	run_TMB_PCAWG

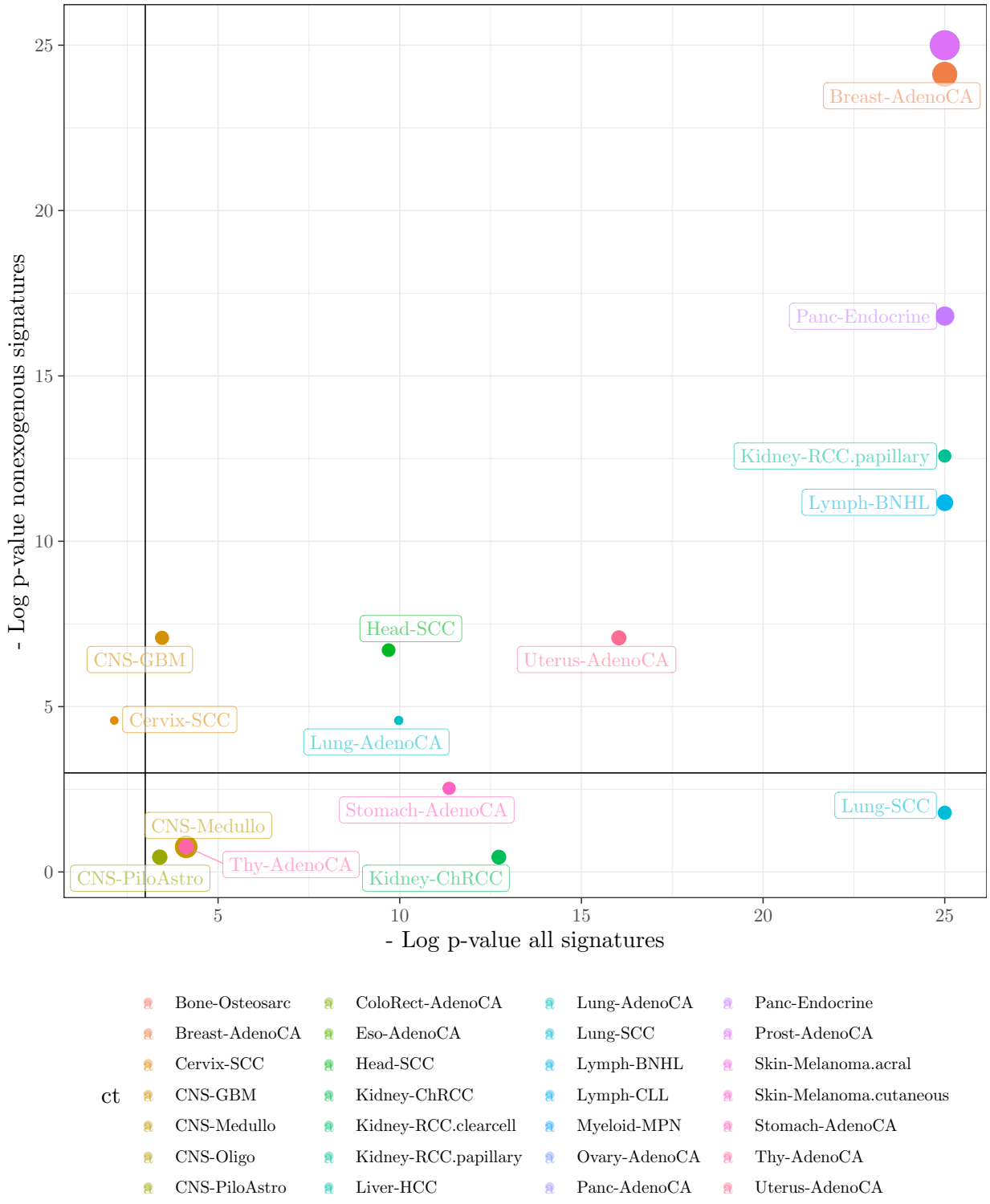
General results of all models

Check the results of all of the models



P-values for all cancer types

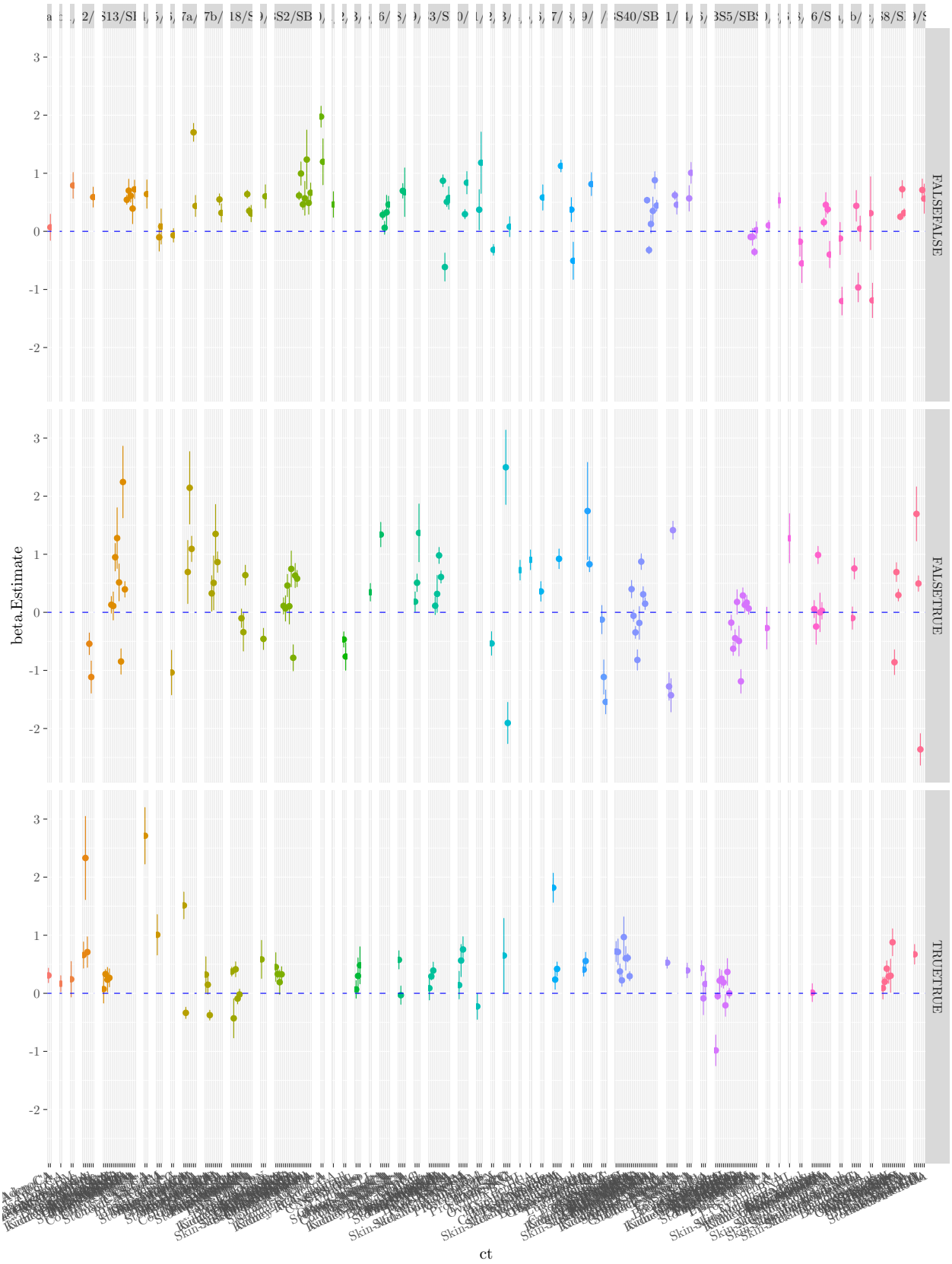


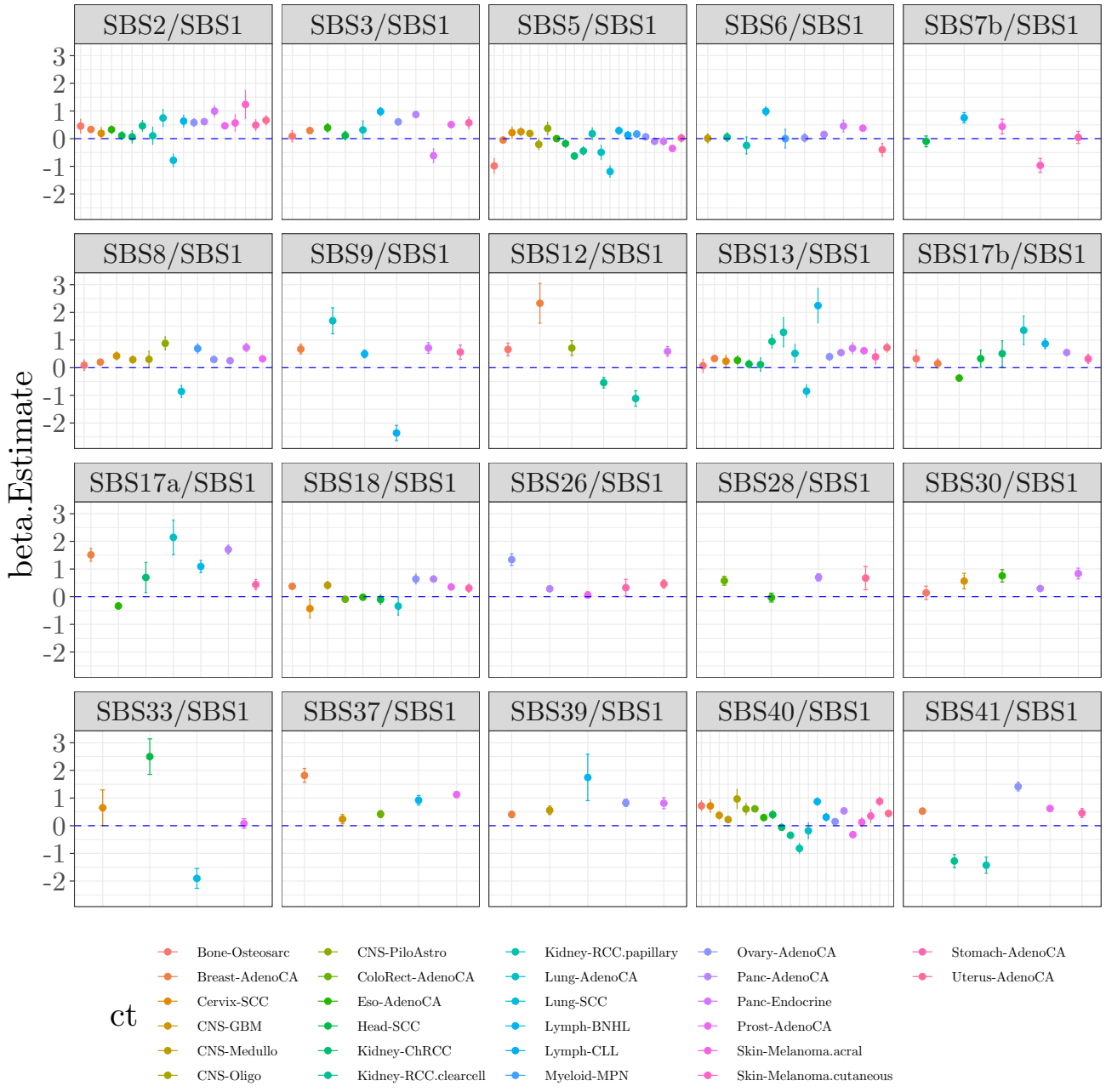


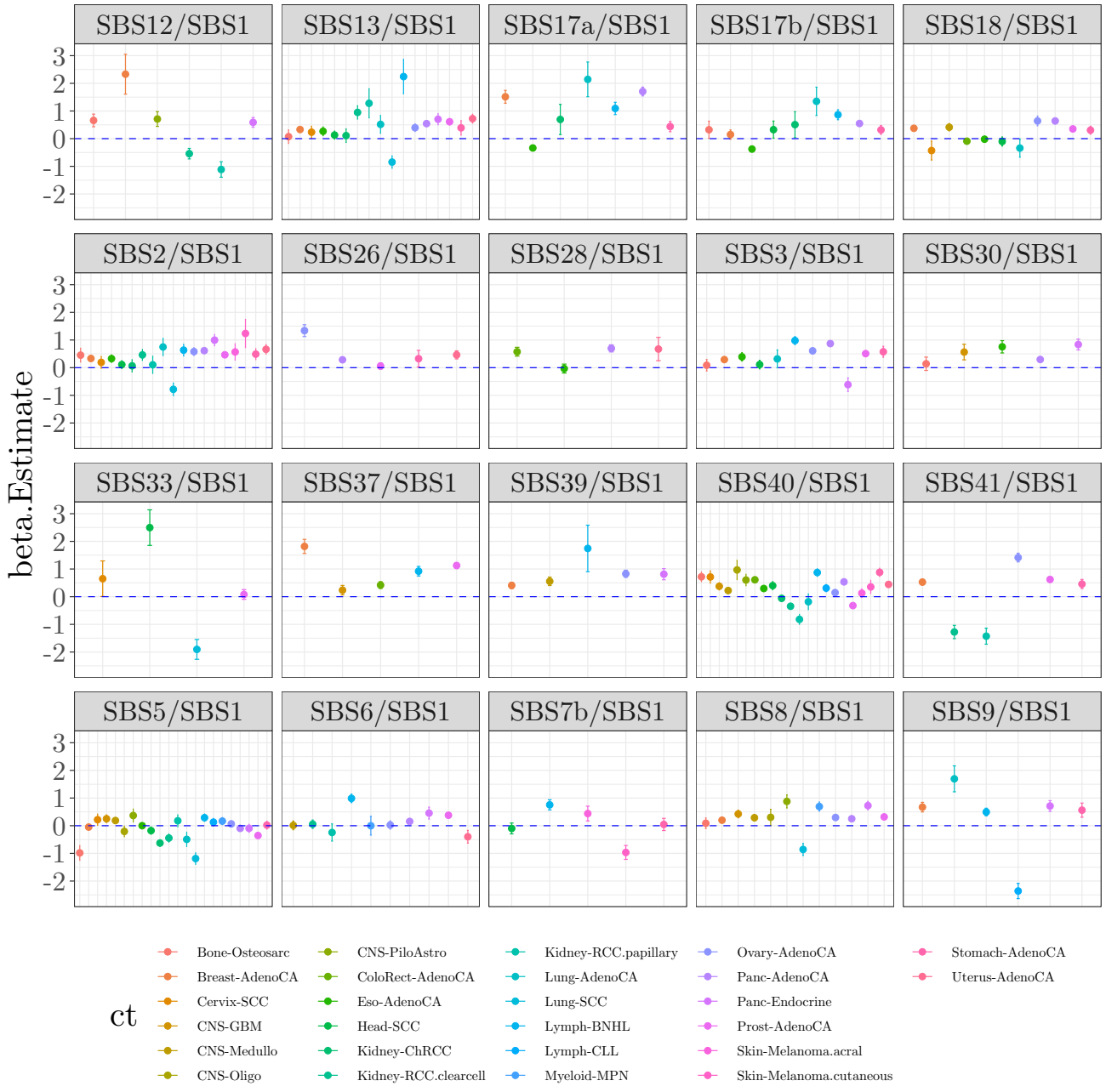
All betas with SBS1 as baseline

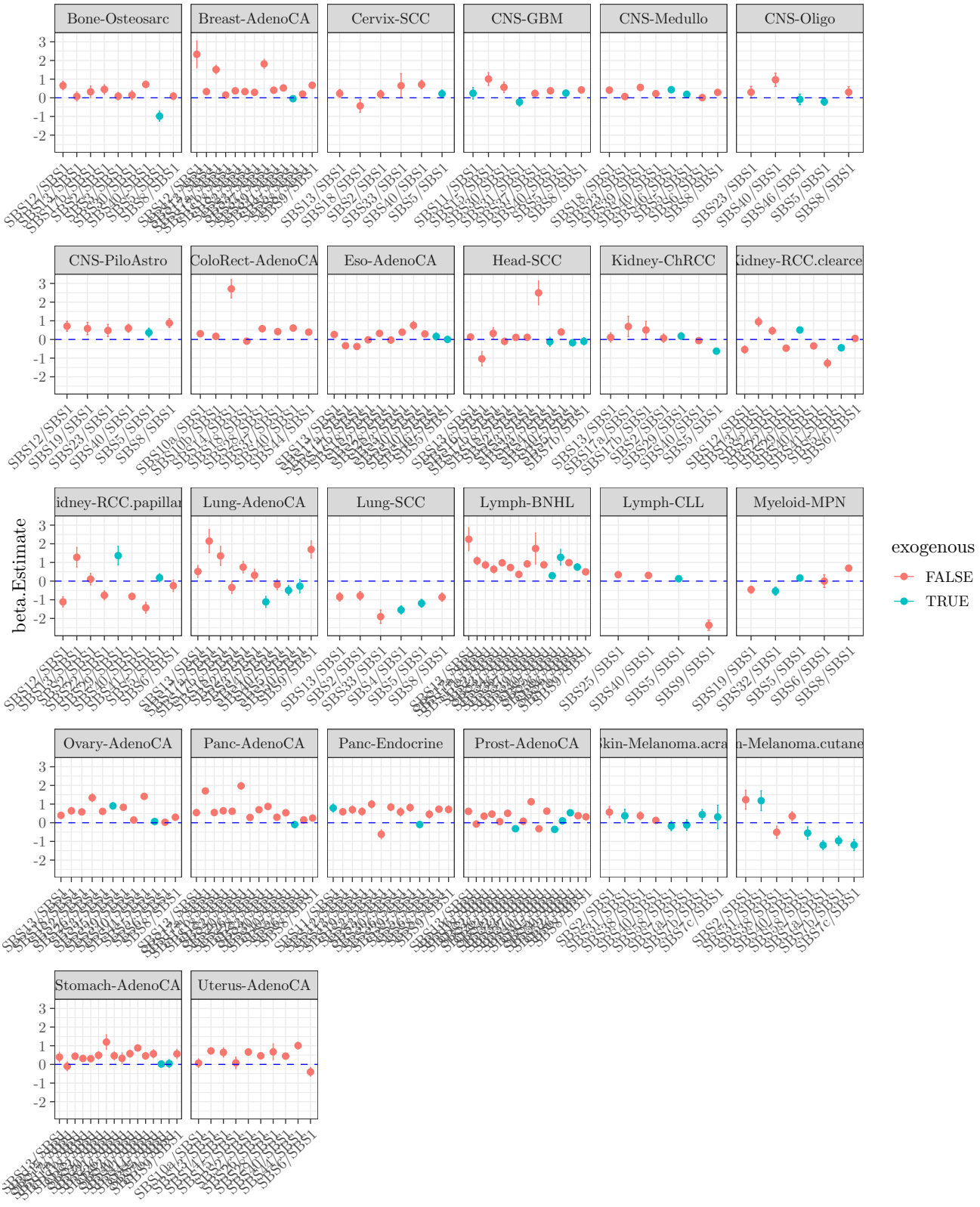
```
##
## FALSEFALSE  FALSETRUE   TRUETRUE
```

```
##          85          89          74
```


FALSEFALSE
FALSETRUE
TRUETRUE
beta.Estimate
ct


SBS2/SBS1
SBS3/SBS1
SBS5/SBS1
SBS6/SBS1
SBS7b/SBS1
SBS8/SBS1
SBS9/SBS1
SBS12/SBS1
SBS13/SBS1
SBS17b/SBS1
SBS17a/SBS1
SBS18/SBS1
SBS26/SBS1
SBS28/SBS1
SBS30/SBS1
SBS33/SBS1
SBS37/SBS1
SBS39/SBS1
SBS40/SBS1
SBS41/SBS1
beta.Estimate
3
2
1
0
-1
-2
ct
Bone-Osteosarc
Breast-AdenoCA
Cervix-SCC
CNS-GBM
CNS-Medullo
CNS-Oligo
CNS-PiloAstro
ColoRect-AdenoCA
Eso-AdenoCA
Head-SCC
Kidney-ChRCC
Kidney-RCC.clearcell
Kidney-RCC.papillary
Lung-AdenoCA
Lung-SCC
Lymph-BNHL
Lymph-CLL
Myeloid-MPN
Ovary-AdenoCA
Panc-AdenoCA
Panc-Endocrine
Prost-AdenoCA
Skin-Melanoma.acral
Skin-Melanoma.cutaneous
Stomach-AdenoCA
Uterus-AdenoCA

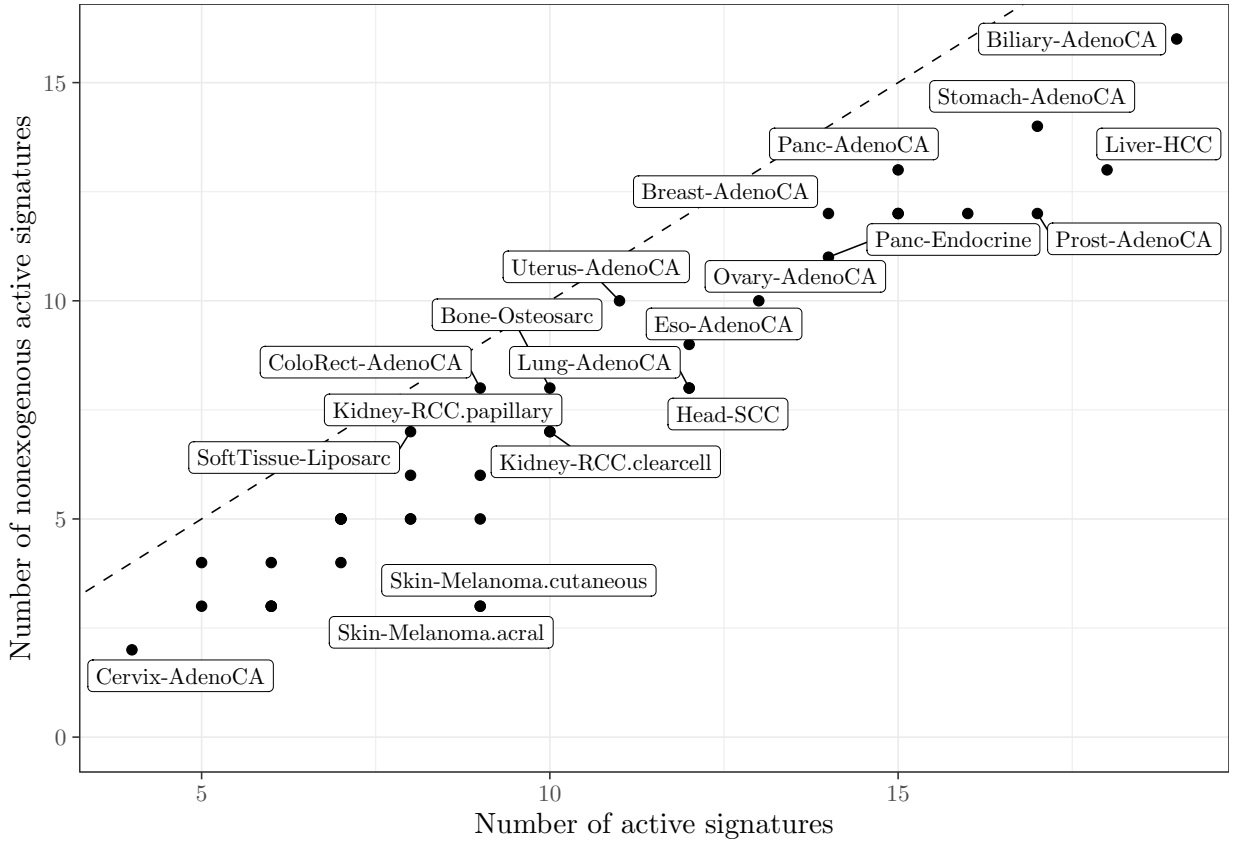

SBS12/SBS1
SBS13/SBS1
SBS17a/SBS1
SBS17b/SBS1
SBS18/SBS1
SBS2/SBS1
SBS26/SBS1
SBS28/SBS1
SBS3/SBS1
SBS30/SBS1
SBS33/SBS1
SBS37/SBS1
SBS39/SBS1
SBS40/SBS1
SBS41/SBS1
SBS5/SBS1
SBS6/SBS1
SBS7b/SBS1
SBS8/SBS1
SBS9/SBS1
beta.Estimate
3
2
1
0
-1
-2
ct
Bone-Osteosarc
Breast-AdenoCA
Cervix-SCC
CNS-GBM
CNS-Medullo
CNS-Oligo
CNS-PiloAstro
ColoRect-AdenoCA
Eso-AdenoCA
Head-SCC
Kidney-ChRCC
Kidney-RCC.clearcell
Kidney-RCC.papillary
Lung-AdenoCA
Lung-SCC
Lymph-BNHL
Lymph-CLL
Myeloid-MPN
Ovary-AdenoCA
Panc-AdenoCA
Panc-Endocrine
Prost-AdenoCA
Skin-Melanoma.acral
Skin-Melanoma.cutaneous
Stomach-AdenoCA
Uterus-AdenoCA


Bone-Osteosarc
Breast-AdenoCA
Cervix-SCC
CNS-GBM
CNS-Medullo
CNS-Oligo
CNS-PiloAstro
ColoRect-AdenoCA
Eso-AdenoCA
Head-SCC
Kidney-ChRCC
idney-RCC.clearce
idney-RCC.papillar
Lung-AdenoCA
Lung-SCC
Lymph-BNHL
Lymph-CLL
Myeloid-MPN
Ovary-AdenoCA
Panc-AdenoCA
Panc-Endocrine
Prost-AdenoCA
kin-Melanoma.acra
-Melanoma.cutane
Stomach-AdenoCA
Uterus-AdenoCA
beta.Estimate
exogenous
FALSE
TRUE
logR

How many signatures so we have in total and how many nonexogenous ones?

```
## Error in slot(i, "count_matrices_active") :
##   cannot get a slot ("count_matrices_active") from an object of type "logical"
```

```
## Error in signature_roo_active[[j]][[1]][, !(colnames(signature_roo_active[[j]][[1]]) %in%  :
##   incorrect number of dimensions
## Error in signature_roo_active[[j]][[1]][, !(colnames(signature_roo_active[[j]][[1]]) %in%  :
##   incorrect number of dimensions
```

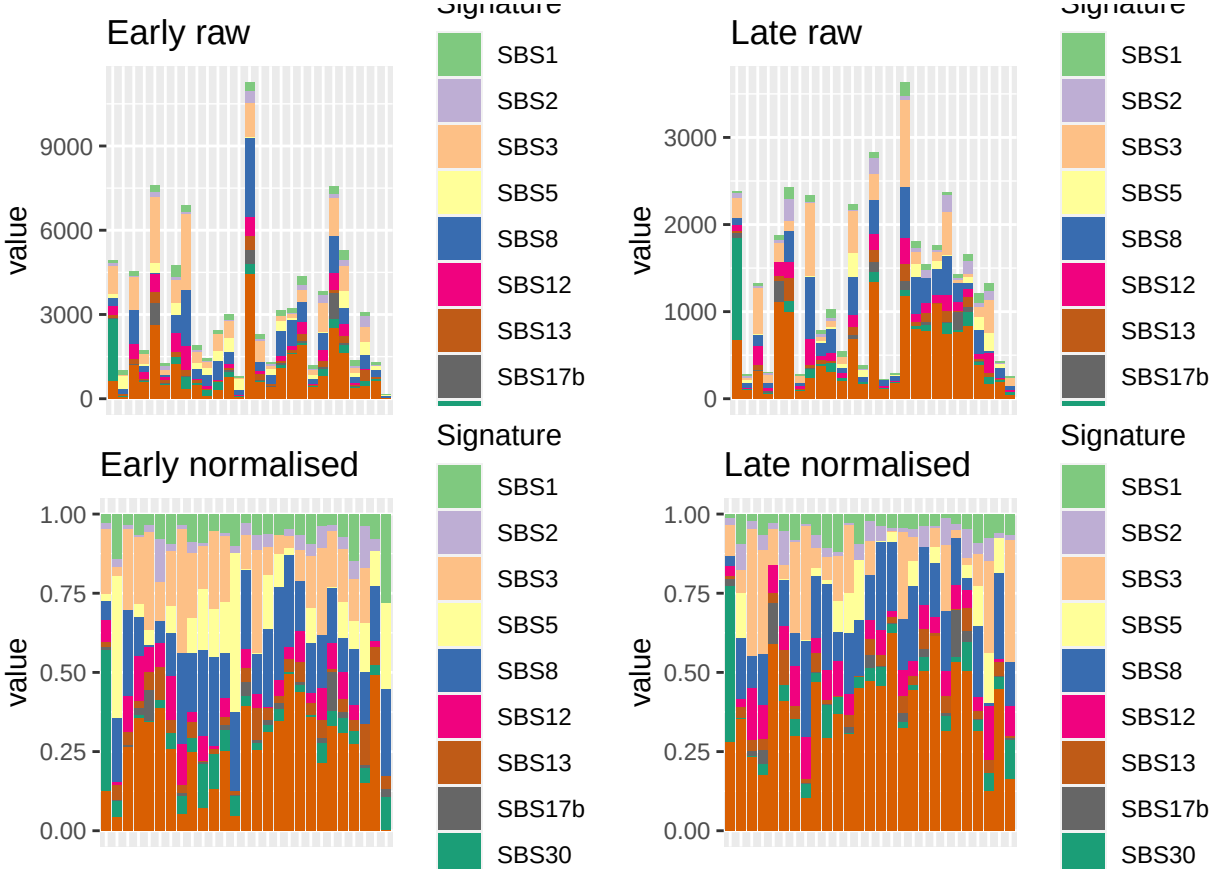


Analysis per cancer type

Bone osteosarcoma

Barplot and general statistics

```
## [1] 27
```



The number of samples and signatures is:

```
## [1] 54 10
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS8"   "SBS12"  "SBS13"  "SBS17b"
##  [9] "SBS30"  "SBS40"
```

Convergence table

We only have converged results for the multinomial with full RE, and the DM with a single lambda (diag and full RE). It is the same for nonexogenous signatures.

```
##             value                                L2                  L1
## 1  Bone-Osteosarc    hessian_positivedefinite_bool          diagRE_M
## 2  Bone-Osteosarc    hessian_positivedefinite_bool          fullRE_M
## 3  Bone-Osteosarc hessian_nonpositivedefinite_bool       diagRE_DMDL
## 4  Bone-Osteosarc hessian_nonpositivedefinite_bool     fullRE_halfDM
## 5  Bone-Osteosarc hessian_nonpositivedefinite_bool       fullRE_DMDL
## 6  Bone-Osteosarc    hessian_positivedefinite_bool       diagRE_DMSL
## 7  Bone-Osteosarc    hessian_positivedefinite_bool     sparseRE_DMSL
## 8  Bone-Osteosarc hessian_nonpositivedefinite_bool       fullRE_DMSL
## 9  Bone-Osteosarc hessian_nonpositivedefinite_bool  fullRE_DMSL_SBS1
## 10 Bone-Osteosarc    hessian_positivedefinite_bool    fullRE_M_nonexo
## 11 Bone-Osteosarc    hessian_positivedefinite_bool diagRE_DMSL_nonexo
```

```
## 12 Bone-Osteosarc                           Timeout     sparseRE_DMSL_nonexo
## 13 Bone-Osteosarc hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 Bone-Osteosarc hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Bone-Osteosarc hessian_nonpositivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

If we use the values of the fullRE M as initial values for the fullRE DM, we also don't get convergence:

```
## [1] FALSE
```

Potentially problematic signatures

We notice that we have several signatures with low exposures, and many zero exposures

```
colSums(obj_Bone_Osteosarc$Y == 0)/nrow(obj_Bone_Osteosarc$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5       SBS8      SBS12      SBS13
## 0.00000000 0.03703704 0.14814815 0.37037037 0.01851852 0.09259259 0.00000000
##     SBS17b      SBS30      SBS40
## 0.37037037 0.12962963 0.01851852
```

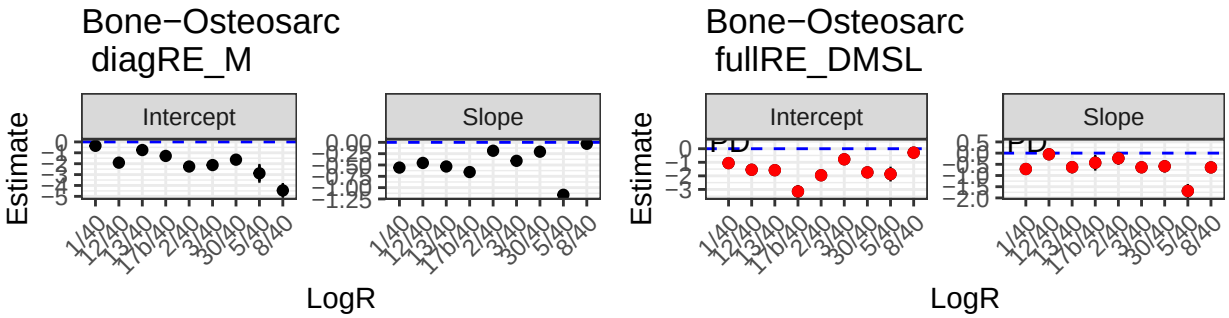
```
colSums(obj_Bone_Osteosarc$Y)/sum(obj_Bone_Osteosarc$Y)
```

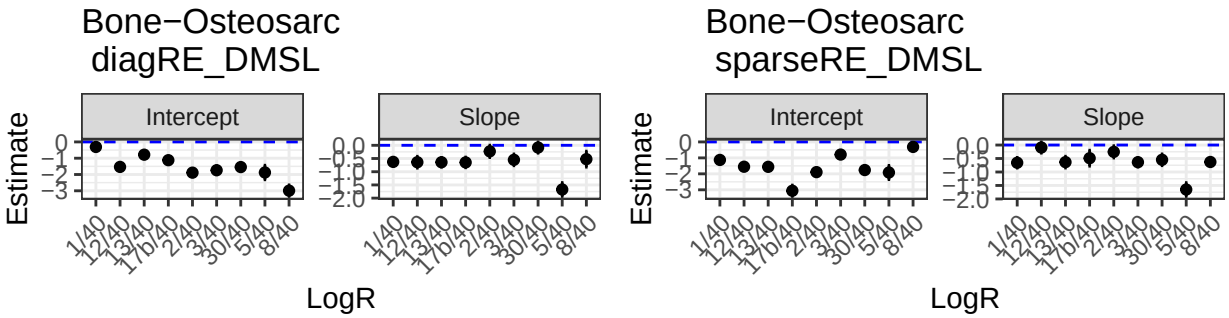
```
##       SBS1       SBS2       SBS3       SBS5       SBS8      SBS12      SBS13
## 0.05099661 0.03376971 0.17876022 0.05053018 0.17164713 0.07538325 0.04159022
##     SBS17b      SBS30      SBS40
## 0.02866227 0.06128922 0.30737119
```

E.g.

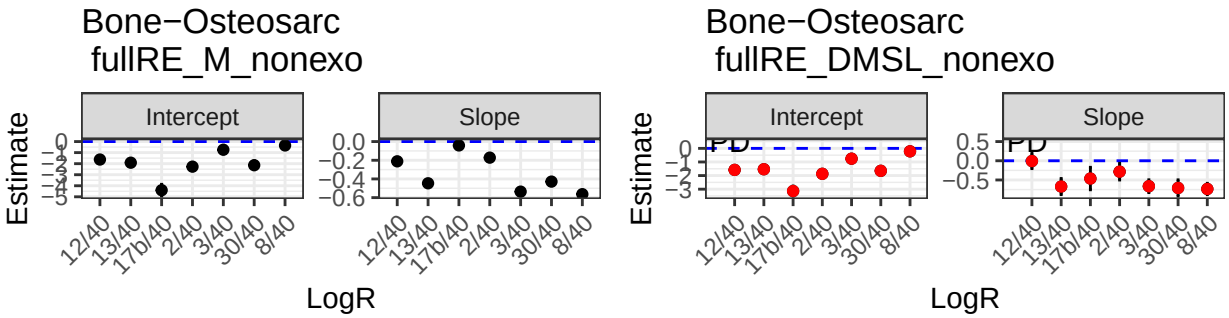
- SBS17b is 0 in 37% of cases and has an overall exposure of 2.9%
- SBS30 is 0 in 13% of cases and overal has an exposure of only 6.1%
- SBS5 is 0 in 37% of cases and has an overall exposure of 5.1%

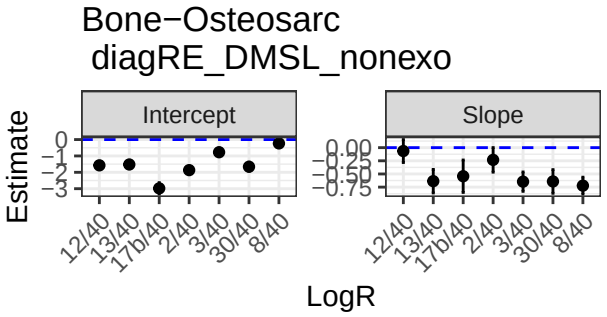
Betas






```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```





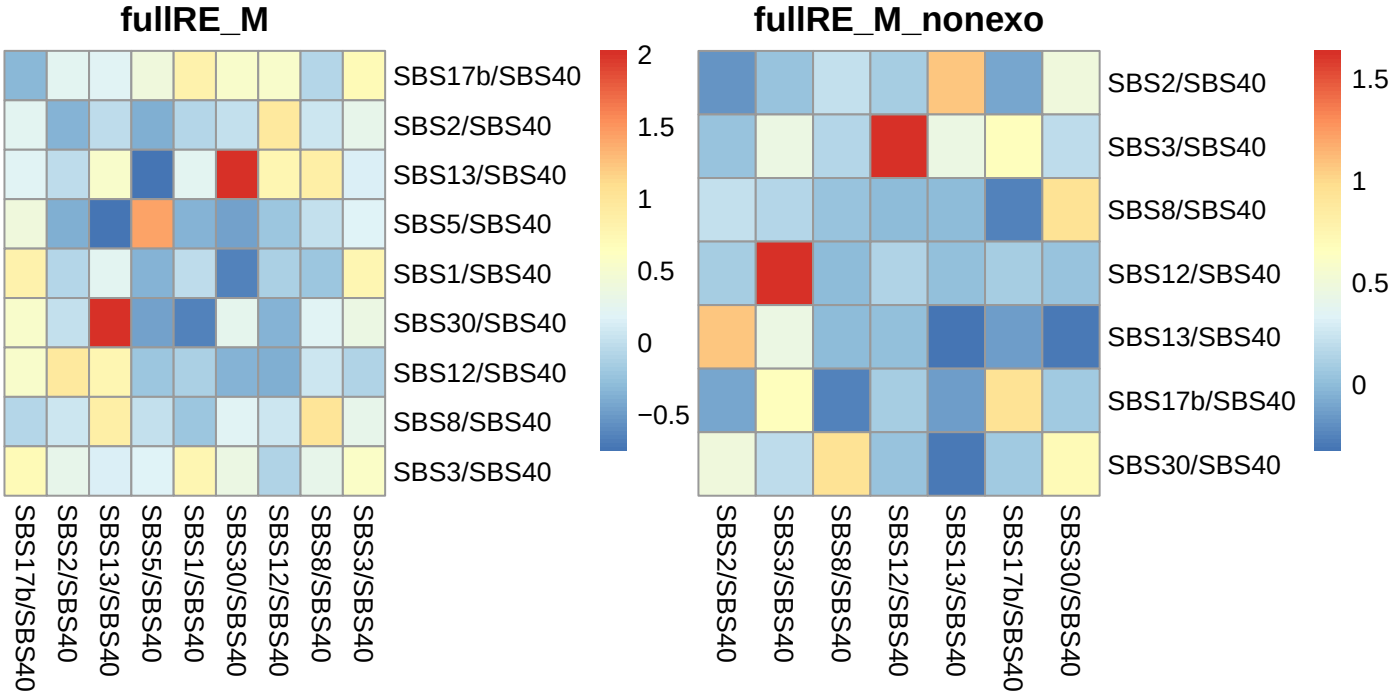
We use the results from the diagonal single lambda DM to test for differential abundance, giving a p-value of 3.8923434×10^{-5}.

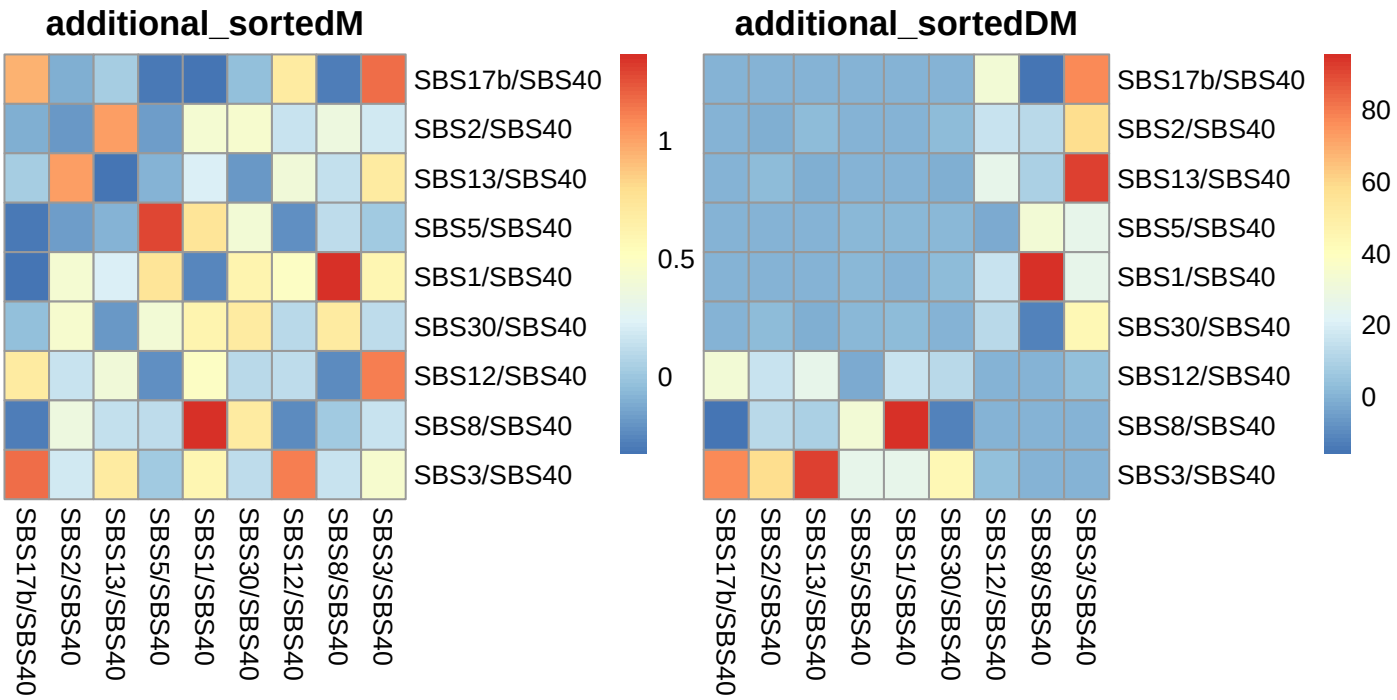
Covariance matrices

Note that sortedDM did not convergence.

Nevertheless, both versions of fullRE M – both of which converged and use the same baseline – give very different covariances matrices.

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```





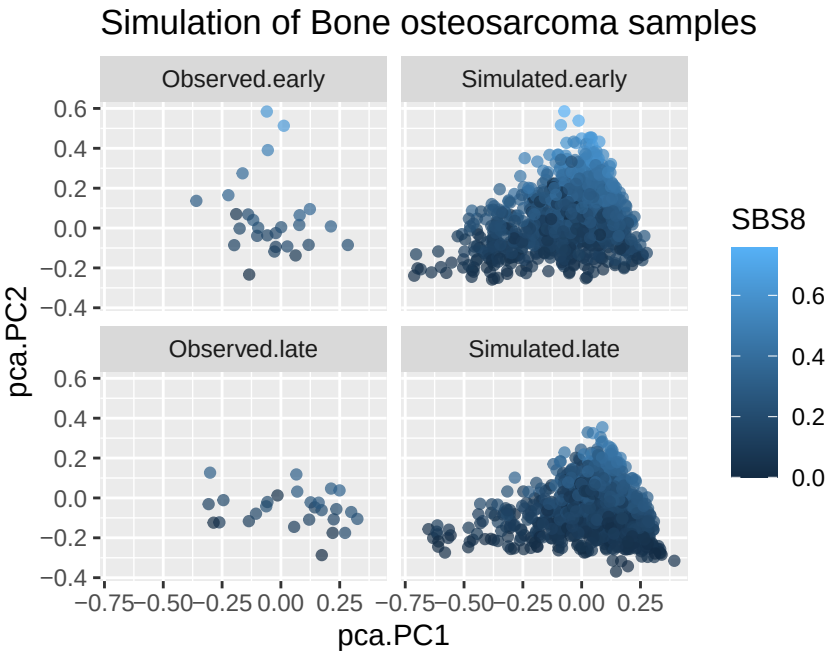
Simulation under inferred data

Simulating with diagRE DMSL nonexo.

```
## [1] 27
```

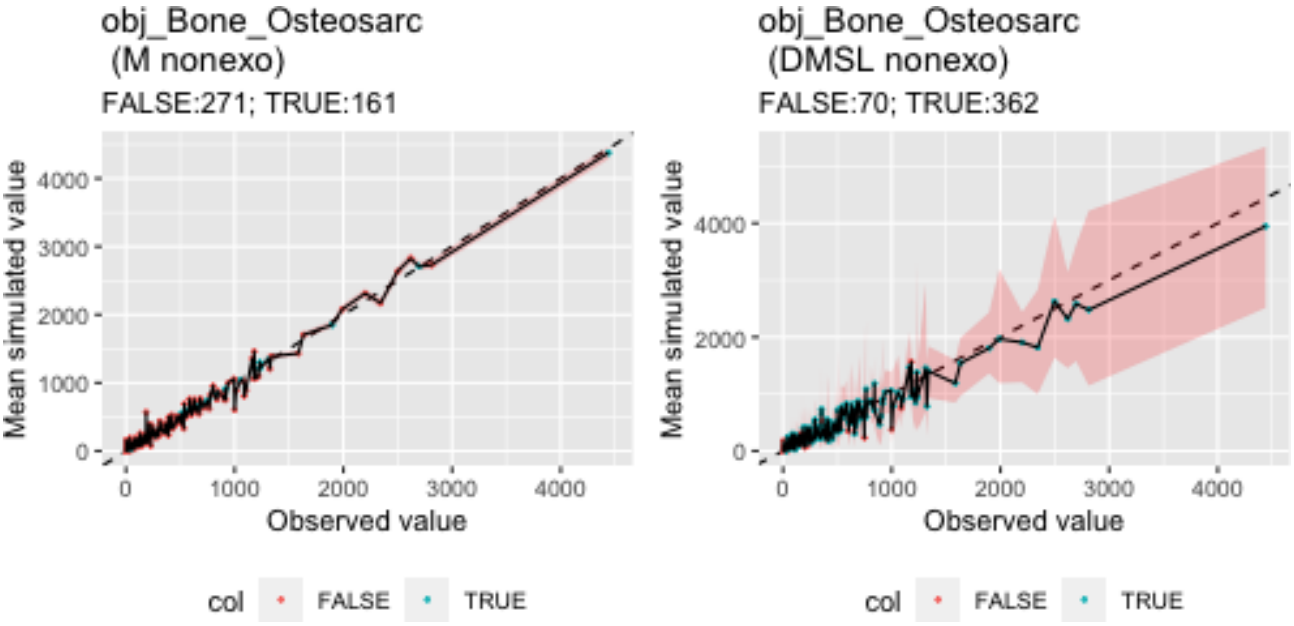
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



Ranked plot for coverage

Note that fullRE DMSL nonexo has not converged!

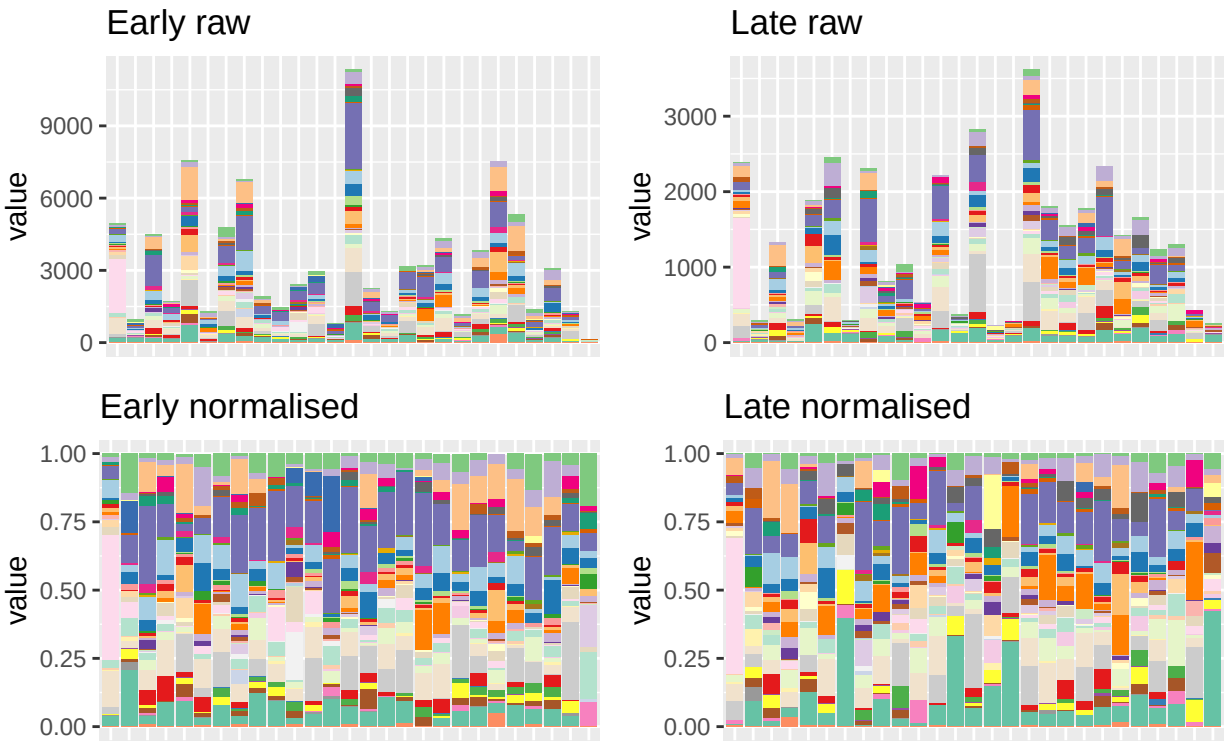


73/359=20% of values are not included in the confidence interval of the DMSL.

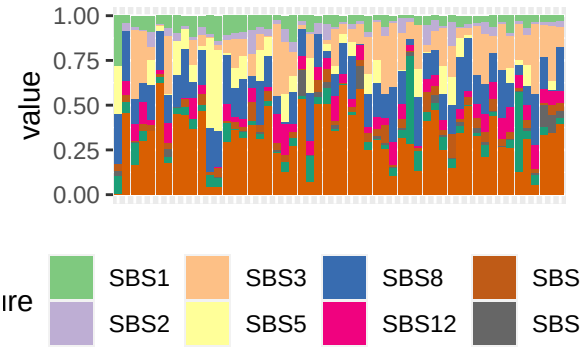
Signatures from mutSigExtractor

The signatures from mutSigExtractor are a bit more chaotic:

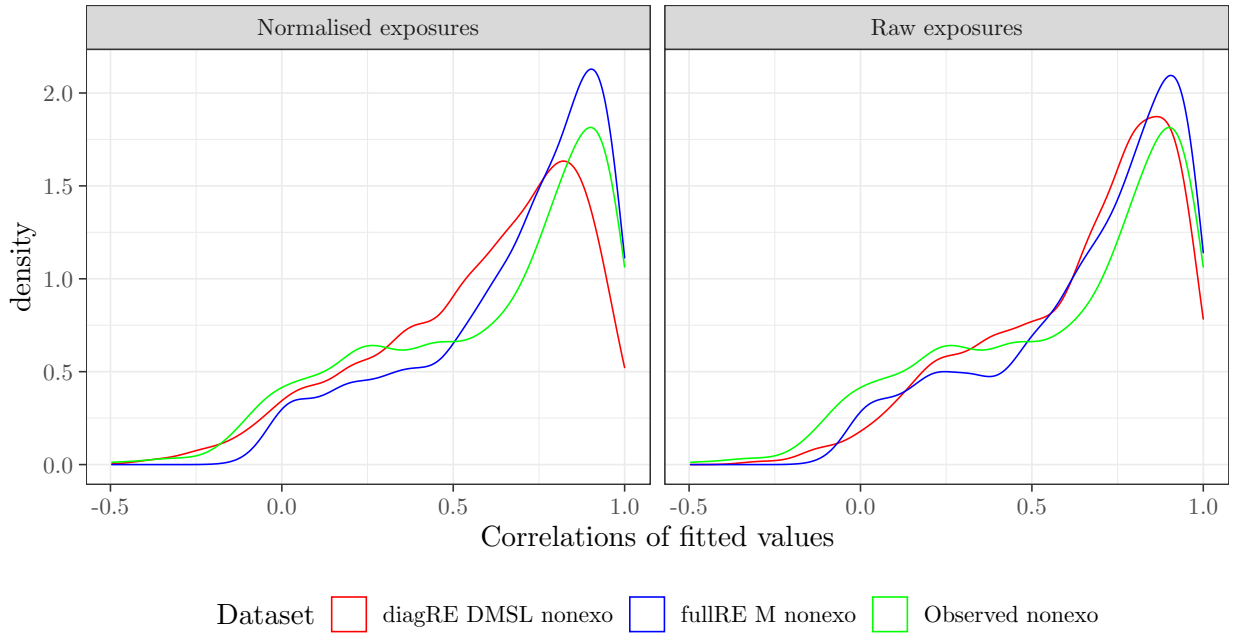
```
## [1] 27
```



Exposures sorted by increasing number of mutations: there is no trend



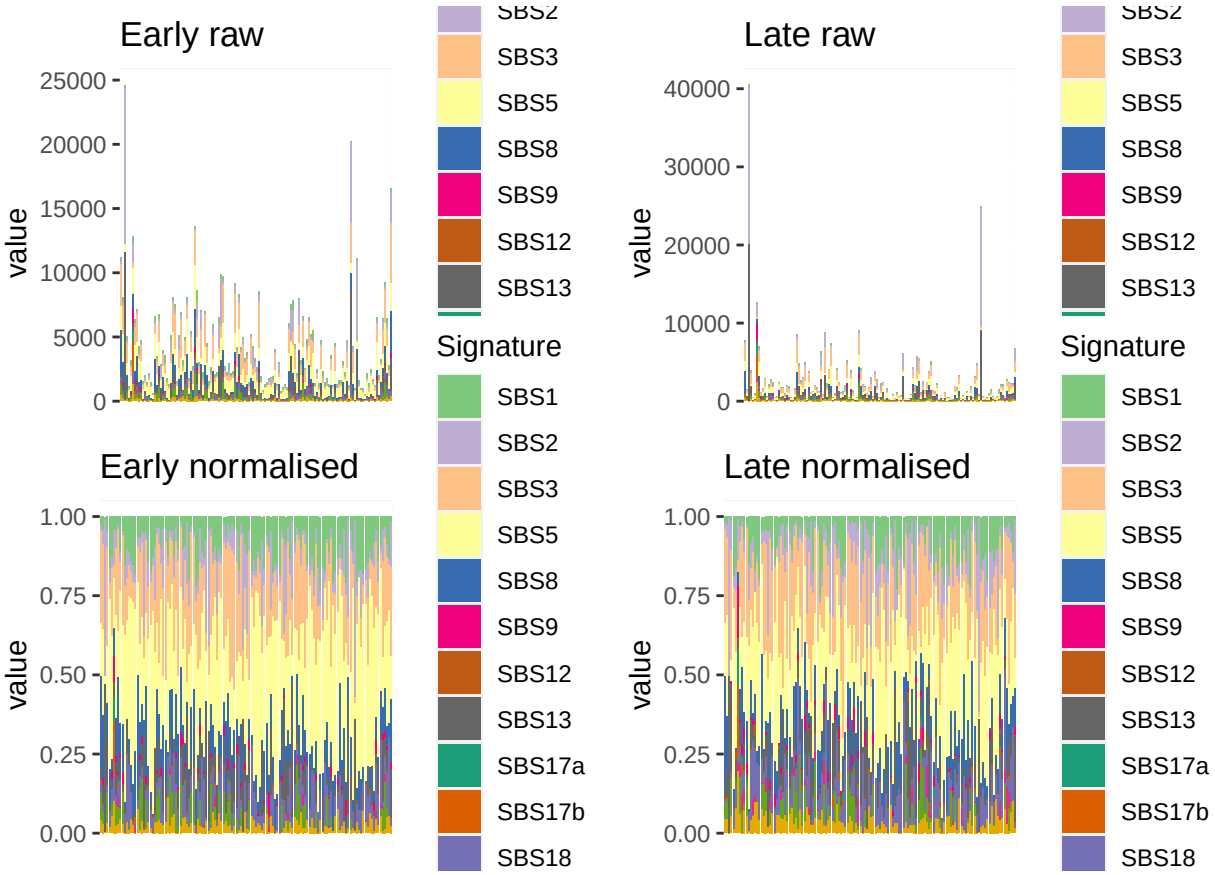
Correlations of signatures



Breast-AdenoCA

Barplot and general statistics

```
## [1] 136
```



There are many signatures, and also many samples.

The number of samples and signatures is:

```
## [1] 272  14
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS8"   "SBS9"   "SBS12"  "SBS13"
##  [9] "SBS17a" "SBS17b" "SBS18"  "SBS37"  "SBS39"  "SBS41"
```

Convergence table

We only have converged results for the diagRE_DMSL, with diagonal or sparse covariance structure, and diagonal M. This is probably due to the very high number of signatures, which make it impossible to infer the whole covariance structure.

```
##             value                                L2                  L1
## 1  Breast-AdenoCA    hessian_positivedefinite_bool          diagRE_M
## 2  Breast-AdenoCA hessian_nonpositivedefinite_bool          fullRE_M
## 3  Breast-AdenoCA hessian_nonpositivedefinite_bool       diagRE_DMDL
## 4  Breast-AdenoCA hessian_nonpositivedefinite_bool      fullRE_halfDM
```

```
## 5  Breast-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  Breast-AdenoCA    hessian_positivedefinite_bool              diagRE_DMSL
## 7  Breast-AdenoCA    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  Breast-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  Breast-AdenoCA hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Breast-AdenoCA hessian_nonpositivedefinite_bool          fullRE_M_nonexo
## 11 Breast-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Breast-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Breast-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 Breast-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Breast-AdenoCA                          Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

If we use the values of the diagRE M as initial values for the diagRE DM, we that it has converged. This is probably due to a combination of things: we are using the optimiser nlminb (better in general than the alternative, optim) and we are starting with these - better - values, and we are sorting the columns so that the category with highest total value is the baseline.

```
## [1] TRUE
```

Potentially problematic signatures

We notice that we have several signatures with low exposures, and many zero exposures

```
colSums(obj_Breast_AdenoCA$Y == 0)/nrow(obj_Breast_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5        SBS8        SBS9
## 0.000000000 0.000000000 0.025735294 0.007352941 0.088235294 0.562500000
##       SBS12       SBS13      SBS17a      SBS17b       SBS18       SBS37
## 0.955882353 0.073529412 0.709558824 0.500000000 0.036764706 0.772058824
##       SBS39       SBS41
## 0.599264706 0.084558824
```

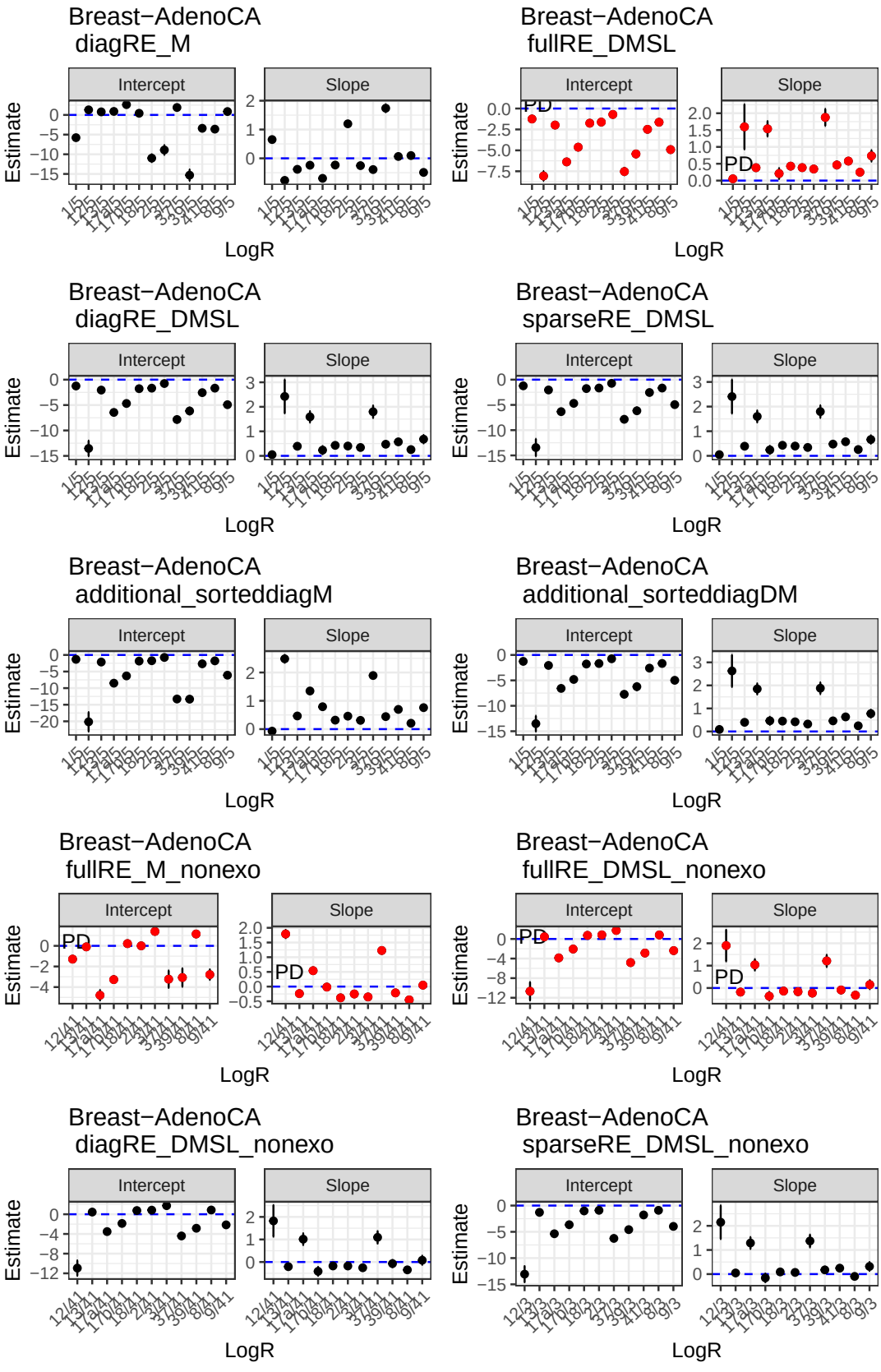
```
colSums(obj_Breast_AdenoCA$Y)/sum(obj_Breast_AdenoCA$Y)
```

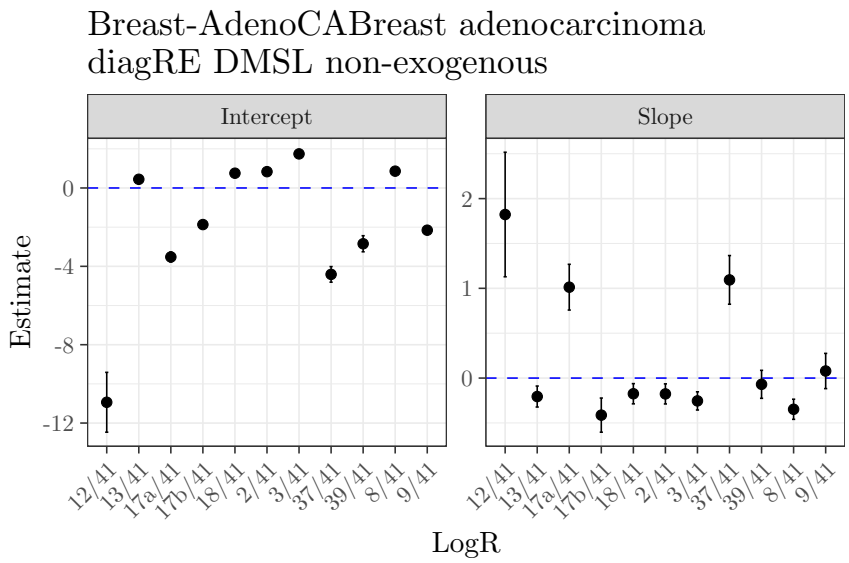
```
##         SBS1         SBS2         SBS3         SBS5         SBS8         SBS9
## 0.0553410311 0.1376261991 0.1993274971 0.2185906789 0.0969490005 0.0132833987
##        SBS12        SBS13       SBS17a       SBS17b        SBS18        SBS37
## 0.0003532317 0.1360853961 0.0036266519 0.0081714966 0.0531199688 0.0057240307
##        SBS39        SBS41
## 0.0402034279 0.0315979909
```

E.g.

- SBS9 is 0 in 56.2% of cases and has an overall exposure of 1.3%
- SBS12 is 0 in 95.6% of cases and has an overall exposure of 0%
- SBS17a is 0 in 71% of cases and has an overall exposure of 0.4%
- SBS17b is 0 in 50% of cases and has an overall exposure of 0.8%
- SBS37 is 0 in 77.2% of cases and has an overall exposure of 0.6%
- SBS39 is 0 in 59.9% of cases and has an overall exposure of 4%

Betas

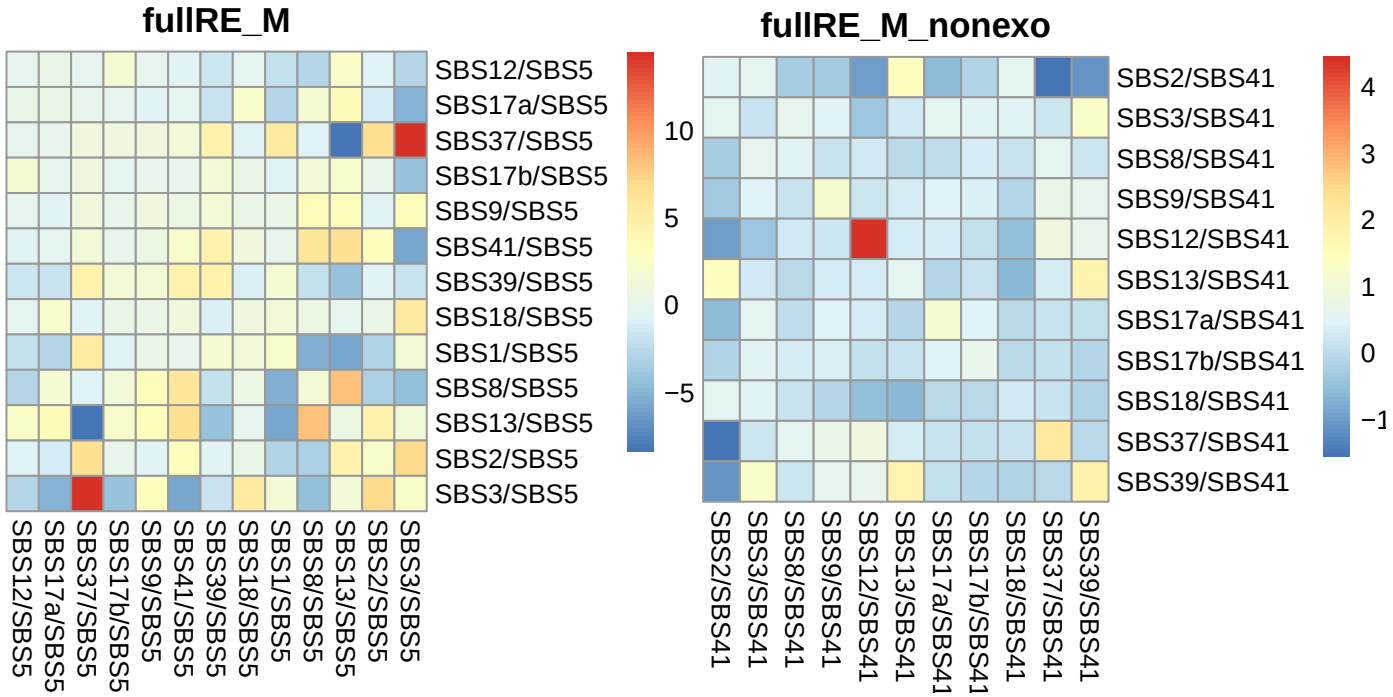

Breast−AdenoCA
diagRE_M
Breast−AdenoCA
fullRE_DMSL
Breast−AdenoCA
diagRE_DMSL
Breast−AdenoCA
sparseRE_DMSL
Breast−AdenoCA
additional_sorteddiagM
Breast−AdenoCA
additional_sorteddiagDM
Breast−AdenoCA
fullRE_M_nonexo
Breast−AdenoCA
fullRE_DMSL_nonexo
Breast−AdenoCA
diagRE_DMSL_nonexo
Breast−AdenoCA
sparseRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR
PD



We use the results from the diagonal single lambda DM to test for differential abundance, giving a p-value of 7.748574×10^{-12}.

Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



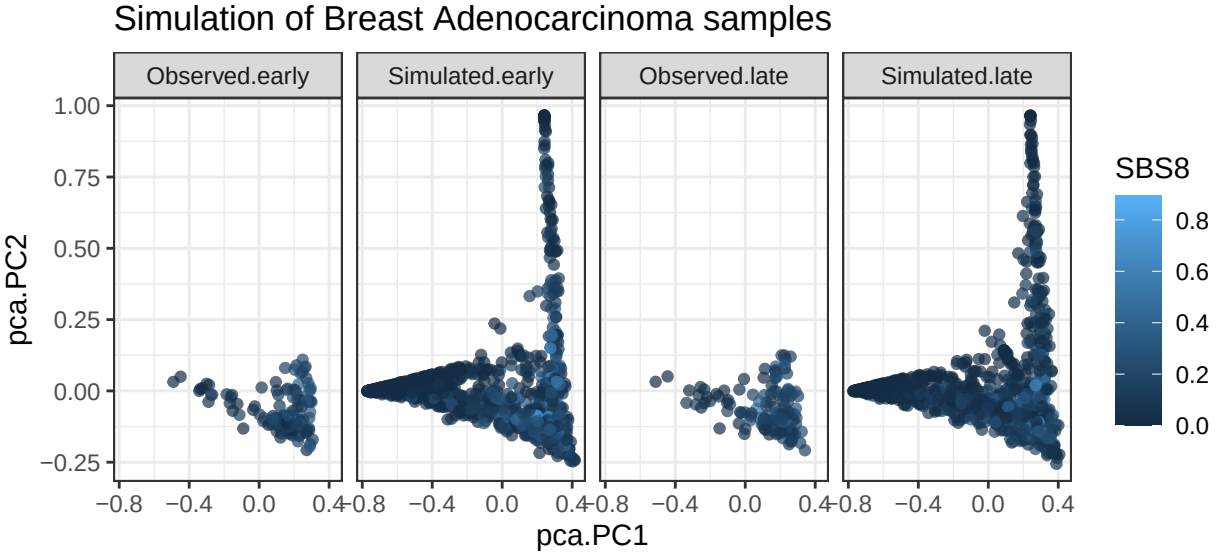
Simulation under inferred data

Sorting the object as we are using sparseRE_DM.

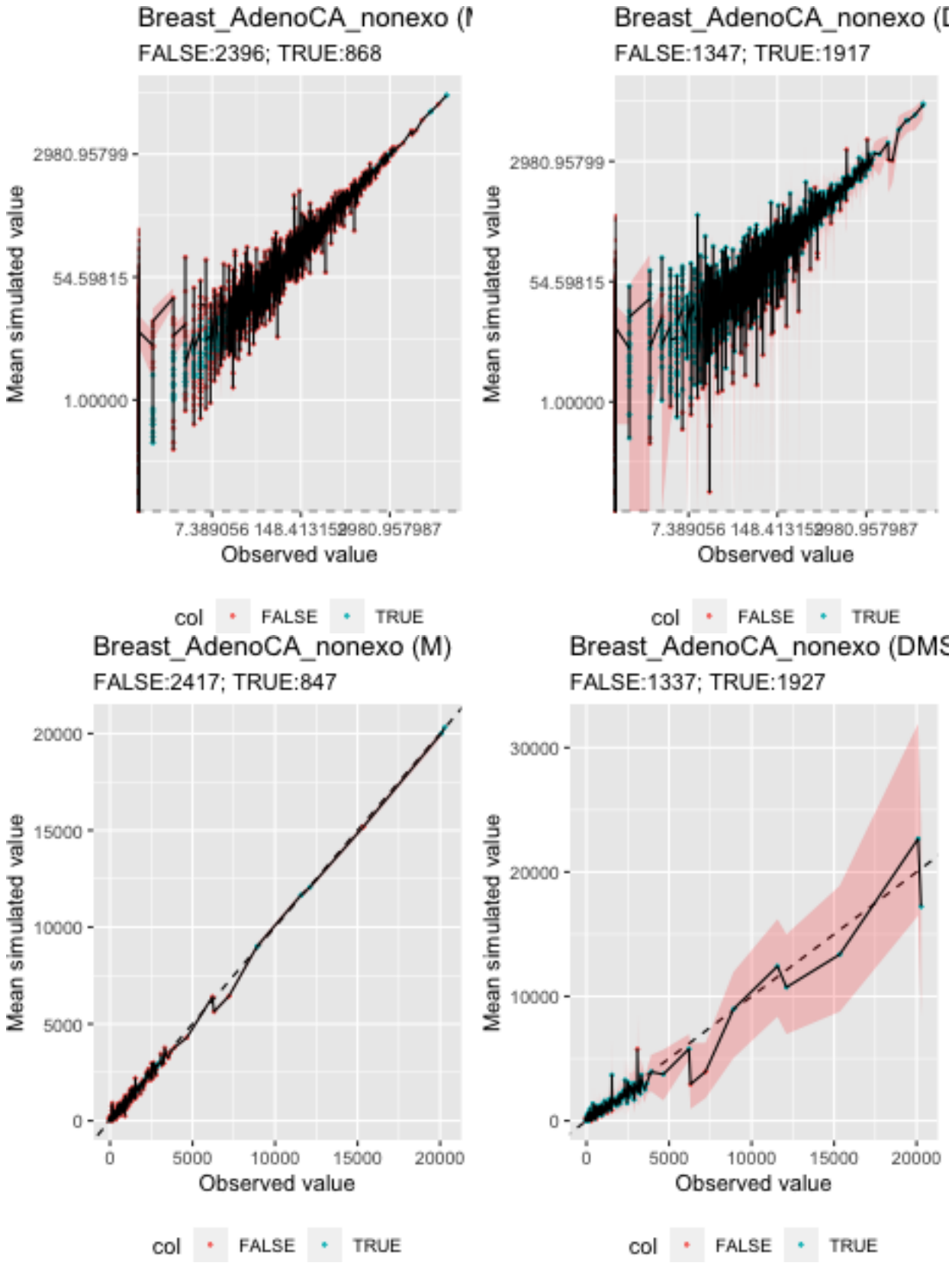
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```

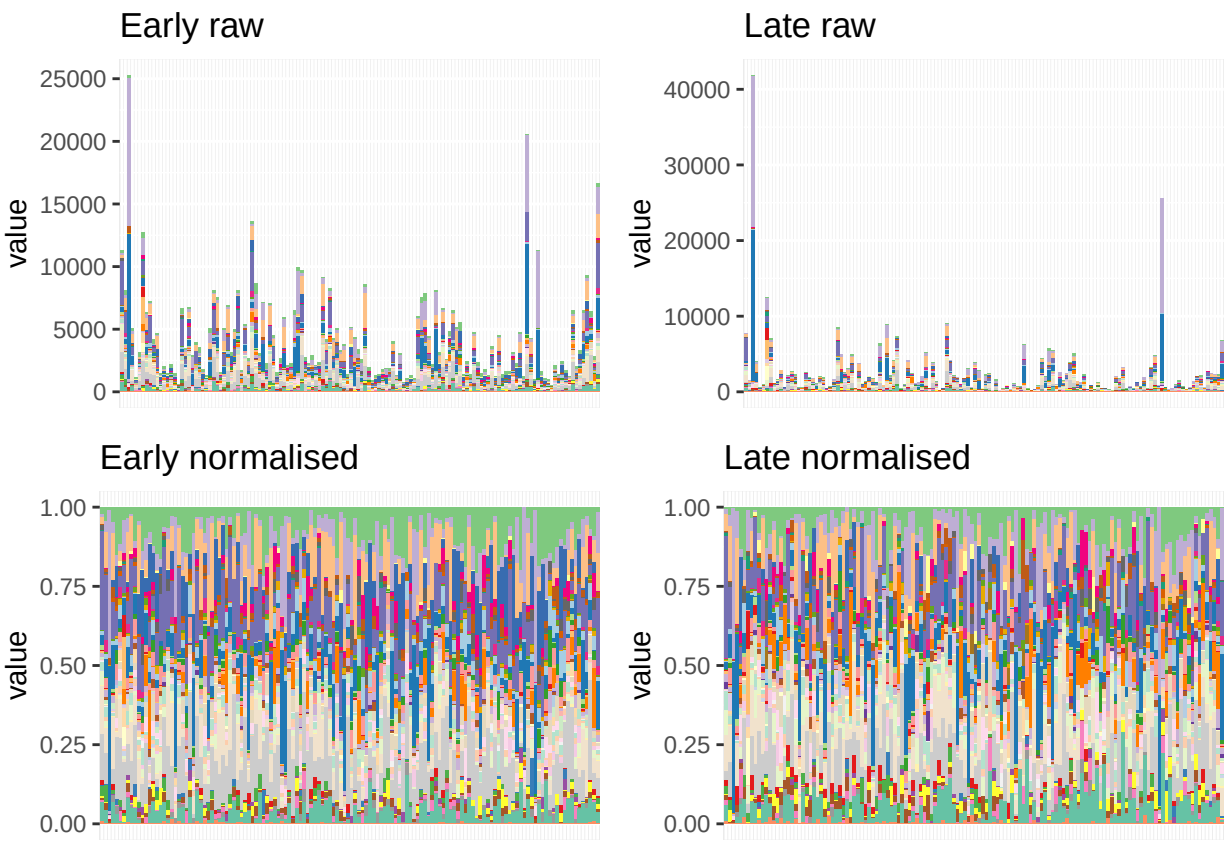


Ranked plot for coverage



Signatures from mutSigExtractor

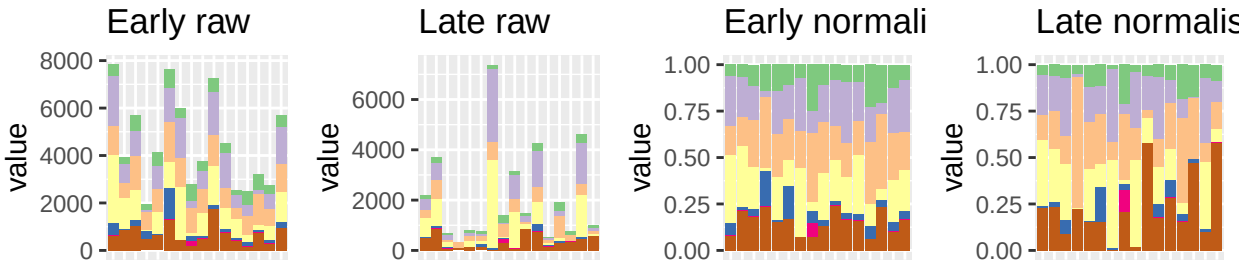
```
## [1] 136
```



Cervix-SCC

Barplot and general statistics

```
## [1] 16
```



The number of samples and signatures is:

```
## [1] 32  7
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS5"  "SBS13" "SBS18" "SBS33" "SBS40"
```

Convergence table

```
## [1] value L2    L1
## <0 rows> (or 0-length row.names)
```

Potentially problematic signatures

SBS33 is a potentially problematic signature, being 0 in 81.2% of cases and with an overall exposure of 0.4%.

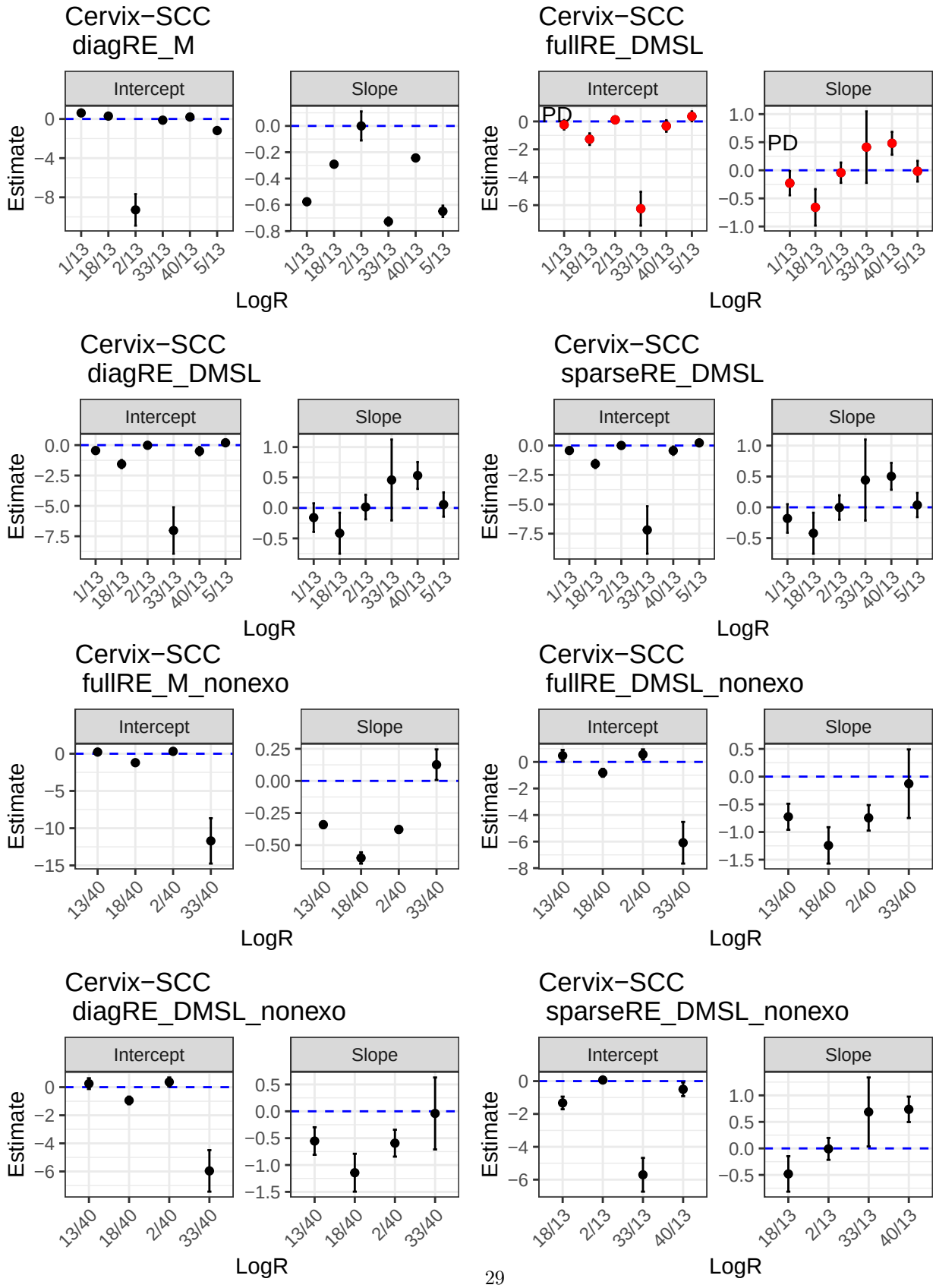
```
colSums(obj_Cervix_SCC$Y == 0)/nrow(obj_Cervix_SCC$Y)
```

```
##    SBS1    SBS2    SBS5   SBS13   SBS18   SBS33   SBS40
## 0.00000 0.00000 0.00000 0.03125 0.15625 0.81250 0.03125
```

```
colSums(obj_Cervix_SCC$Y)/sum(obj_Cervix_SCC$Y)
```

```
##        SBS1        SBS2        SBS5       SBS13       SBS18       SBS33
## 0.099164517 0.235000561 0.211562185 0.250439236 0.046577698 0.003560615
##       SBS40
## 0.153695189
```

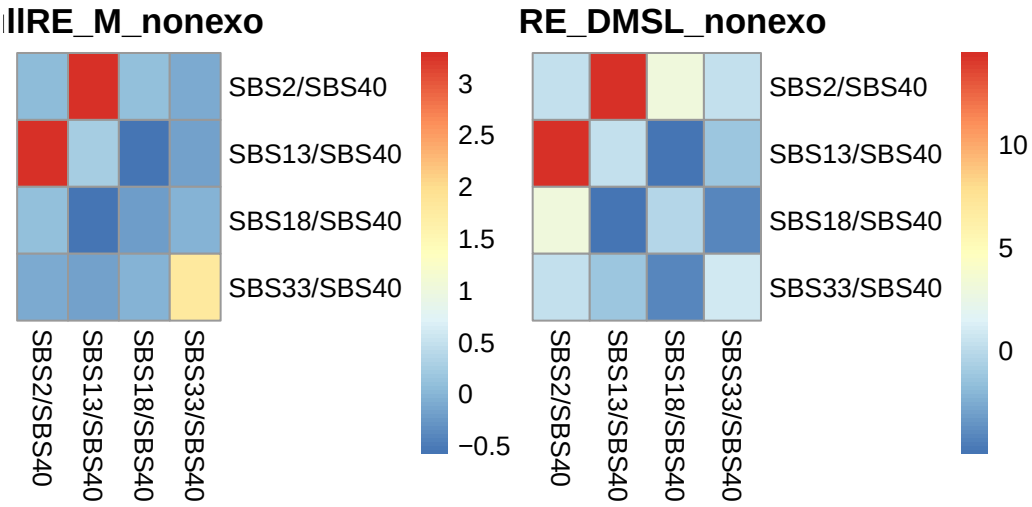
Betas


Cervix−SCC
diagRE_M
Intercept
Slope
Estimate
LogR
1/13
18/13
2/13
33/13
40/13
5/13
Cervix−SCC
fullRE_DMSL
PD
PD
Cervix−SCC
diagRE_DMSL
Cervix−SCC
sparseRE_DMSL
Cervix−SCC
fullRE_M_nonexo
13/40
18/40
2/40
33/40
Cervix−SCC
fullRE_DMSL_nonexo
Cervix−SCC
diagRE_DMSL_nonexo
Cervix−SCC
sparseRE_DMSL_nonexo

We use the results from the fullRE single lambda DM to test for differential abundance, giving a p-value of 3.8923434×10^{-5}.

Covariance matrices

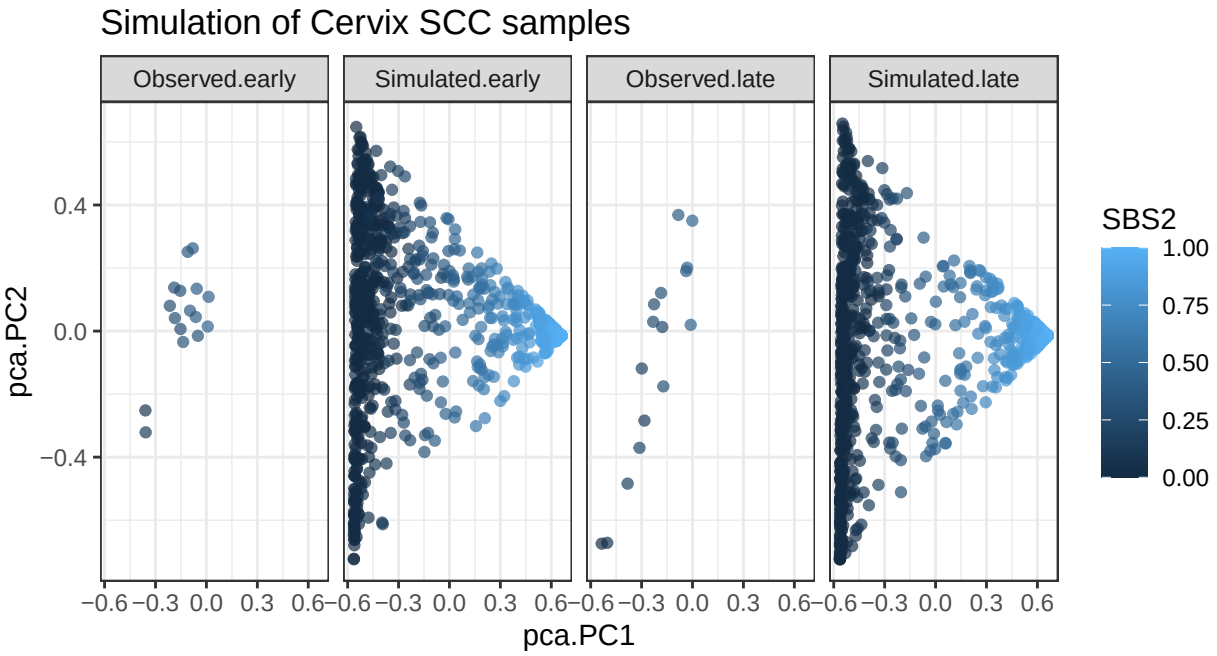
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



Simulation under inferred data

We are usaing the sorted version of the exposures, as we are using sparseRE DMSL.

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



Comparison of the estimated covariance matrix and the closest positive semidefinite matrix.

```
ct <- "Cervix-SCC"

dmin1_Cervix_SCC_cov_mat_sparse <- length(python_like_select_name(sparseRE_DMSL_nonexo[[ct]]$par.fixed, '

cov_vec_Cervix_SCC_cov_mat_sparse = rep(0, (dmin1_Cervix_SCC_cov_mat_sparse**2-dmin1_Cervix_SCC_cov_mat_s
cov_vec_Cervix_SCC_cov_mat_sparse[as.numeric(strsplit(subset_sigs_sparse_cov_idx_nonexo[subset_sigs_spars

cov_mat_Cervix_SCC <- fill_covariance_matrix(arg_d = dmin1_Cervix_SCC_cov_mat_sparse,
                                     arg_entries_var =  exp(python_like_select_name(sparseRE_DMSL_nonexo[[
                                     arg_entries_cov = cov_vec_Cervix_SCC_cov_mat_sparse)

cov_mat_Cervix_SCC
```

```
##           [,1]       [,2]       [,3]      [,4]
## [1,] 2.690655   1.554016   1.337713 0.0000000
## [2,] 1.554016   1.246623 103.048586 0.0000000
## [3,] 1.337713 103.048586   1.490150 0.0000000
## [4,] 0.000000   0.000000   0.000000 0.2029568
```

```
Matrix::nearPD(cov_mat_Cervix_SCC)
```

```
## $mat
## 4 x 4 Matrix of class "dpoMatrix"
##           [,1]      [,2]      [,3]      [,4]
## [1,] 2.690877  1.447736  1.443865 0.0000000
## [2,] 1.447736 52.148325 52.208484 0.0000000
## [3,] 1.443865 52.208484 52.268725 0.0000000
## [4,] 0.000000  0.000000  0.000000 0.2029568
##
## $eigenvalues
```

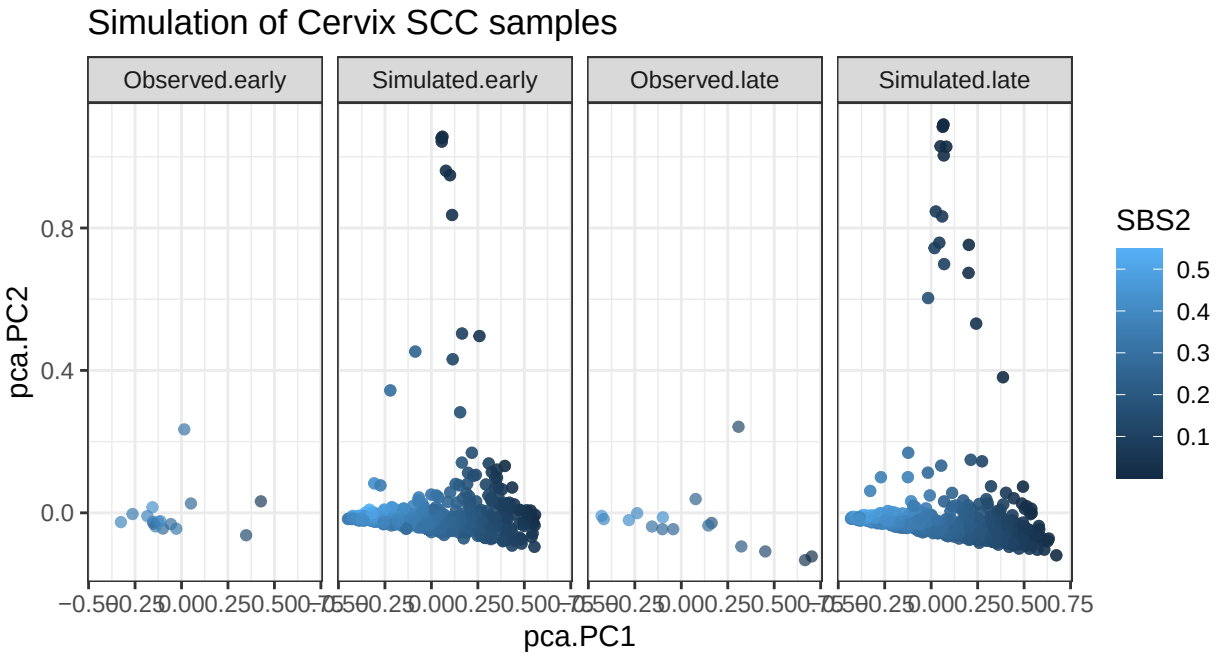
```
## [1] 1.044581e+02 2.649802e+00 2.029568e-01 1.044581e-06
##
## $corr
## [1] FALSE
##
## $normF
## [1] 101.6805
##
## $iterations
## [1] 2
##
## $rel.tol
## [1] 0
##
## $converged
## [1] TRUE
##
## attr(,"class")
## [1] "nearPD"
```

Now with fullRE M:

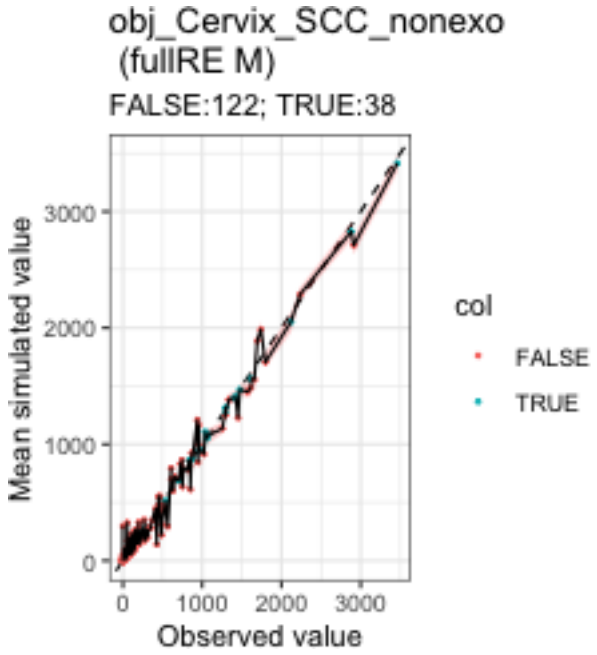
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

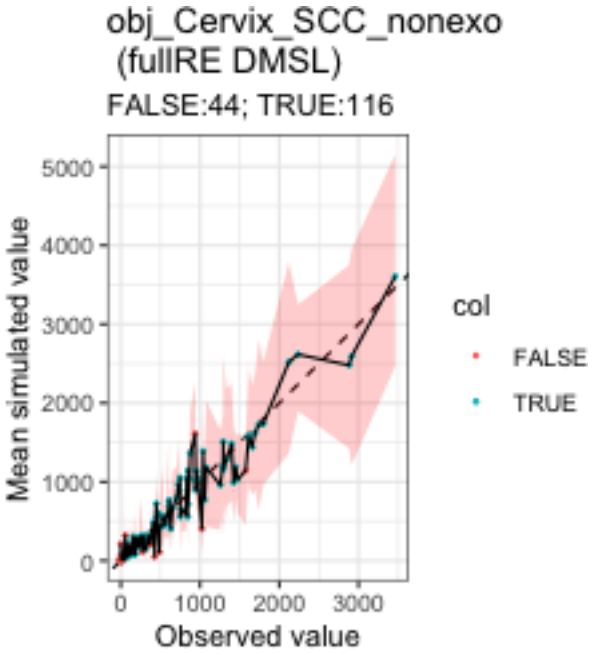
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



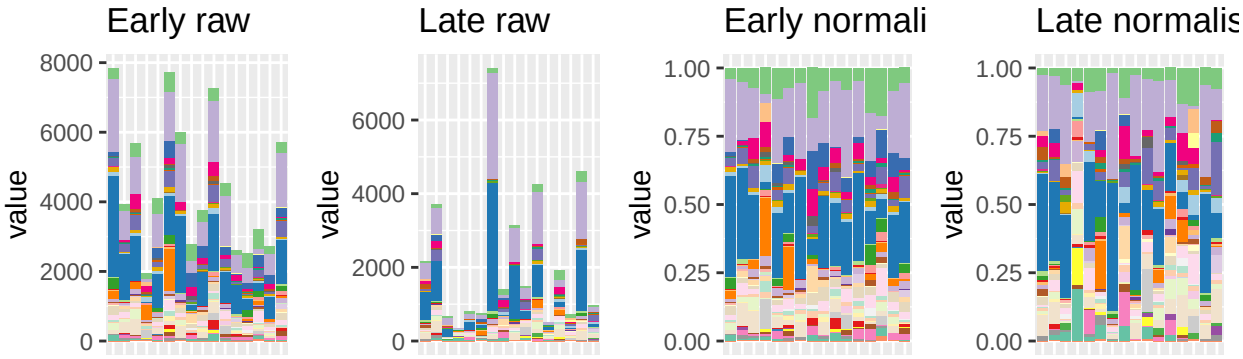
Ranked plot for coverage





Signatures from mutSigExtractor

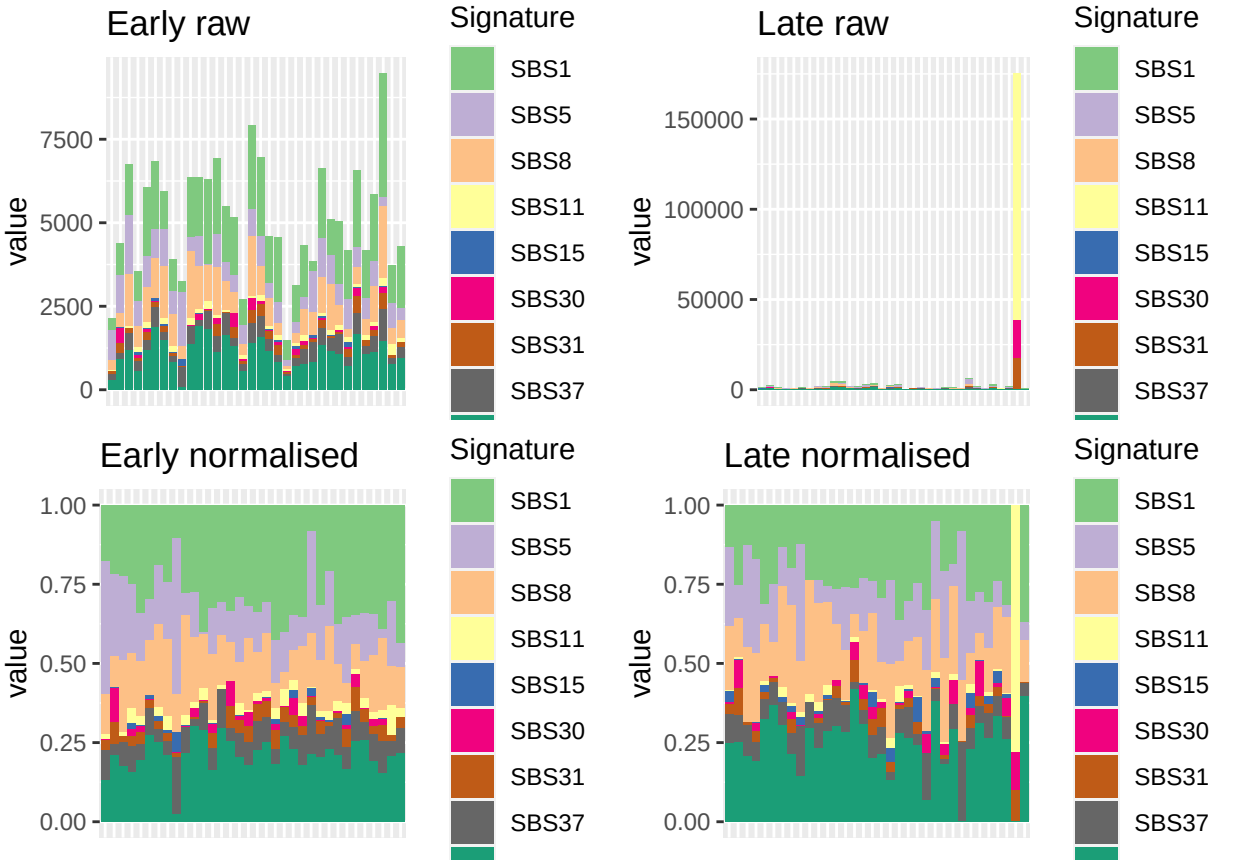
```
## [1] 16
```



CNS-GBM

Barplot and general statistics

```
## [1] 34
```



The number of samples and signatures is:

```
## [1] 68  9
```

The signatures are:

```
## [1] "SBS1"  "SBS5"  "SBS8"  "SBS11" "SBS15" "SBS30" "SBS31" "SBS37" "SBS40"
```

Convergence table

We only have converged results for the multinomial with full RE, and the DM with a single lambda (diag and sparse RE). It is the same for nonexogenous signatures.

```
##      value                              L2                      L1
## 1  CNS-GBM    hessian_positivedefinite_bool                diagRE_M
## 2  CNS-GBM    hessian_positivedefinite_bool                fullRE_M
## 3  CNS-GBM hessian_nonpositivedefinite_bool             diagRE_DMDL
## 4  CNS-GBM hessian_nonpositivedefinite_bool            fullRE_halfDM
## 5  CNS-GBM hessian_nonpositivedefinite_bool             fullRE_DMDL
## 6  CNS-GBM    hessian_positivedefinite_bool             diagRE_DMSL
## 7  CNS-GBM    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  CNS-GBM hessian_nonpositivedefinite_bool             fullRE_DMSL
## 9  CNS-GBM hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 CNS-GBM    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 CNS-GBM    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 CNS-GBM    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 CNS-GBM hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 CNS-GBM hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 CNS-GBM                          Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do converge:

```
## [1] TRUE
```

Potentially problematic signatures

We notice that there are no truly problematic signatures (SBS15 has the most zeros; 50%).

```
colSums(obj_CNS_GBM$Y == 0)/nrow(obj_CNS_GBM$Y)
```

```
##       SBS1       SBS5       SBS8      SBS11      SBS15      SBS30      SBS31
## 0.01470588 0.02941176 0.01470588 0.20588235 0.50000000 0.33823529 0.13235294
##      SBS37      SBS40
## 0.01470588 0.02941176
```

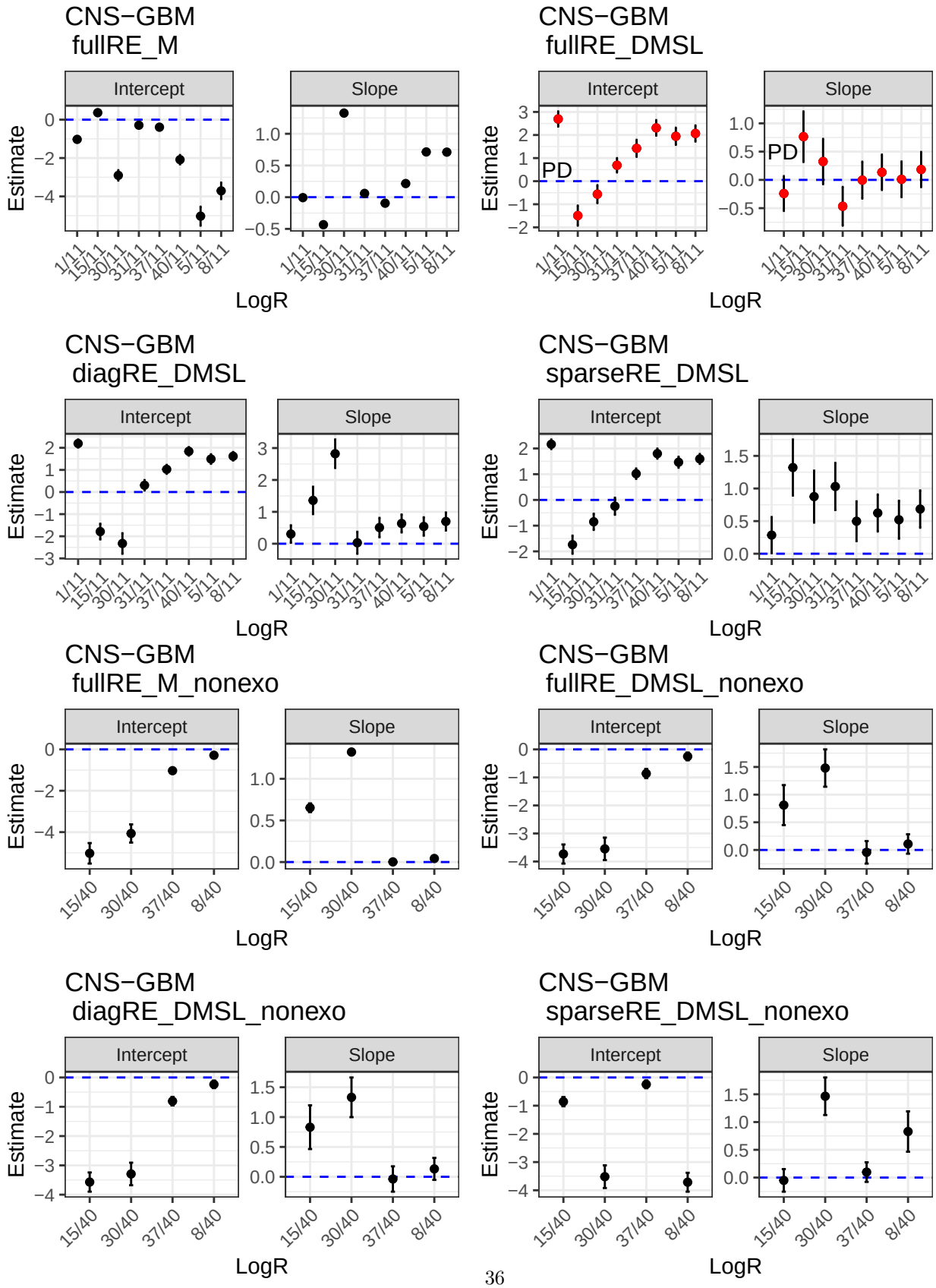
```
colSums(obj_CNS_GBM$Y)/sum(obj_CNS_GBM$Y)
```

```
##        SBS1        SBS5        SBS8       SBS11       SBS15       SBS30
## 0.164856854 0.087757118 0.103223676 0.345294365 0.004258098 0.060917020
##       SBS31       SBS37       SBS40
## 0.060793210 0.046931329 0.125968329
```

```
additional_sortedMnonexo <- list()
additional_sortedDMSLnonexo <- list()
```

```
additional_sortedMnonexo[["CNS-GBM"]] <- sortedM_CNSGBM
additional_sortedDMSLnonexo[["CNS-GBM"]] <- sortedDM_CNSGBM
```

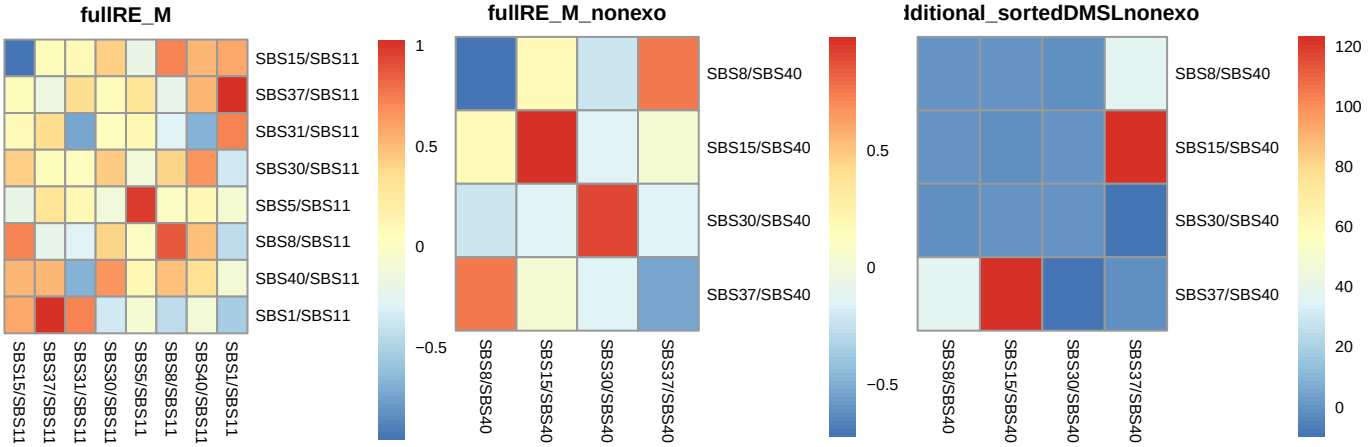
Betas


CNS−GBM
fullRE_M
Intercept
Slope
Estimate
LogR
CNS−GBM
fullRE_DMSL
PD
CNS−GBM
diagRE_DMSL
CNS−GBM
sparseRE_DMSL
CNS−GBM
fullRE_M_nonexo
CNS−GBM
fullRE_DMSL_nonexo
CNS−GBM
diagRE_DMSL_nonexo
CNS−GBM
sparseRE_DMSL_nonexo
1/11
15/11
30/11
31/11
37/11
40/11
5/11
8/11
15/40
30/40
37/40
8/40

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 6.6827492×10^{-5}.

Covariance matrices

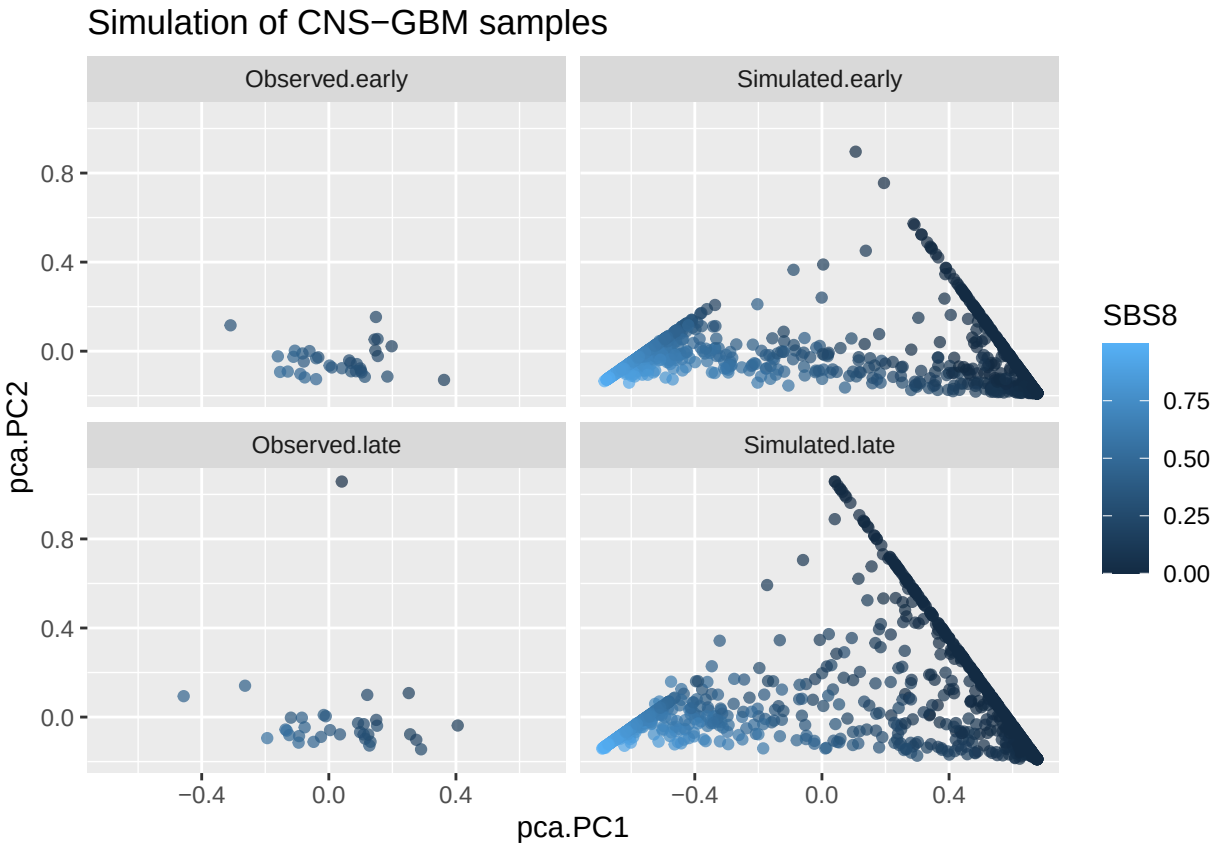
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



Simulation under inferred data

With fullRE DMSL:

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```

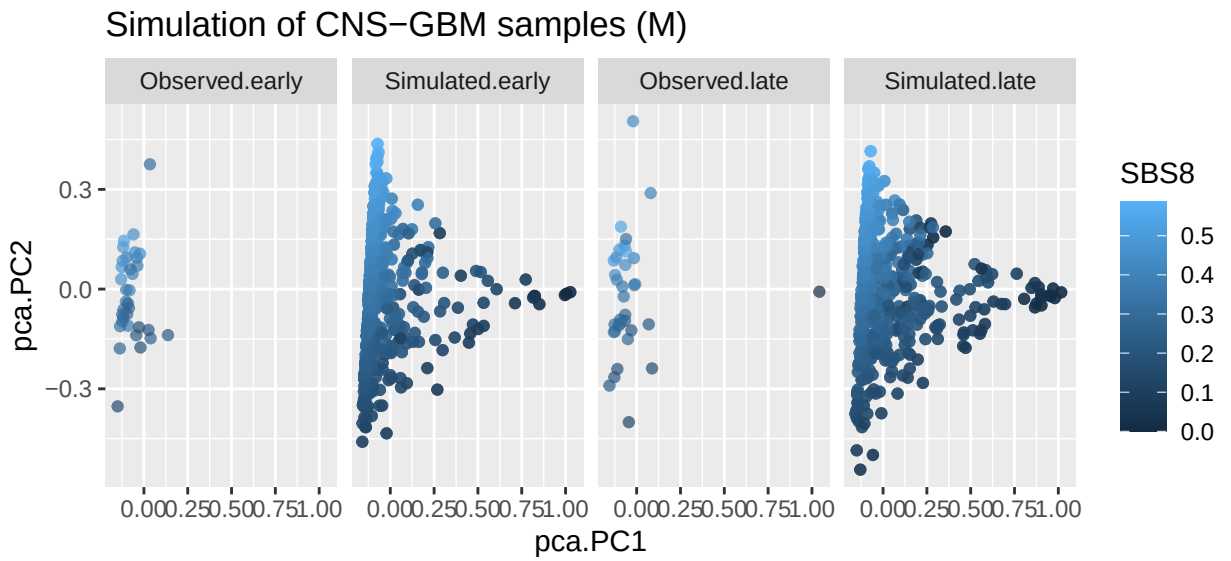


With fullRE M, this time the covariance matrix is positive semi-definite:

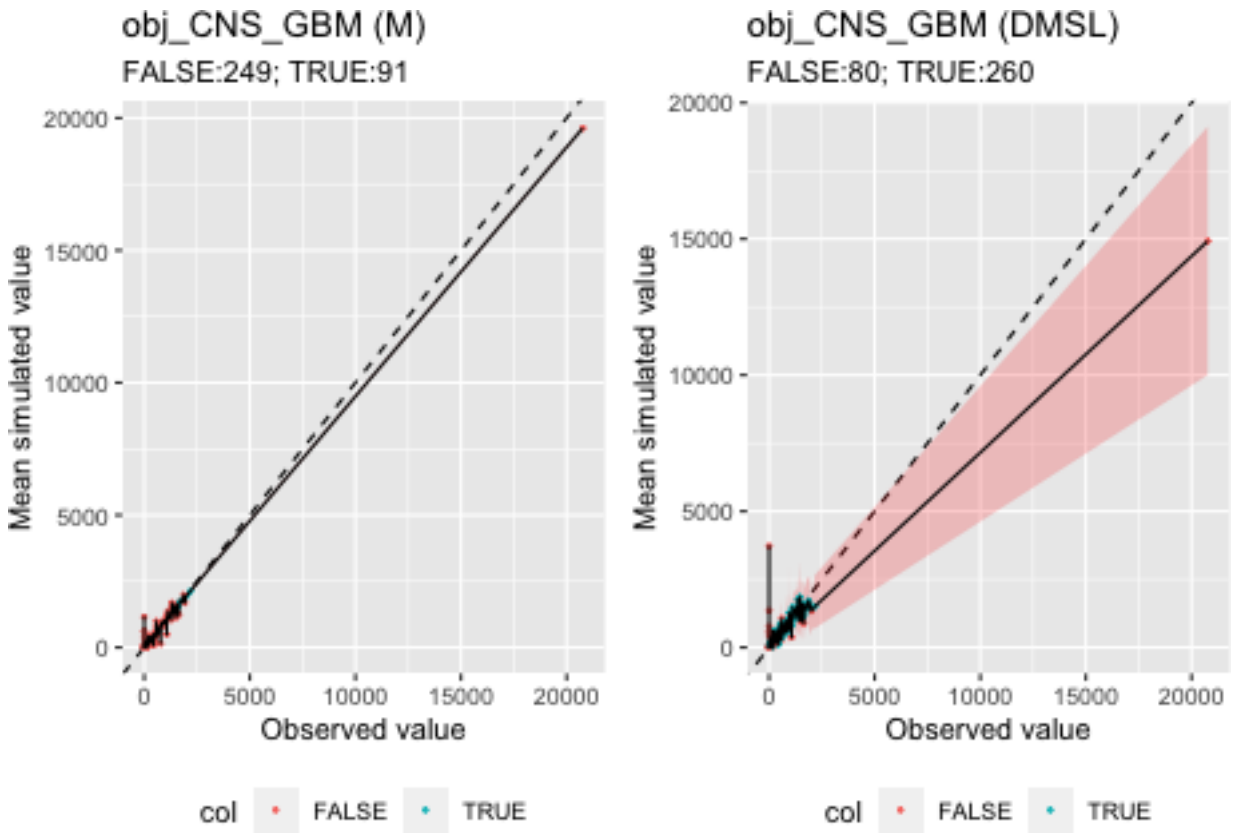
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



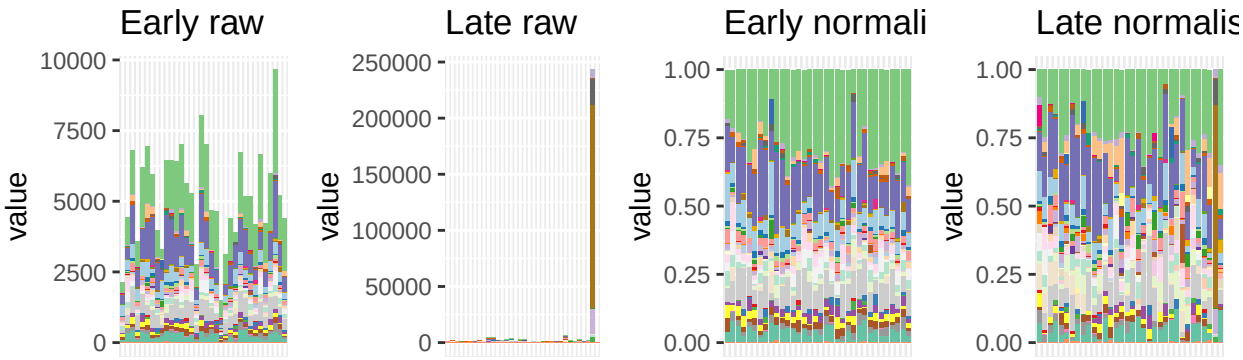
Ranked plot for coverage



Signatures from mutSigExtractor

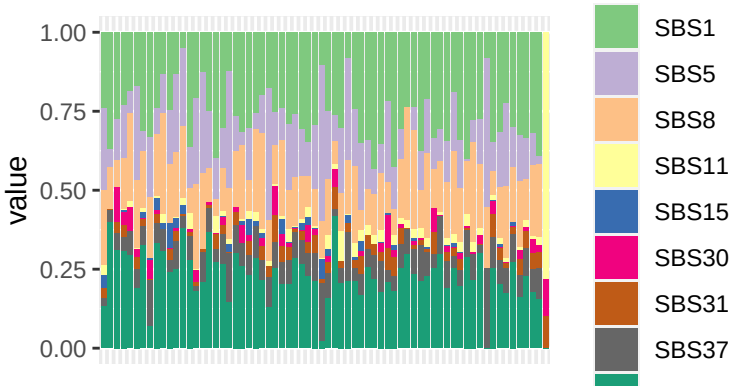
The signatures from mutSigExtractor are a bit more chaotic:

```
## [1] 34
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.

```
## Creating plot... it might take some time if the data are large. Number of samples: 68
```

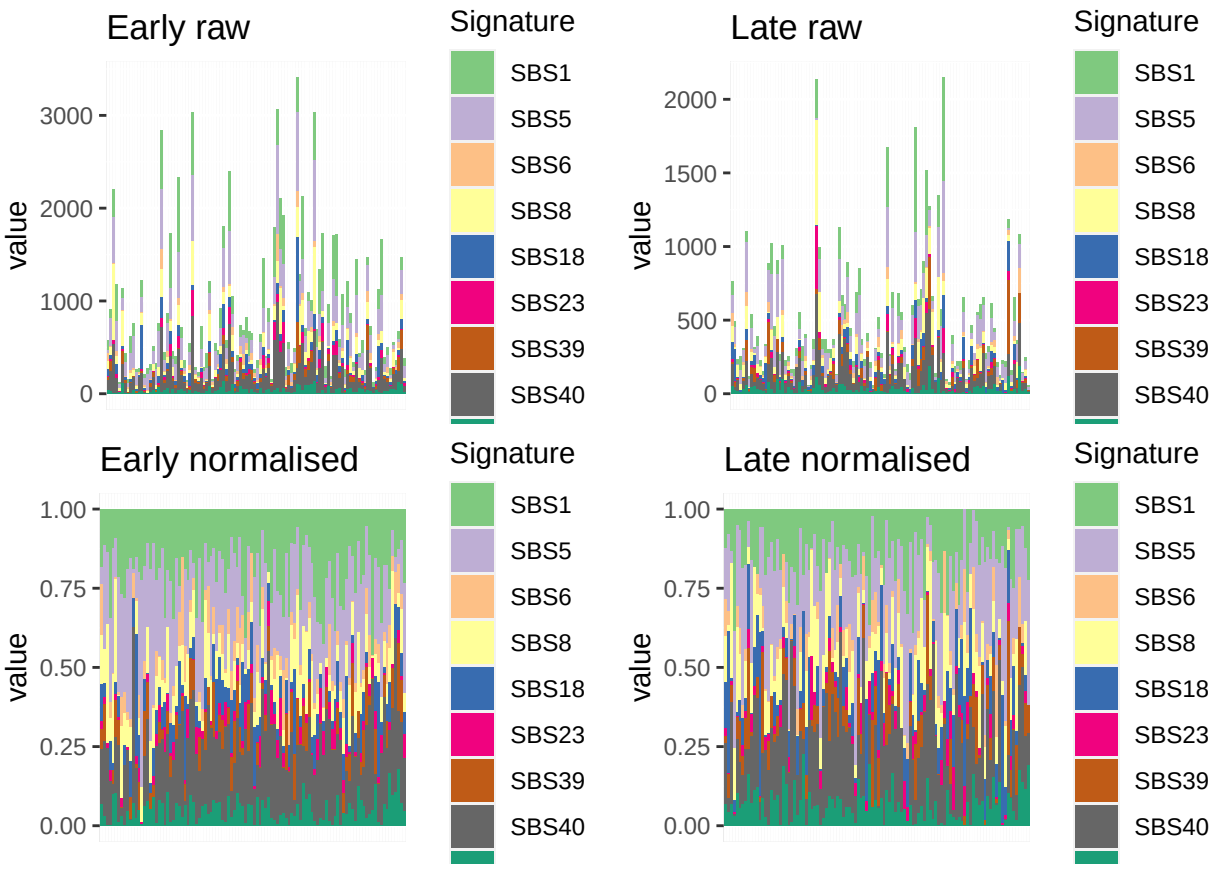


CNS-Medullo

Barplot and general statistics

```
## [1] 106
```

```
## Creating plot... it might take some time if the data are large. Number of samples: 106
## Creating plot... it might take some time if the data are large. Number of samples: 106
## Creating plot... it might take some time if the data are large. Number of samples: 106
## Creating plot... it might take some time if the data are large. Number of samples: 106
```



The number of samples and signatures is:

```
## [1] 212   9
```

The signatures are:

```
## [1] "SBS1"  "SBS5"  "SBS6"  "SBS8"  "SBS18" "SBS23" "SBS39" "SBS40" "SBS46"
```

Convergence table

Pretty much everything has converged in this case

```
##          value                                L2                    L1
## 1  CNS-Medullo    hessian_positivedefinite_bool             diagRE_M
## 2  CNS-Medullo    hessian_positivedefinite_bool             fullRE_M
## 3  CNS-Medullo    hessian_positivedefinite_bool           diagRE_DMDL
## 4  CNS-Medullo hessian_nonpositivedefinite_bool         fullRE_halfDM
## 5  CNS-Medullo hessian_nonpositivedefinite_bool           fullRE_DMDL
## 6  CNS-Medullo    hessian_positivedefinite_bool           diagRE_DMSL
## 7  CNS-Medullo    hessian_positivedefinite_bool         sparseRE_DMSL
## 8  CNS-Medullo hessian_nonpositivedefinite_bool           fullRE_DMSL
## 9  CNS-Medullo hessian_nonpositivedefinite_bool      fullRE_DMSL_SBS1
## 10 CNS-Medullo    hessian_positivedefinite_bool       fullRE_M_nonexo
## 11 CNS-Medullo    hessian_positivedefinite_bool     diagRE_DMSL_nonexo
## 12 CNS-Medullo    hessian_positivedefinite_bool   sparseRE_DMSL_nonexo
## 13 CNS-Medullo    hessian_positivedefinite_bool     fullRE_DMSL_nonexo
```

```
## 14 CNS-Medullo    hessian_positivedefinite_bool       fullRE_DMDL_nonexo
## 15 CNS-Medullo hessian_nonpositivedefinite_bool fullRE_DMDL_sortednonexo
```

As nonexo DMSL has already converged, we don't re-run anything.

Potentially problematic signatures

We notice that there are no truly problematic signatures

```
colSums(obj_CNS_Medullo$Y == 0)/nrow(obj_CNS_Medullo$Y)
```

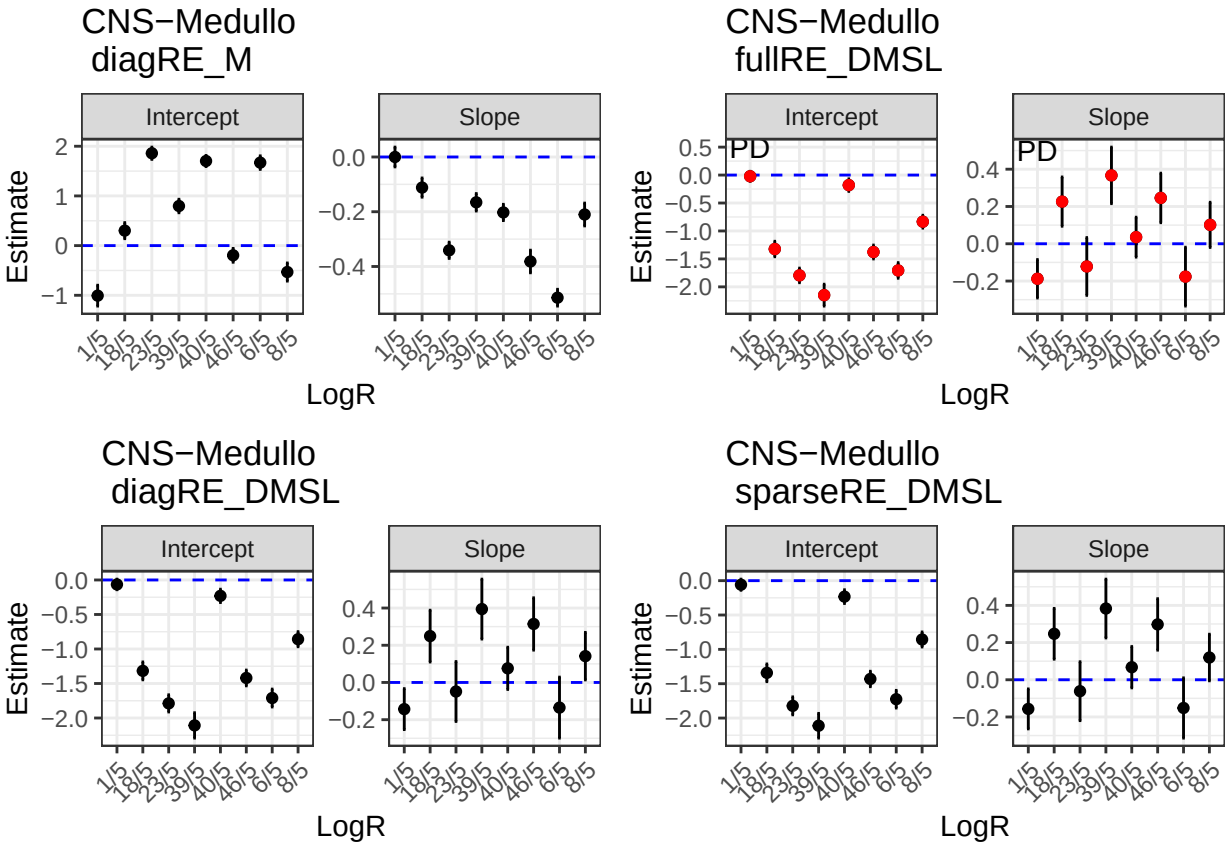
```
##        SBS1        SBS5        SBS6        SBS8       SBS18       SBS23
## 0.004716981 0.056603774 0.264150943 0.089622642 0.155660377 0.235849057
##       SBS39       SBS40       SBS46
## 0.353773585 0.066037736 0.099056604
```

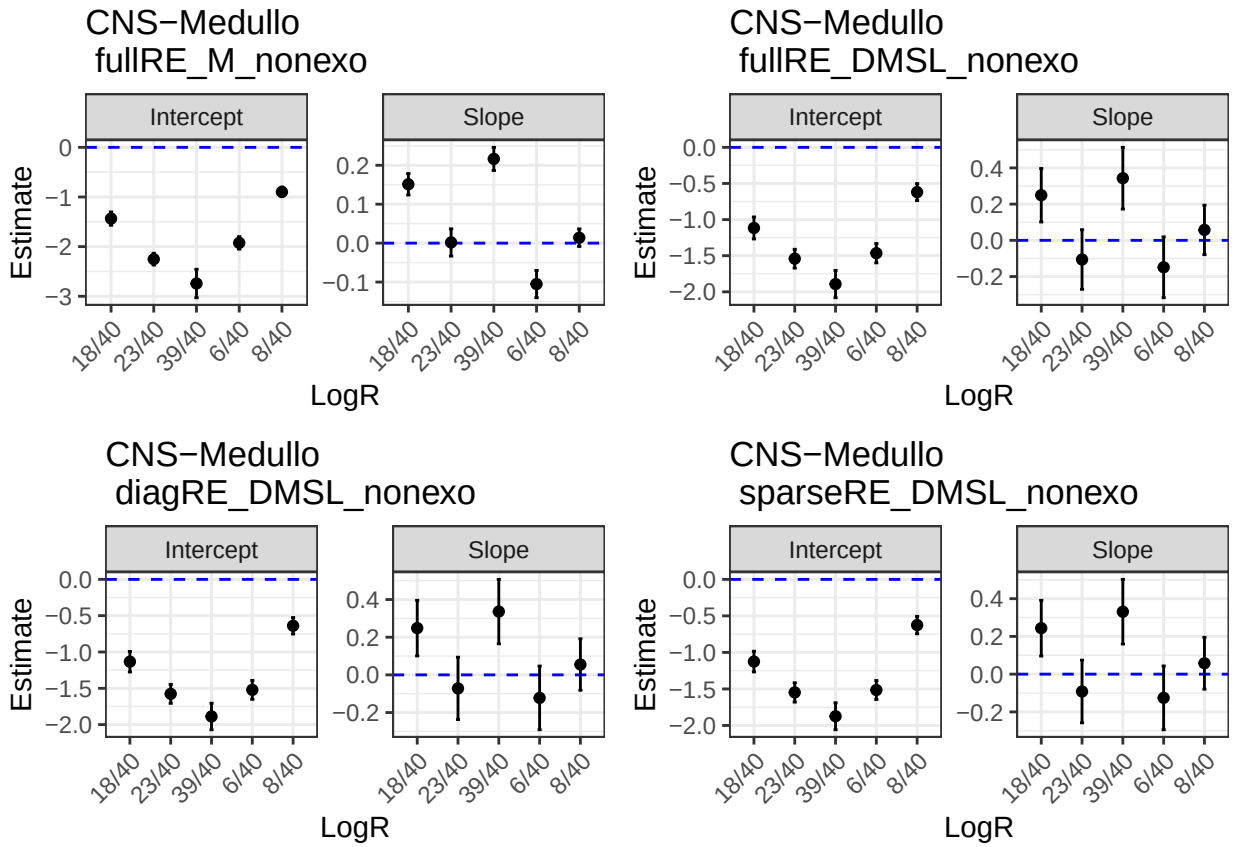
```
colSums(obj_CNS_Medullo$Y)/sum(obj_CNS_Medullo$Y)
```

```
##       SBS1       SBS5       SBS6       SBS8      SBS18      SBS23      SBS39
## 0.19177483 0.22946904 0.03737123 0.11614418 0.07466844 0.03836035 0.05498025
##      SBS40      SBS46
## 0.21065558 0.04657610
```

Betas

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

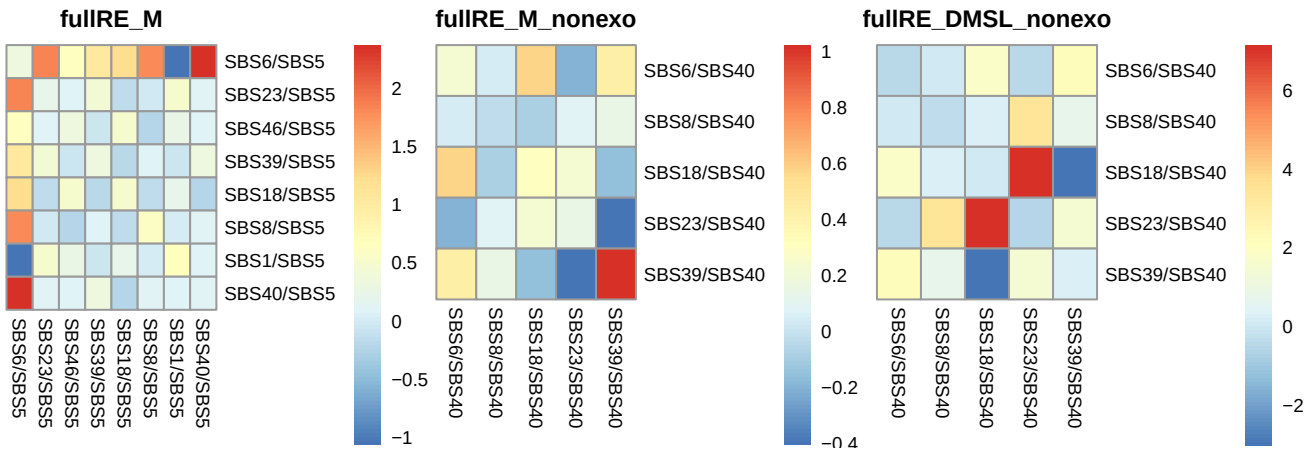




We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.0677062.

Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



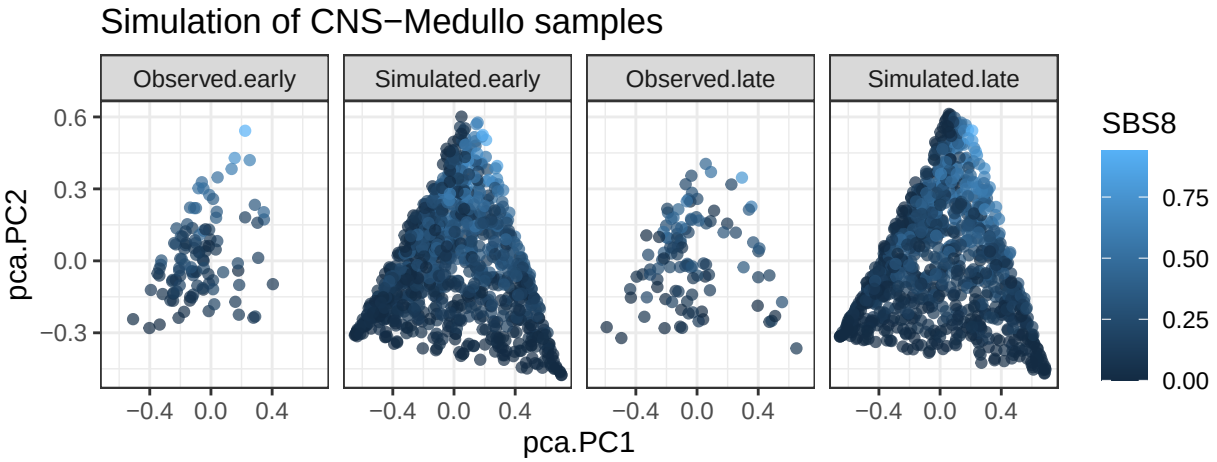
Simulation under inferred data

```
obj_CNS_Medullo_nonexo <- give_subset_sigs_TMBobj(obj_CNS_Medullo, nonexogenous$V1)
```

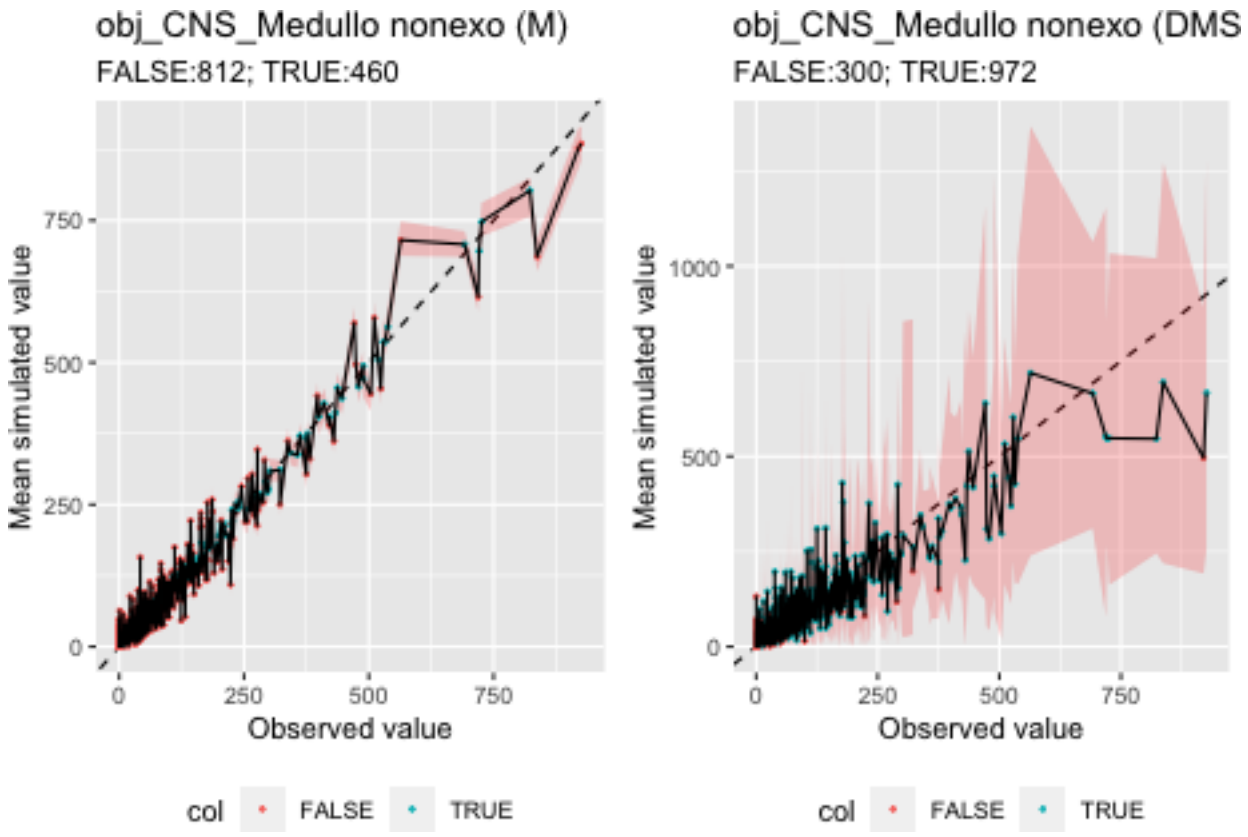
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



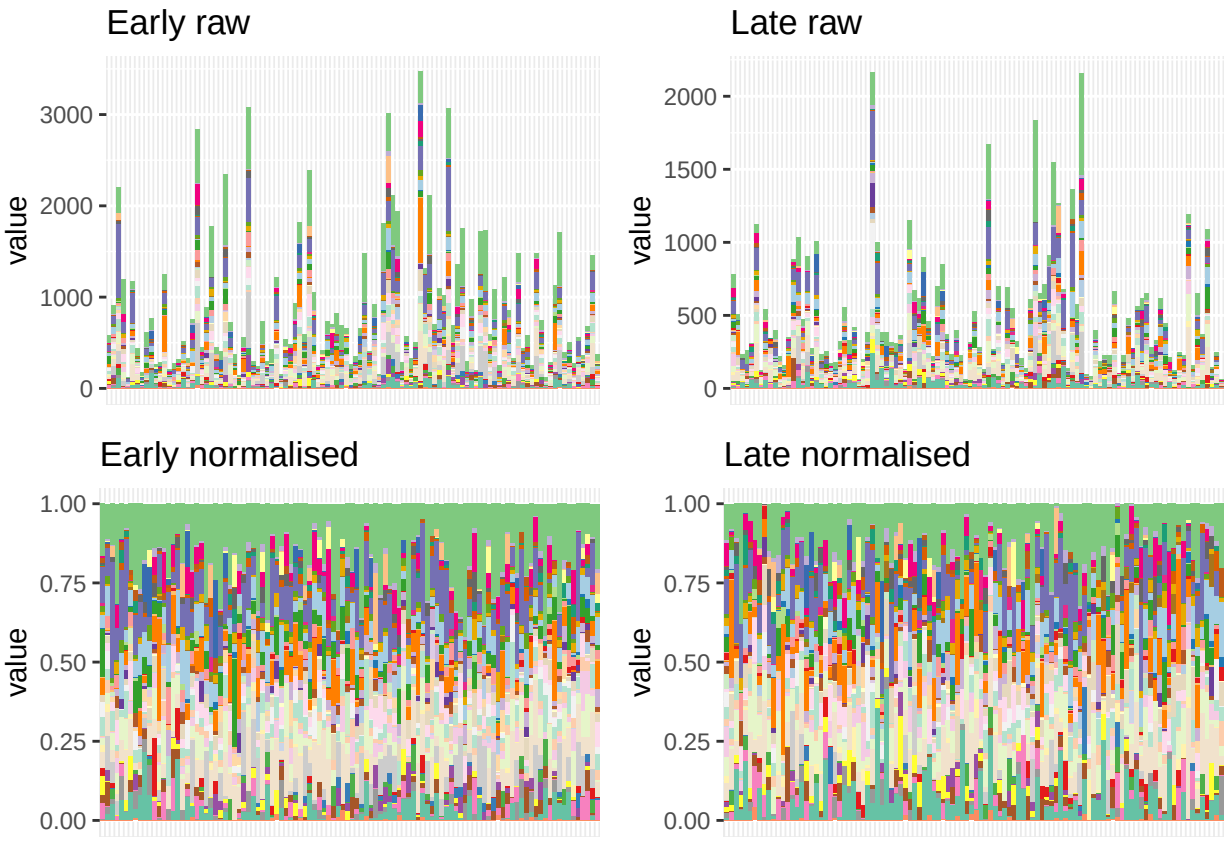
Ranked plot for coverage



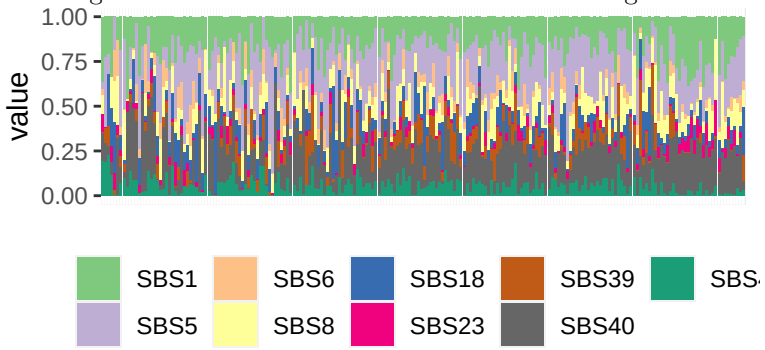
Signatures from mutSigExtractor

The signatures from mutSigExtractor are a bit more chaotic:

```
## [1] 106
```



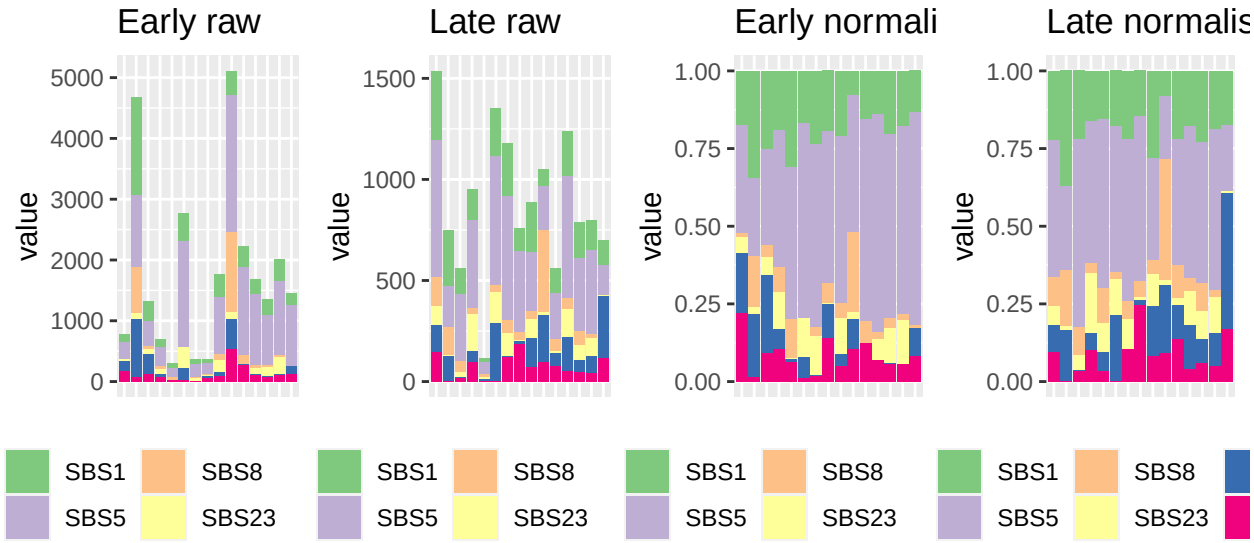
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



CNS-Oligo

Barplot and general statistics

```
## [1] 15
```



The number of samples and signatures is:

```
## [1] 30  6
```

The signatures are:

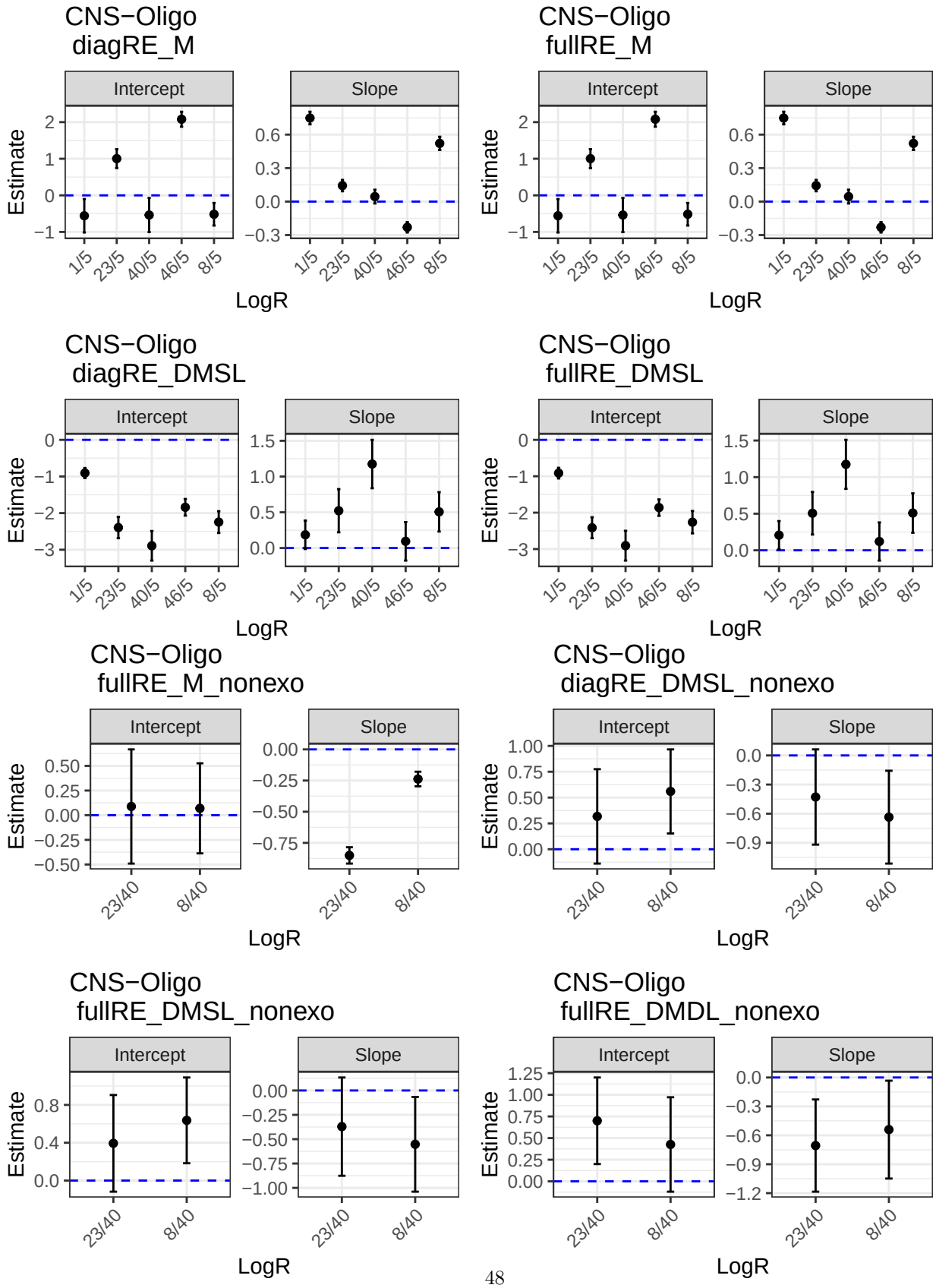
```
## [1] "SBS1"  "SBS5"  "SBS8"  "SBS23" "SBS40" "SBS46"
```

Convergence table

Pretty much everything has converged

```
## [1] value L2    L1
## <0 rows> (or 0-length row.names)
```

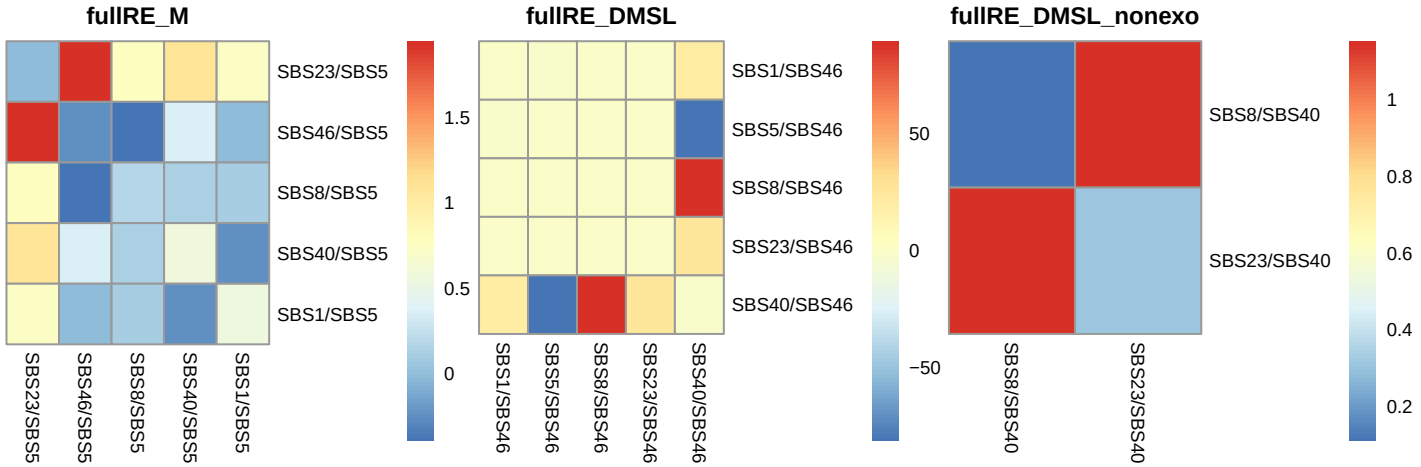
Betas


CNS−Oligo
diagRE_M
Intercept
Slope
Estimate
LogR
1/5
23/5
40/5
46/5
8/5
CNS−Oligo
fullRE_M
CNS−Oligo
diagRE_DMSL
CNS−Oligo
fullRE_DMSL
CNS−Oligo
fullRE_M_nonexo
23/40
8/40
CNS−Oligo
diagRE_DMSL_nonexo
CNS−Oligo
fullRE_DMSL_nonexo
CNS−Oligo
fullRE_DMDL_nonexo

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.5220955.

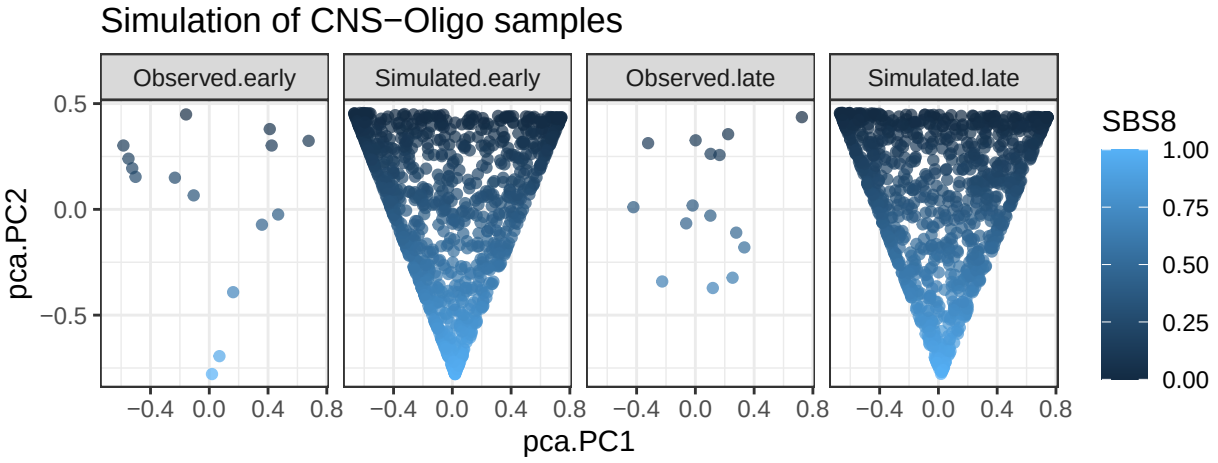
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



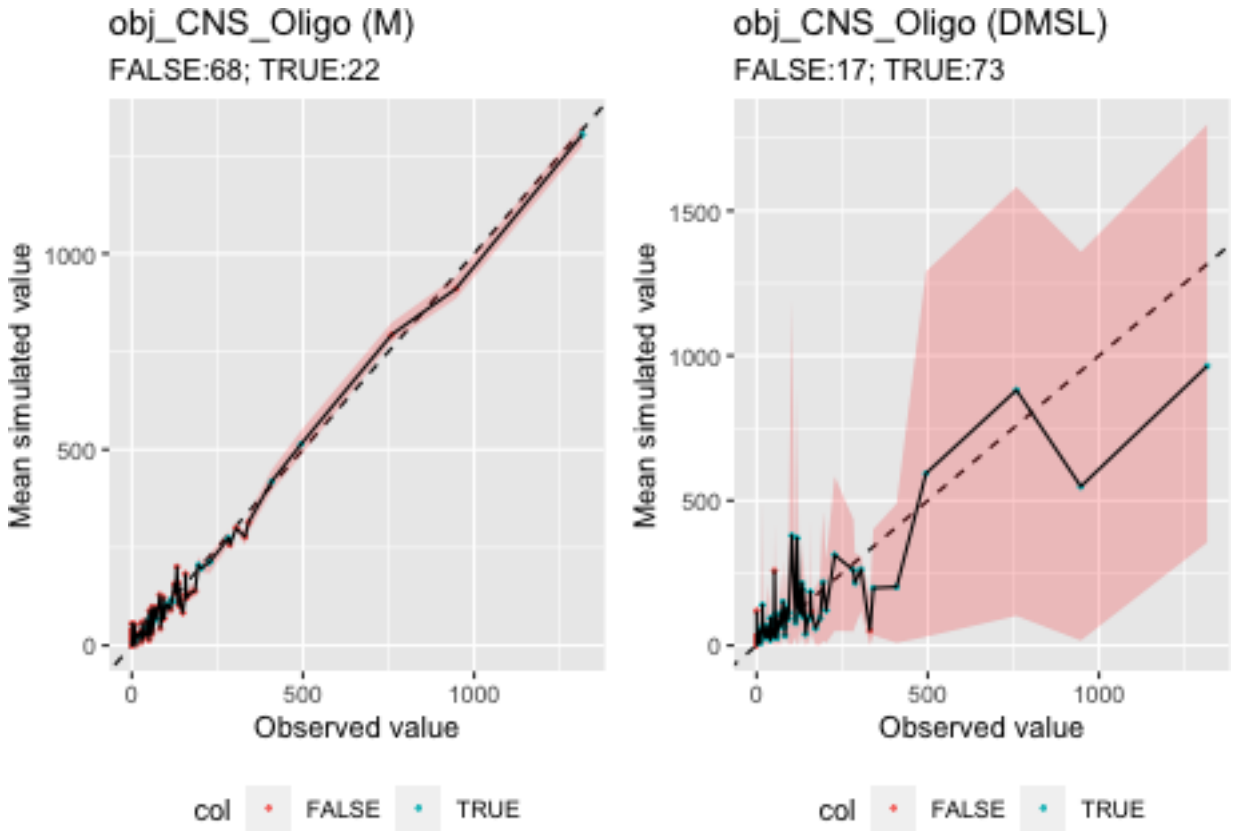
Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



Ranked plot for coverage

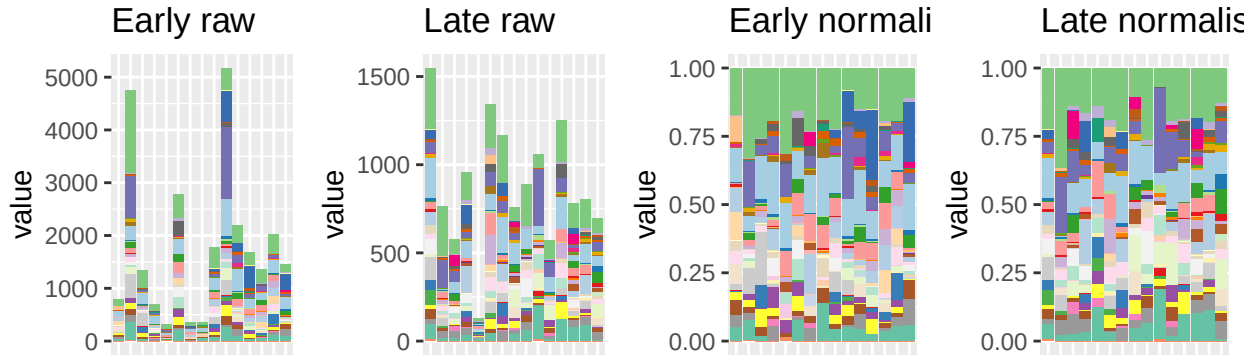
The values for DMSL nonexo look considerably better than for M nonexo.



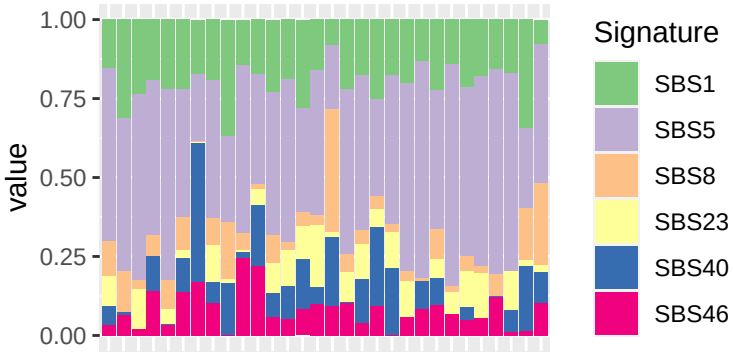
Signatures from mutSigExtractor

These are the signatures from mutSigExtractor:

```
## [1] 15
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.

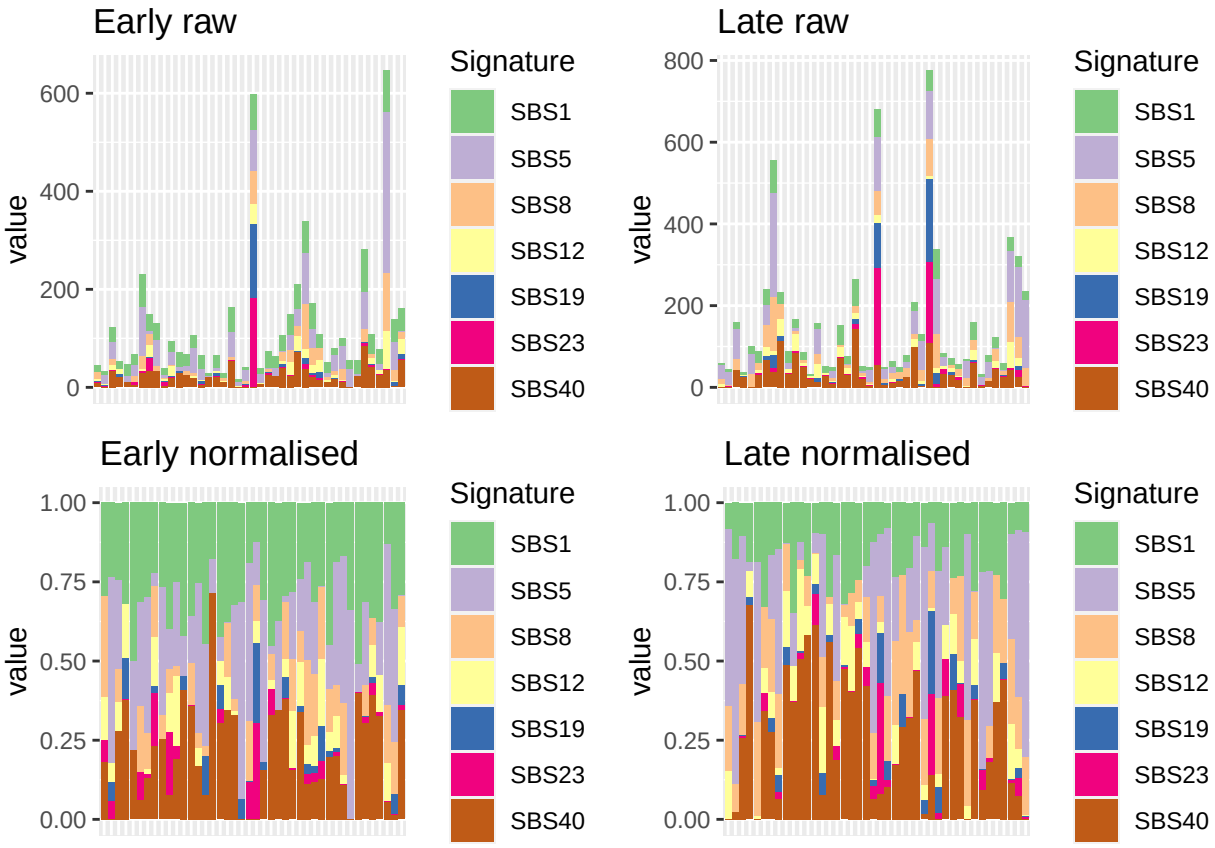


CNS-PiloAstro

CNS-PiloAstro

Barplot and general statistics

```
## [1] 42
```



The number of samples and signatures is:

```
## [1] 84  7
```

The signatures are:

```
## [1] "SBS1"  "SBS5"  "SBS8"  "SBS12" "SBS19" "SBS23" "SBS40"
```

Convergence table

We have converged results for everything except for full RE DM, in the case of all signatures (with only nonexo everything has).

```
##            value                                L2                     L1
## 1  CNS-PiloAstro    hessian_positivedefinite_bool                diagRE_M
## 2  CNS-PiloAstro    hessian_positivedefinite_bool                fullRE_M
## 3  CNS-PiloAstro    hessian_positivedefinite_bool             diagRE_DMDL
## 4  CNS-PiloAstro hessian_nonpositivedefinite_bool            fullRE_halfDM
## 5  CNS-PiloAstro hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  CNS-PiloAstro    hessian_positivedefinite_bool              diagRE_DMSL
## 7  CNS-PiloAstro    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  CNS-PiloAstro hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  CNS-PiloAstro hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 CNS-PiloAstro    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 CNS-PiloAstro    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 CNS-PiloAstro    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 CNS-PiloAstro    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 CNS-PiloAstro hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 CNS-PiloAstro    hessian_positivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M to fit fullRE_DMSL (all sigs, as the one with nonexo has already converged)

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

If we use the values of the fullRE M as initial values for the fullRE DMSL still do not converge:

```
## [1] FALSE
```

Potentially problematic signatures

We notice that there are no truly problematic signatures (SBS15 has the most zeros; 50%).

```
colSums(obj_CNS_PiloAstro$Y == 0)/nrow(obj_CNS_PiloAstro$Y)
```

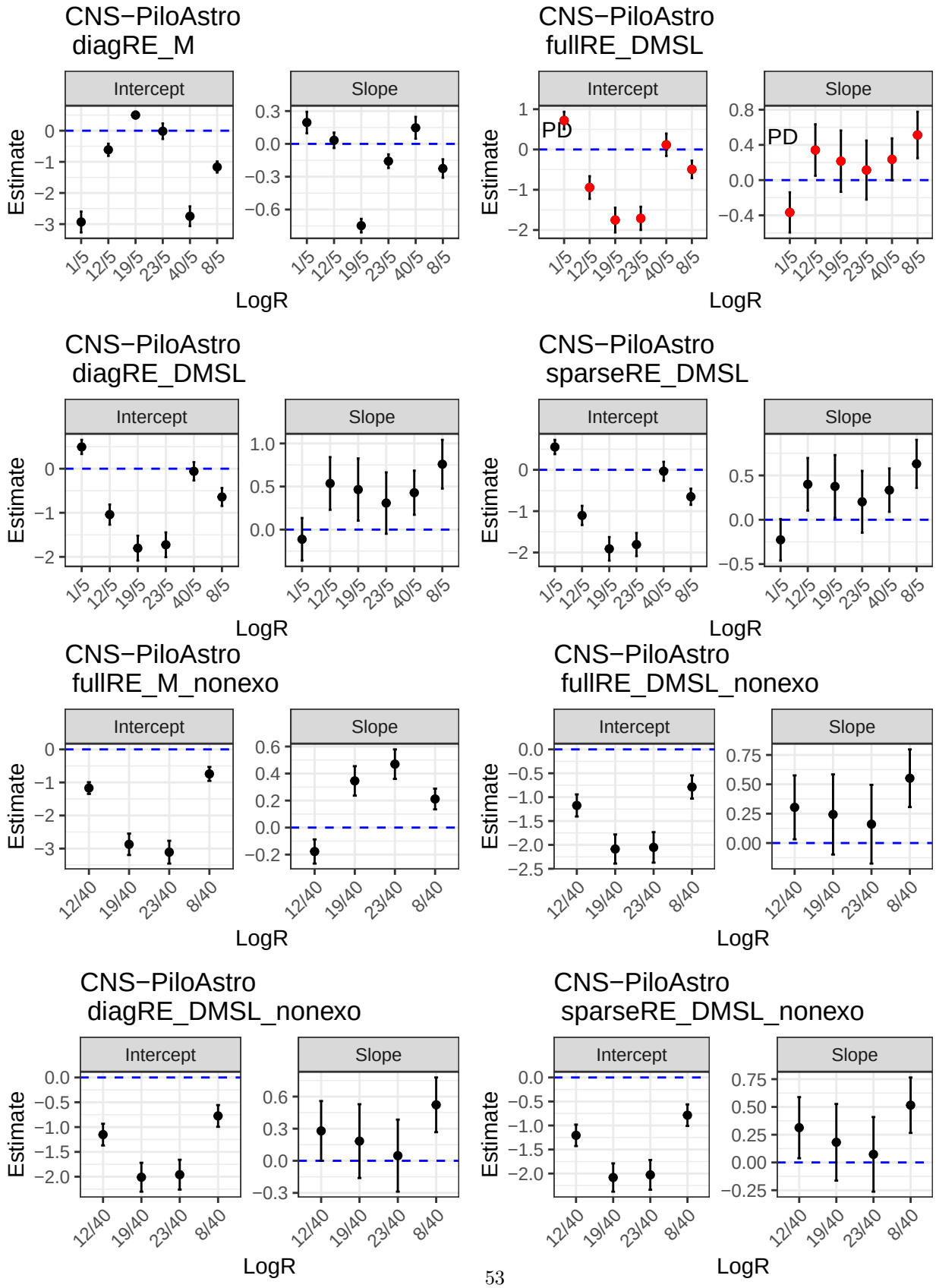
```
##      SBS1      SBS5      SBS8     SBS12     SBS19     SBS23     SBS40
## 0.0000000 0.2261905 0.2023810 0.2857143 0.5119048 0.5000000 0.1190476
```

```
colSums(obj_CNS_PiloAstro$Y)/sum(obj_CNS_PiloAstro$Y)
```

```
##       SBS1       SBS5       SBS8      SBS12      SBS19      SBS23      SBS40
## 0.19840611 0.26357297 0.14212187 0.07313631 0.05894073 0.06749128 0.19633073
```

SBS19 and SBS23 are quite sparse.

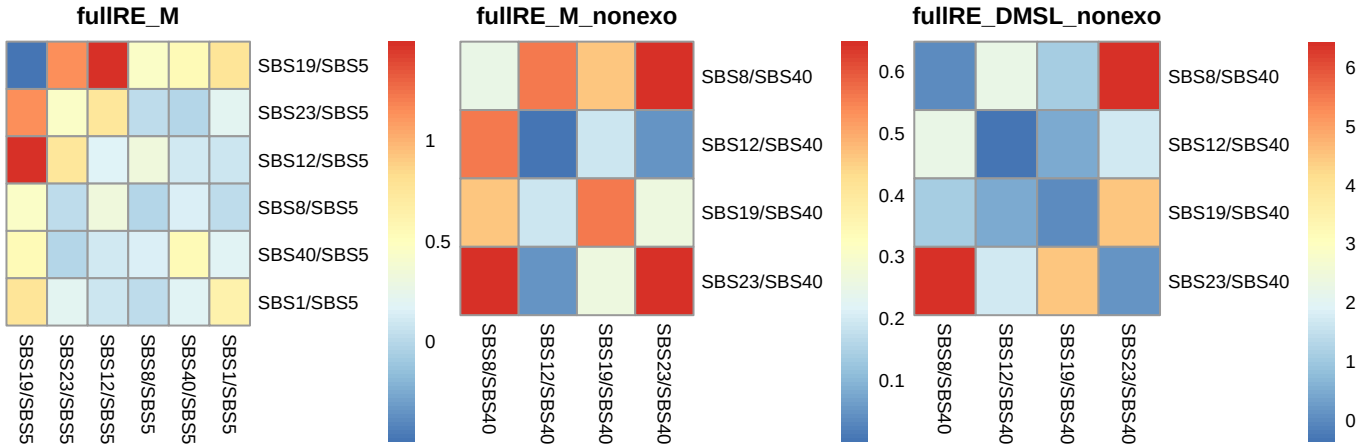
Betas


CNS−PiloAstro
diagRE_M
Intercept
Slope
Estimate
LogR
1/5
12/5
19/5
23/5
40/5
8/5
CNS−PiloAstro
fullRE_DMSL
PD
CNS−PiloAstro
diagRE_DMSL
CNS−PiloAstro
sparseRE_DMSL
CNS−PiloAstro
fullRE_M_nonexo
12/40
19/40
23/40
8/40
CNS−PiloAstro
fullRE_DMSL_nonexo
CNS−PiloAstro
diagRE_DMSL_nonexo
CNS−PiloAstro
sparseRE_DMSL_nonexo

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.2632004.

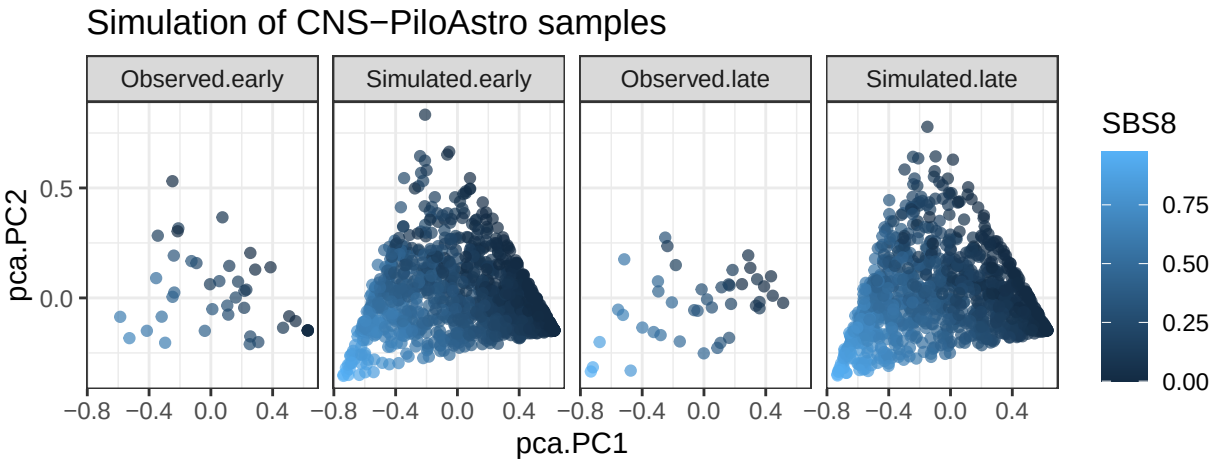
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



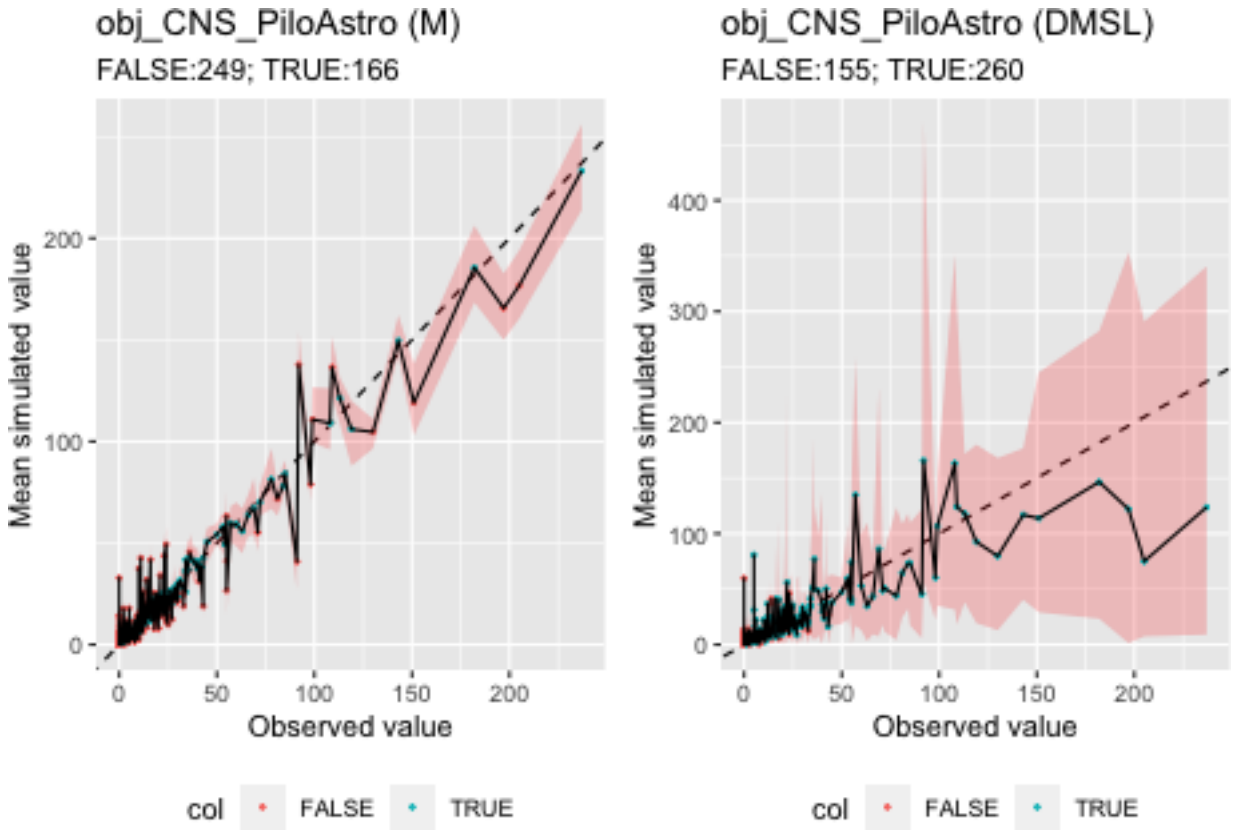
Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



Ranked plot for coverage

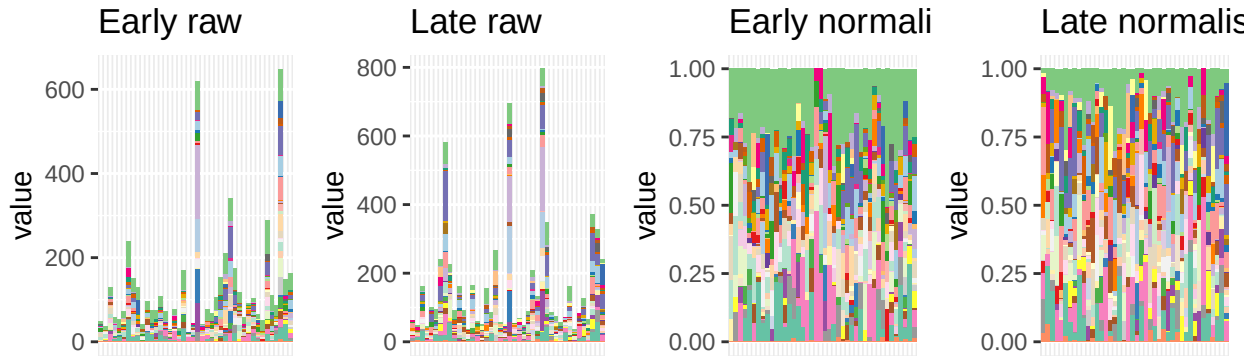
The strange pattern in DMSL is worth pointing out. The beta coefficients are too very different between M and DMSL for nonexo.



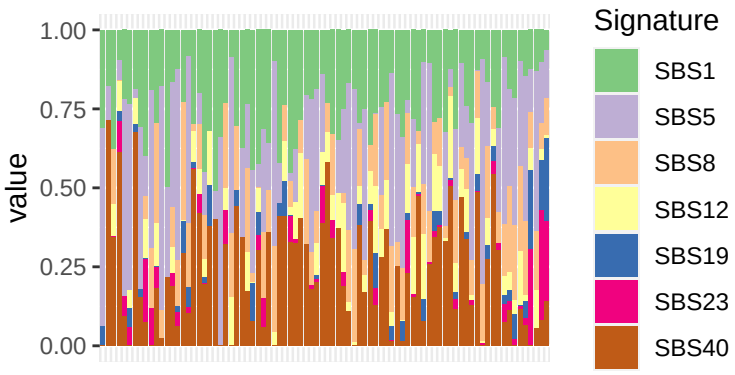
Signatures from mutSigExtractor

The signatures from mutSigExtractor are a bit more chaotic:

```
## [1] 42
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations, with perhaps SBS9 being slightly found in the rightmost side preferentially.

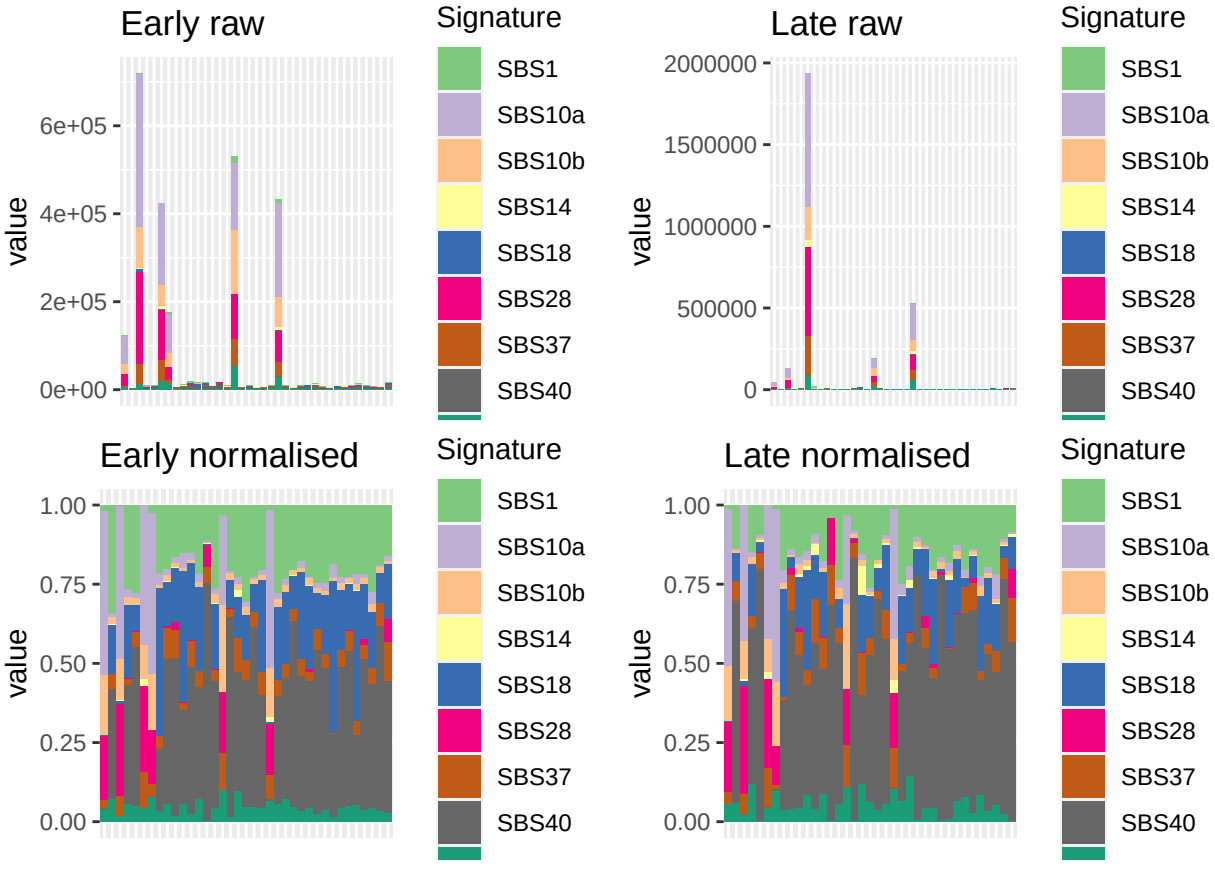


ColoRect-AdenoCA

ColoRect-AdenoCA

Barplot and general statistics

```
## [1] 37
```



The number of samples and signatures is:

```
## [1] 74  9
```

The signatures are:

```
## [1] "SBS1"   "SBS10a" "SBS10b" "SBS14"  "SBS18"  "SBS28"  "SBS37"  "SBS40"
## [9] "SBS44"
```

Convergence table

We only have converged results for the multinomial with diag RE, when including all mutations. For exogenous mutations, full DMSL is has not converged.

```
##               value                                L2                    L1
## 1  ColoRect-AdenoCA    hessian_positivedefinite_bool                diagRE_M
## 2  ColoRect-AdenoCA hessian_nonpositivedefinite_bool                fullRE_M
## 3  ColoRect-AdenoCA hessian_nonpositivedefinite_bool             diagRE_DMDL
## 4  ColoRect-AdenoCA hessian_nonpositivedefinite_bool            fullRE_halfDM
## 5  ColoRect-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  ColoRect-AdenoCA    hessian_positivedefinite_bool              diagRE_DMSL
## 7  ColoRect-AdenoCA    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  ColoRect-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  ColoRect-AdenoCA hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 ColoRect-AdenoCA    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 ColoRect-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 ColoRect-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 ColoRect-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 ColoRect-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 ColoRect-AdenoCA                          Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Now the fullM doesn't converge (even though the original fullRE M nonexo did converge?), so I cannot use all the parameters to find the starting parameters of the DM, as some are NA. I can however use some, such as beta.

What parameters are NA? Betas, logsd and covariances are not NA. Therefore, we use these values as starting values, and give an empty random effects matrix.

I get the error "gradient function must rerurn a number vector of length 43" for some reason I don't understand - it's as though the initial values I am giving are not correct.

Potentially problematic signatures

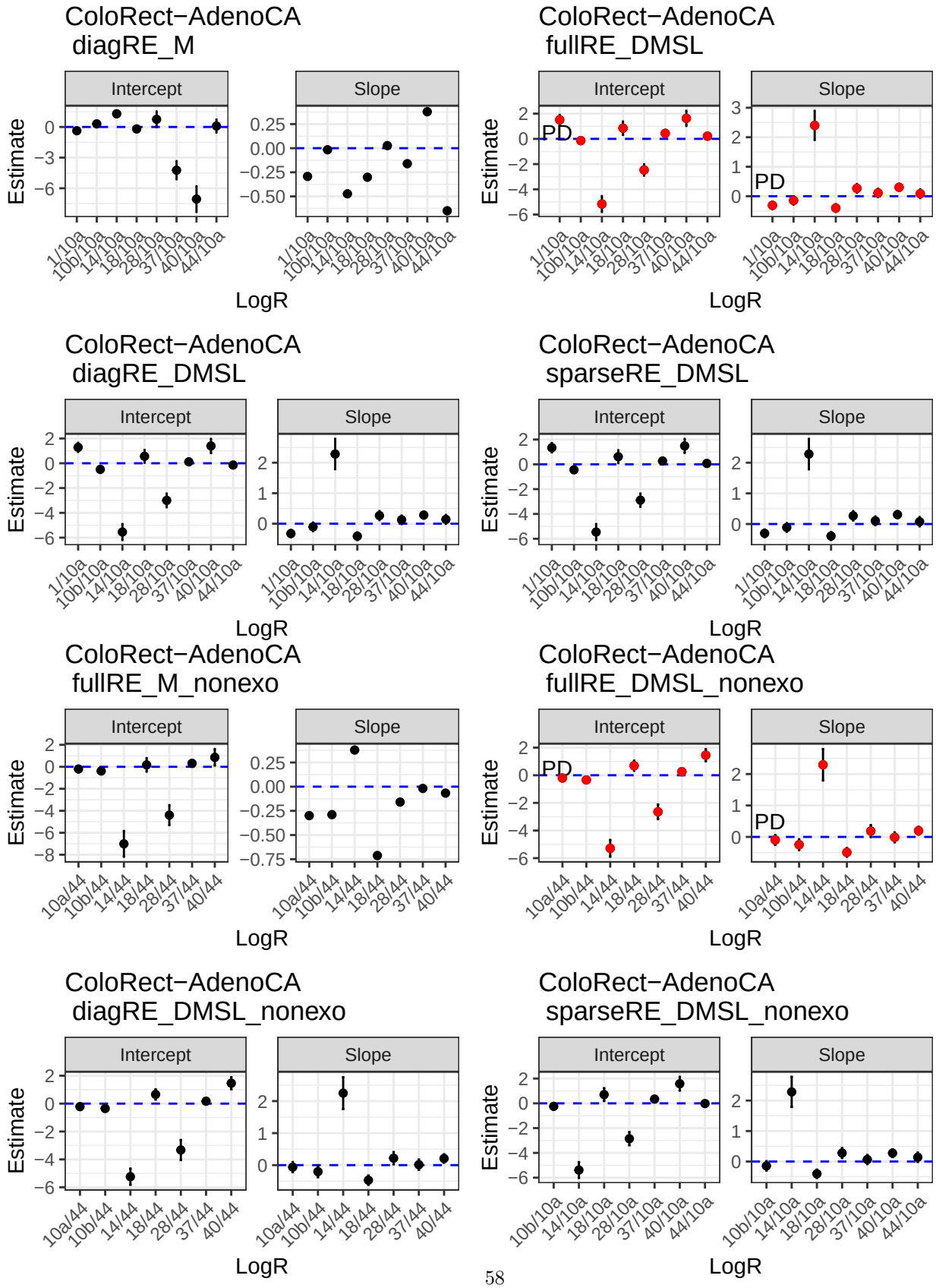
```
colSums(obj_ColoRect_AdenoCA$Y == 0)/nrow(obj_ColoRect_AdenoCA$Y)
```

```
##       SBS1     SBS10a     SBS10b      SBS14      SBS18      SBS28      SBS37
## 0.02702703 0.04054054 0.02702703 0.68918919 0.13513514 0.52702703 0.04054054
##      SBS40      SBS44
## 0.09459459 0.05405405
```

```
colSums(obj_ColoRect_AdenoCA$Y)/sum(obj_ColoRect_AdenoCA$Y)
```

```
##       SBS1     SBS10a     SBS10b      SBS14      SBS18      SBS28      SBS37
## 0.02342633 0.39302667 0.13415977 0.01502674 0.01674129 0.22524153 0.09998130
##      SBS40      SBS44
## 0.03731777 0.05507859
```

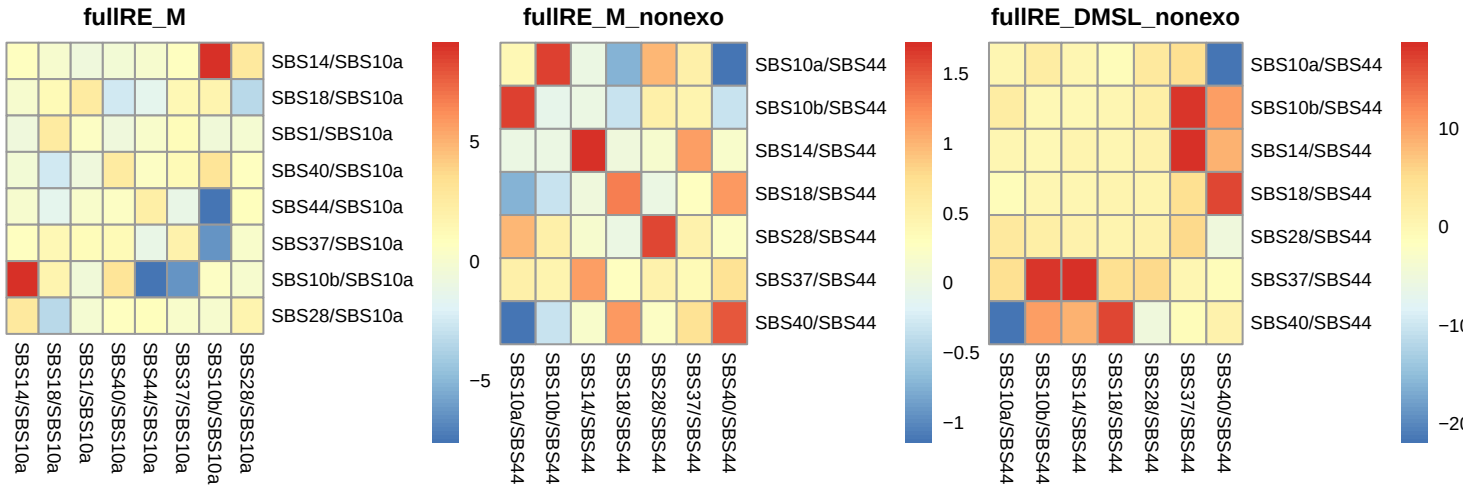
Betas


ColoRect−AdenoCA
diagRE_M
ColoRect−AdenoCA
fullRE_DMSL
ColoRect−AdenoCA
diagRE_DMSL
ColoRect−AdenoCA
sparseRE_DMSL
ColoRect−AdenoCA
fullRE_M_nonexo
ColoRect−AdenoCA
fullRE_DMSL_nonexo
ColoRect−AdenoCA
diagRE_DMSL_nonexo
ColoRect−AdenoCA
sparseRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR
PD
1/10a
10b/10a
14/10a
18/10a
28/10a
37/10a
40/10a
44/10a
10a/44
10b/44
14/44
18/44
28/44
37/44
40/44

We use the results from the diagonal RE single lambda DM nonexo to test for differential abundance, giving a p-value of $8.8714208 \times 10^{-16}$.

Covariance matrices

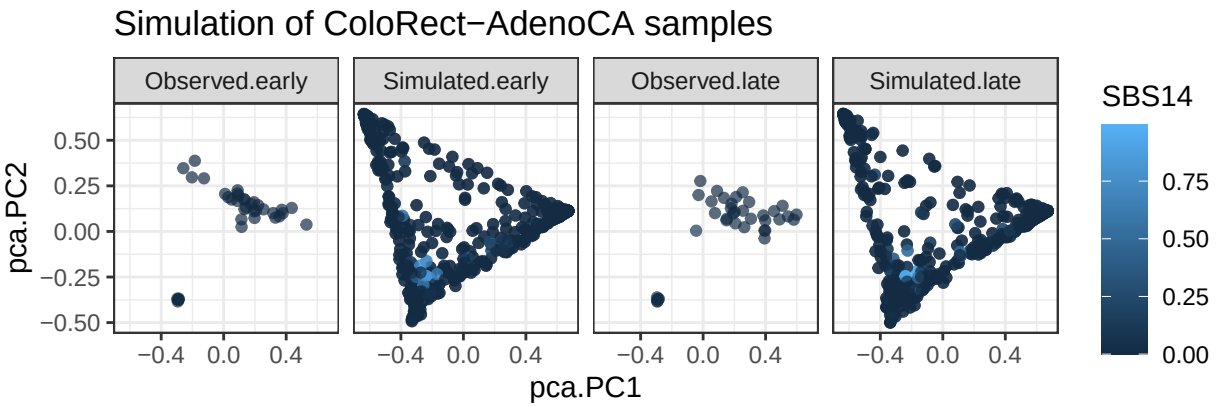
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



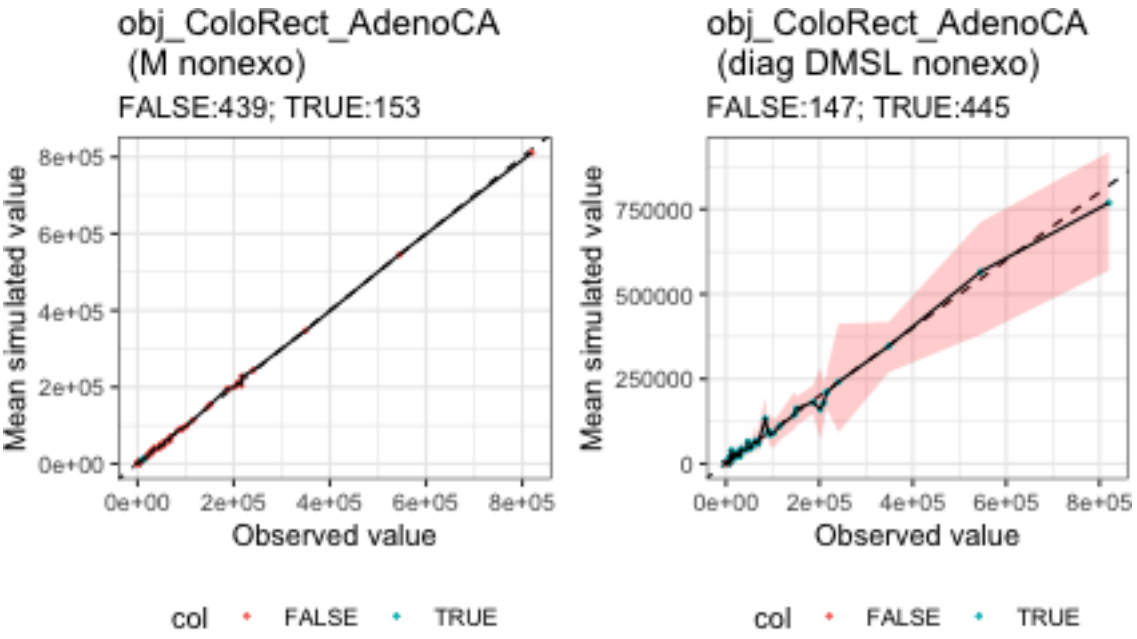
Simulation under inferred data

I am simulating using the full effects multinomial, because the function needs to be adapted to diagDMSL.

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```

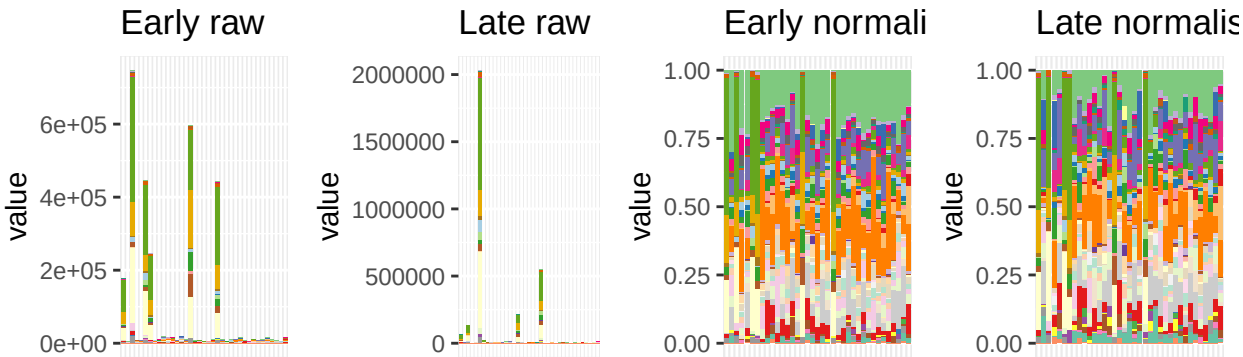


Ranked plot for coverage



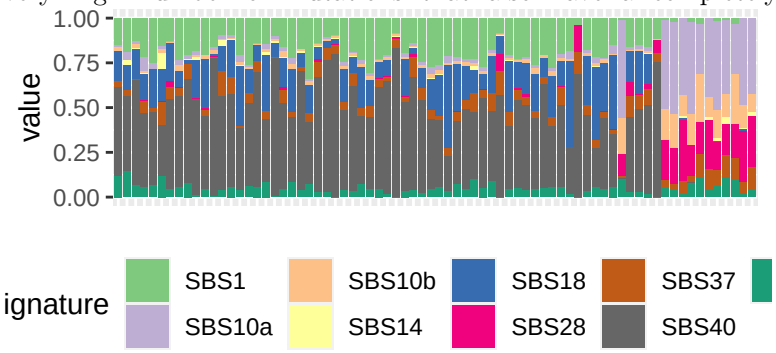
Signatures from mutSigExtractor

```
## [1] 37
```



Exposures sorted by increasing number of mutations: very clearly there are a few samples with

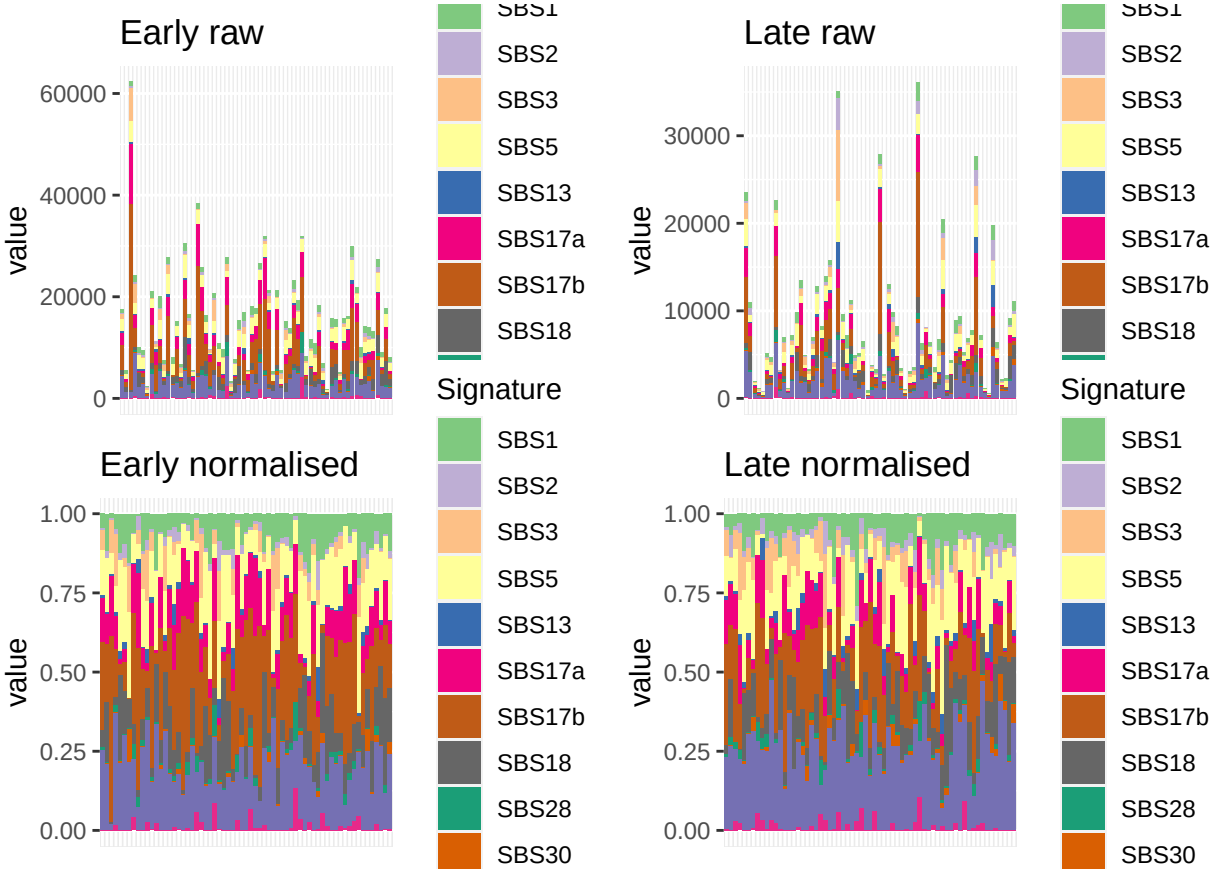
very high number of mutations that also have a completely different mutational signature exposure.



Eso-AdenoCA

Barplot and general statistics

```
## [1] 65
```



The number of samples and signatures is:

```
## [1] 130  12
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS13"  "SBS17a" "SBS17b" "SBS18"
```

```
##  [9] "SBS28"  "SBS30"  "SBS40"  "SBS46"
```

Convergence table

None of the fullRE have converged when including all signatures. When including nonexo, all but fullRE_DMSL_nonexo (using either the highest absolute signature or SBS1) have converged.

```
##          value                                L2                      L1
## 1  Eso-AdenoCA    hessian_positivedefinite_bool                 diagRE_M
## 2  Eso-AdenoCA hessian_nonpositivedefinite_bool                 fullRE_M
## 3  Eso-AdenoCA hessian_nonpositivedefinite_bool              diagRE_DMDL
## 4  Eso-AdenoCA hessian_nonpositivedefinite_bool            fullRE_halfDM
## 5  Eso-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  Eso-AdenoCA    hessian_positivedefinite_bool              diagRE_DMSL
## 7  Eso-AdenoCA    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  Eso-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  Eso-AdenoCA hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Eso-AdenoCA    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Eso-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Eso-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Eso-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 Eso-AdenoCA    hessian_positivedefinite_bool       fullRE_DMDL_nonexo
## 15 Eso-AdenoCA    hessian_positivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo

which has a positive-semidefinite covariance matrix, i.e. has converged

```
## [1] TRUE
```

The fullRE DMSL hasn't, though:

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

```
## [1] FALSE
```

Running fullRE DMSL, this time without sorting, doesn't converge either:

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Bafflingly, using two lambdas does:

Potentially problematic signatures

We notice that there are no truly problematic signatures (SBS30 has the most zeros; 54.6%).

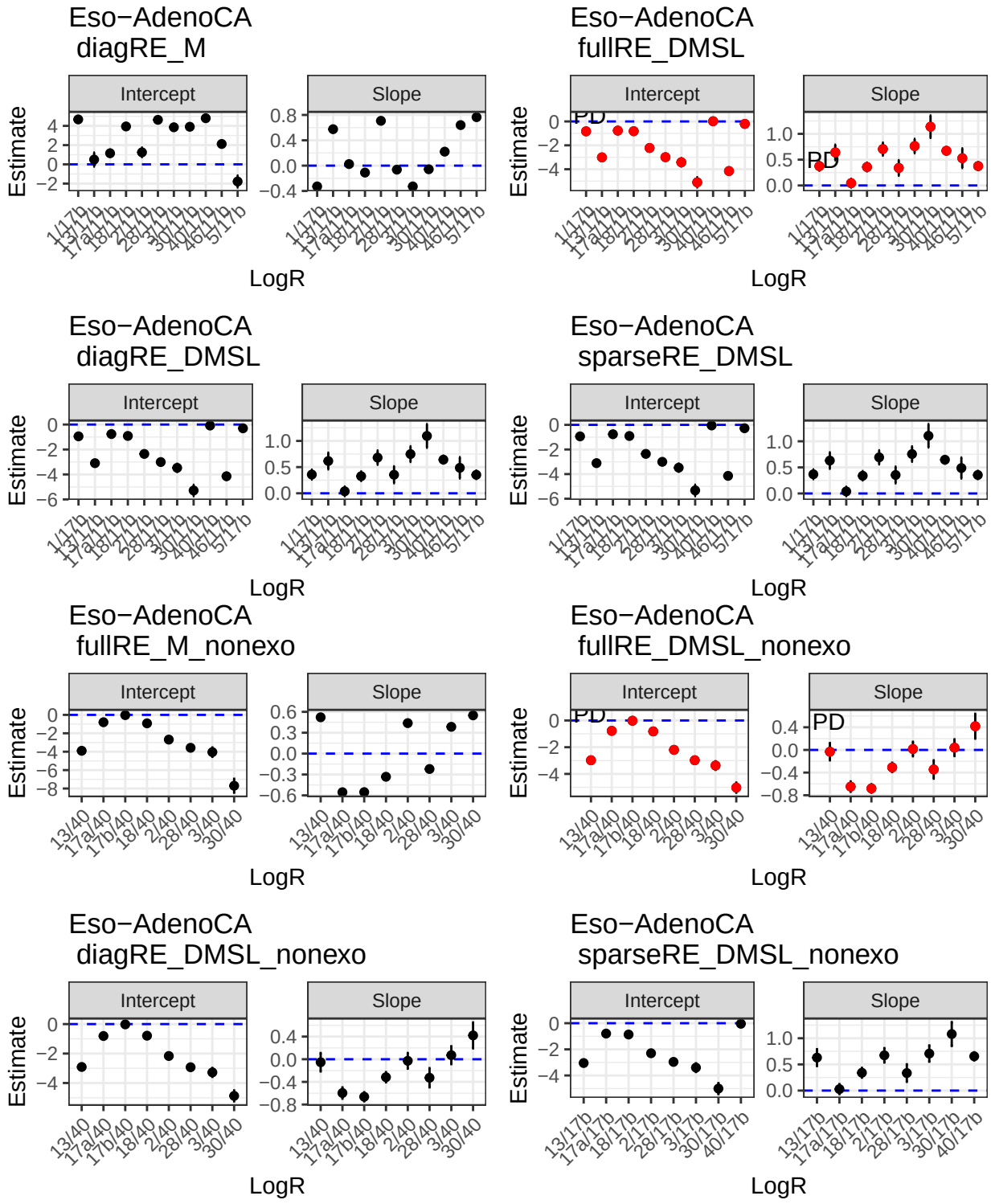
```
colSums(obj_Eso_AdenoCA$Y == 0)/nrow(obj_Eso_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5       SBS13      SBS17a
## 0.000000000 0.023076923 0.392307692 0.000000000 0.215384615 0.038461538
##      SBS17b       SBS18       SBS28       SBS30       SBS40       SBS46
## 0.007692308 0.038461538 0.238461538 0.546153846 0.000000000 0.476923077
```

```
colSums(obj_Eso_AdenoCA$Y)/sum(obj_Eso_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5       SBS13      SBS17a
## 0.069743929 0.022981455 0.042124010 0.135767687 0.017294837 0.118553858
##      SBS17b       SBS18       SBS28       SBS30       SBS40       SBS46
## 0.265599550 0.088385597 0.020133817 0.006873288 0.198223396 0.014318577
```

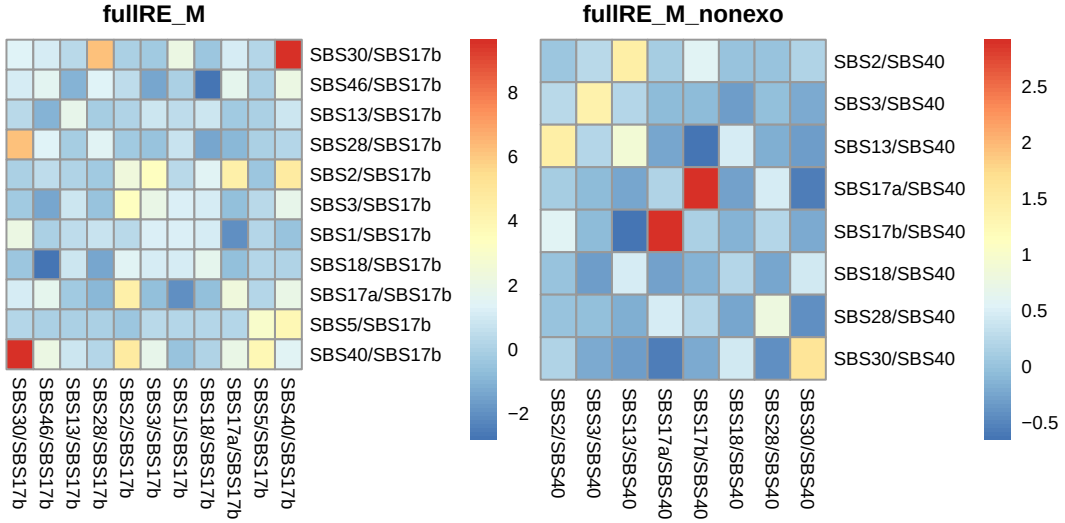
Betas



We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of $2.4465743 \times 10^{-18}$.

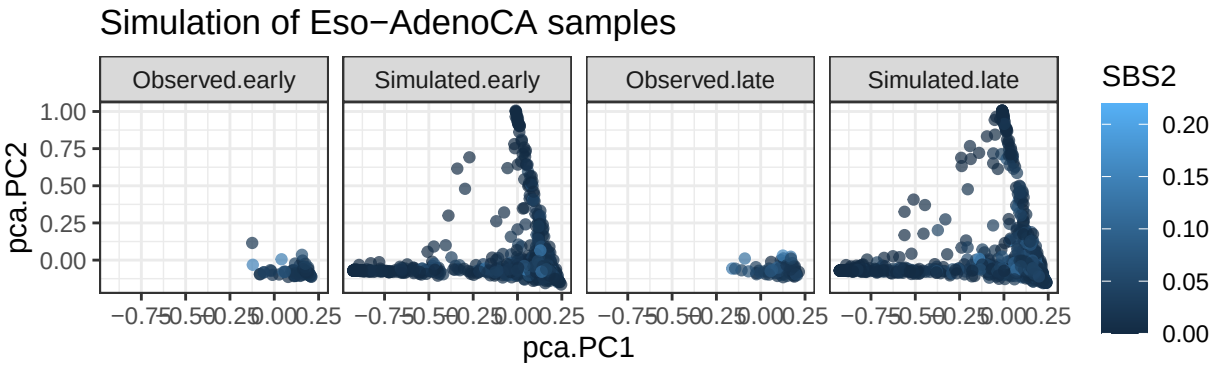
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

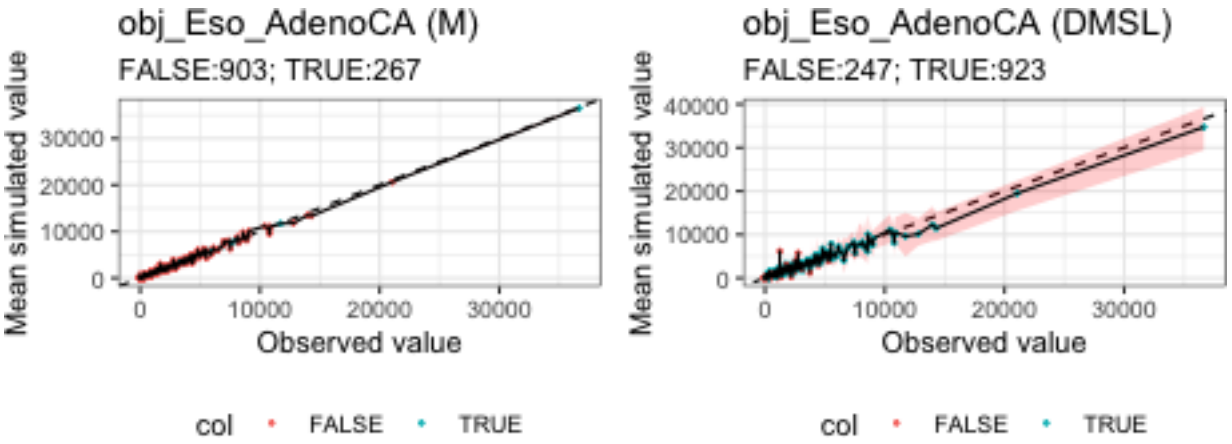


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



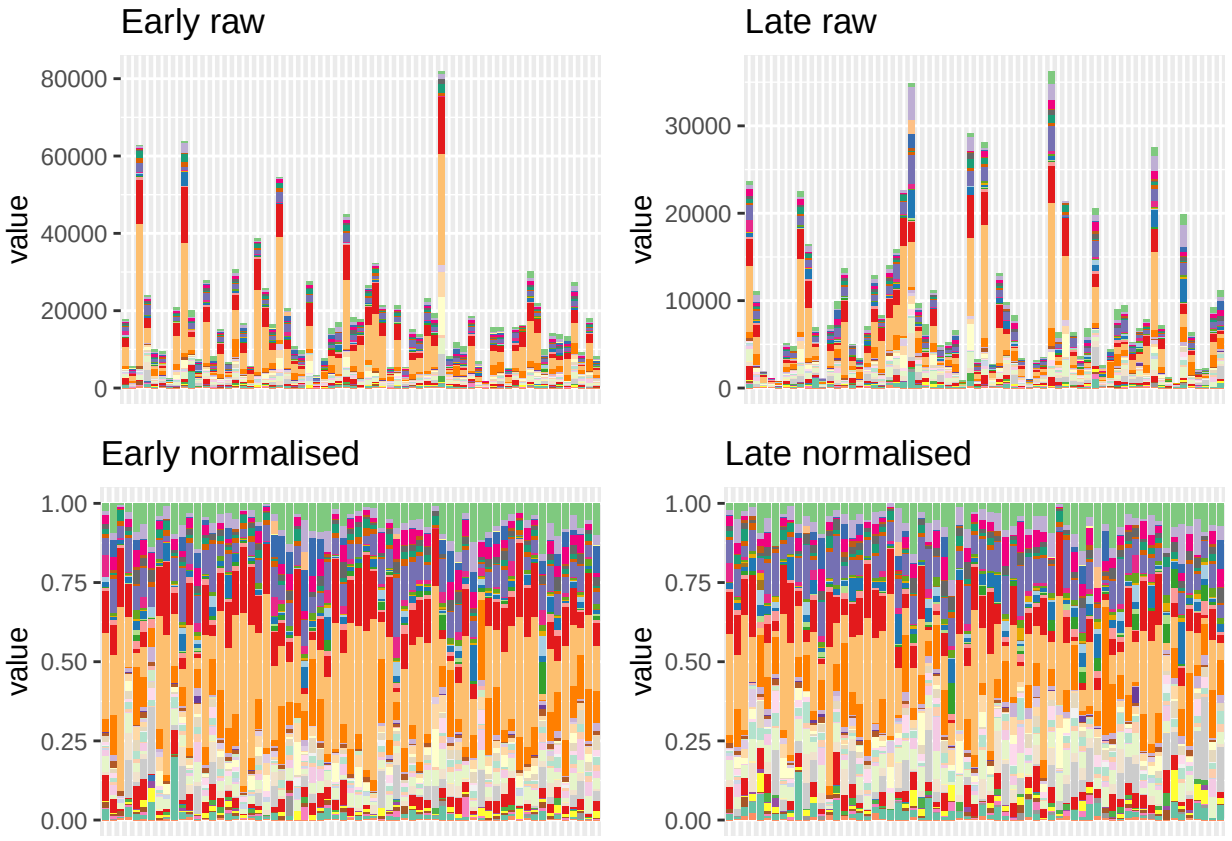
Ranked plot for coverage



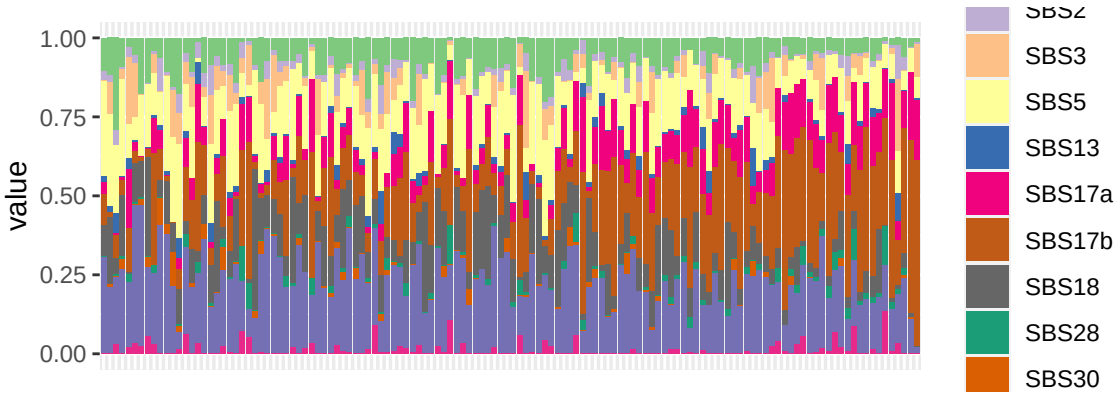
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 65
```



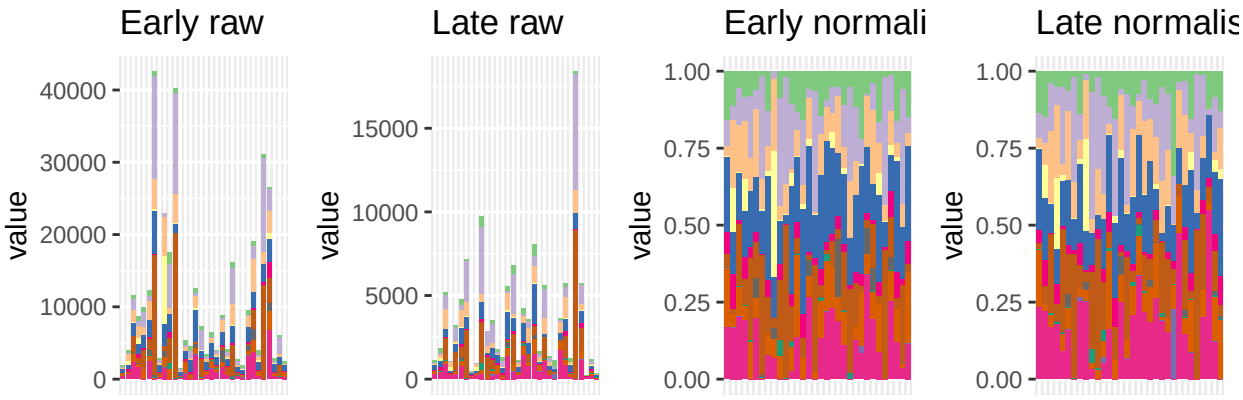
Exposures sorted by increasing number of mutations: there is a trend of samples with more mutations having more SBS17b and less SBS5, relatively.



Head-SCC

Barplot and general statistics

```
## [1] 32
```



The number of samples and signatures is:

```
## [1] 64 12
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS4"   "SBS5"   "SBS7b"  "SBS13"  "SBS16"
##  [9] "SBS17b" "SBS18"  "SBS33"  "SBS40"
```

Convergence table

We don't have converged results for the multinomial with full RE, but for nonexogenous signatures everything has.

```
##       value                                L2                  L1
## 1  Head-SCC    hessian_positivedefinite_bool          diagRE_M
## 2  Head-SCC hessian_nonpositivedefinite_bool          fullRE_M
## 3  Head-SCC hessian_nonpositivedefinite_bool        diagRE_DMDL
## 4  Head-SCC hessian_nonpositivedefinite_bool       fullRE_halfDM
## 5  Head-SCC hessian_nonpositivedefinite_bool        fullRE_DMDL
## 6  Head-SCC    hessian_positivedefinite_bool        diagRE_DMSL
## 7  Head-SCC    hessian_positivedefinite_bool      sparseRE_DMSL
## 8  Head-SCC hessian_nonpositivedefinite_bool        fullRE_DMSL
```

```
## 9  Head-SCC hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Head-SCC    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Head-SCC    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Head-SCC    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Head-SCC    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 Head-SCC hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Head-SCC                         Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

We don't need refitting, as the results have already converged.

Potentially problematic signatures

SBS33 is likely to be problematic.

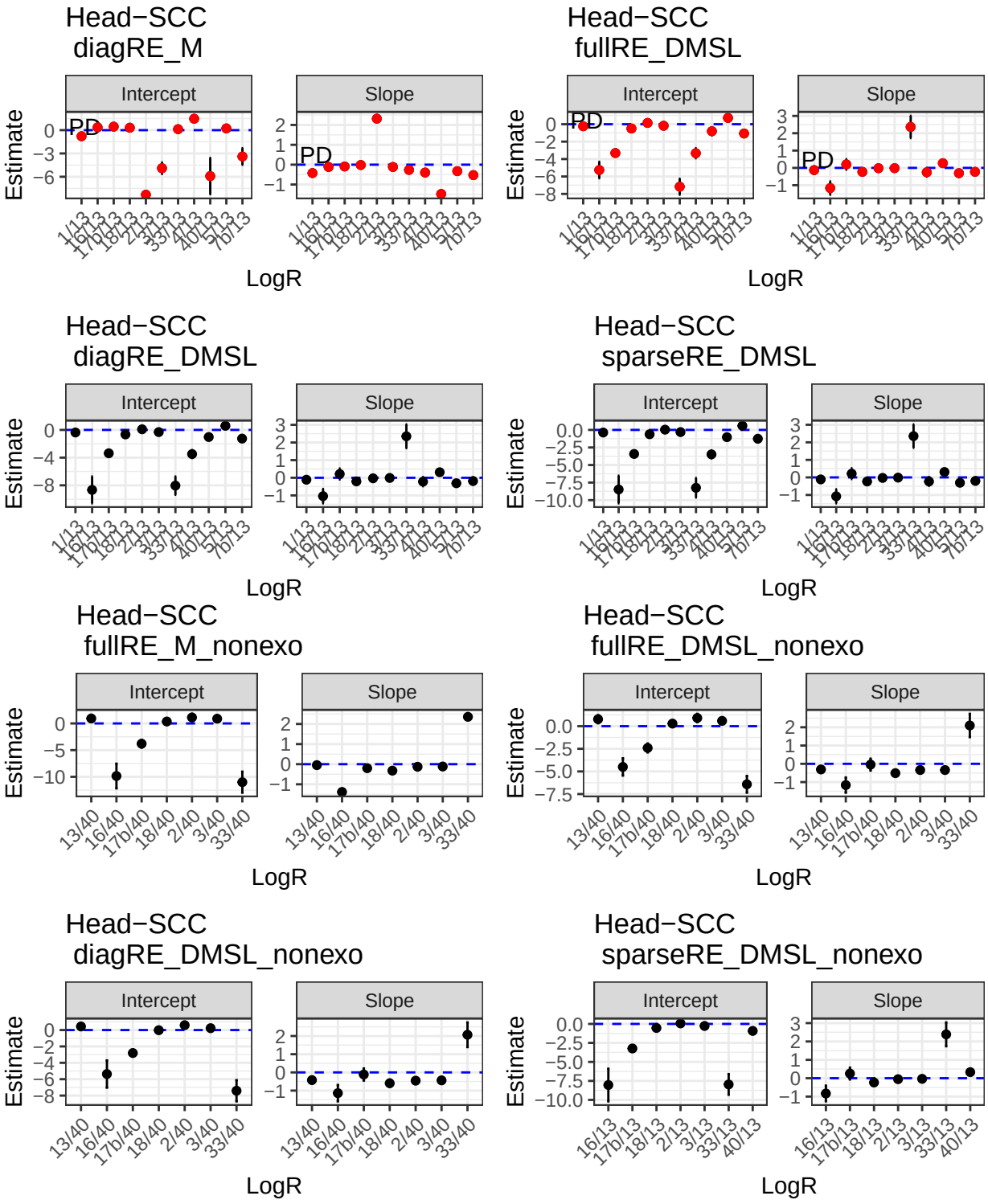
```
colSums(obj_Head_SCC$Y == 0)/nrow(obj_Head_SCC$Y)
```

```
##    SBS1    SBS2    SBS3    SBS4    SBS5   SBS7b   SBS13   SBS16  SBS17b   SBS18
## 0.00000 0.00000 0.06250 0.50000 0.00000 0.06250 0.00000 0.75000 0.40625 0.09375
##   SBS33   SBS40
## 0.81250 0.21875
```

```
colSums(obj_Head_SCC$Y)/sum(obj_Head_SCC$Y)
```

```
##        SBS1        SBS2        SBS3        SBS4        SBS5       SBS7b
## 0.052157398 0.209133263 0.121874082 0.030243442 0.152377754 0.025011542
##       SBS13       SBS16      SBS17b       SBS18       SBS33       SBS40
## 0.225057712 0.017861490 0.005867786 0.056873033 0.001013641 0.102528856
```

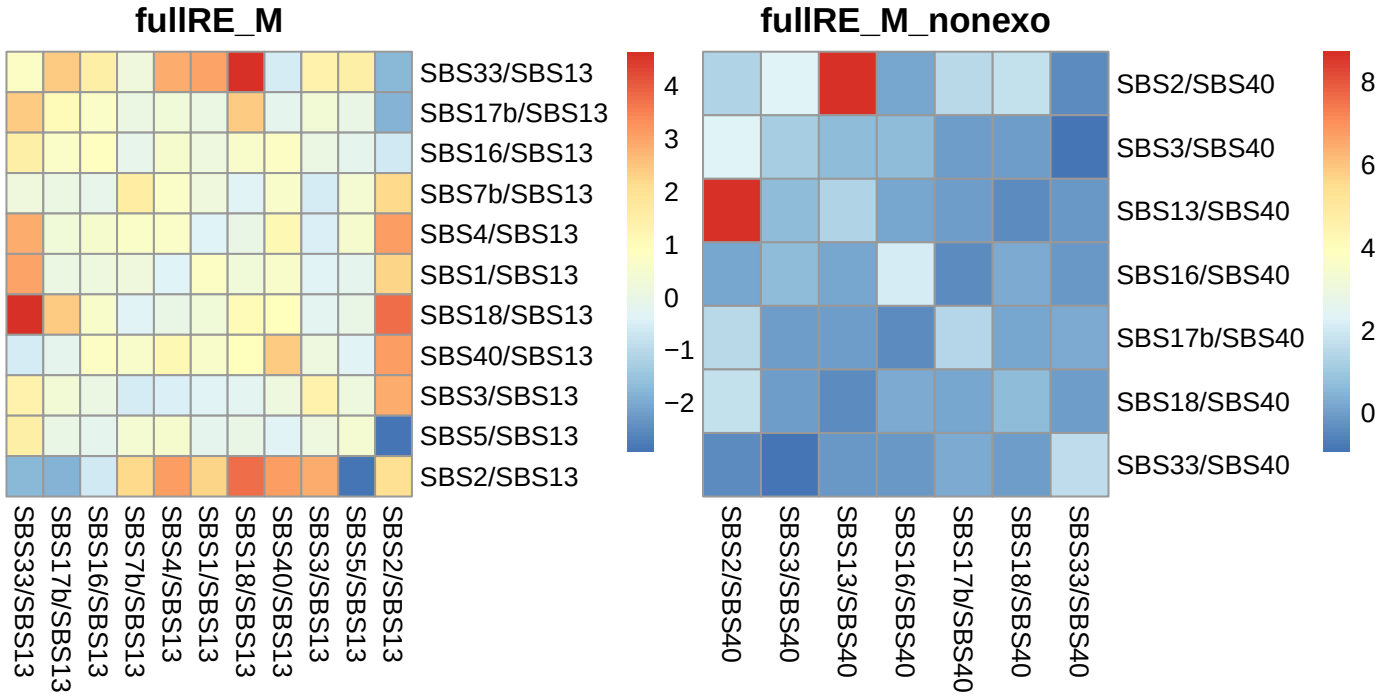
Betas

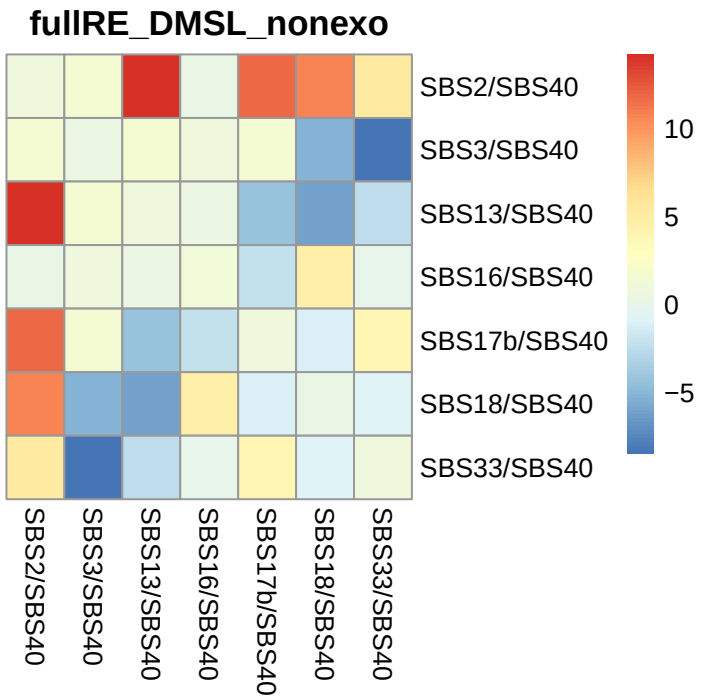


We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 8.4420109×10^{-5}.

Covariance matrices

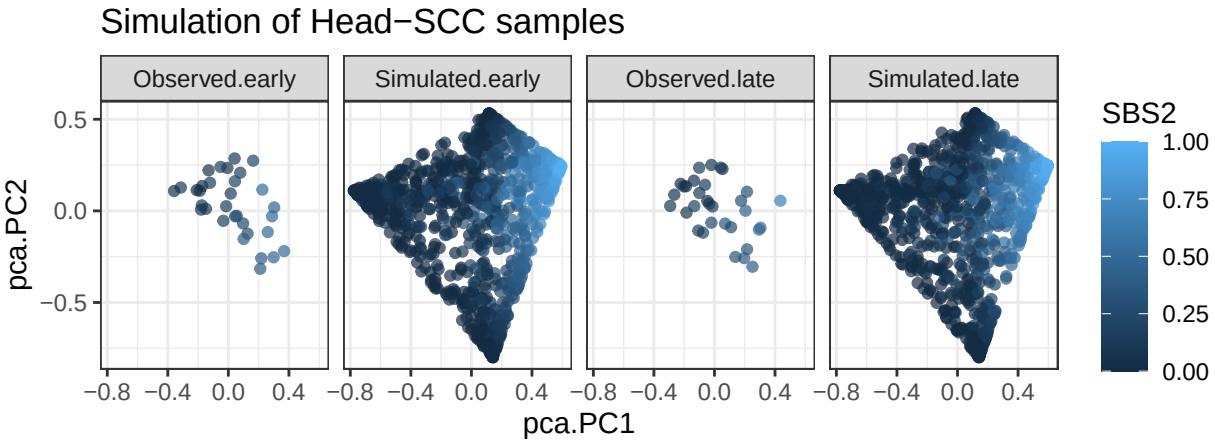
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



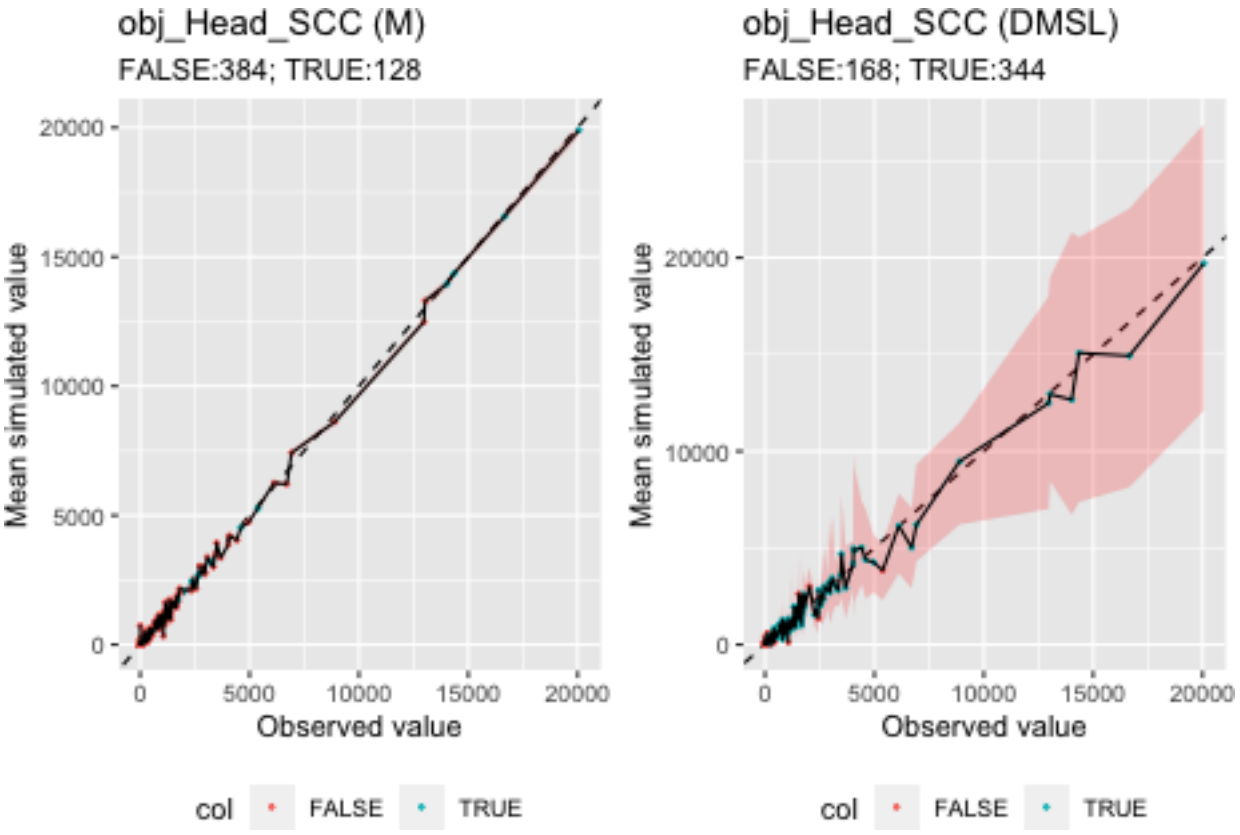


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



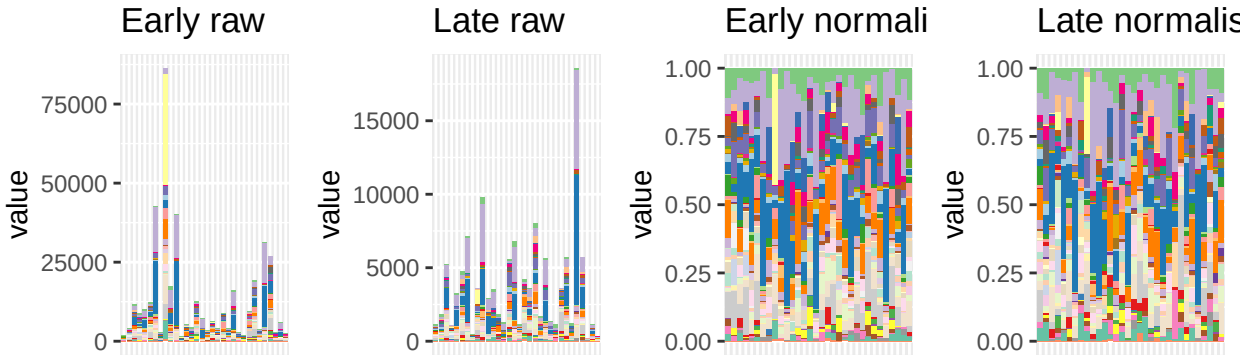
Ranked plot for coverage



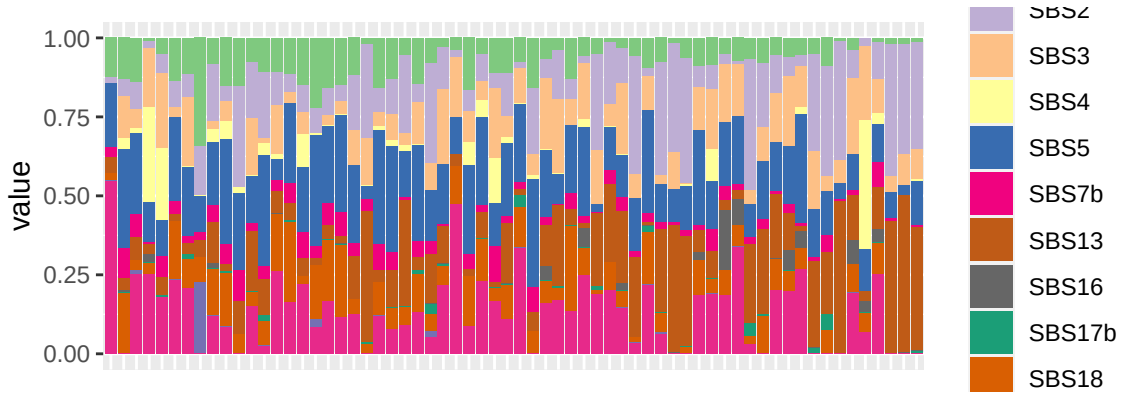
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 32
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the num-

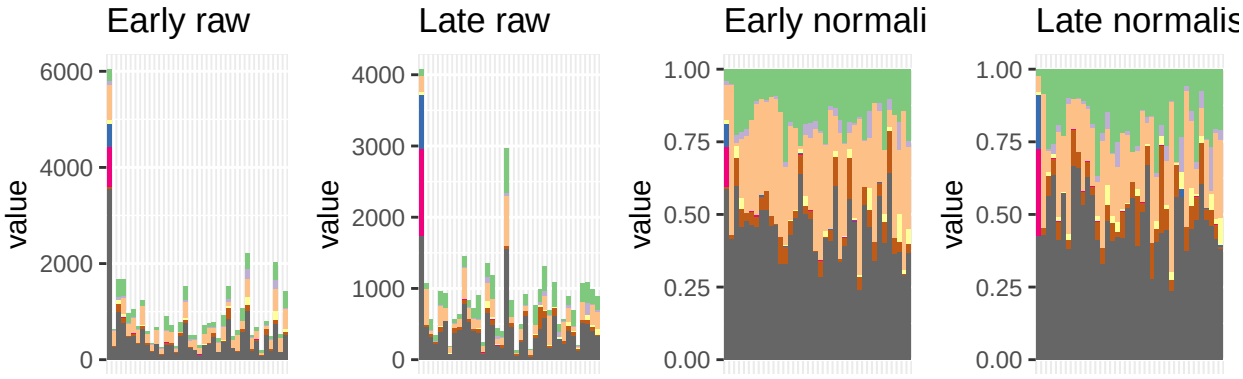


ber of mutations.

Kidney-ChRCC

Barplot and general statistics

```
## [1] 38
```



The number of samples and signatures is:

```
## [1] 76  8
```

The signatures are:

```
## [1] "SBS1"   "SBS2"   "SBS5"   "SBS13"  "SBS17a" "SBS17b" "SBS29"  "SBS40"
```

Convergence table

For all signatures, no fullRE model has converged. For nonexogenous ones, all have.

```
##           value                                L2                   L1
## 1  Kidney-ChRCC hessian_nonpositivedefinite_bool              diagRE_M
## 2  Kidney-ChRCC hessian_nonpositivedefinite_bool              fullRE_M
## 3  Kidney-ChRCC hessian_nonpositivedefinite_bool            diagRE_DMDL
## 4  Kidney-ChRCC hessian_nonpositivedefinite_bool           fullRE_halfDM
## 5  Kidney-ChRCC hessian_nonpositivedefinite_bool            fullRE_DMDL
## 6  Kidney-ChRCC    hessian_positivedefinite_bool            diagRE_DMSL
## 7  Kidney-ChRCC    hessian_positivedefinite_bool           sparseRE_DMSL
## 8  Kidney-ChRCC hessian_nonpositivedefinite_bool            fullRE_DMSL
## 9  Kidney-ChRCC hessian_nonpositivedefinite_bool       fullRE_DMSL_SBS1
## 10 Kidney-ChRCC    hessian_positivedefinite_bool        fullRE_M_nonexo
## 11 Kidney-ChRCC    hessian_positivedefinite_bool      diagRE_DMSL_nonexo
```

```
## 12 Kidney-ChRCC    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Kidney-ChRCC    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 Kidney-ChRCC hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Kidney-ChRCC hessian_nonpositivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

We do not need to re-run any model fitting.

Potentially problematic signatures

We notice that SBS17a and SBS17b are perhaps problematic.

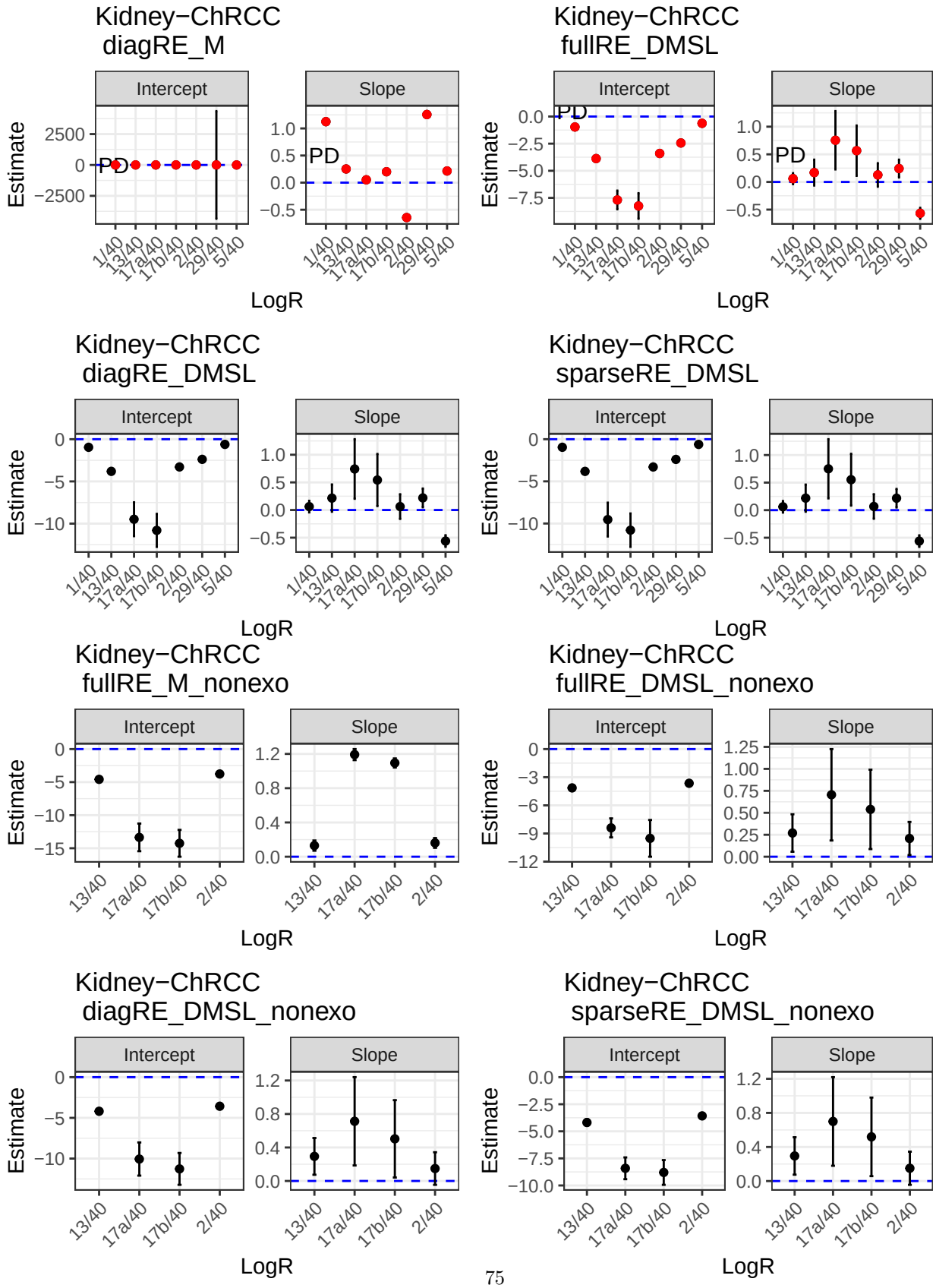
```
colSums(obj_Kidney_ChRCC$Y == 0)/nrow(obj_Kidney_ChRCC$Y)
```

```
##       SBS1       SBS2       SBS5      SBS13     SBS17a     SBS17b      SBS29
## 0.00000000 0.23684211 0.05263158 0.35526316 0.89473684 0.89473684 0.09210526
##      SBS40
## 0.00000000
```

```
colSums(obj_Kidney_ChRCC$Y)/sum(obj_Kidney_ChRCC$Y)
```

```
##       SBS1       SBS2       SBS5      SBS13     SBS17a     SBS17b      SBS29
## 0.17183661 0.02350905 0.21046460 0.02066116 0.01747822 0.02920482 0.04789759
##      SBS40
## 0.47894796
```

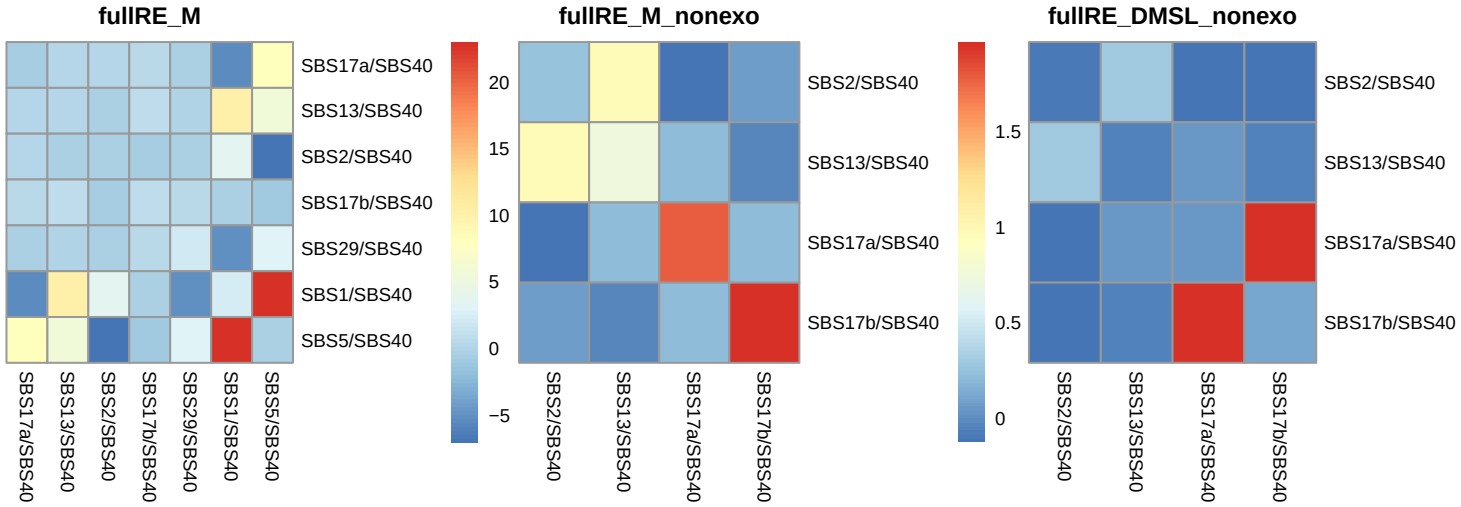
Betas


Kidney−ChRCC
diagRE_M
Intercept
Slope
Estimate
PD
LogR
Kidney−ChRCC
fullRE_DMSL
Kidney−ChRCC
diagRE_DMSL
Kidney−ChRCC
sparseRE_DMSL
Kidney−ChRCC
fullRE_M_nonexo
Kidney−ChRCC
fullRE_DMSL_nonexo
Kidney−ChRCC
diagRE_DMSL_nonexo
Kidney−ChRCC
sparseRE_DMSL_nonexo
1/40
13/40
17a/40
17b/40
2/40
29/40
5/40

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.2664763.

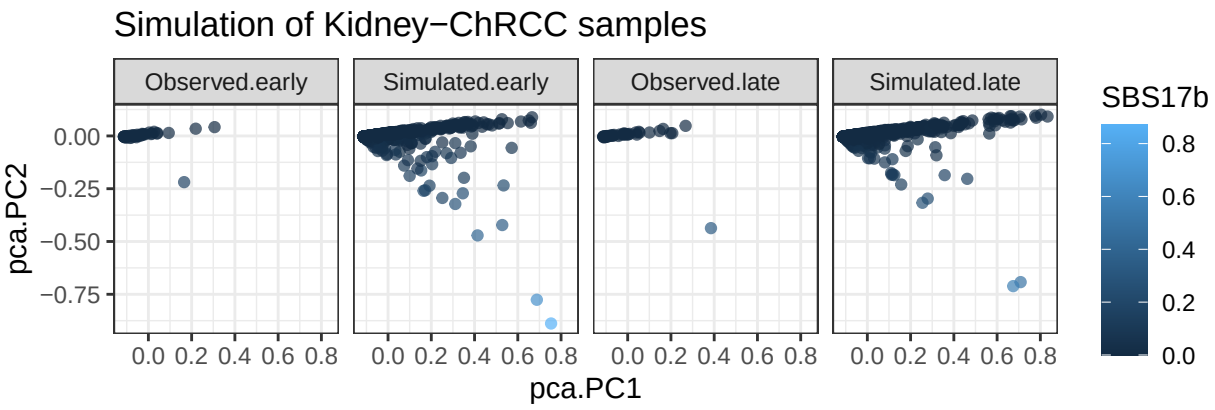
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

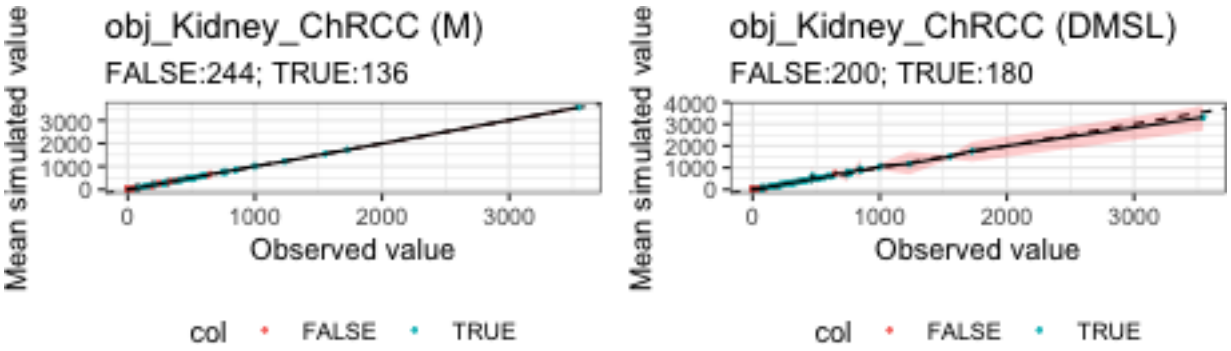


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



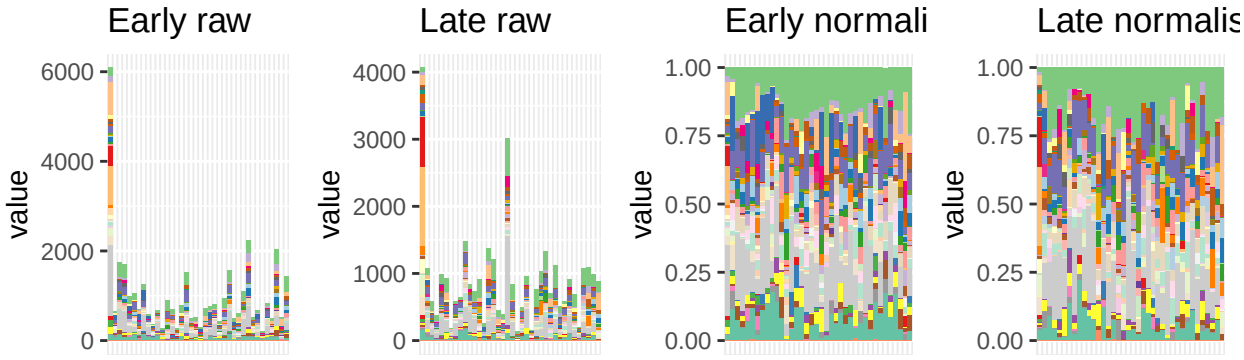
Ranked plot for coverage



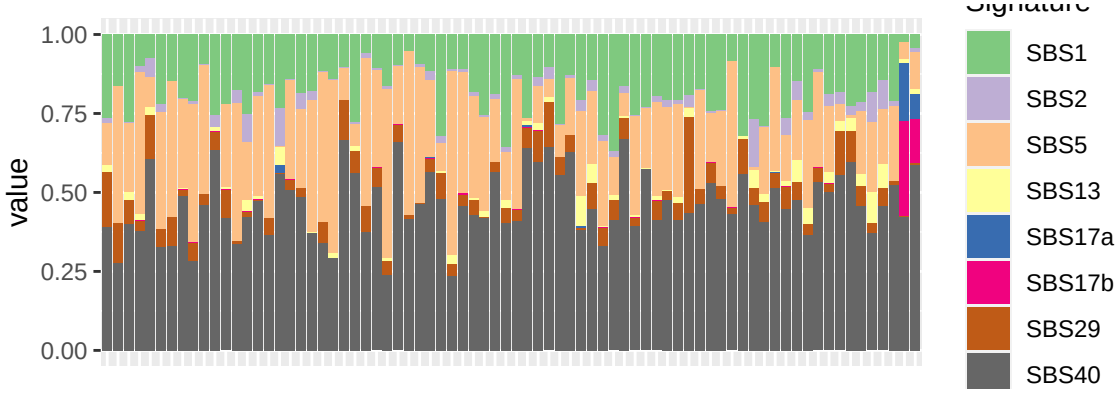
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 38
```



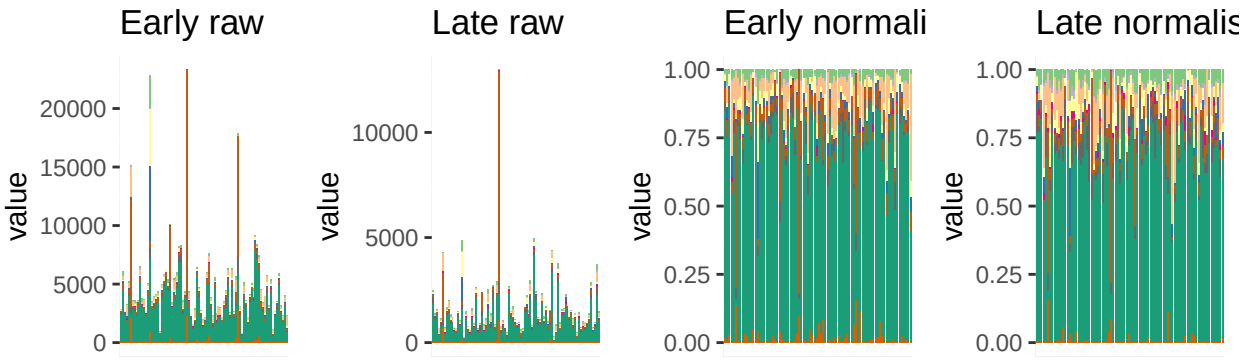
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Kidney-RCC.clearcell

Barplot and general statistics

```
## [1] 86
```



The number of samples and signatures is:

```
## [1] 172  10
```

The signatures are:

```
##  [1] "SBS1"  "SBS2"  "SBS5"  "SBS6"  "SBS12" "SBS13" "SBS22" "SBS29" "SBS40"
## [10] "SBS41"
```

Convergence table

Essentially, everything has converged.

```
##                    value                                 L2
## 1  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 2  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 3  Kidney-RCC.clearcell hessian_nonpositivedefinite_bool
## 4  Kidney-RCC.clearcell                         Timeout
## 5  Kidney-RCC.clearcell hessian_nonpositivedefinite_bool
## 6  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 7  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 8  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 9  Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 10 Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 11 Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 12 Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 13 Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 14 Kidney-RCC.clearcell    hessian_positivedefinite_bool
## 15 Kidney-RCC.clearcell                         Timeout
##                          L1
## 1                  diagRE_M
## 2                  fullRE_M
## 3               diagRE_DMDL
## 4             fullRE_halfDM
## 5               fullRE_DMDL
## 6               diagRE_DMSL
## 7             sparseRE_DMSL
## 8               fullRE_DMSL
## 9          fullRE_DMSL_SBS1
## 10          fullRE_M_nonexo
## 11       diagRE_DMSL_nonexo
## 12     sparseRE_DMSL_nonexo
```

```
## 13       fullRE_DMSL_nonexo
## 14       fullRE_DMDL_nonexo
## 15 fullRE_DMDL_sortednonexo
```

Potentially problematic signatures

There are no problematic signatures.

```
colSums(obj_Kidney_RCCclearcell$Y == 0)/nrow(obj_Kidney_RCCclearcell$Y)
```

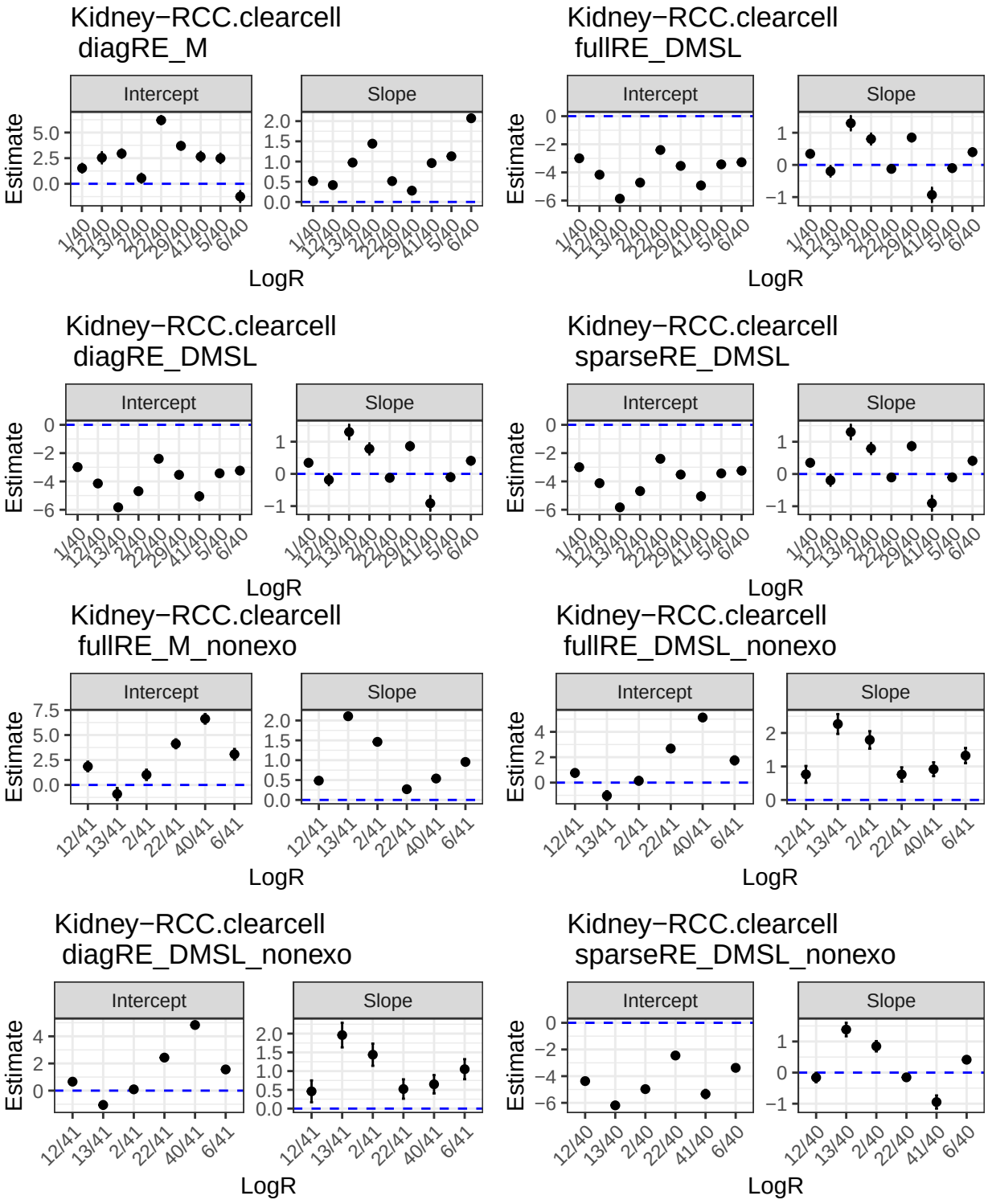
```
##       SBS1       SBS2       SBS5       SBS6      SBS12      SBS13      SBS22
## 0.02906977 0.26744186 0.33139535 0.11627907 0.35465116 0.52325581 0.00000000
##      SBS29      SBS40      SBS41
## 0.13372093 0.00000000 0.58720930
```

```
colSums(obj_Kidney_RCCclearcell$Y)/sum(obj_Kidney_RCCclearcell$Y)
```

```
##        SBS1        SBS2        SBS5        SBS6       SBS12       SBS13
## 0.033881764 0.005632854 0.060208144 0.036168761 0.026758216 0.003701688
##       SBS22       SBS29       SBS40       SBS41
## 0.140375696 0.036802695 0.629055577 0.027414605
```

We would need to run `fullRE_DMSL`, because it timed out.

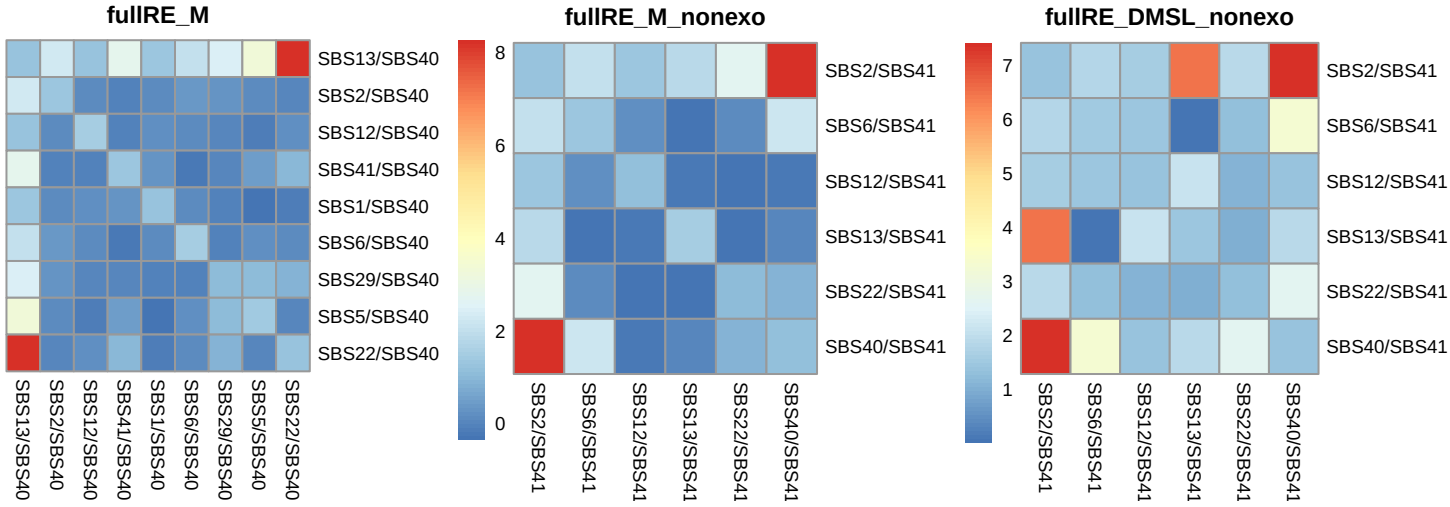
Betas



We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of $3.2572102 \times 10^{-21}$.

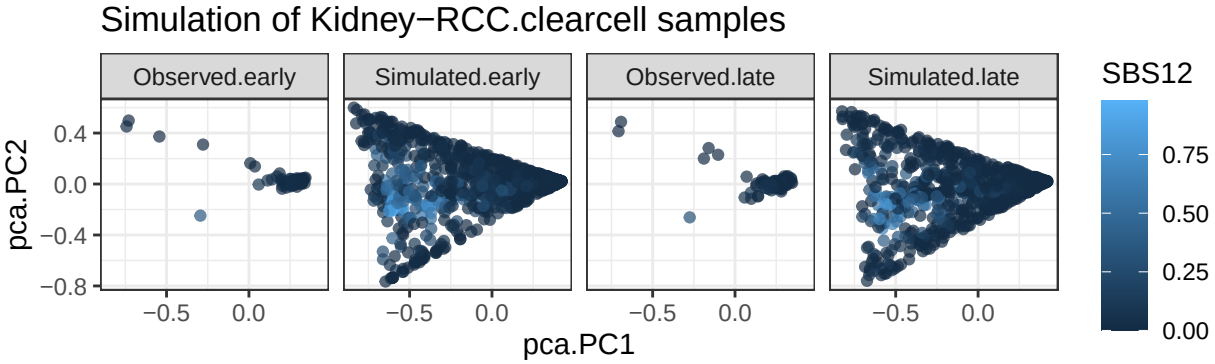
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

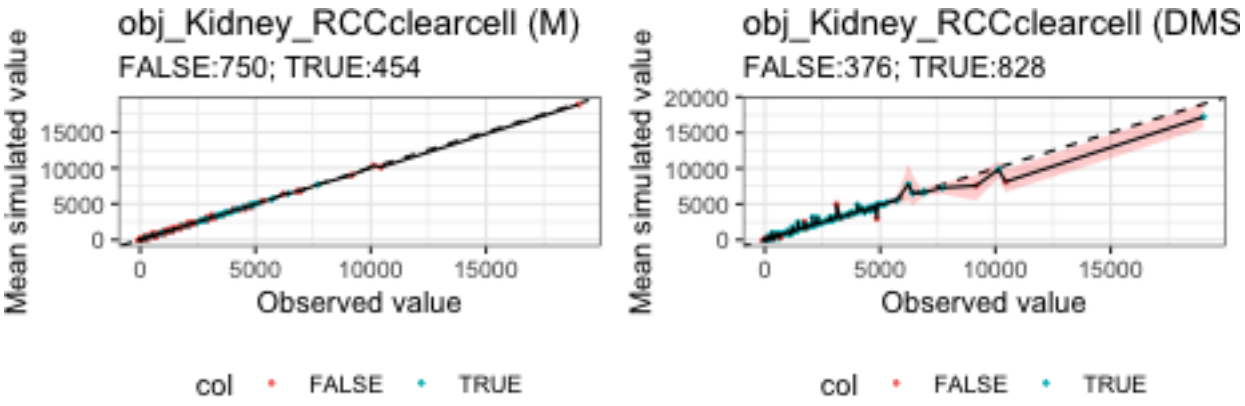


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



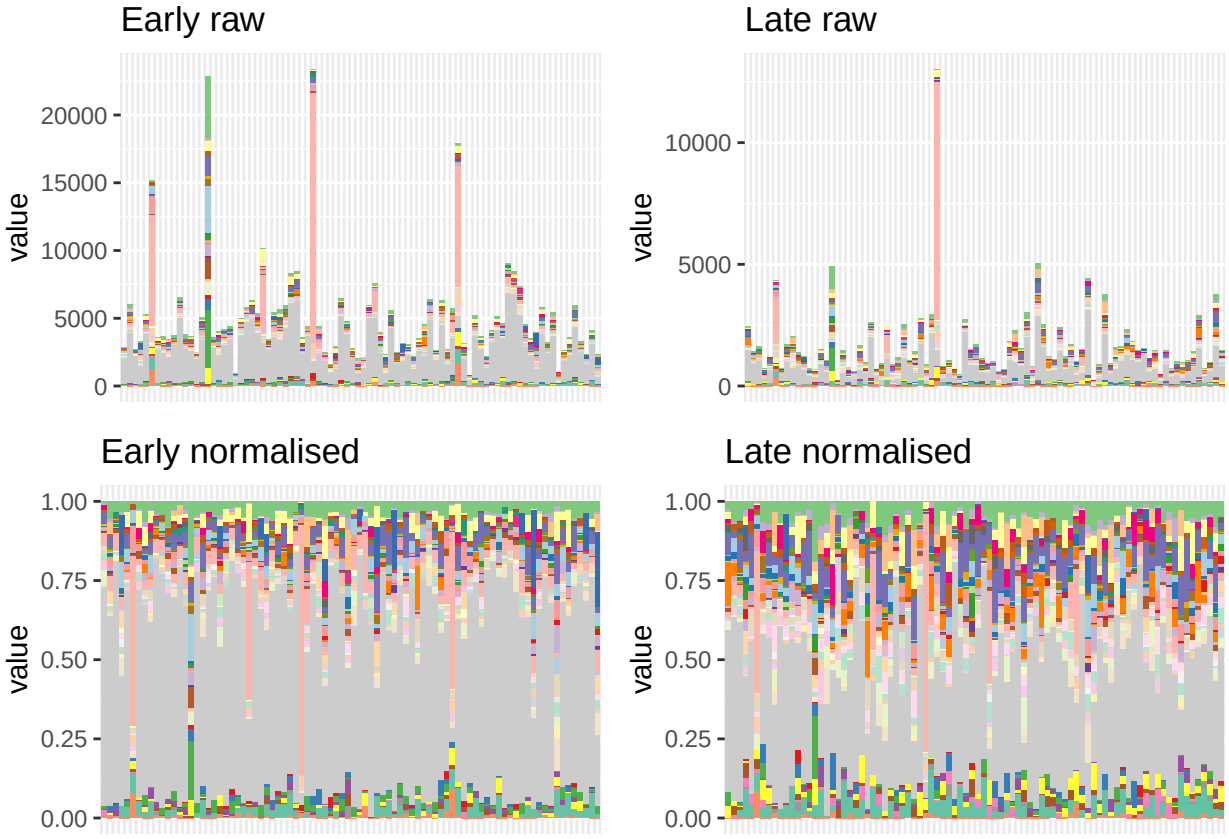
Ranked plot for coverage



Signatures from mutSigExtractor

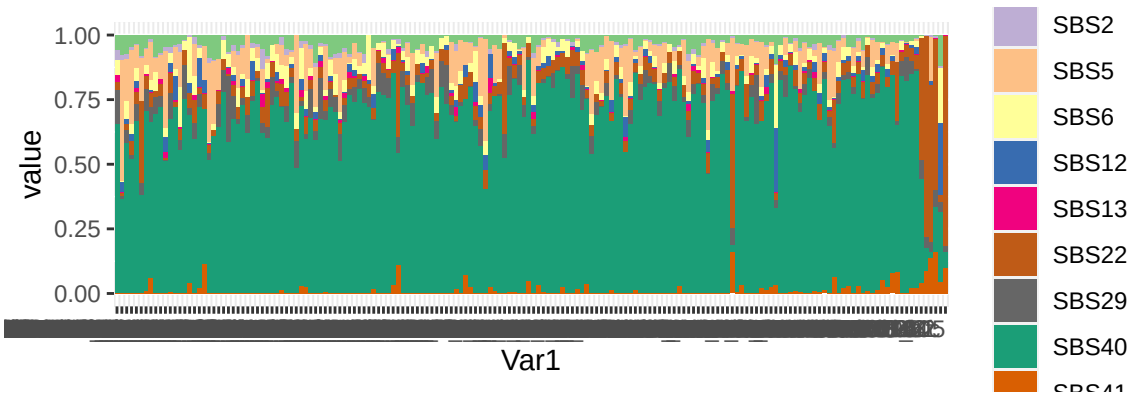
The signatures from mutSigExtractor are as follows:

```
## [1] 86
```



I should check if this grey exposure corresponds to SBS40.

Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations except for perhaps the very few with highest exposure.

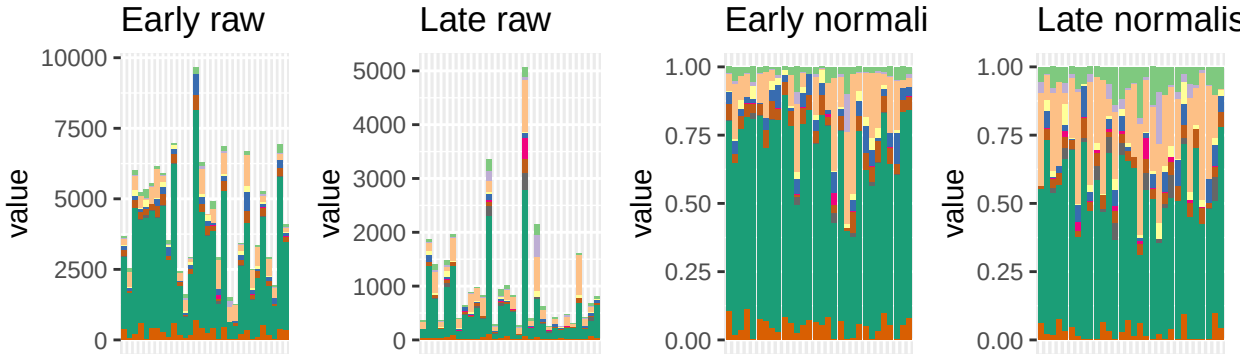


Kidney-RCC.papillary

It looks very similar to clear cell, looking generally at the signatures.

Barplot and general statistics

```
## [1] 30
```



The number of samples and signatures is:

```
## [1] 60 10
```

The signatures are:

```
##  [1] "SBS1"  "SBS2"  "SBS5"  "SBS6"  "SBS12" "SBS13" "SBS22" "SBS29" "SBS40"
## [10] "SBS41"
```

Convergence table

Although fulLRE DMSL has in one case converged, it hasn't when using SBS1 as baseline. The nonexogenous version has not converged, but M has.

```
##                   value                                L2
## 1  Kidney-RCC.papillary    hessian_positivedefinite_bool
## 2  Kidney-RCC.papillary    hessian_positivedefinite_bool
## 3  Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 4  Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 5  Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 6  Kidney-RCC.papillary    hessian_positivedefinite_bool
## 7  Kidney-RCC.papillary    hessian_positivedefinite_bool
```

```
## 8  Kidney-RCC.papillary    hessian_positivedefinite_bool
## 9  Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 10 Kidney-RCC.papillary    hessian_positivedefinite_bool
## 11 Kidney-RCC.papillary    hessian_positivedefinite_bool
## 12 Kidney-RCC.papillary    hessian_positivedefinite_bool
## 13 Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 14 Kidney-RCC.papillary hessian_nonpositivedefinite_bool
## 15 Kidney-RCC.papillary hessian_nonpositivedefinite_bool
##                           L1
## 1                    diagRE_M
## 2                    fullRE_M
## 3                 diagRE_DMDL
## 4               fullRE_halfDM
## 5                 fullRE_DMDL
## 6                 diagRE_DMSL
## 7               sparseRE_DMSL
## 8                 fullRE_DMSL
## 9            fullRE_DMSL_SBS1
## 10            fullRE_M_nonexo
## 11         diagRE_DMSL_nonexo
## 12      sparseRE_DMSL_nonexo
## 13        fullRE_DMSL_nonexo
## 14        fullRE_DMDL_nonexo
## 15 fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo. We first re-run M.

The we use it to re-run DM.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do converge:

```
## [1] TRUE
```

Potentially problematic signatures

We explore whether there are problematic signatures:

```
colSums(obj_Kidney_RCCpapillary$Y == 0)/nrow(obj_Kidney_RCCpapillary$Y)
```

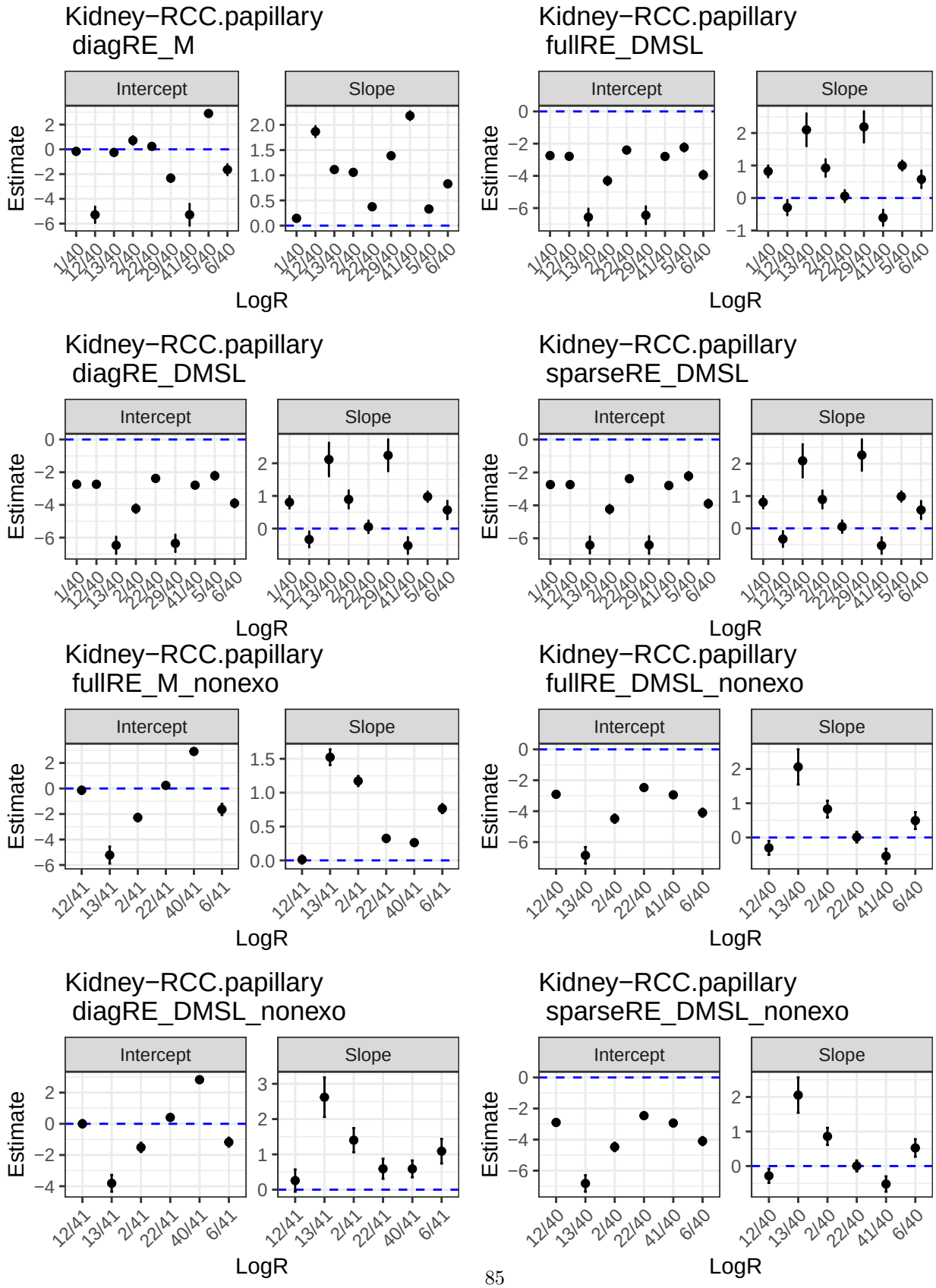
```
##      SBS1      SBS2      SBS5      SBS6     SBS12     SBS13     SBS22     SBS29
## 0.0000000 0.2333333 0.1000000 0.2666667 0.1833333 0.6500000 0.0000000 0.6333333
##     SBS40     SBS41
## 0.0000000 0.2166667
```

```
colSums(obj_Kidney_RCCpapillary$Y)/sum(obj_Kidney_RCCpapillary$Y)
```

```
##        SBS1        SBS2        SBS5        SBS6       SBS12       SBS13
## 0.035736437 0.010720323 0.116403371 0.016521337 0.039719314 0.004438931
##       SBS22       SBS29       SBS40       SBS41
## 0.049284298 0.007187420 0.669488124 0.050500444
```

SBS29 is found in relatively small quantities.

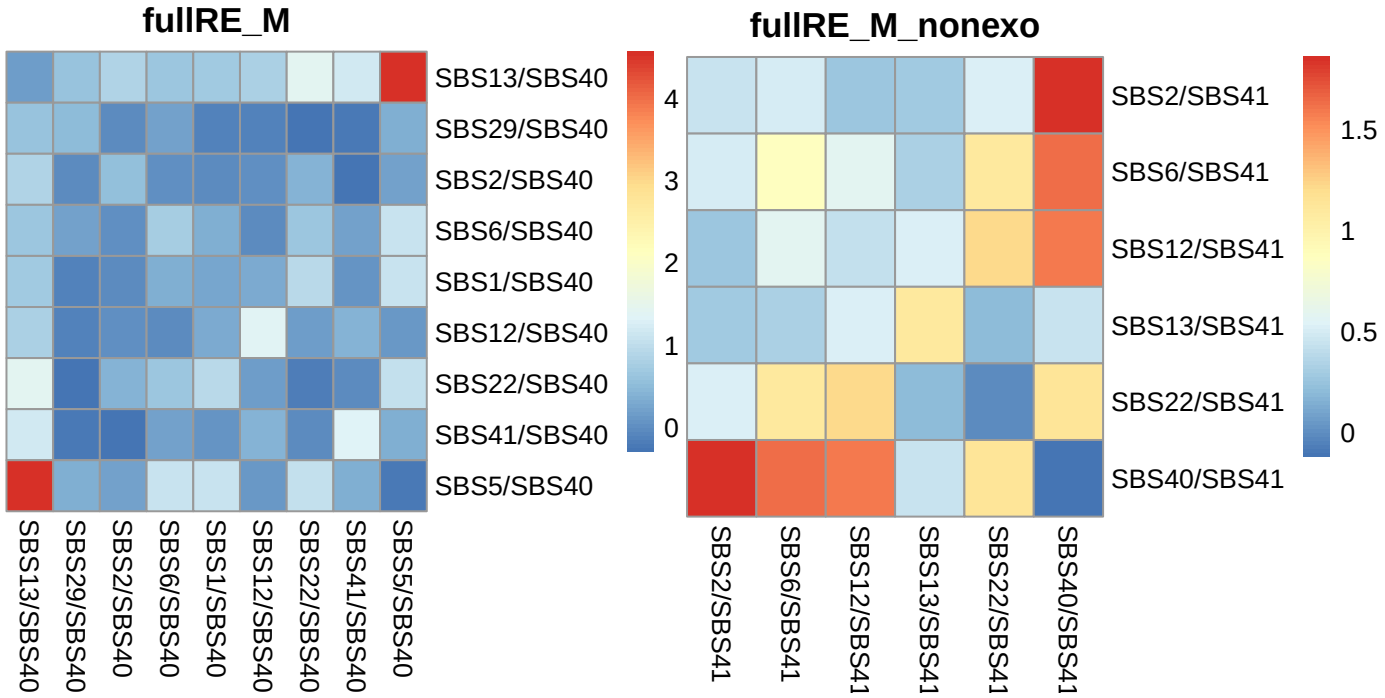
Betas

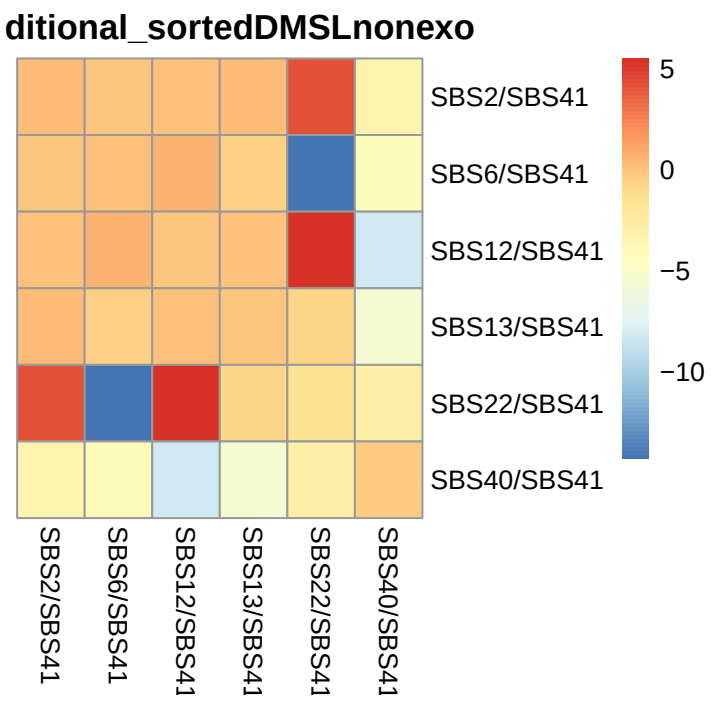

Kidney−RCC.papillary
diagRE_M
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
fullRE_DMSL
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
diagRE_DMSL
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
sparseRE_DMSL
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
fullRE_M_nonexo
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
fullRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
diagRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR
Kidney−RCC.papillary
sparseRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 2.591235×10^{-7}.

Covariance matrices

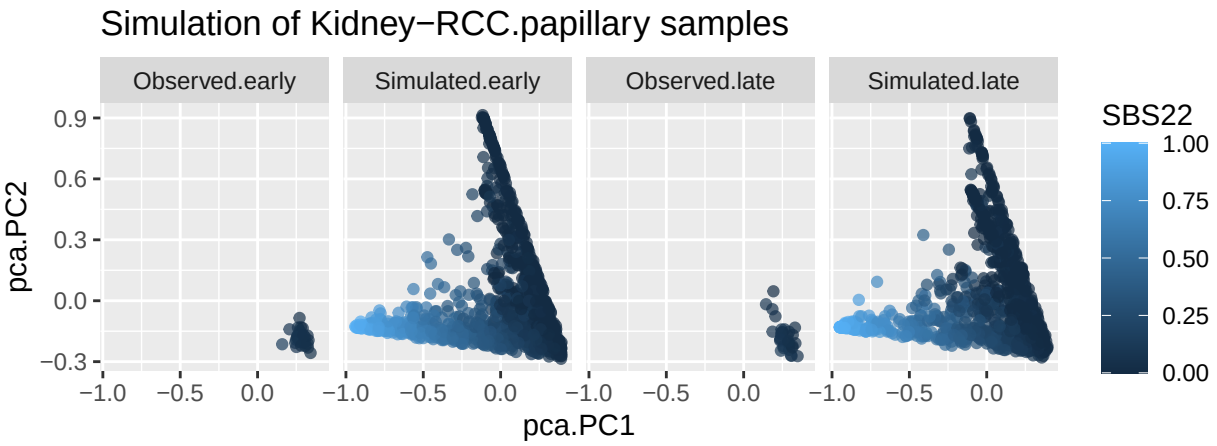
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



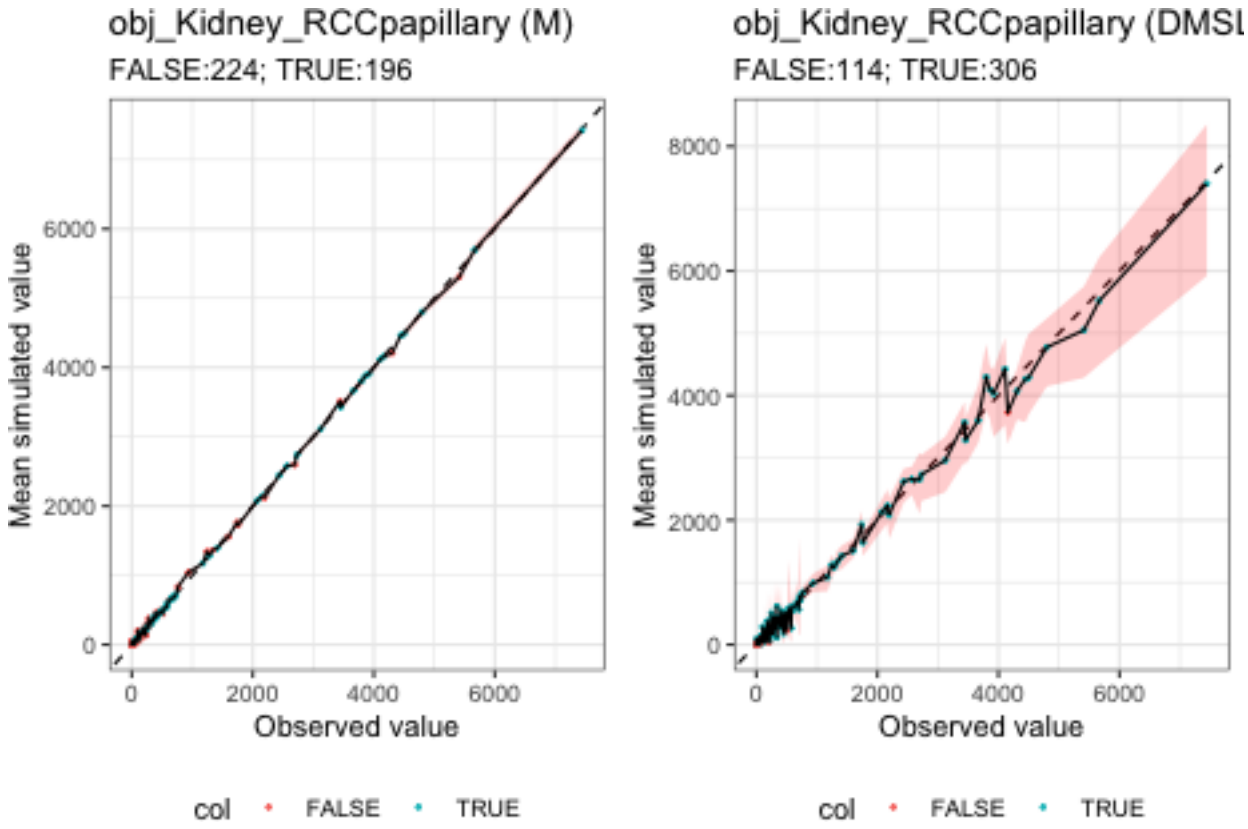


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



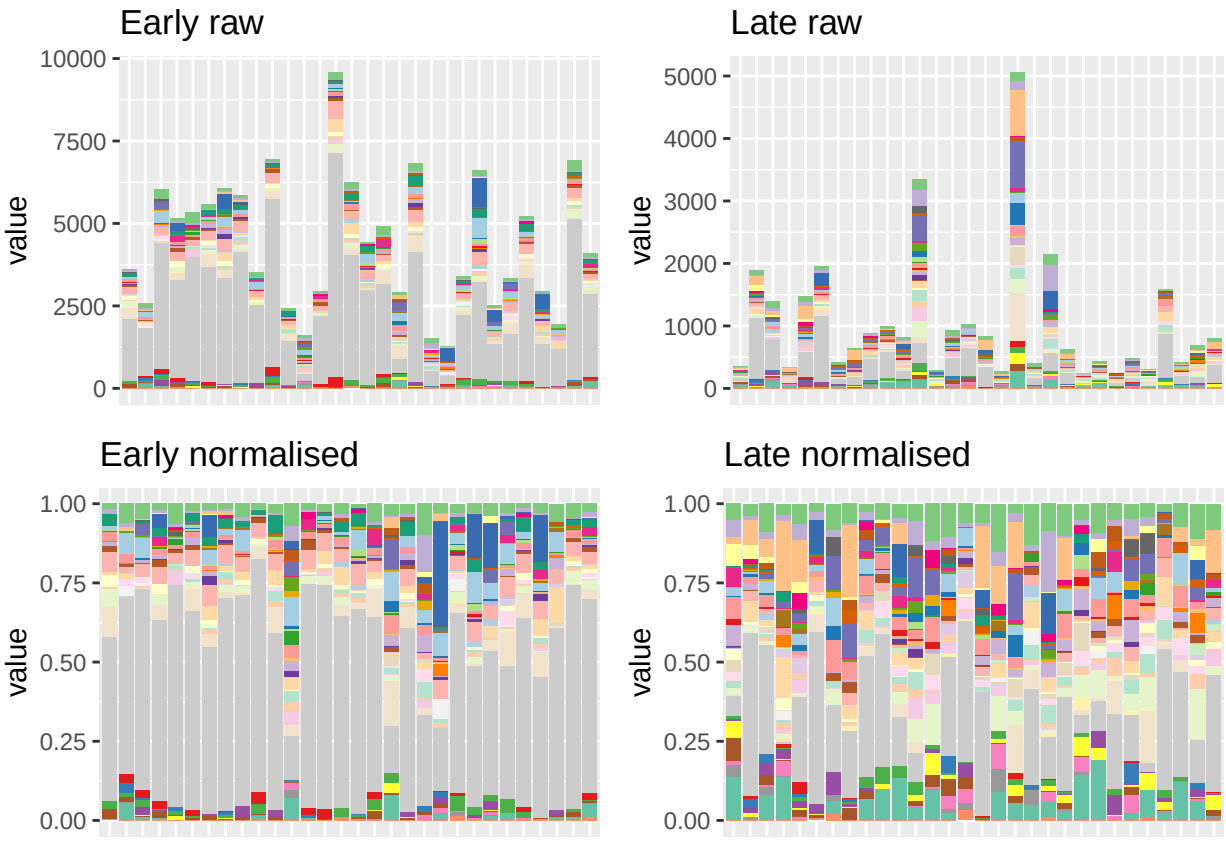
Ranked plot for coverage



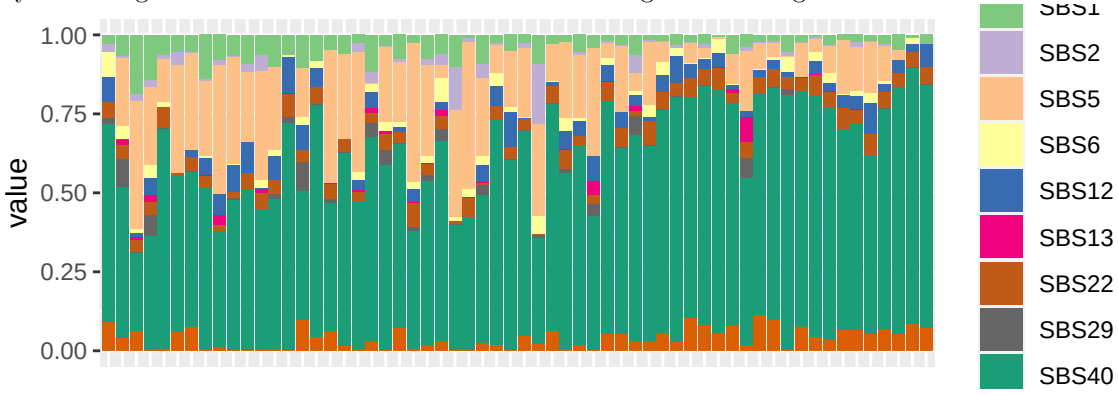
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 30
```



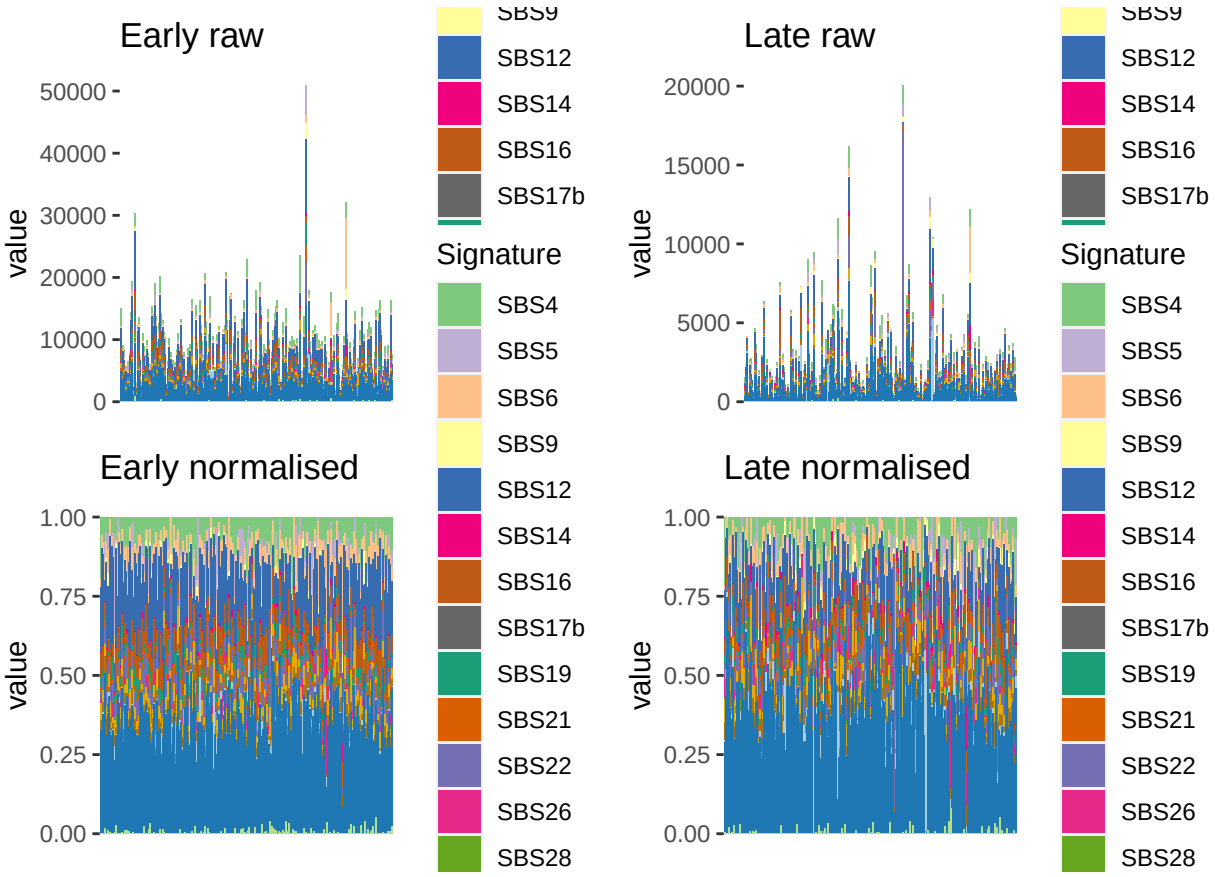
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Liver-HCC

Barplot and general statistics

```
## [1] 207
```



The number of samples and signatures is:

```
## [1] 414  18
```

The signatures are:

```
##  [1] "SBS4"   "SBS5"   "SBS6"   "SBS9"   "SBS12"  "SBS14"  "SBS16"  "SBS17b"
##  [9] "SBS19"  "SBS21"  "SBS22"  "SBS26"  "SBS28"  "SBS29"  "SBS30"  "SBS35"
## [17] "SBS40"  "SBS54"
```

Convergence table

The fullRE versions with all signatures have not converged. Neither has `fullRE_M_nonexo`, but `fullRE_DMSL_nonexo` has.

```
##        value                                L2                  L1
## 1  Liver-HCC    hessian_positivedefinite_bool          diagRE_M
## 2  Liver-HCC hessian_nonpositivedefinite_bool          fullRE_M
## 3  Liver-HCC hessian_nonpositivedefinite_bool        diagRE_DMDL
## 4  Liver-HCC                           Timeout       fullRE_halfDM
## 5  Liver-HCC hessian_nonpositivedefinite_bool        fullRE_DMDL
## 6  Liver-HCC    hessian_positivedefinite_bool        diagRE_DMSL
## 7  Liver-HCC    hessian_positivedefinite_bool       sparseRE_DMSL
## 8  Liver-HCC hessian_nonpositivedefinite_bool        fullRE_DMSL
## 9  Liver-HCC    hessian_positivedefinite_bool  fullRE_DMSL_SBS1
## 10 Liver-HCC hessian_nonpositivedefinite_bool   fullRE_M_nonexo
```

```
## 11 Liver-HCC    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Liver-HCC    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Liver-HCC    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 Liver-HCC hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Liver-HCC                          Timeout fullRE_DMDL_sortednonexo
```

Potentially problematic signatures

We explore whether there are problematic signatures:

```
colSums(obj_Liver_HCC$Y == 0)/nrow(obj_Liver_HCC$Y)
```

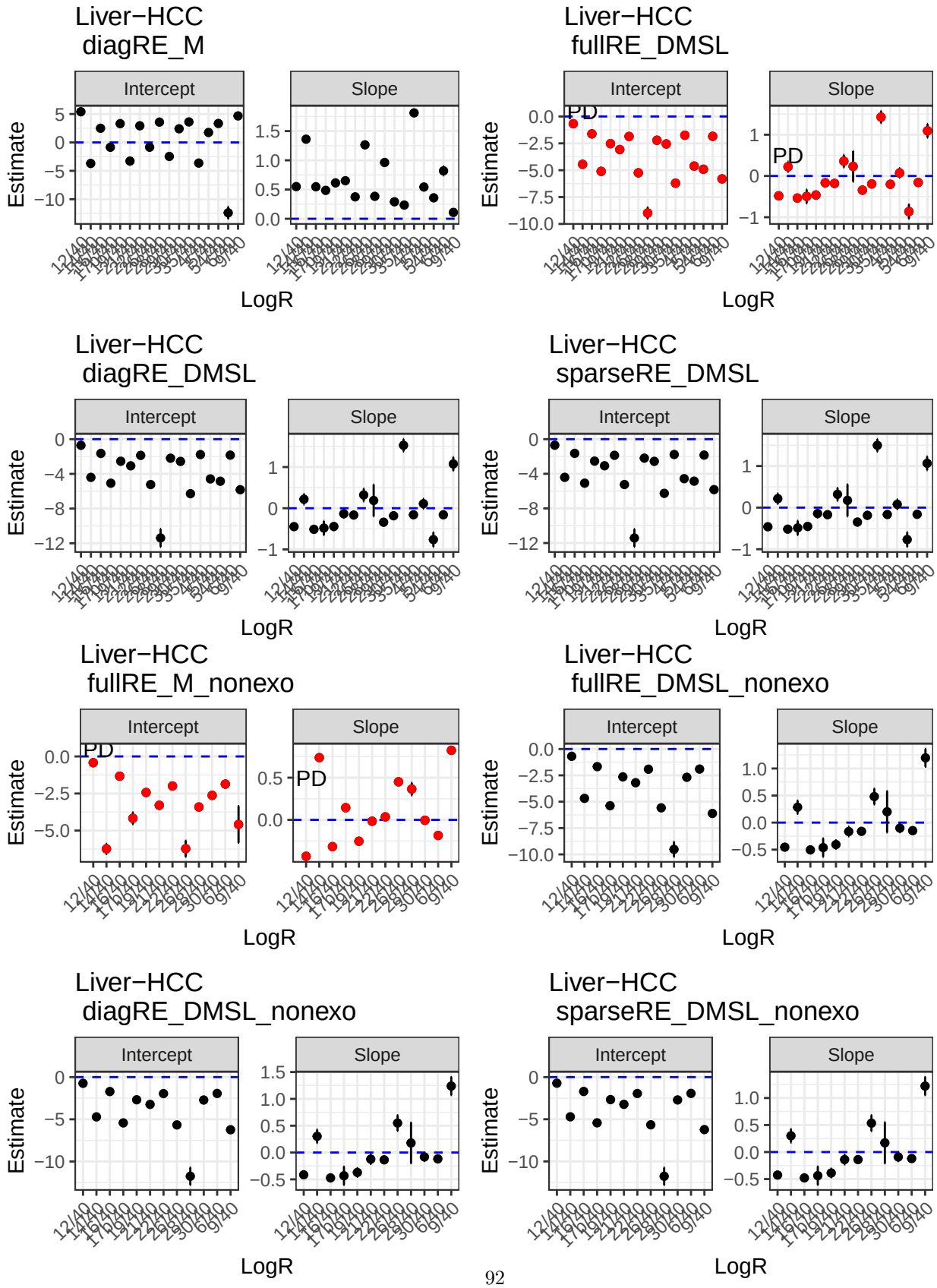
```
##        SBS4        SBS5        SBS6        SBS9       SBS12       SBS14
## 0.084541063 0.548309179 0.007246377 0.642512077 0.026570048 0.434782609
##       SBS16      SBS17b       SBS19       SBS21       SBS22       SBS26
## 0.048309179 0.649758454 0.120772947 0.176328502 0.012077295 0.613526570
##       SBS28       SBS29       SBS30       SBS35       SBS40       SBS54
## 0.934782609 0.096618357 0.113526570 0.628019324 0.007246377 0.649758454
```

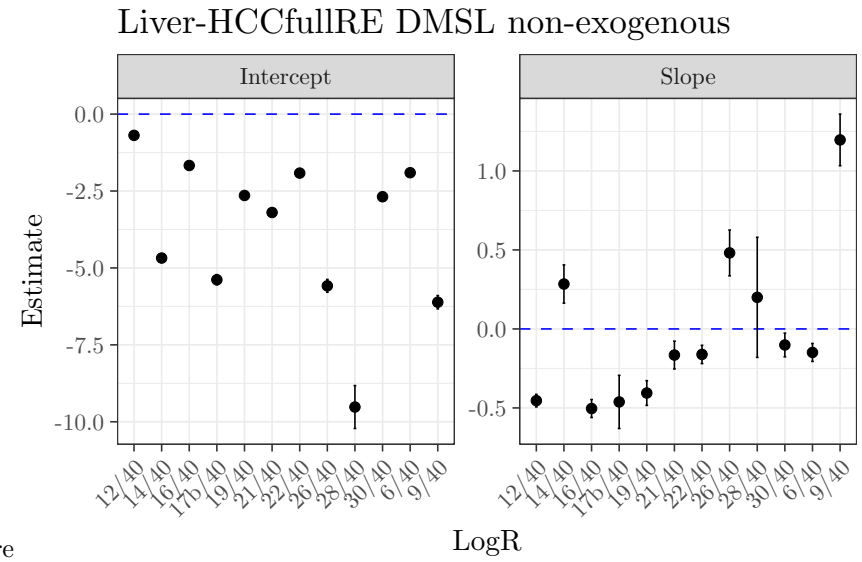
```
colSums(obj_Liver_HCC$Y)/sum(obj_Liver_HCC$Y)
```

```
##        SBS4        SBS5        SBS6        SBS9       SBS12       SBS14
## 0.081740962 0.026465897 0.047036092 0.007670584 0.206143096 0.005884273
##       SBS16      SBS17b       SBS19       SBS21       SBS22       SBS26
## 0.068804779 0.001835835 0.028215636 0.016888238 0.058186725 0.011387208
##       SBS28       SBS29       SBS30       SBS35       SBS40       SBS54
## 0.000603818 0.036248511 0.025723410 0.015247453 0.357387133 0.004530350
```

SBS28 is only present in 7% of samples and has extremely low exposure - we could consider removing it.

Betas


Liver−HCC
diagRE_M
Intercept
Slope
Estimate
LogR
Liver−HCC
fullRE_DMSL
PD
Liver−HCC
diagRE_DMSL
Liver−HCC
sparseRE_DMSL
Liver−HCC
fullRE_M_nonexo
Liver−HCC
fullRE_DMSL_nonexo
Liver−HCC
diagRE_DMSL_nonexo
Liver−HCC
sparseRE_DMSL_nonexo

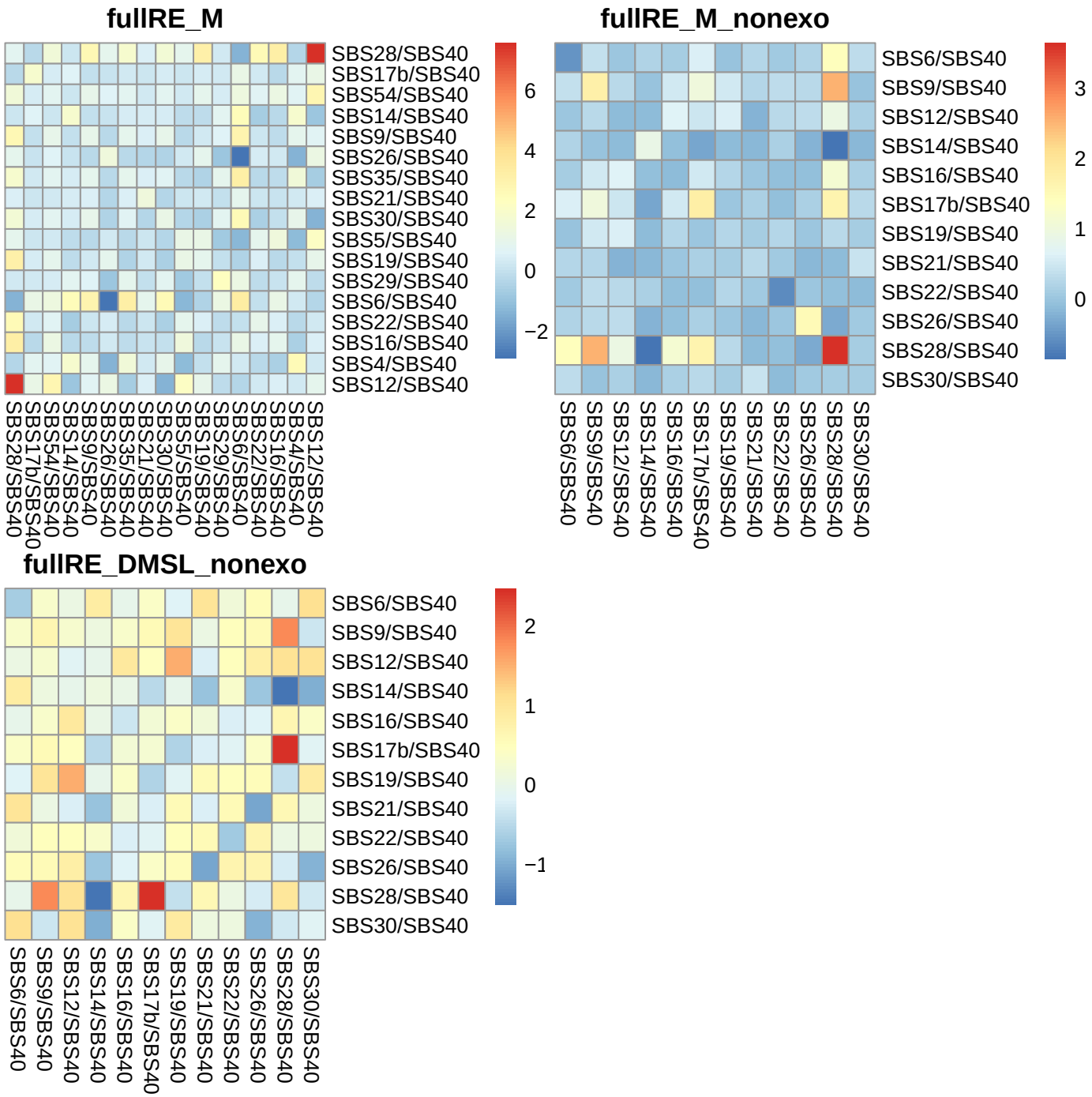


Difficult to see the labels, so I replot it here

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of $1.0407591 \times 10^{-55}$.

Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

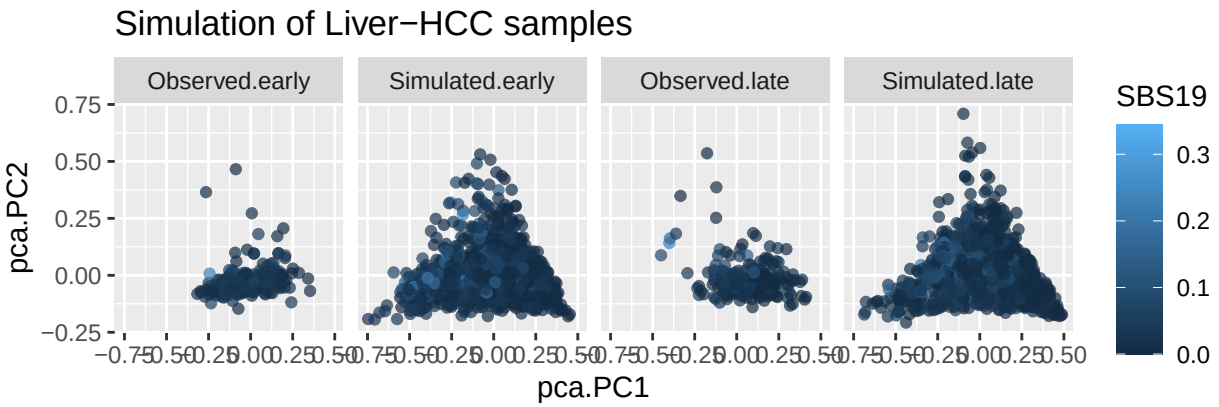


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

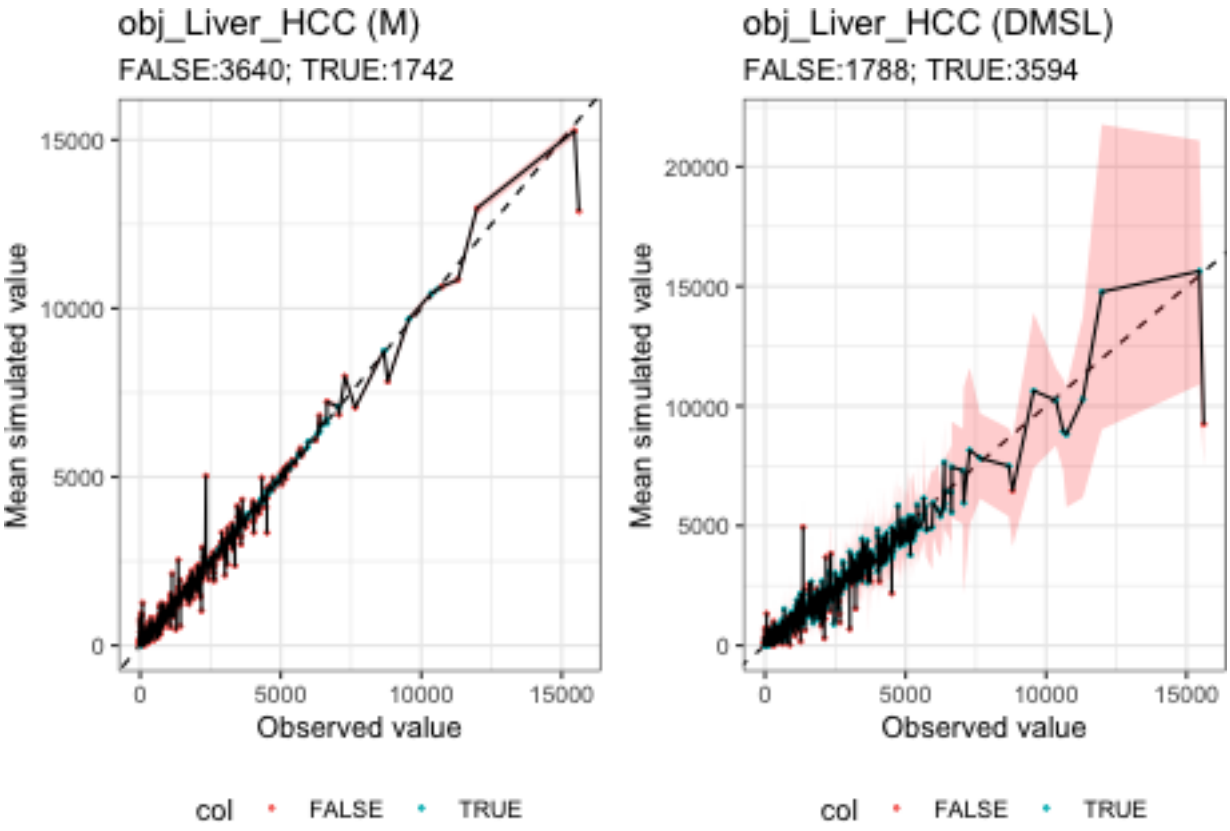
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



Ranked plot for coverage

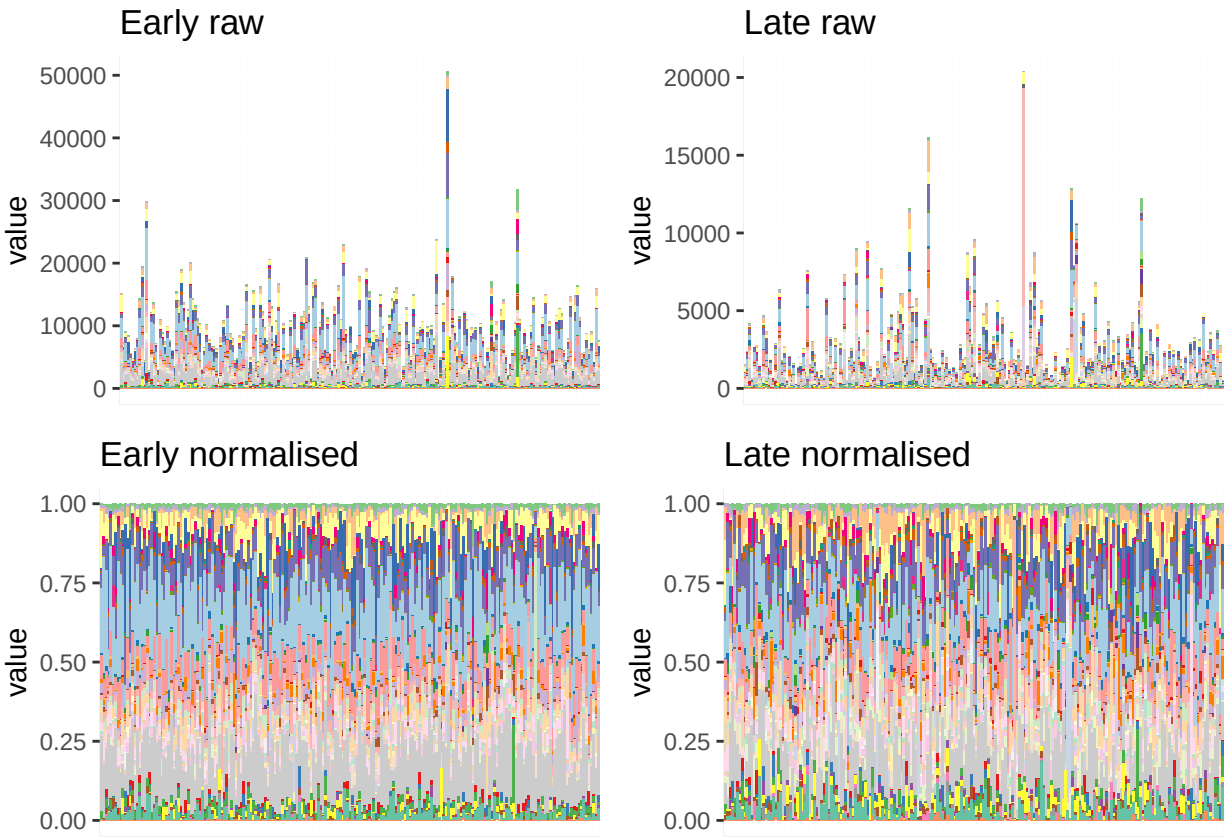
Remember that fullRE M has not converged, and it should be re-run:



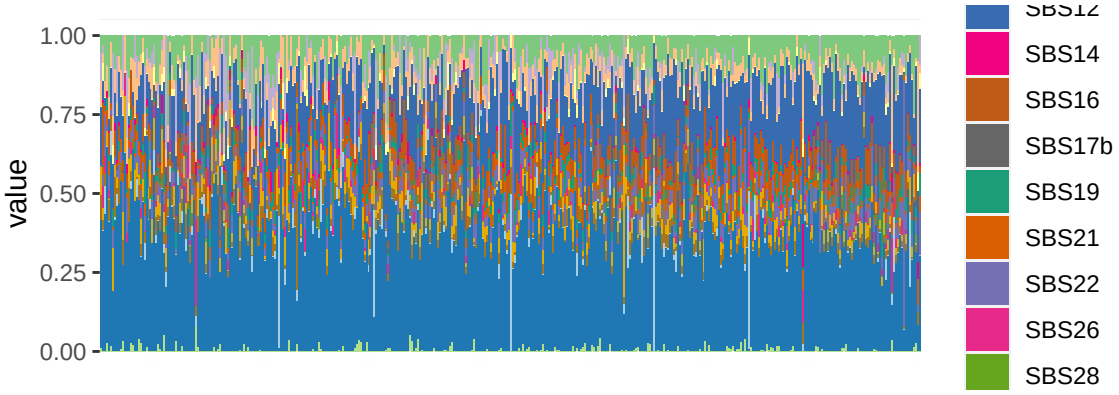
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 207
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.

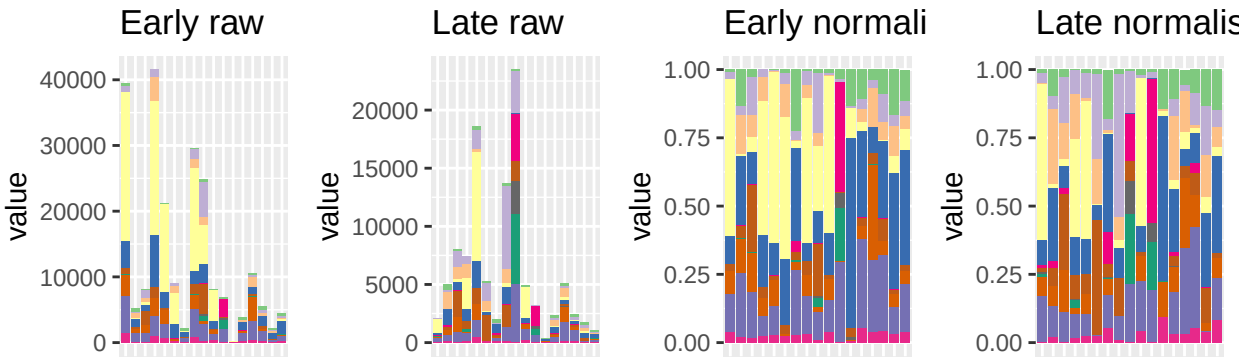


Lung-AdenoCA

How can we have such few samples?

Barplot and general statistics

```
## [1] 17
```



The number of samples and signatures is:

```
## [1] 34 12
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS4"   "SBS5"   "SBS9"   "SBS13"  "SBS17a"
##  [9] "SBS17b" "SBS18"  "SBS40"  "SBS50"
```

Convergence table

No fullRE DMSL have converged.

```
## [1] value L2    L1
## <0 rows> (or 0-length row.names)
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo. We re-run fullRE_M_nonexo and it has converged:

But fullRE DMSL hasn't:

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo does not converge:

```
## [1] FALSE
```

Potentially problematic signatures

We explore whether there are problematic signatures:

```
colSums(obj_Lung_AdenoCA$Y == 0)/nrow(obj_Lung_AdenoCA$Y)
```

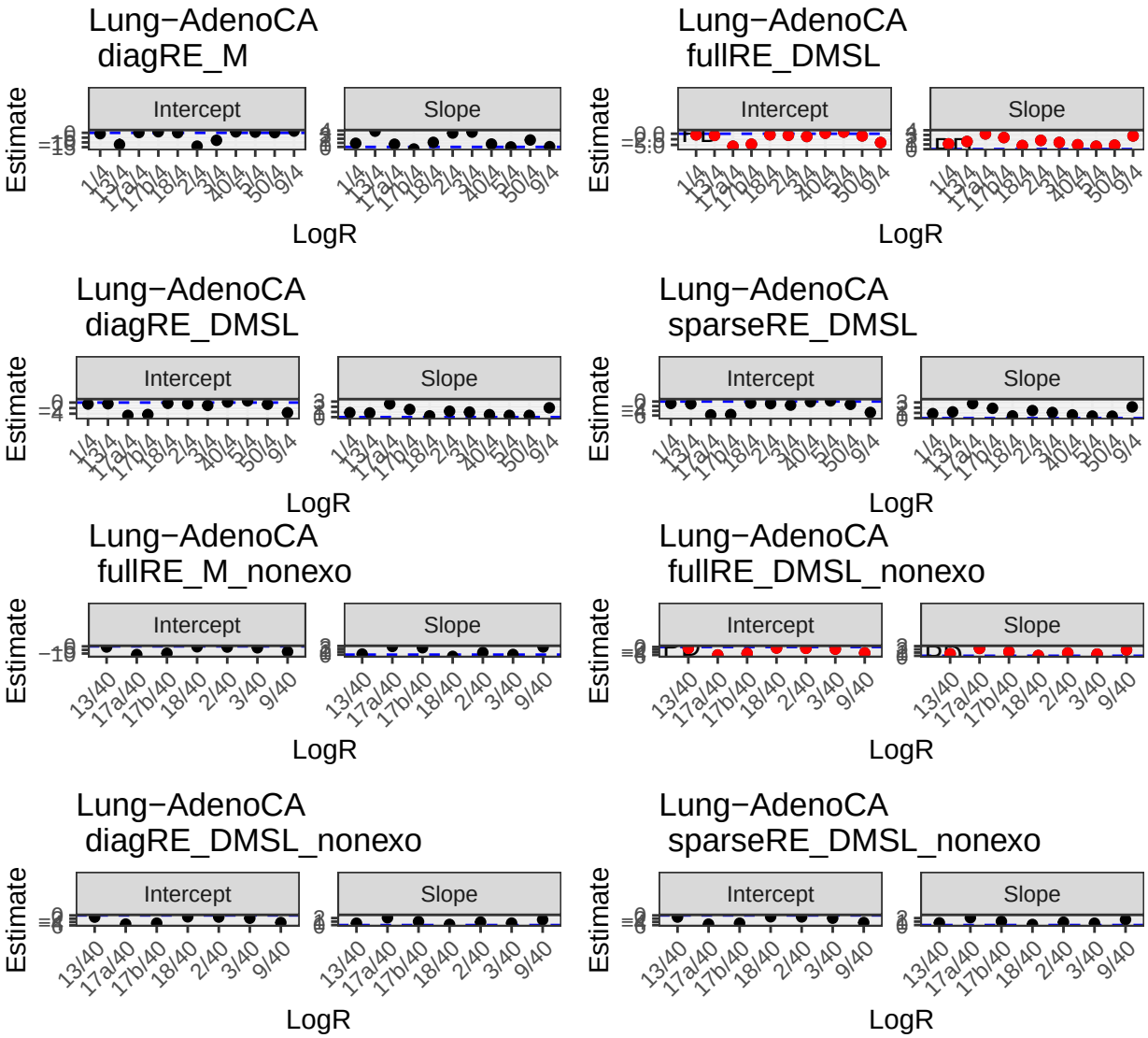
```
##       SBS1       SBS2       SBS3       SBS4       SBS5       SBS9      SBS13
## 0.02941176 0.02941176 0.26470588 0.17647059 0.08823529 0.58823529 0.08823529
##     SBS17a     SBS17b      SBS18      SBS40      SBS50
## 0.64705882 0.64705882 0.14705882 0.14705882 0.11764706
```

```
colSums(obj_Lung_AdenoCA$Y)/sum(obj_Lung_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS3       SBS4       SBS5       SBS9      SBS13
## 0.02550030 0.09137609 0.05699124 0.32022800 0.13218445 0.02860308 0.07430021
##     SBS17a     SBS17b      SBS18      SBS40      SBS50
## 0.01139098 0.02722878 0.07136089 0.13654347 0.02429249
```

None seem to be problematic.

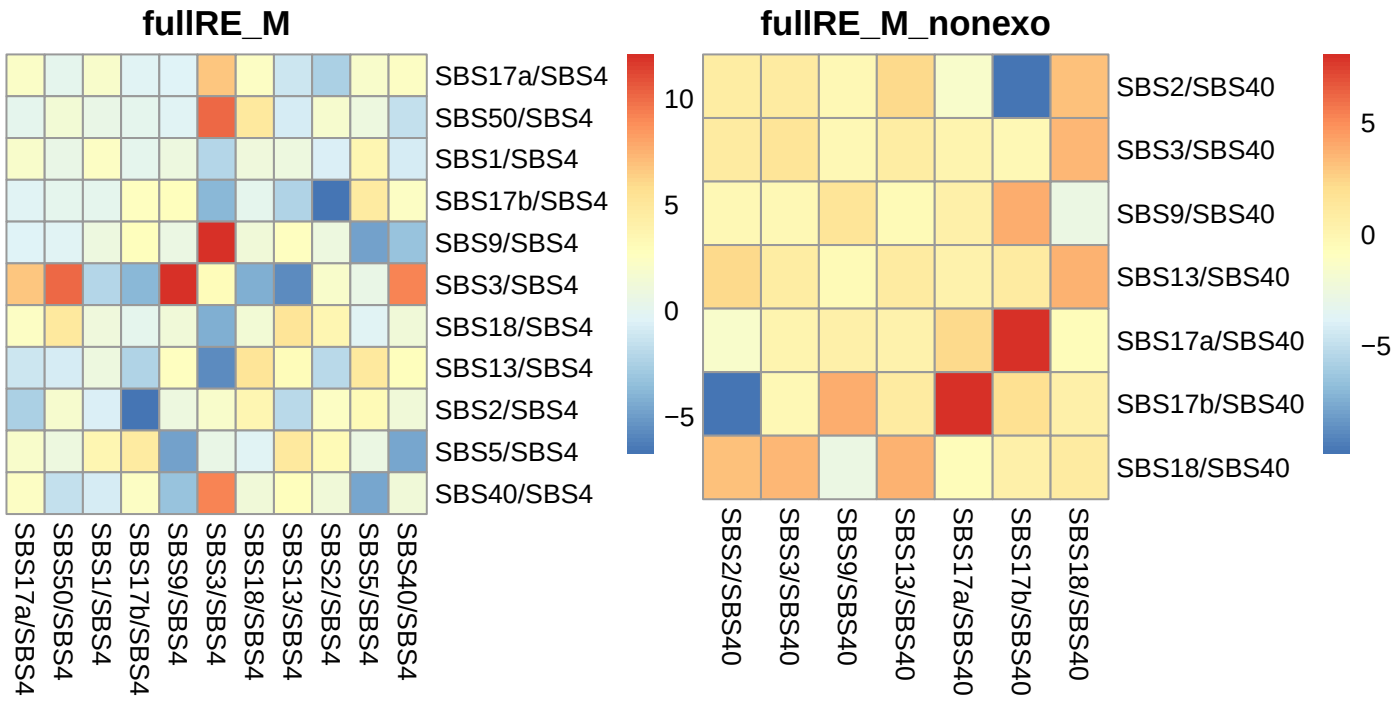
Betas



We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 0.0034356.

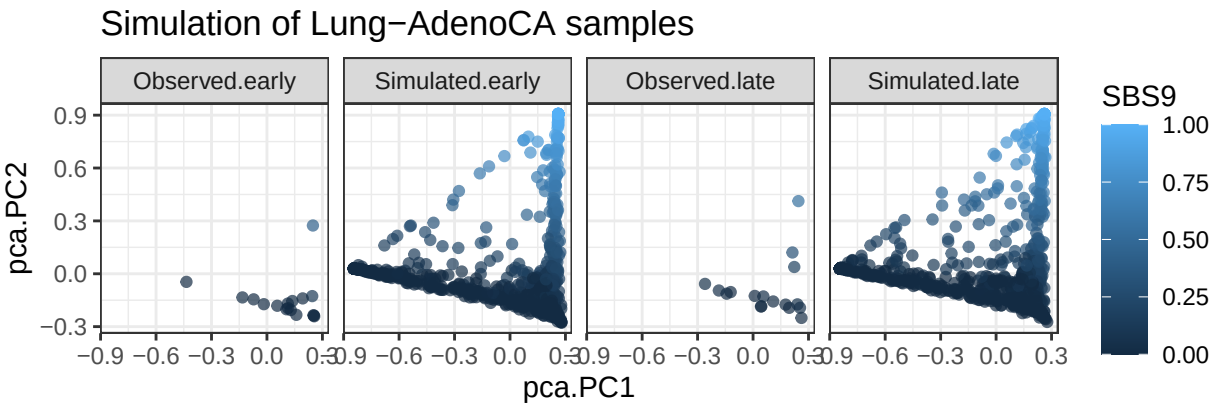
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

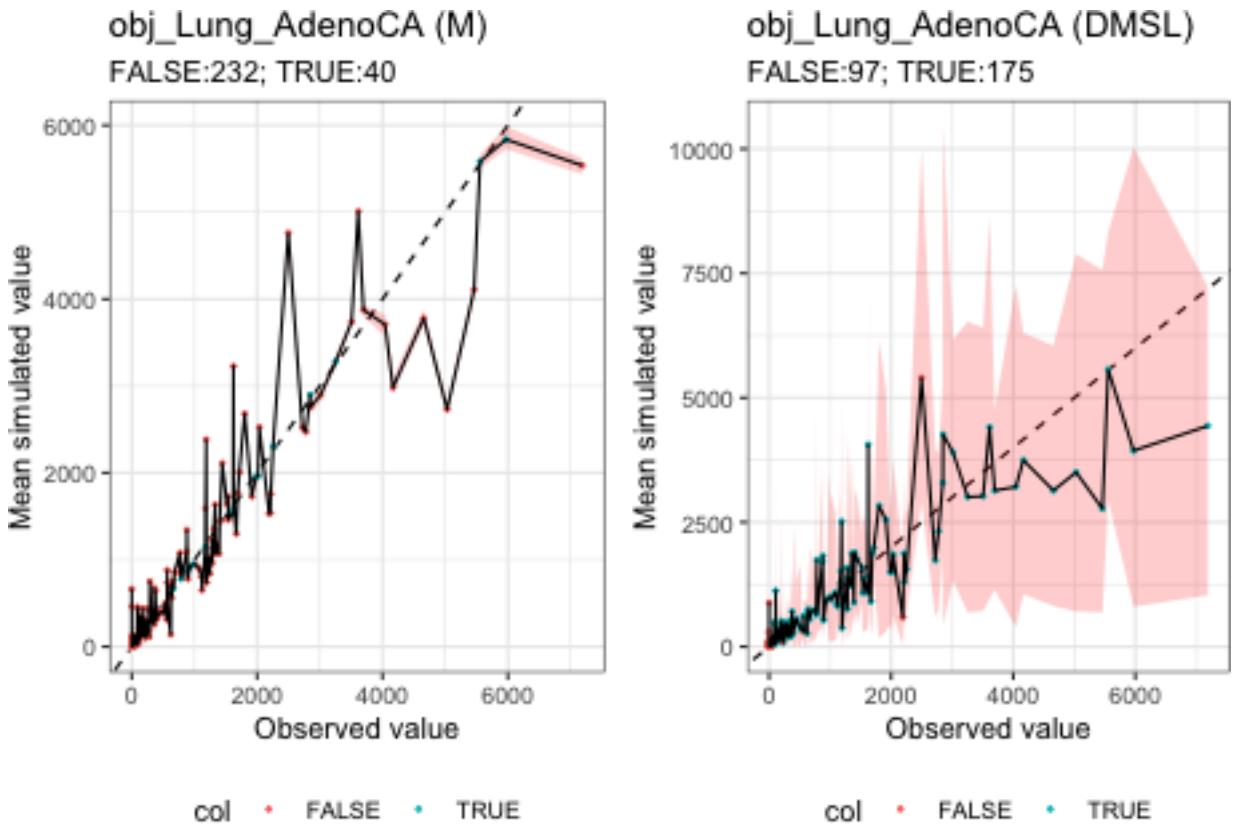


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



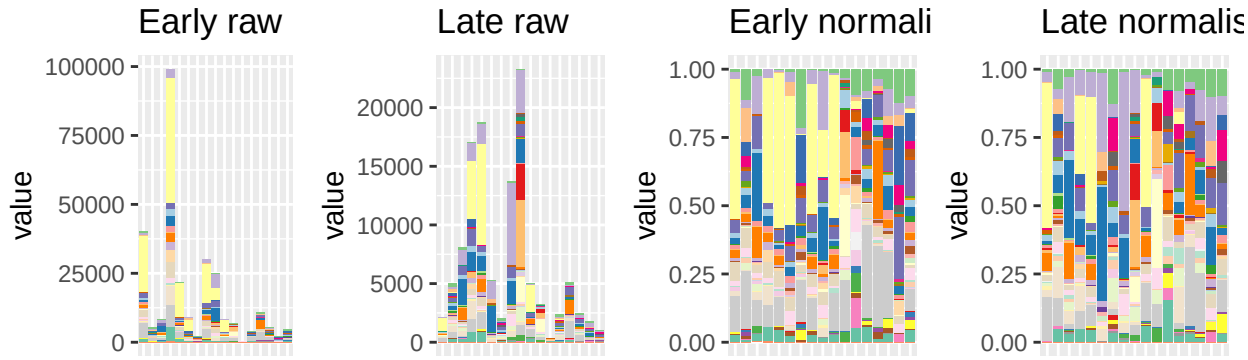
Ranked plot for coverage



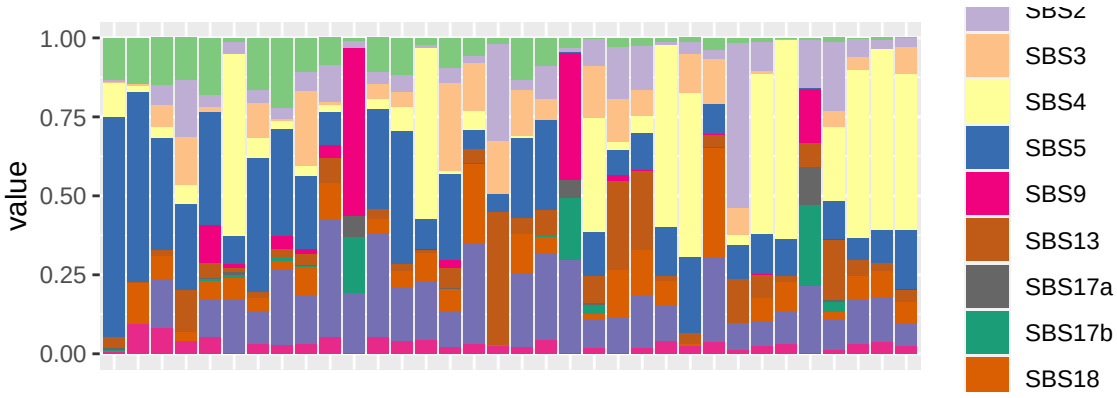
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 17
```



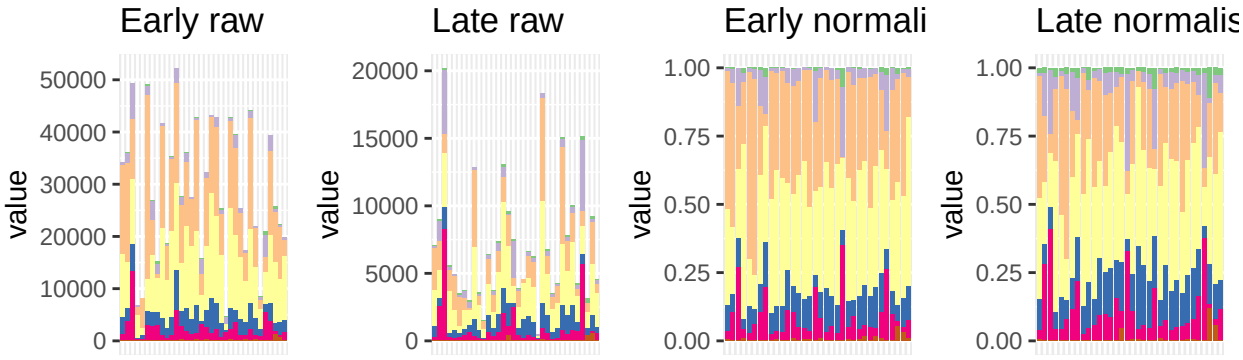
Exposures sorted by increasing number of mutations: there is a trend in which SBS5 decreases and SBS4 increases with the number of mutations.



Lung-SCC

Barplot and general statistics

```
## [1] 34
```



The number of samples and signatures is:

```
## [1] 68  7
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS4"  "SBS5"  "SBS8"  "SBS13" "SBS33"
```

Convergence table

We have converged results in most cases, and all in nonexo.

```
##       value                                L2                   L1
## 1  Lung-SCC    hessian_positivedefinite_bool               diagRE_M
## 2  Lung-SCC    hessian_positivedefinite_bool               fullRE_M
## 3  Lung-SCC hessian_nonpositivedefinite_bool            diagRE_DMDL
## 4  Lung-SCC                           Timeout           fullRE_halfDM
## 5  Lung-SCC hessian_nonpositivedefinite_bool            fullRE_DMDL
## 6  Lung-SCC    hessian_positivedefinite_bool            diagRE_DMSL
## 7  Lung-SCC    hessian_positivedefinite_bool           sparseRE_DMSL
## 8  Lung-SCC    hessian_positivedefinite_bool            fullRE_DMSL
## 9  Lung-SCC hessian_nonpositivedefinite_bool        fullRE_DMSL_SBS1
## 10 Lung-SCC    hessian_positivedefinite_bool         fullRE_M_nonexo
## 11 Lung-SCC hessian_nonpositivedefinite_bool      diagRE_DMSL_nonexo
```

```
## 12 Lung-SCC    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Lung-SCC    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 Lung-SCC    hessian_positivedefinite_bool       fullRE_DMDL_nonexo
## 15 Lung-SCC                          Timeout fullRE_DMDL_sortednonexo
```

Potentially problematic signatures

We explore whether there are problematic signatures; none are.

```
colSums(obj_Lung_SCC$Y == 0)/nrow(obj_Lung_SCC$Y)
```

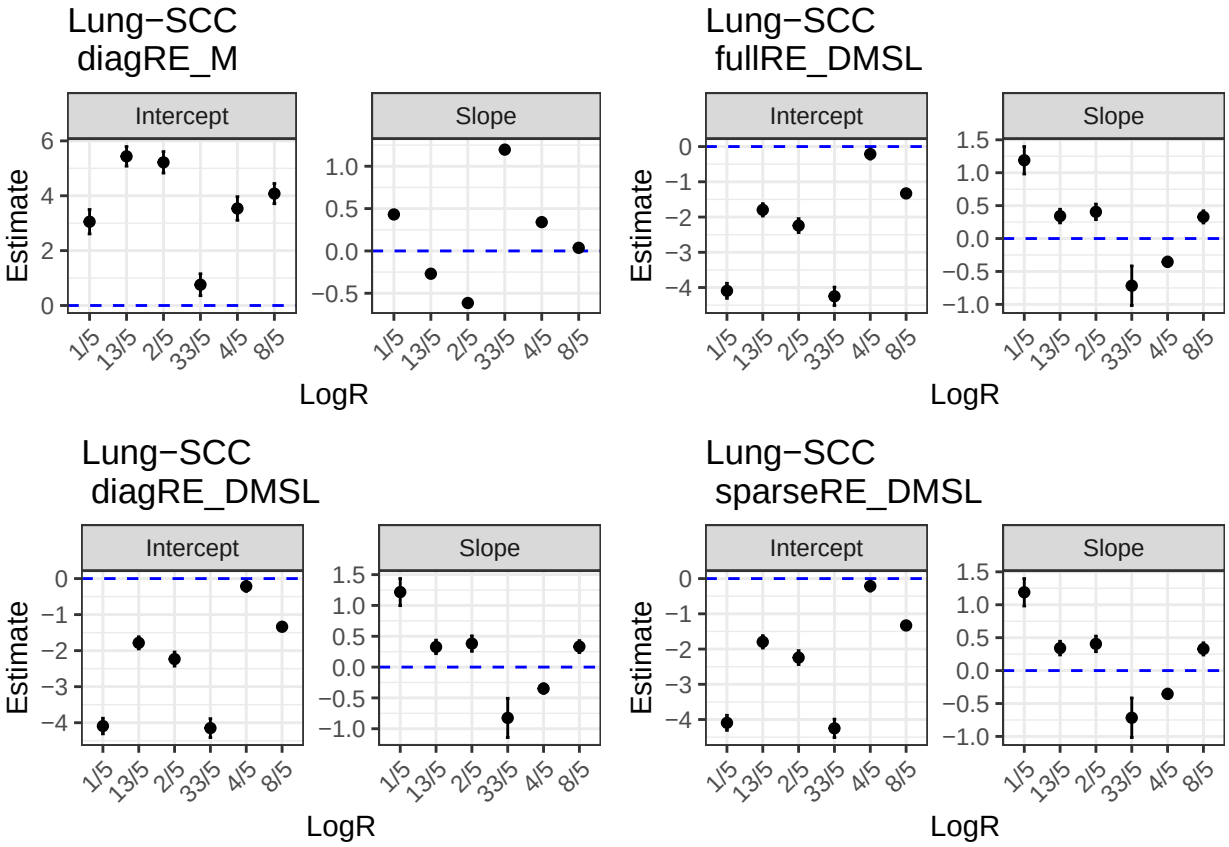
```
##       SBS1       SBS2       SBS4       SBS5       SBS8      SBS13      SBS33
## 0.11764706 0.01470588 0.01470588 0.00000000 0.00000000 0.00000000 0.35294118
```

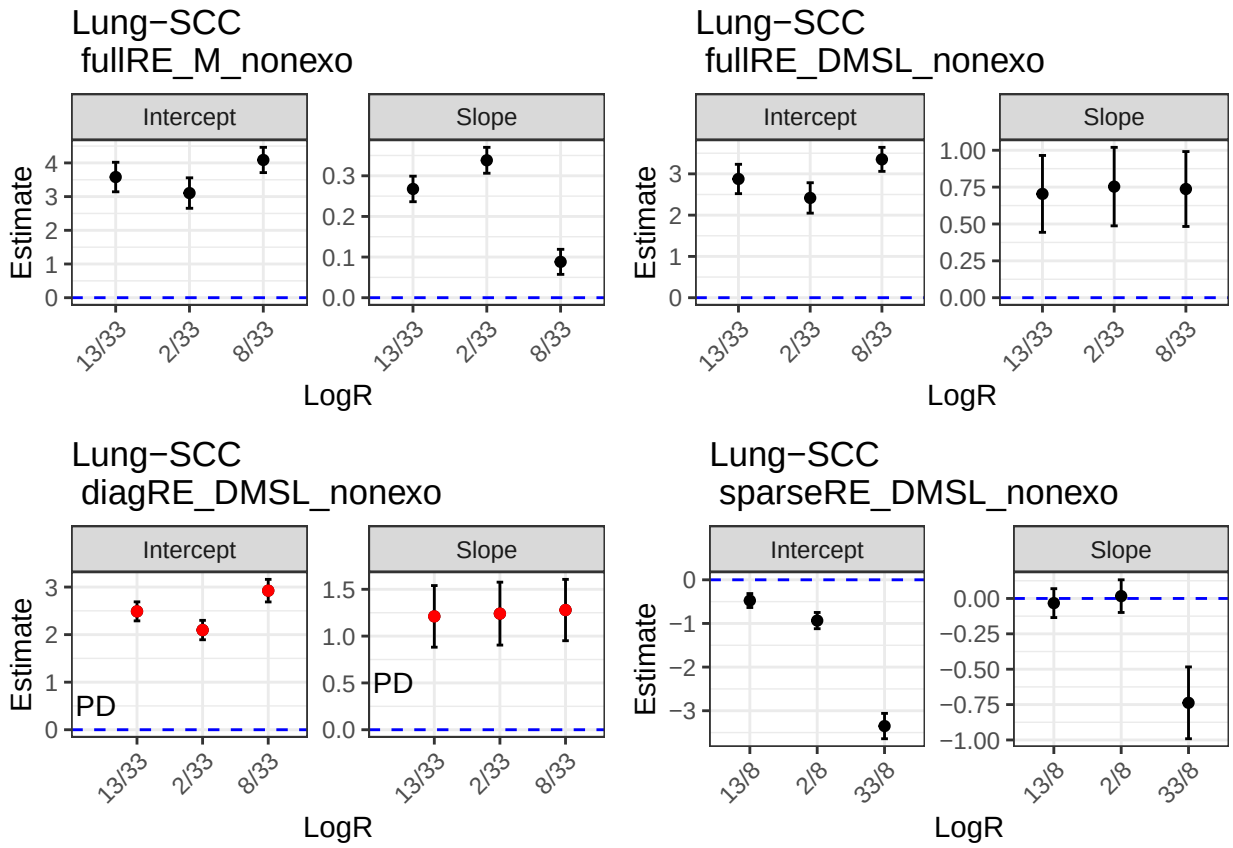
```
colSums(obj_Lung_SCC$Y)/sum(obj_Lung_SCC$Y)
```

```
##        SBS1        SBS2        SBS4        SBS5        SBS8       SBS13
## 0.007157222 0.057307486 0.361344268 0.373780803 0.107939657 0.086773493
##       SBS33
## 0.005697071
```

Betas

Very clear example of only one signature changing:

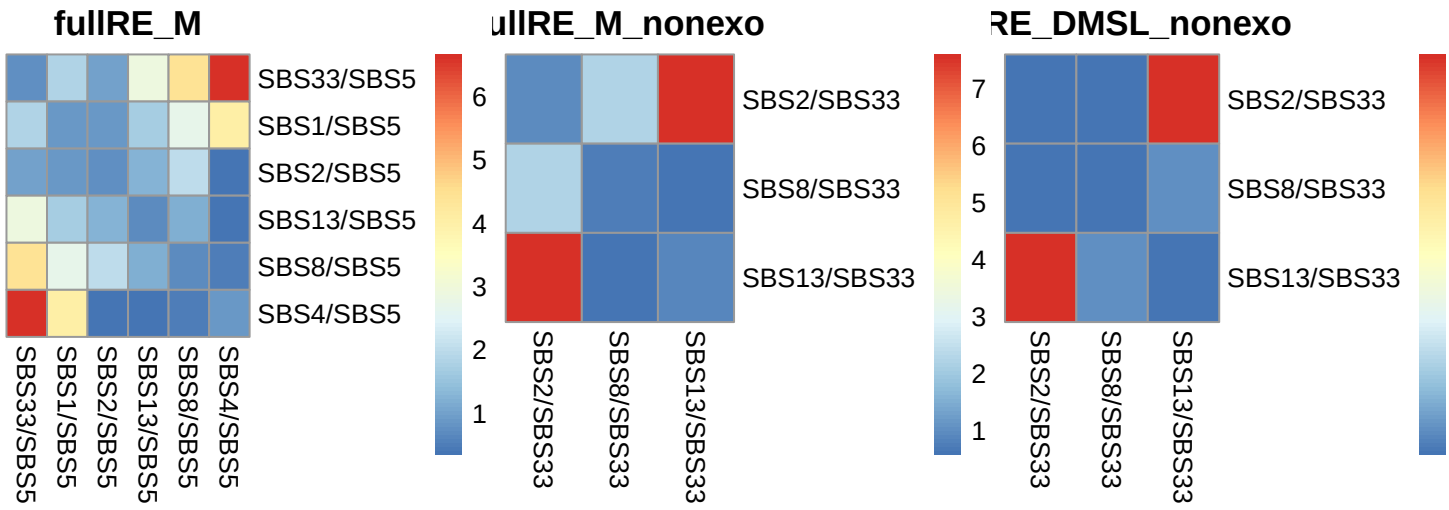




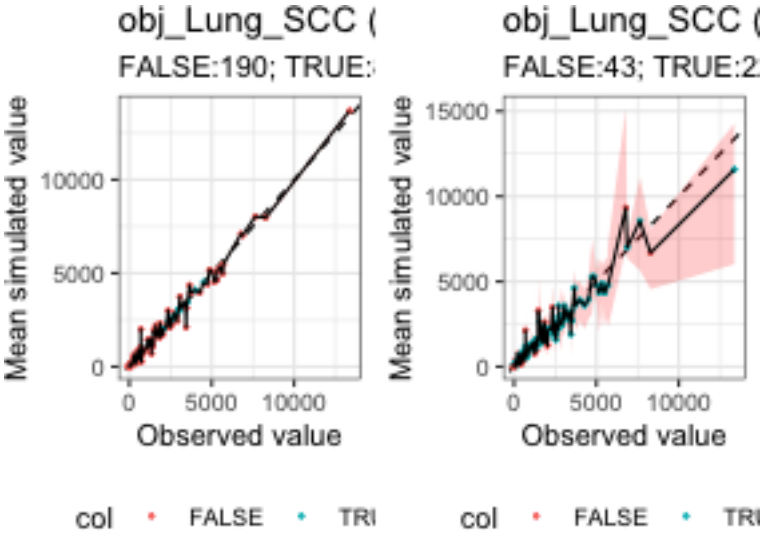
We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.0338506.

Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



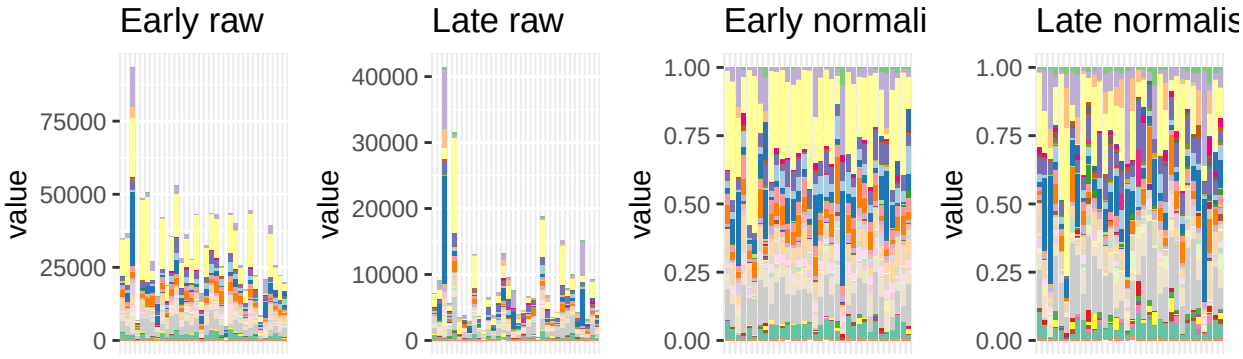
Ranked plot for coverage



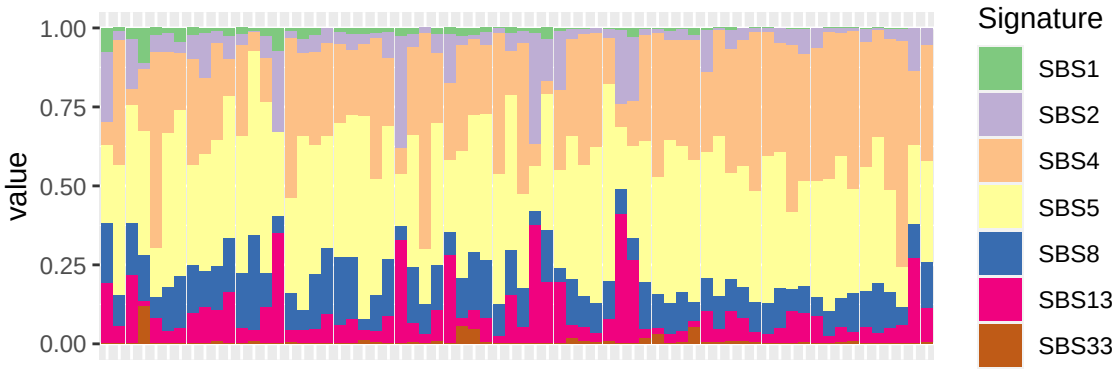
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 34
```



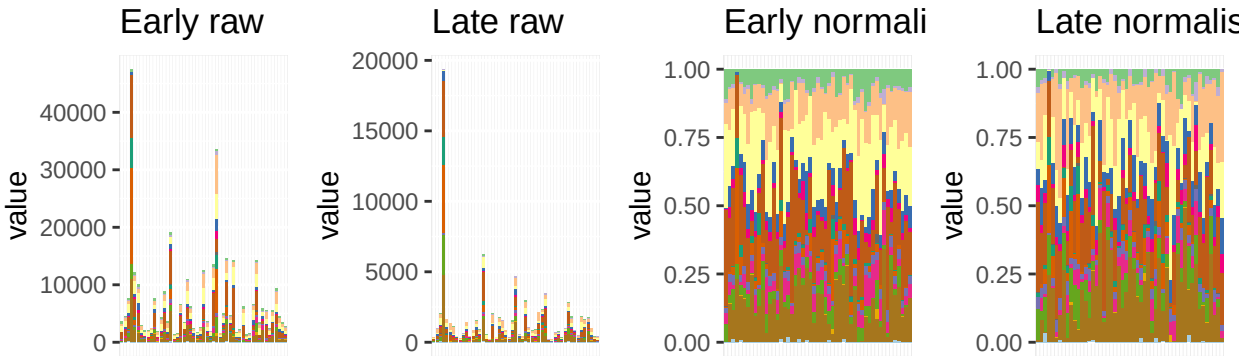
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Lymph-BNHL

Barplot and general statistics

```
## [1] 51
```



The number of samples and signatures is:

```
## [1] 102  16
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS6"   "SBS7b"  "SBS9"   "SBS13"
##  [9] "SBS17a" "SBS17b" "SBS34"  "SBS36"  "SBS37"  "SBS39"  "SBS40"  "SBS56"
```

Convergence table

fullRE_DMSL_nonexo had not run, and fullRE_M_nonexo didn't converge.

```
##         value                                L2                  L1
## 1  Lymph-BNHL    hessian_positivedefinite_bool          diagRE_M
## 2  Lymph-BNHL hessian_nonpositivedefinite_bool          fullRE_M
## 3  Lymph-BNHL    hessian_positivedefinite_bool        diagRE_DMDL
## 4  Lymph-BNHL                          Timeout      fullRE_halfDM
## 5  Lymph-BNHL                          Timeout        fullRE_DMDL
## 6  Lymph-BNHL    hessian_positivedefinite_bool        diagRE_DMSL
## 7  Lymph-BNHL    hessian_positivedefinite_bool      sparseRE_DMSL
## 8  Lymph-BNHL hessian_nonpositivedefinite_bool        fullRE_DMSL
```

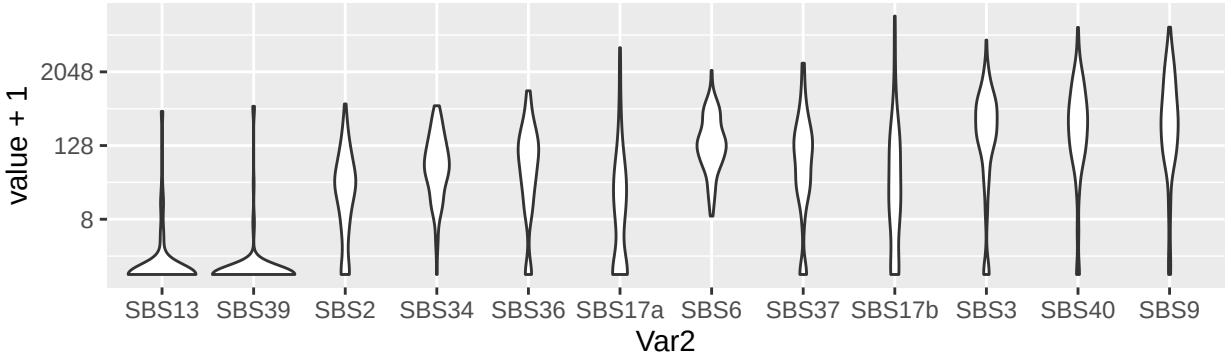
```
## 9  Lymph-BNHL hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Lymph-BNHL hessian_nonpositivedefinite_bool          fullRE_M_nonexo
## 11 Lymph-BNHL    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Lymph-BNHL    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Lymph-BNHL                          Timeout       fullRE_DMSL_nonexo
## 14 Lymph-BNHL hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Lymph-BNHL    hessian_positivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo. Very clearly there are too many signatures.

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Which signatures should be omitted from the analysis?



SBS13 and SBS39 should definitely be removed.

Has fullRE M now converged? converge:

```
## [1] TRUE
```

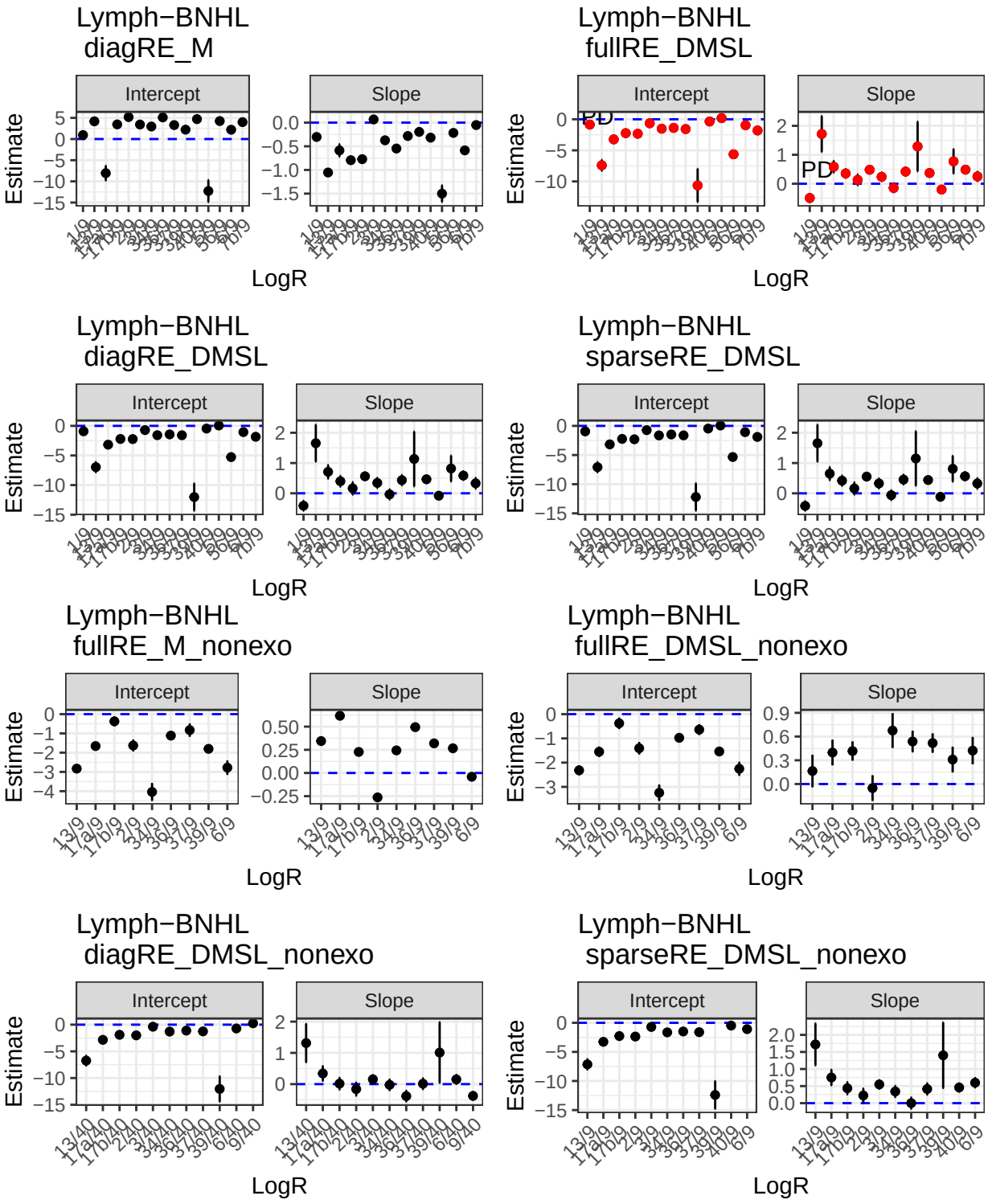
it has. I now run DM with this subset

Its convergence is as follows:

```
## [1] TRUE
```

it has also converged

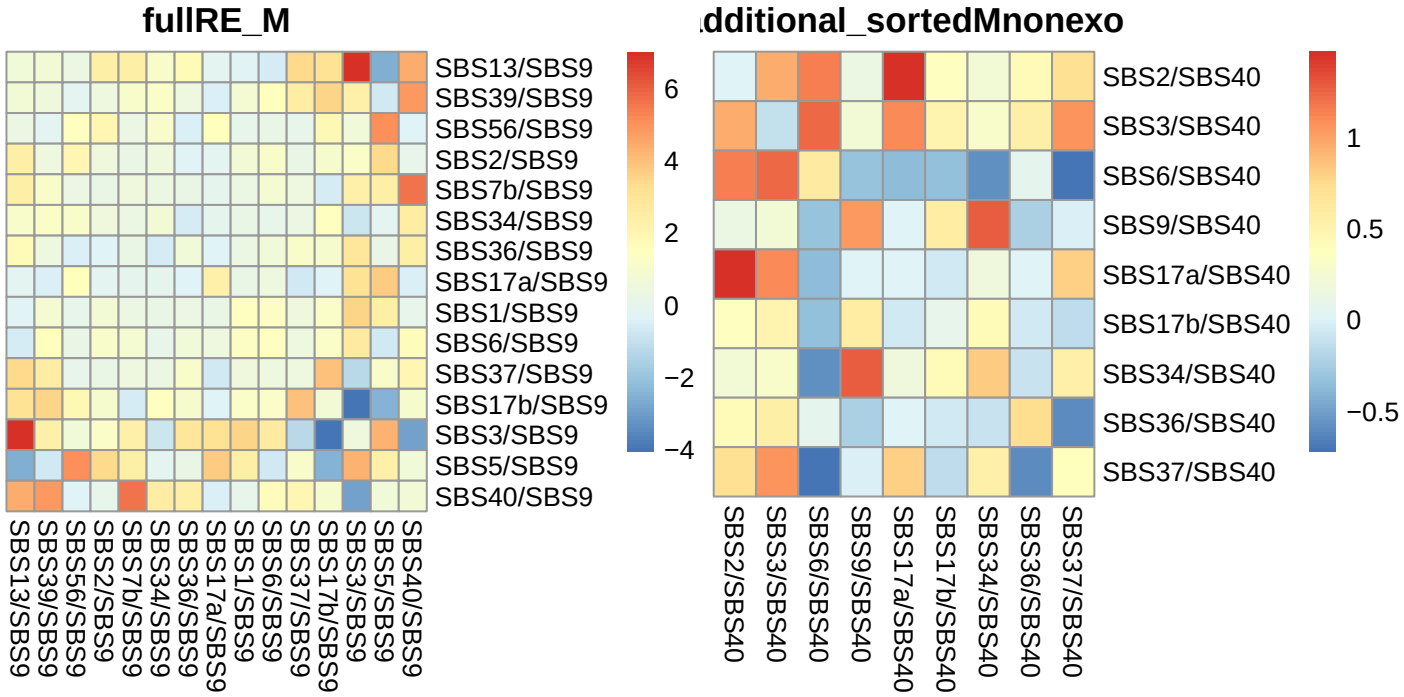
Betas

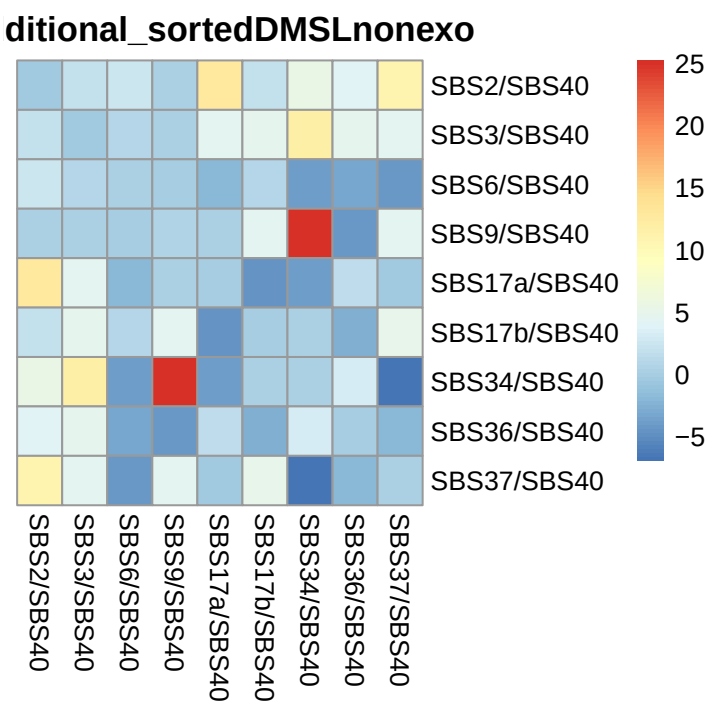


We use the results from the full RE single lambda DM with the subset of signatures (removing thw two problematic ones) to test for differential abundance, giving a p-value of 9.5835037×10^{-7}.

Covariance matrices

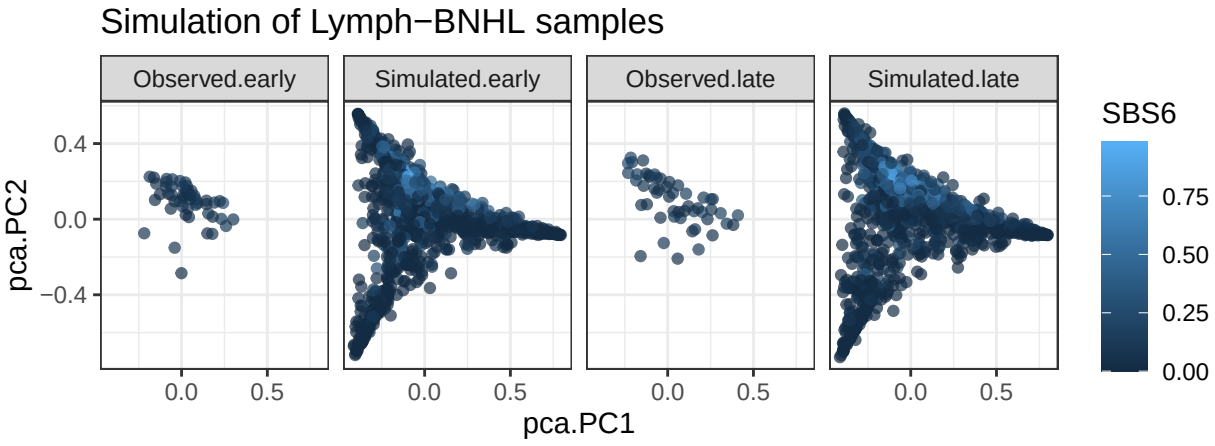
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



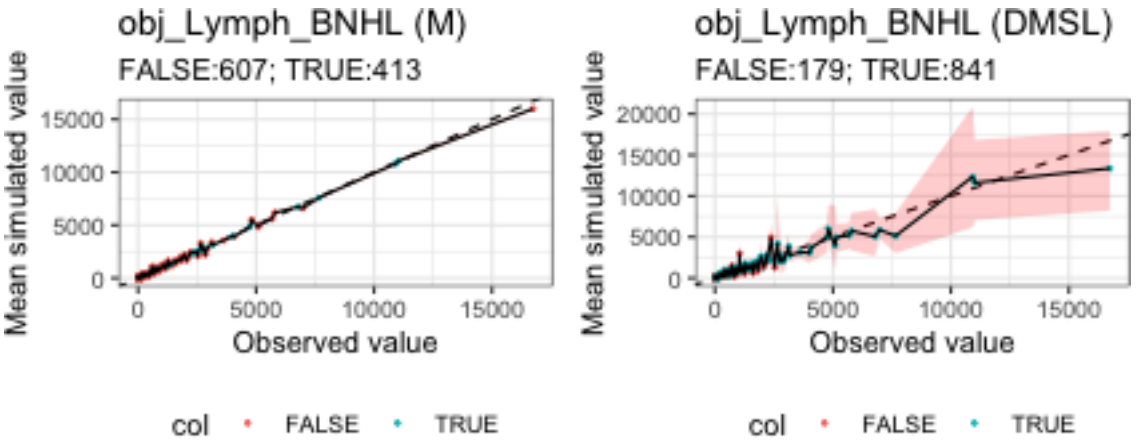


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)

## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



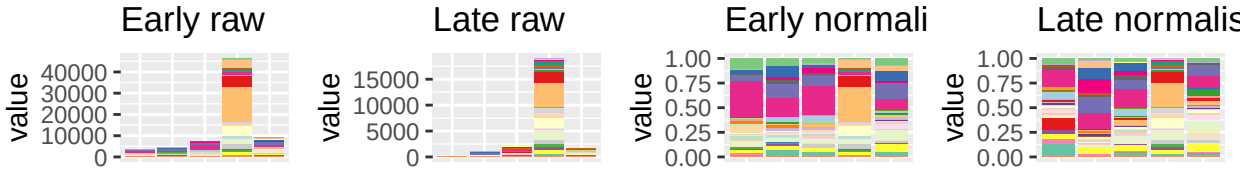
Ranked plot for coverage



Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 5
```

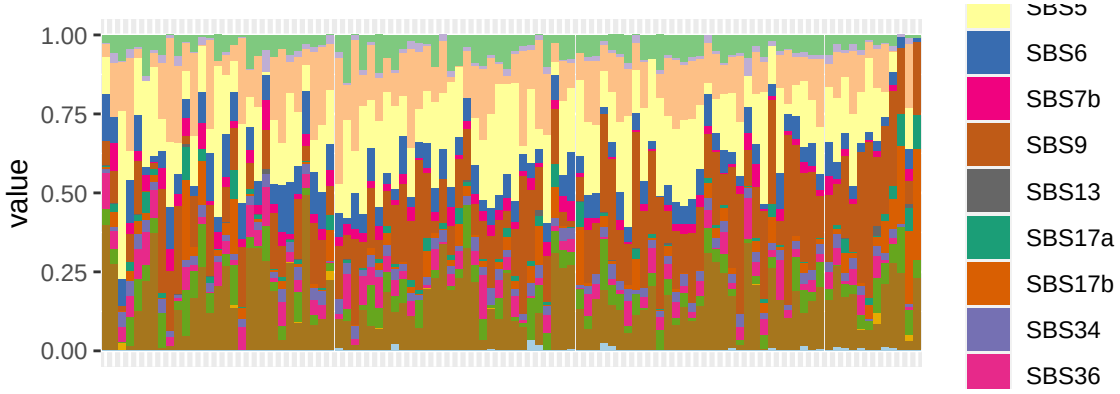


something must have gone wrong here. Dims of Y for mutsigextractor and normal signatures:

```
## [1] 10 53
```

```
## [1] 102  16
```

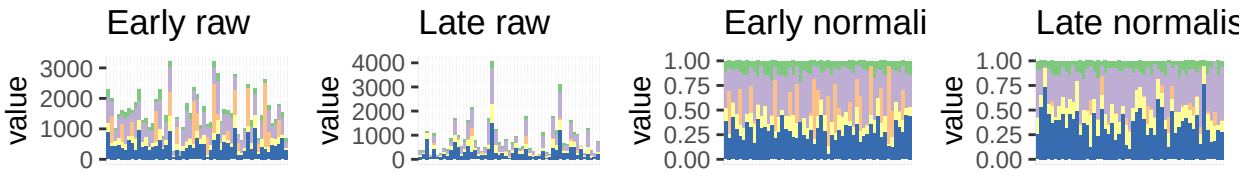
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Lymph-CLL

Barplot and general statistics

```
## [1] 53
```



The number of samples and signatures is:

```
## [1] 106   5
```

The signatures are:

```
## [1] "SBS1"  "SBS5"  "SBS9"  "SBS25" "SBS40"
```

Convergence table

We have converged results in most cases

```
##        value                                L2                     L1
## 1  Lymph-CLL    hessian_positivedefinite_bool                diagRE_M
## 2  Lymph-CLL    hessian_positivedefinite_bool                fullRE_M
## 3  Lymph-CLL    hessian_positivedefinite_bool             diagRE_DMDL
## 4  Lymph-CLL                          Timeout            fullRE_halfDM
## 5  Lymph-CLL hessian_nonpositivedefinite_bool             fullRE_DMDL
## 6  Lymph-CLL    hessian_positivedefinite_bool             diagRE_DMSL
## 7  Lymph-CLL    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  Lymph-CLL hessian_nonpositivedefinite_bool             fullRE_DMSL
## 9  Lymph-CLL hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Lymph-CLL    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Lymph-CLL    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Lymph-CLL                          Timeout     sparseRE_DMSL_nonexo
## 13 Lymph-CLL    hessian_positivedefinite_bool       fullRE_DMSL_nonexo
## 14 Lymph-CLL    hessian_positivedefinite_bool       fullRE_DMDL_nonexo
## 15 Lymph-CLL                          Timeout fullRE_DMDL_sortednonexo
```

Potentially problematic signatures

SBS9 has quite a lot of zeros.

```
colSums(obj_Lymph_CLL$Y == 0)/nrow(obj_Lymph_CLL$Y)
```

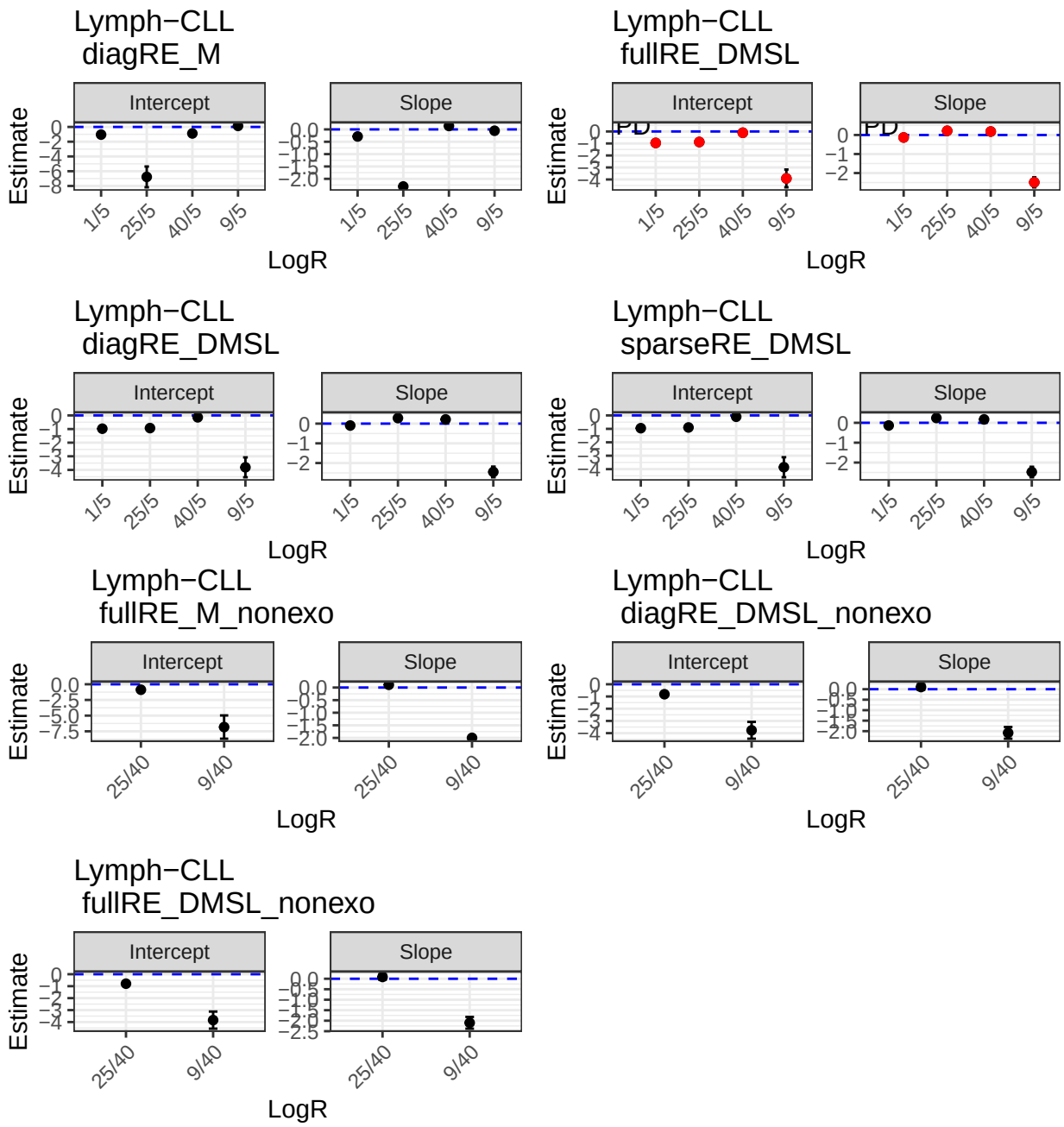
```
##        SBS1        SBS5        SBS9       SBS25       SBS40
## 0.000000000 0.028301887 0.613207547 0.009433962 0.000000000
```

```
colSums(obj_Lymph_CLL$Y)/sum(obj_Lymph_CLL$Y)
```

```
##       SBS1       SBS5       SBS9      SBS25      SBS40
## 0.09712712 0.33726681 0.12275176 0.13805198 0.30480234
```

Betas

It's interesting the very high correlation between intercept and slope betas.

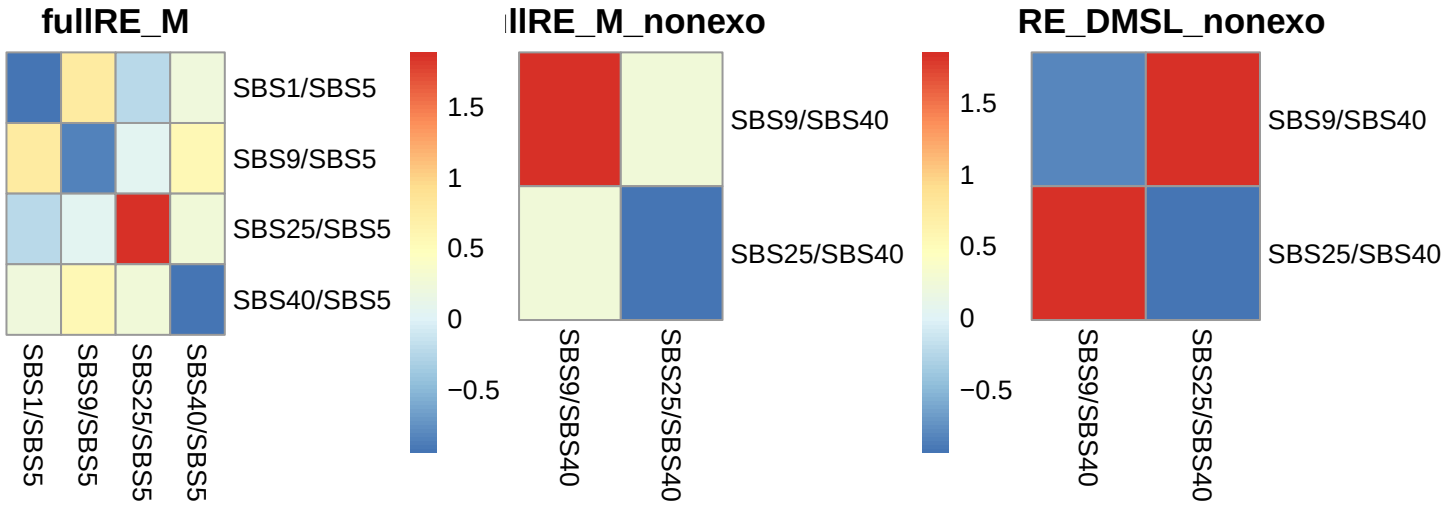


We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of $6.1779312 \times 10^{-14}$.

Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

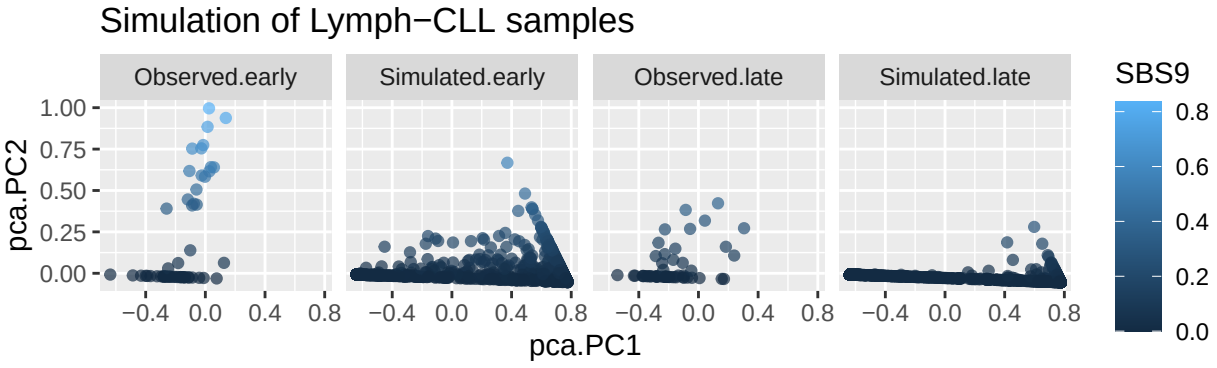


Simulation under inferred data

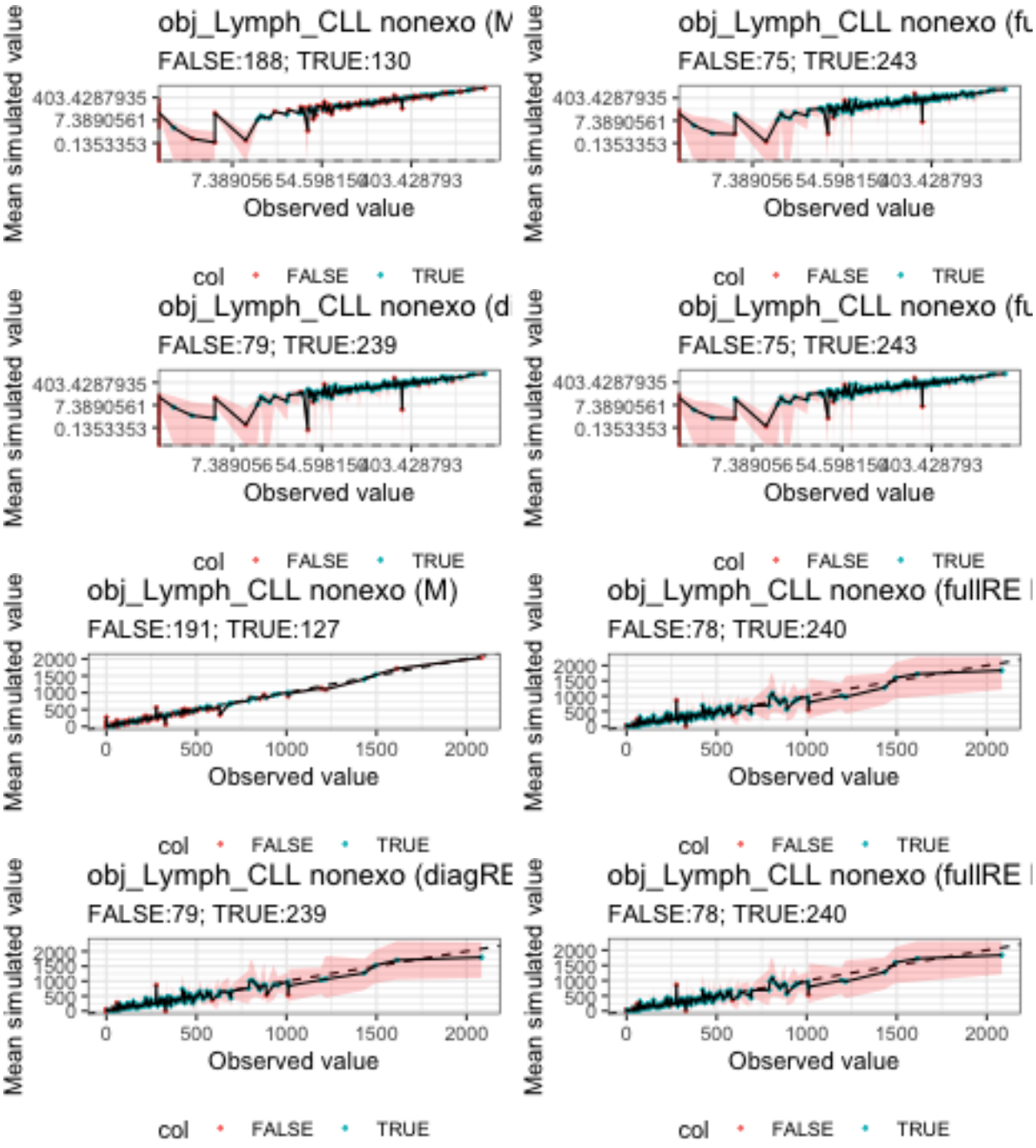
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



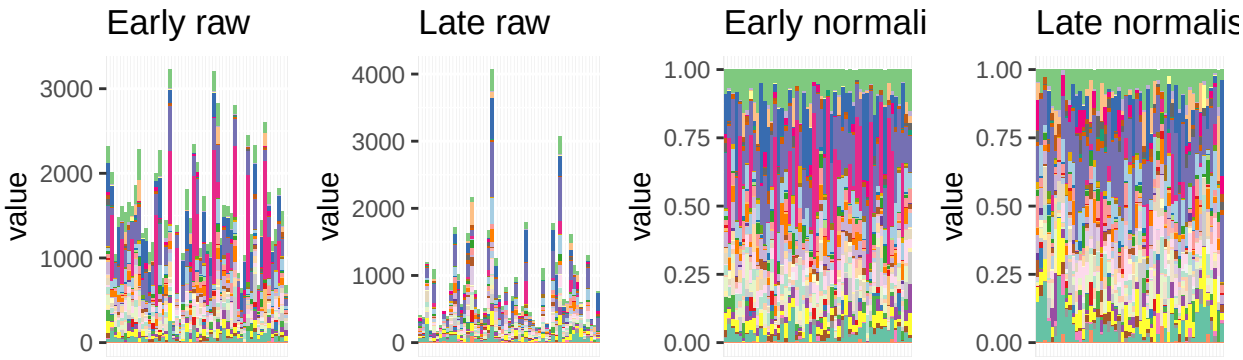
Ranked plot for coverage



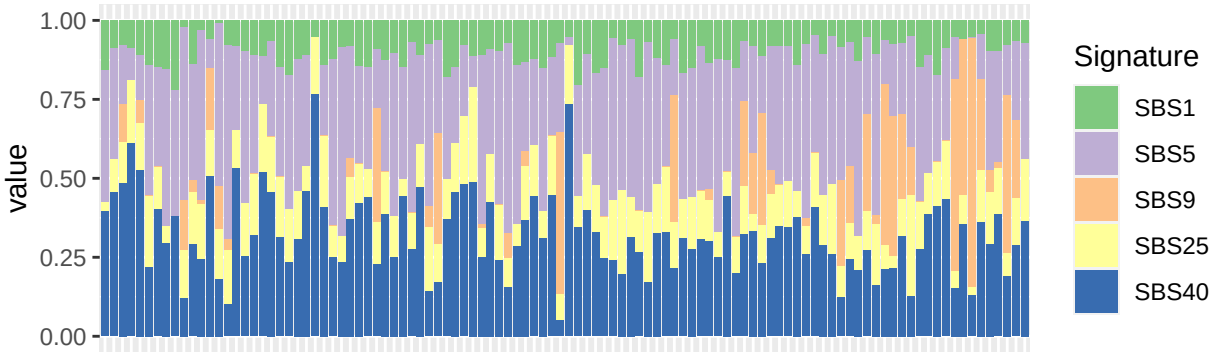
Signatures from mutSigExtractor

These are the signatures from mutSigExtractor:

```
## [1] 53
```



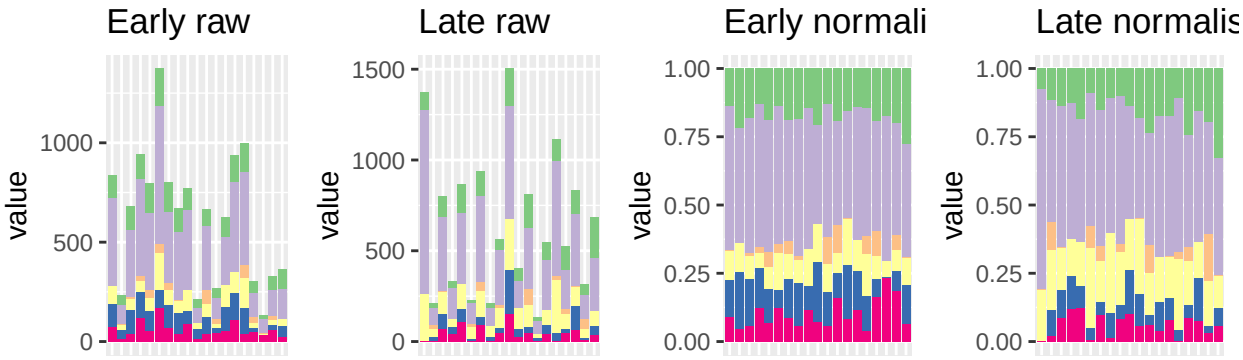
Exposures sorted by increasing number of mutations: SBS9 and SBS25 seem to be somewhat associated with samples with a high number of mutations.



Myeloid-MPN

Barplot and general statistics

```
## [1] 19
```



The number of samples and signatures is:

```
## [1] 38  6
```

The signatures are:

```
## [1] "SBS1"  "SBS5"  "SBS6"  "SBS8"  "SBS19" "SBS32"
```

Convergence table

These are the results for the convergence of models fits. The fullRE DMSL have not converged, or have not run.

```
## [1] value L2    L1
## <0 rows> (or 0-length row.names)
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do converge:

```
## [1] TRUE
```

Potentially problematic signatures

We explore whether there are problematic signatures. There are none.

```
colSums(obj_Myeloid_MPN$Y == 0)/nrow(obj_Myeloid_MPN$Y)
```

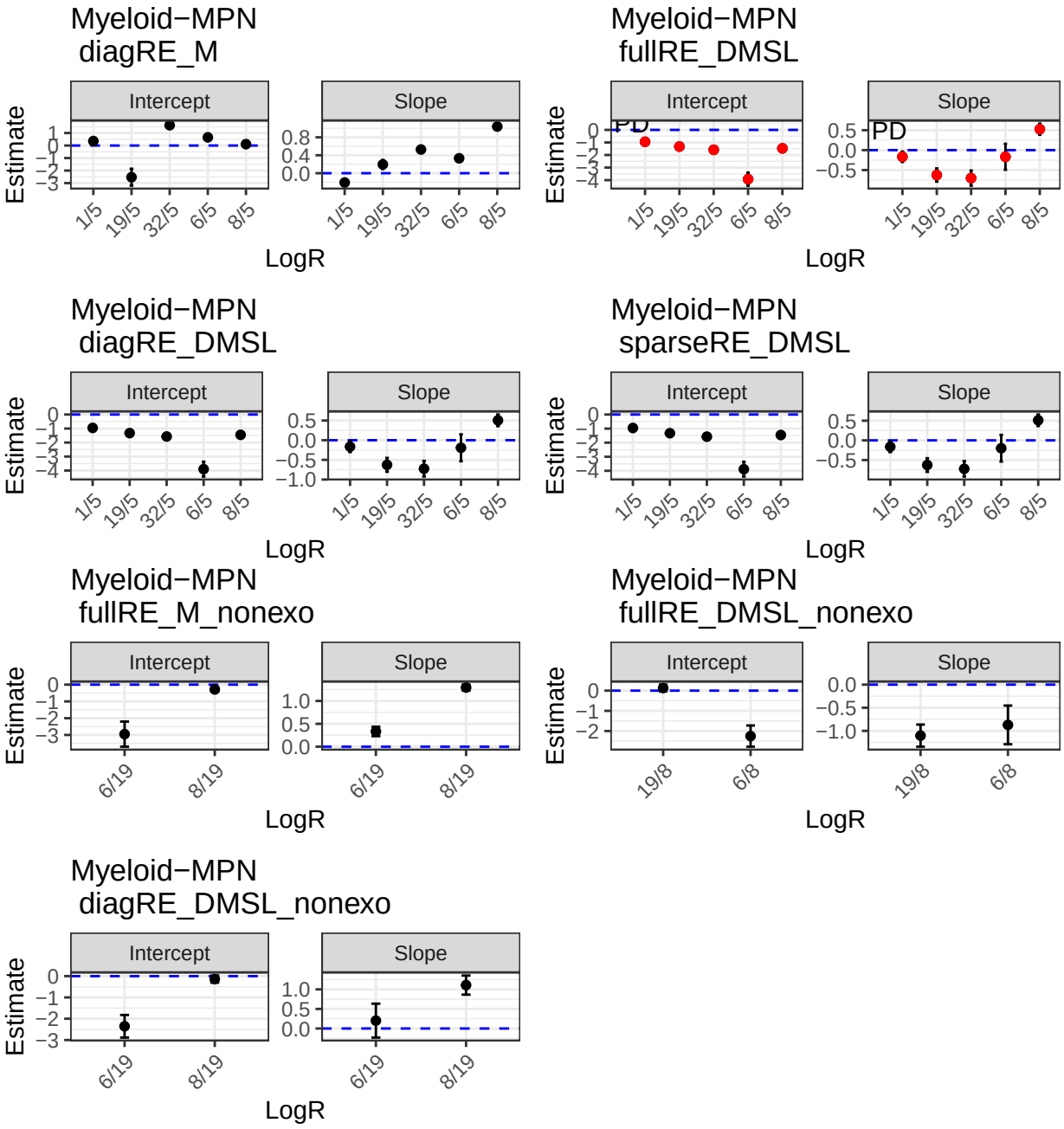
```
##       SBS1       SBS5       SBS6       SBS8      SBS19      SBS32
## 0.00000000 0.00000000 0.50000000 0.00000000 0.05263158 0.05263158
```

```
colSums(obj_Myeloid_MPN$Y)/sum(obj_Myeloid_MPN$Y)
```

```
##       SBS1       SBS5       SBS6       SBS8      SBS19      SBS32
## 0.16009042 0.48849157 0.02737361 0.14488286 0.10098644 0.07817509
```

We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 1.367896×10^{-5}.

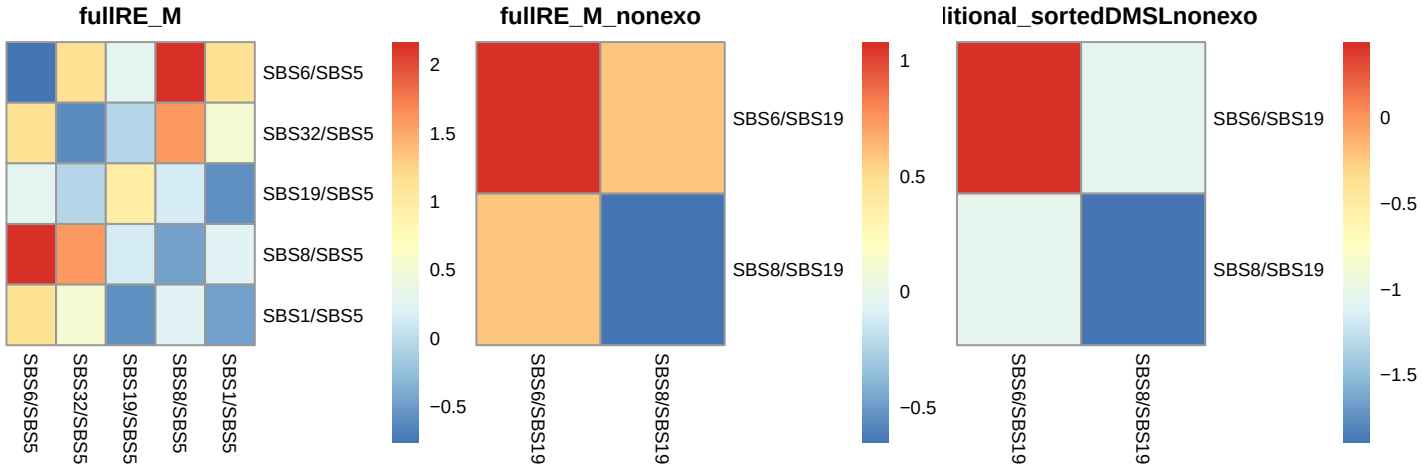
Betas



Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
```

```
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

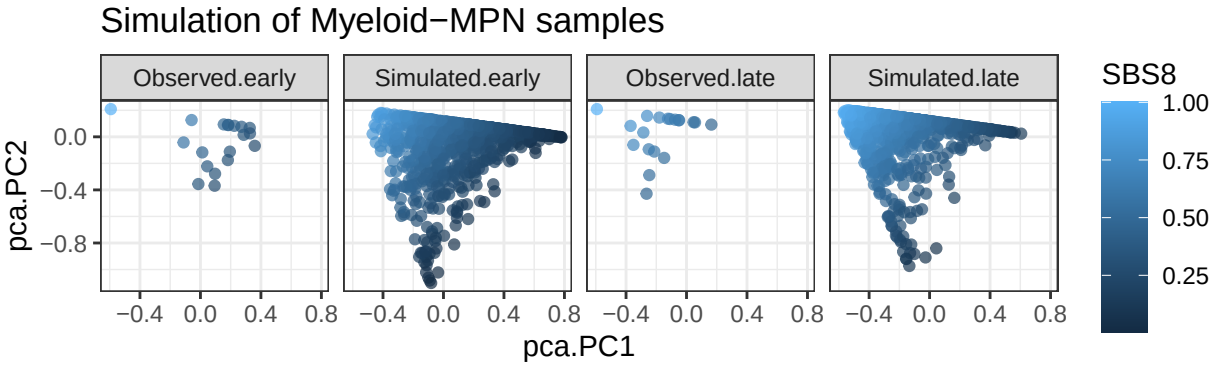


Simulation under inferred data

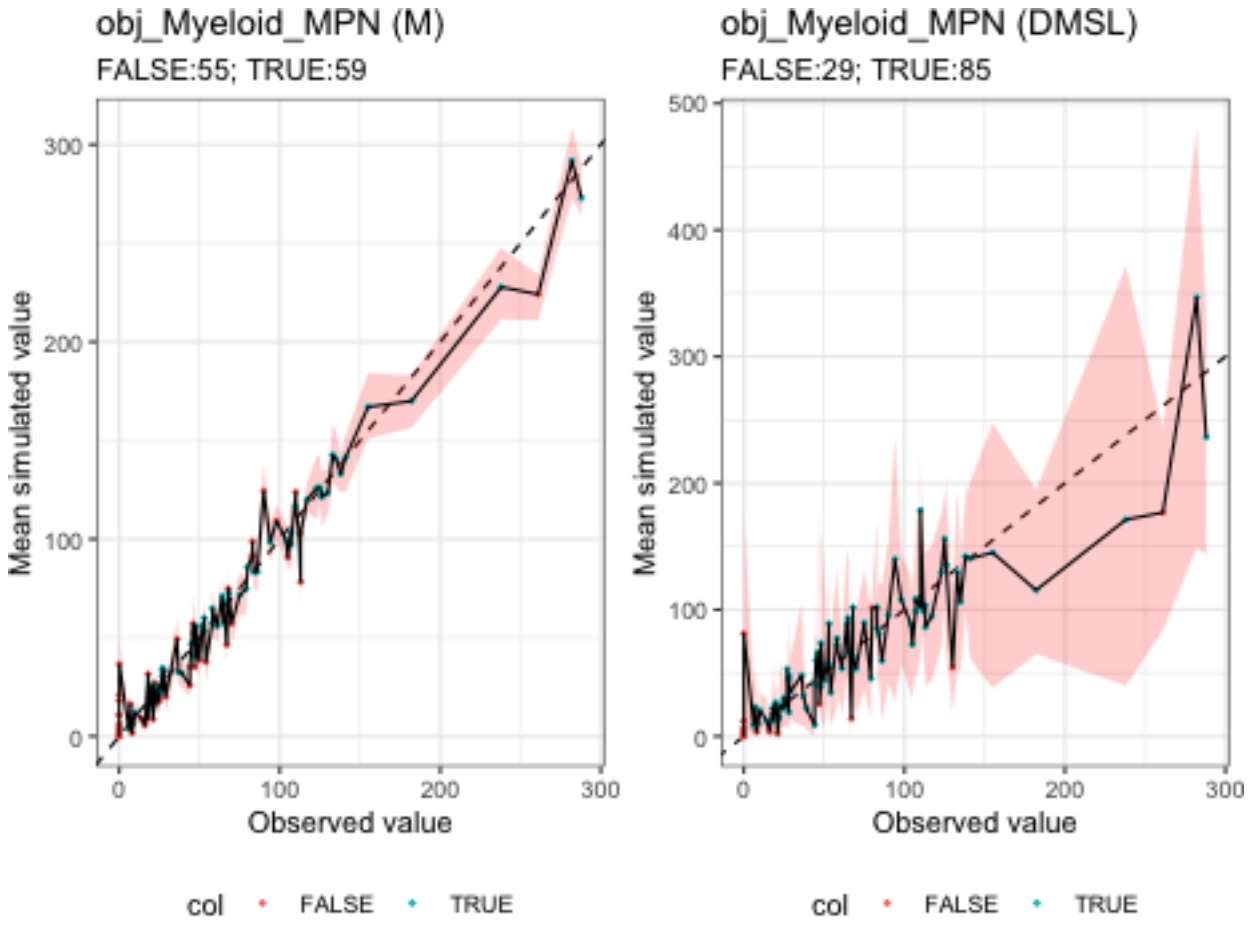
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



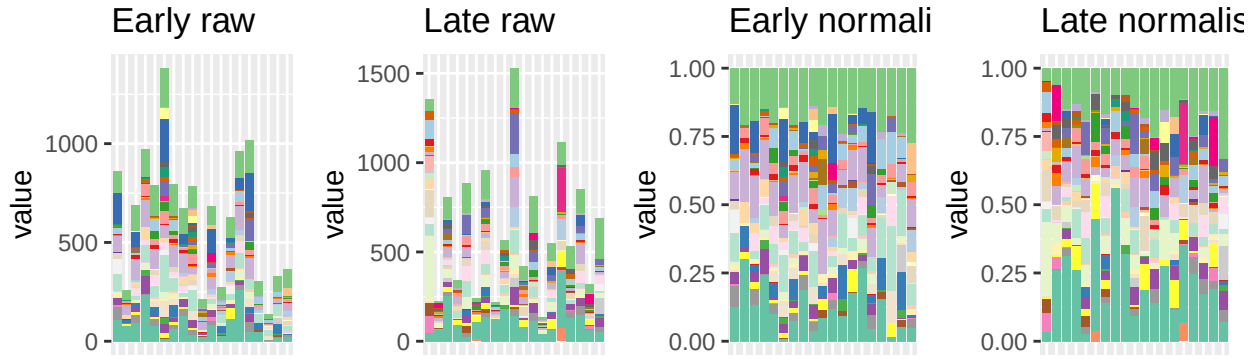
Ranked plot for coverage



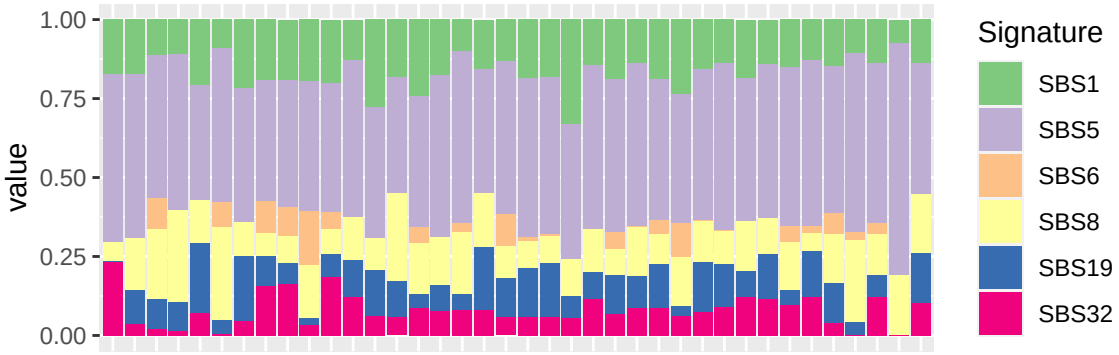
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 19
```



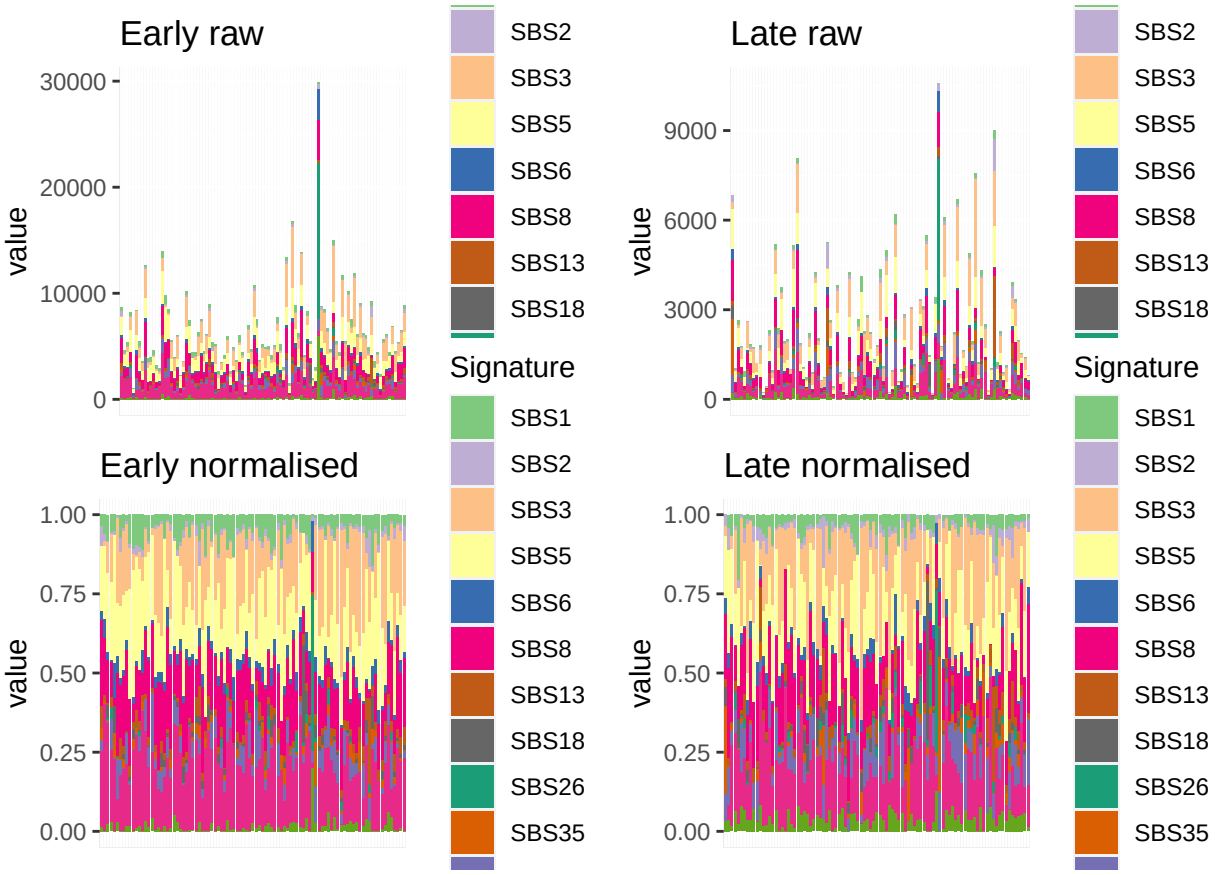
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Ovary-AdenoCA

Barplot and general statistics

```
## [1] 97
```



The number of samples and signatures is:

```
## [1] 194  13
```

The signatures are:

```
##  [1] "SBS1"  "SBS2"  "SBS3"  "SBS5"  "SBS6"  "SBS8"  "SBS13" "SBS18" "SBS26"
## [10] "SBS35" "SBS39" "SBS40" "SBS41"
```

Convergence table

These are the results for the convergence of models fits. None of the all-signatures models converged (we do have many signatures!) but nonexo generally have, except fullRE_DMSL_nonexo.

```
##            value                                L2                      L1
## 1  Ovary-AdenoCA    hessian_positivedefinite_bool                diagRE_M
## 2  Ovary-AdenoCA hessian_nonpositivedefinite_bool                fullRE_M
## 3  Ovary-AdenoCA hessian_nonpositivedefinite_bool             diagRE_DMDL
## 4  Ovary-AdenoCA                         Timeout            fullRE_halfDM
## 5  Ovary-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  Ovary-AdenoCA hessian_nonpositivedefinite_bool              diagRE_DMSL
## 7  Ovary-AdenoCA    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  Ovary-AdenoCA    hessian_positivedefinite_bool              fullRE_DMSL
## 9  Ovary-AdenoCA    hessian_positivedefinite_bool         fullRE_DMSL_SBS1
## 10 Ovary-AdenoCA    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Ovary-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Ovary-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Ovary-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 Ovary-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Ovary-AdenoCA                         Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do not yet converge:

```
## [1] FALSE
```

Potentially problematic signatures

We explore whether there are problematic signatures. There doesn't seem to be.

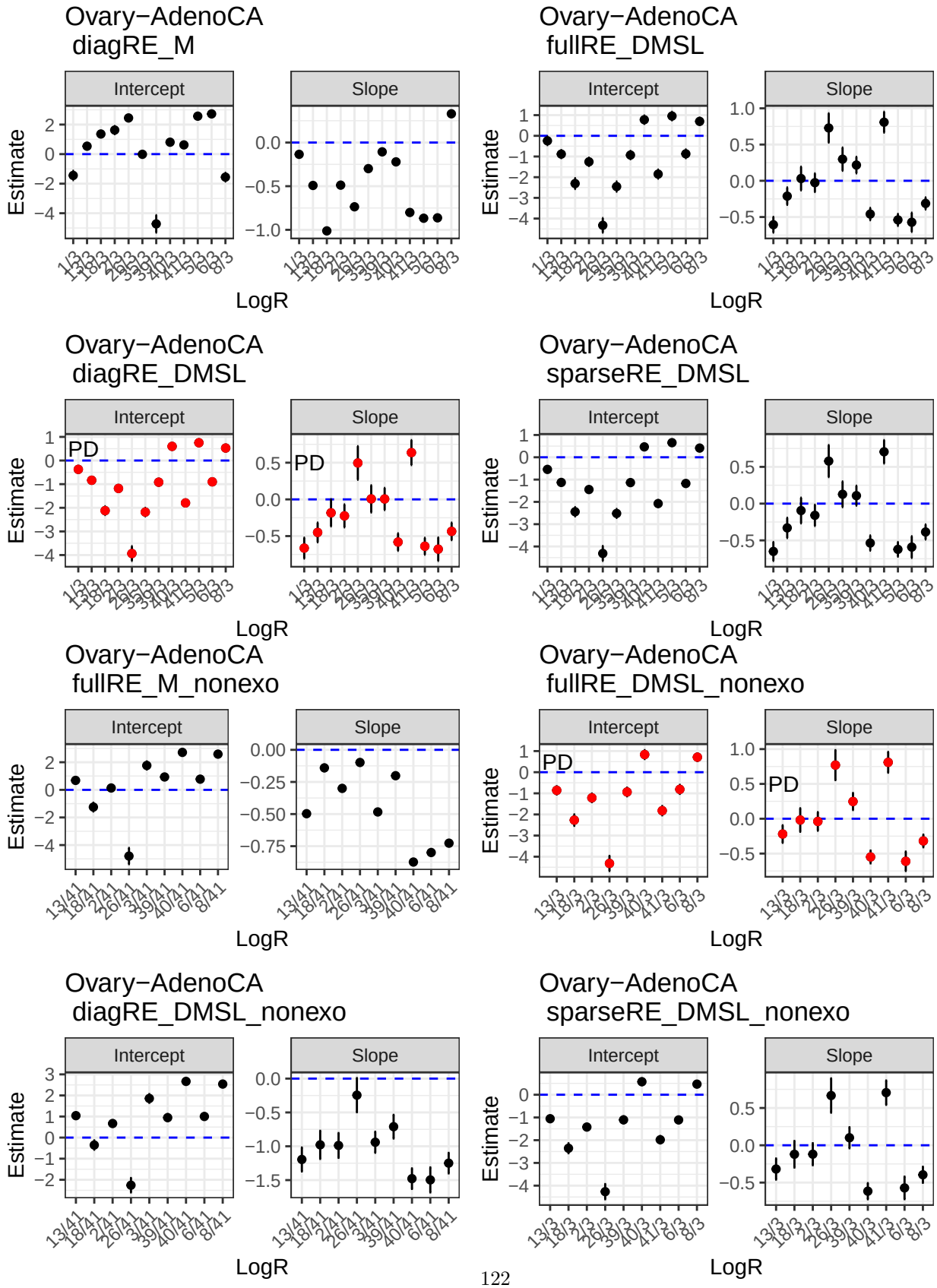
```
colSums(obj_Ovary_AdenoCA$Y == 0)/nrow(obj_Ovary_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5       SBS6       SBS8      SBS13
## 0.02061856 0.04639175 0.15463918 0.04639175 0.11855670 0.01546392 0.04123711
##      SBS18      SBS26      SBS35      SBS39      SBS40      SBS41
## 0.36597938 0.64432990 0.35567010 0.16494845 0.07216495 0.19587629
```

```
colSums(obj_Ovary_AdenoCA$Y)/sum(obj_Ovary_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5       SBS6       SBS8      SBS13
## 0.04443406 0.01997521 0.18513114 0.18242339 0.02948011 0.16654912 0.03420887
##      SBS18      SBS26      SBS35      SBS39      SBS40      SBS41
## 0.01553666 0.03041662 0.01913467 0.06447749 0.17833631 0.02989634
```

Betas

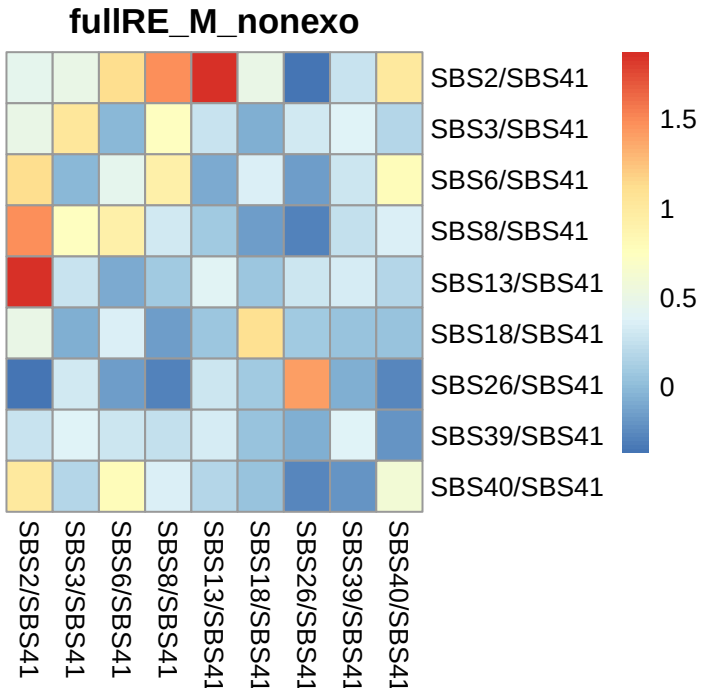

Ovary−AdenoCA
diagRE_M
Ovary−AdenoCA
fullRE_DMSL
Ovary−AdenoCA
diagRE_DMSL
Ovary−AdenoCA
sparseRE_DMSL
Ovary−AdenoCA
fullRE_M_nonexo
Ovary−AdenoCA
fullRE_DMSL_nonexo
Ovary−AdenoCA
diagRE_DMSL_nonexo
Ovary−AdenoCA
sparseRE_DMSL_nonexo
Intercept
Slope
Estimate
LogR
PD

We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of $2.6852565 \times 10^{-28}$.

Covariance matrices

Keep in mind that fullRE DMSL nonexo has not converged.

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

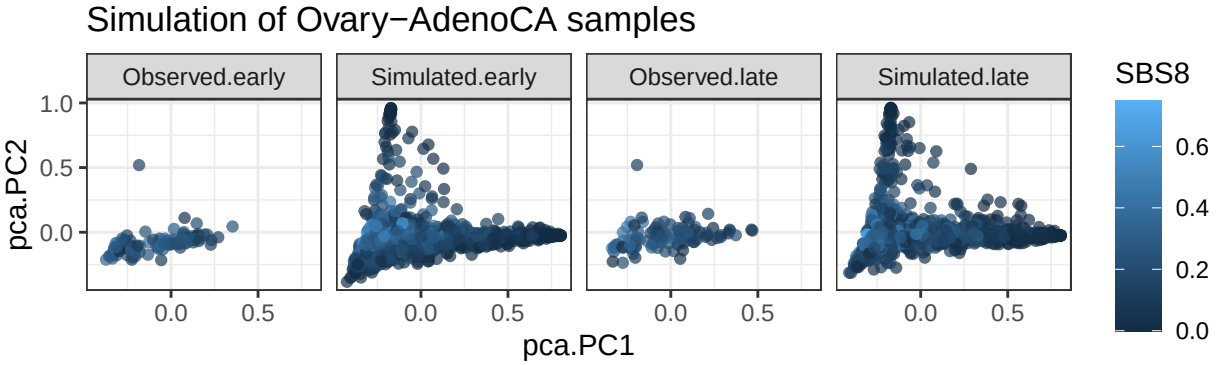


Simulation under inferred data

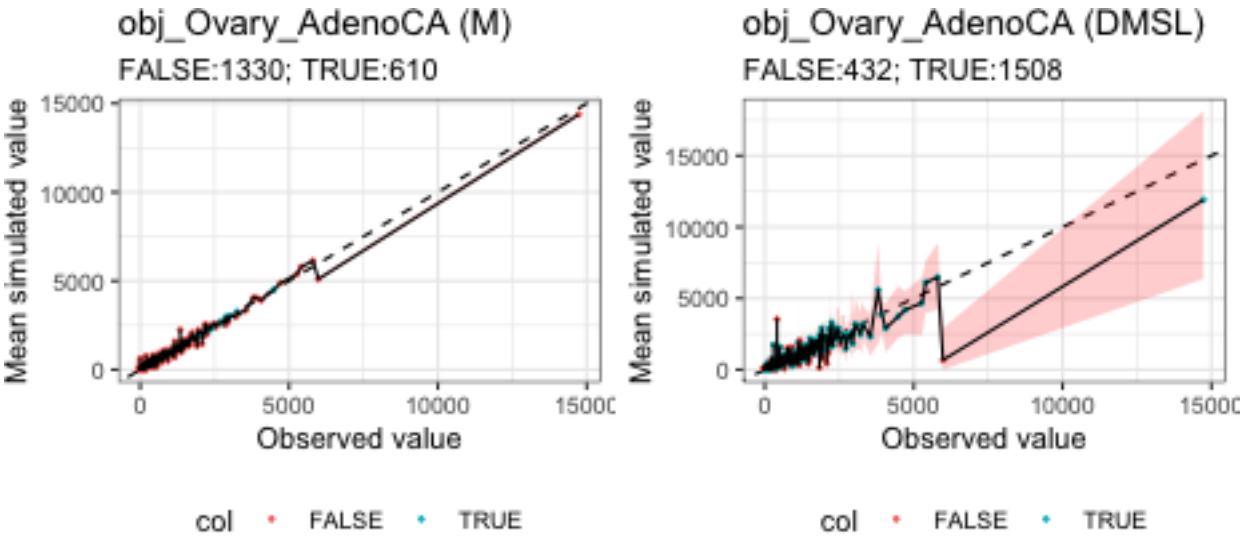
Using diagRE DMSL nonexo.

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



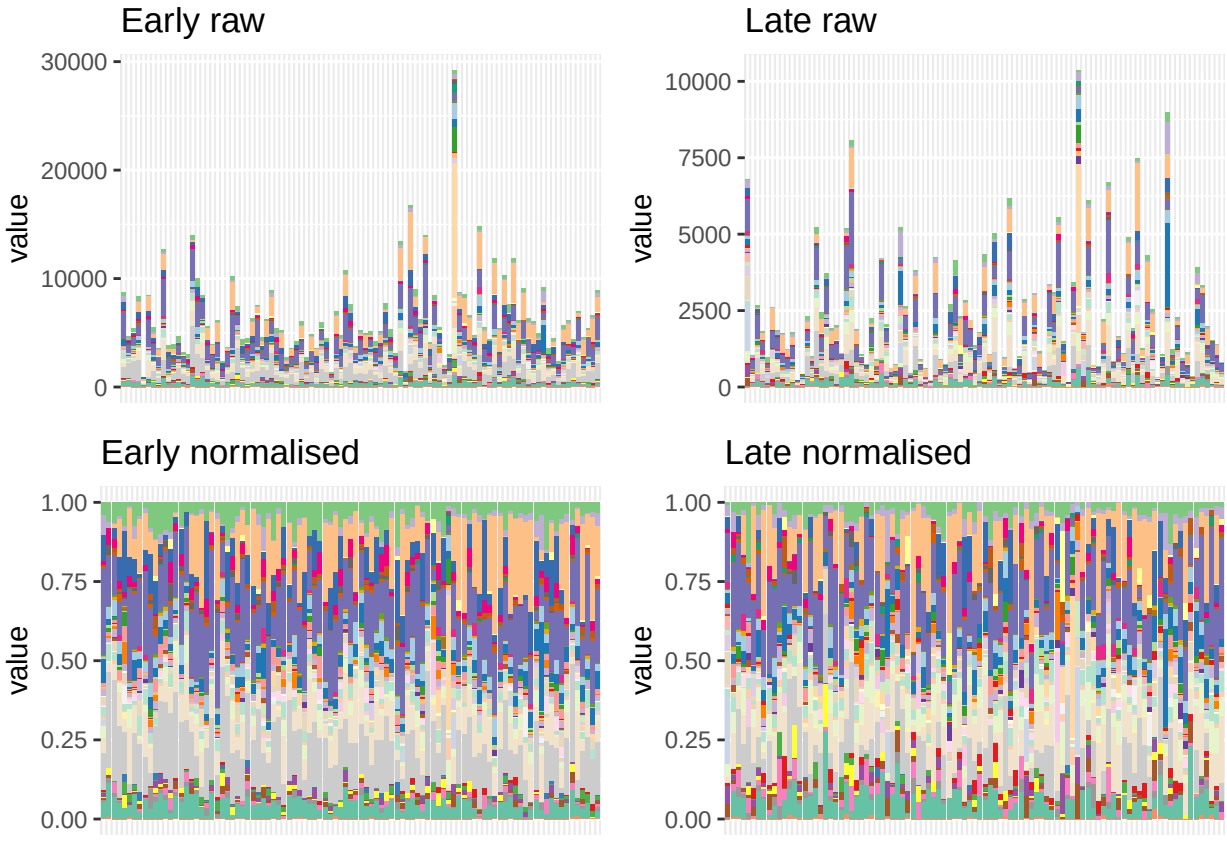
Ranked plot for coverage



Signatures from mutSigExtractor

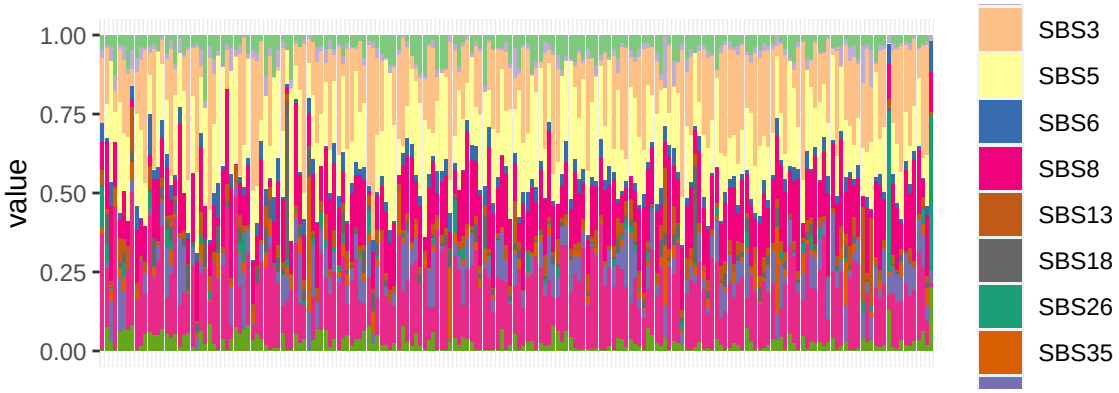
The signatures from mutSigExtractor are as follows:

```
## [1] 97
```



Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the

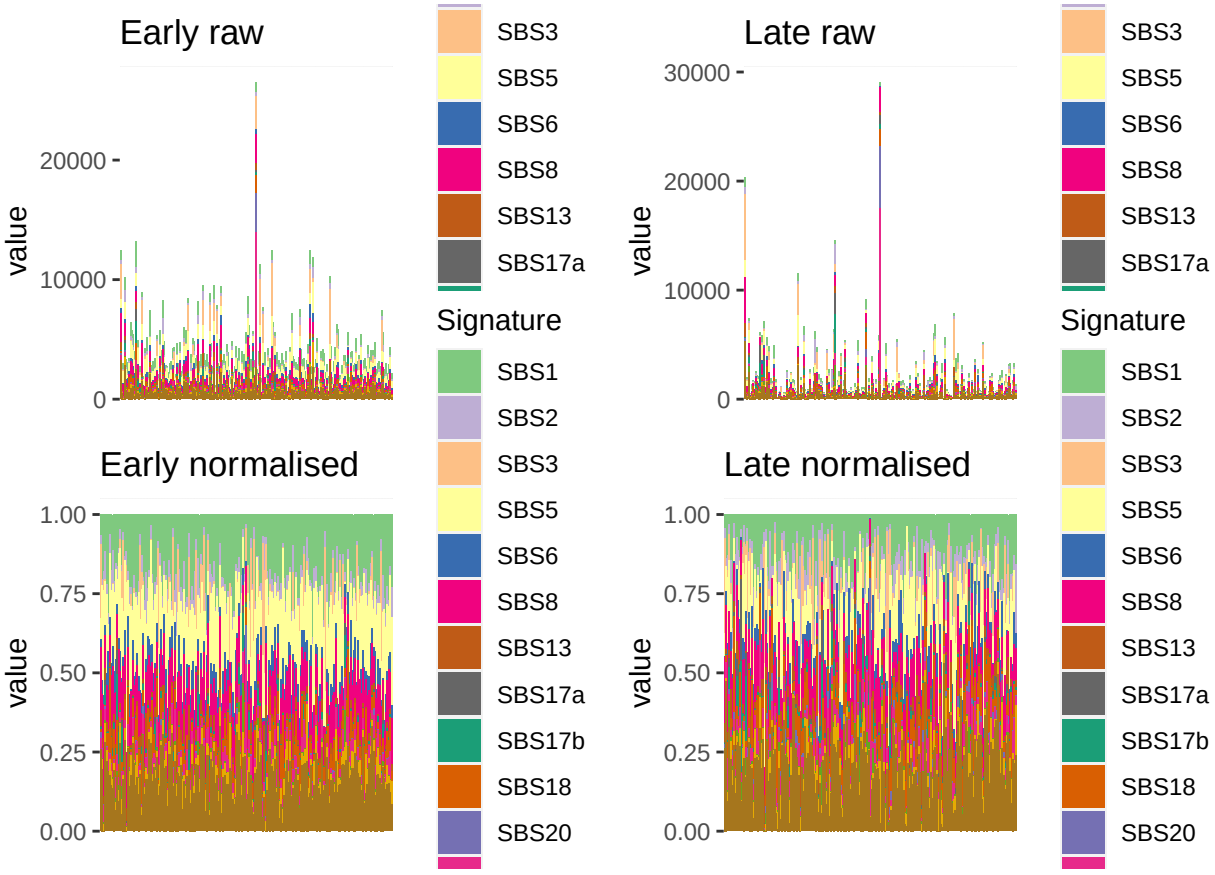
number of mutations.



Panc-AdenoCA

Barplot and general statistics

```
## [1] 193
```



The number of samples and signatures is:

```
## [1] 386  15
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS6"   "SBS8"   "SBS13"  "SBS17a"
##  [9] "SBS17b" "SBS18"  "SBS20"  "SBS26"  "SBS28"  "SBS30"  "SBS40"
```

Convergence table

These are the results for the convergence of models fits. Most runs have converged. fullRE_DMSL_nonexo hadn't run.

```
##           value                                L2                     L1
## 1  Panc-AdenoCA    hessian_positivedefinite_bool                diagRE_M
## 2  Panc-AdenoCA hessian_nonpositivedefinite_bool                fullRE_M
## 3  Panc-AdenoCA hessian_nonpositivedefinite_bool              diagRE_DMDL
## 4  Panc-AdenoCA                          Timeout             fullRE_halfDM
## 5  Panc-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMDL
## 6  Panc-AdenoCA    hessian_positivedefinite_bool              diagRE_DMSL
## 7  Panc-AdenoCA    hessian_positivedefinite_bool             sparseRE_DMSL
## 8  Panc-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  Panc-AdenoCA hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Panc-AdenoCA    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Panc-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Panc-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Panc-AdenoCA                          Timeout       fullRE_DMSL_nonexo
## 14 Panc-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Panc-AdenoCA    hessian_positivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo. M hasn't converged.

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Potentially problematic signatures

We explore whether there are problematic signatures. SBS17a is potentially problematic.

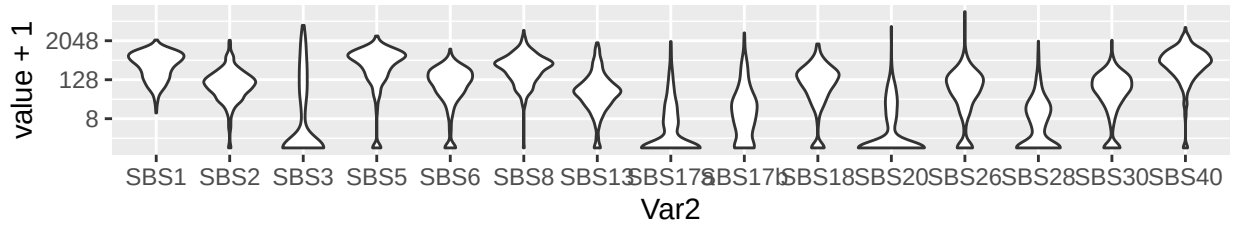
```
colSums(obj_Panc_AdenoCA$Y == 0)/nrow(obj_Panc_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5        SBS6        SBS8
## 0.000000000 0.012953368 0.634715026 0.051813472 0.069948187 0.005181347
##       SBS13      SBS17a      SBS17b       SBS18       SBS20       SBS26
## 0.036269430 0.575129534 0.183937824 0.093264249 0.647668394 0.134715026
##       SBS28       SBS30       SBS40
## 0.409326425 0.124352332 0.023316062
```

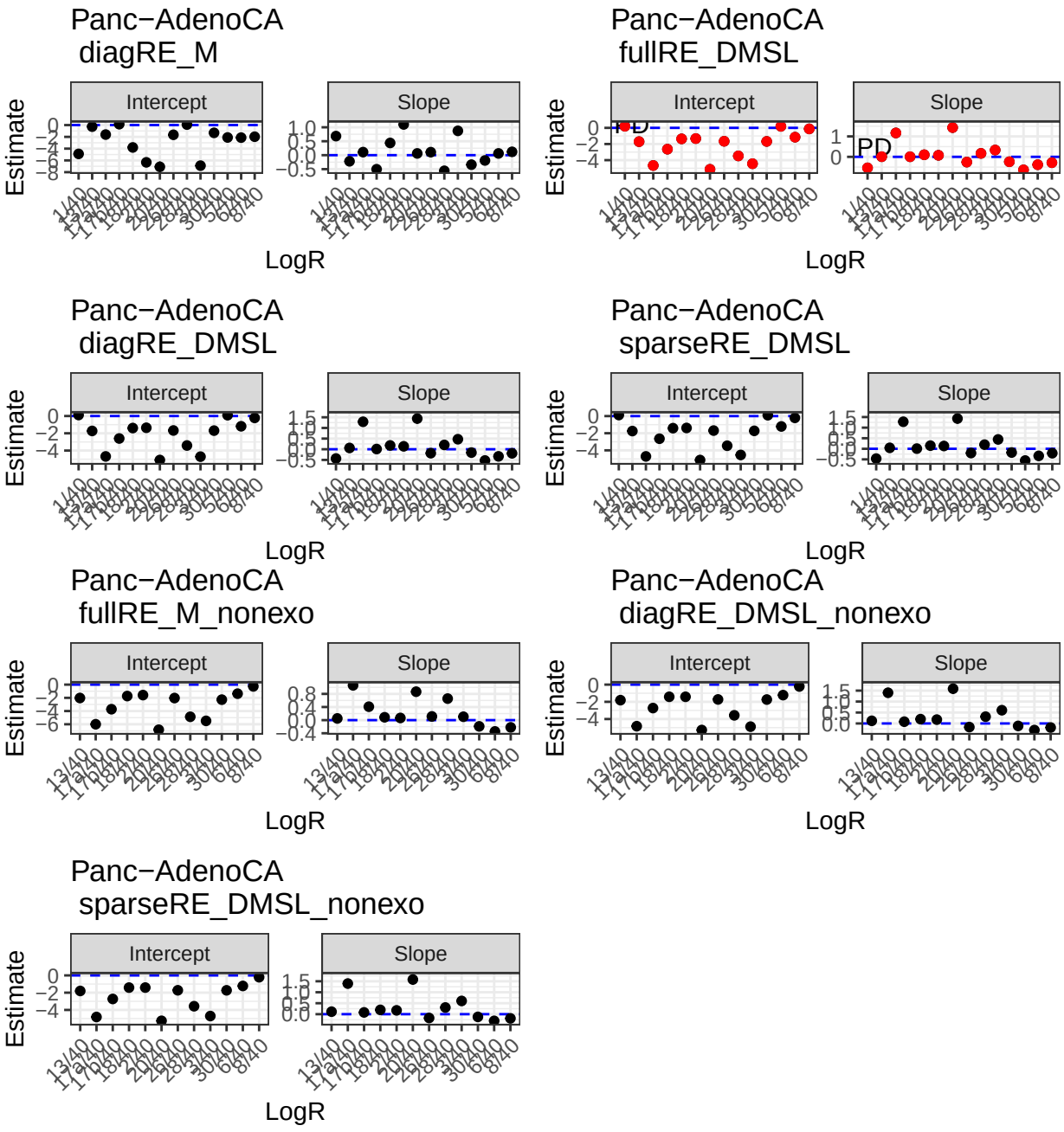
```
colSums(obj_Panc_AdenoCA$Y)/sum(obj_Panc_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5        SBS6        SBS8
## 0.146510755 0.044043854 0.068370611 0.164694624 0.043276473 0.126954041
##       SBS13      SBS17a      SBS17b       SBS18       SBS20       SBS26
## 0.035397768 0.009365890 0.021818955 0.052439007 0.011446261 0.054377412
##       SBS28       SBS30       SBS40
## 0.008232467 0.029313201 0.183758680
```

From the violin plot, none seem too problematic.



Betas

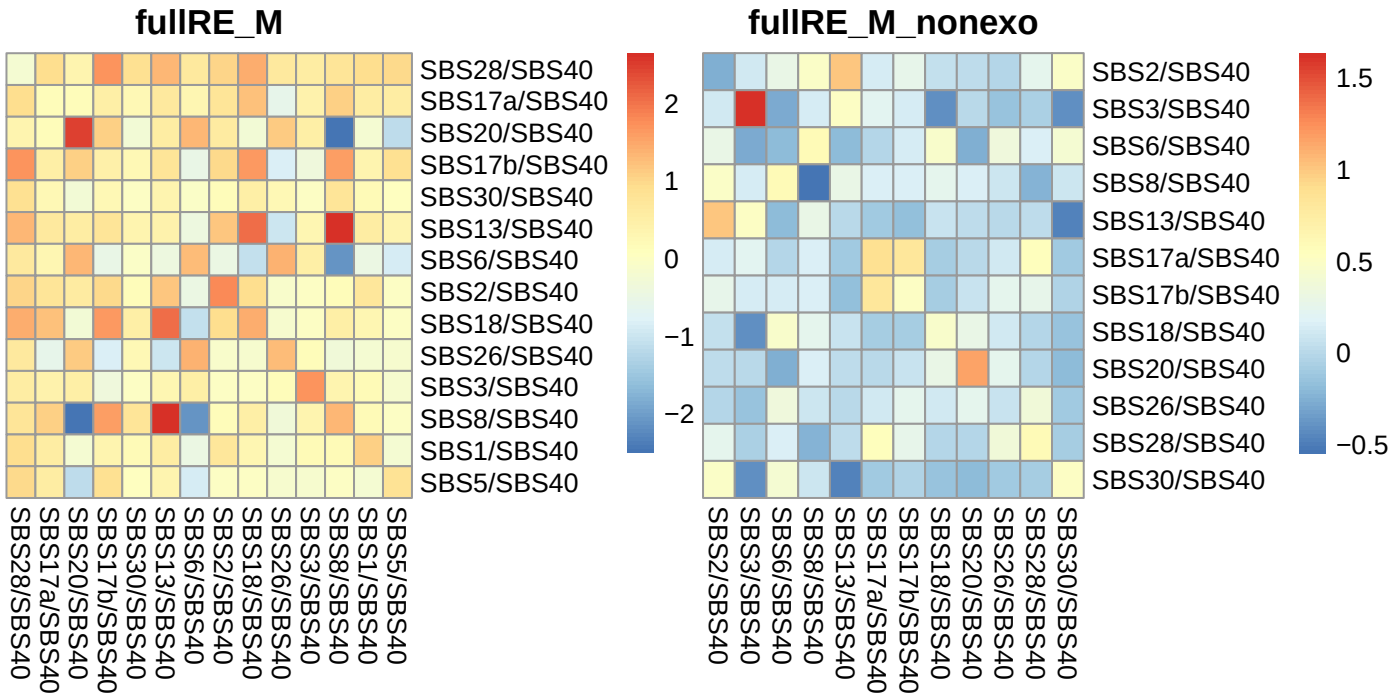


We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of

$6.8710408 \times 10^{-46}$.

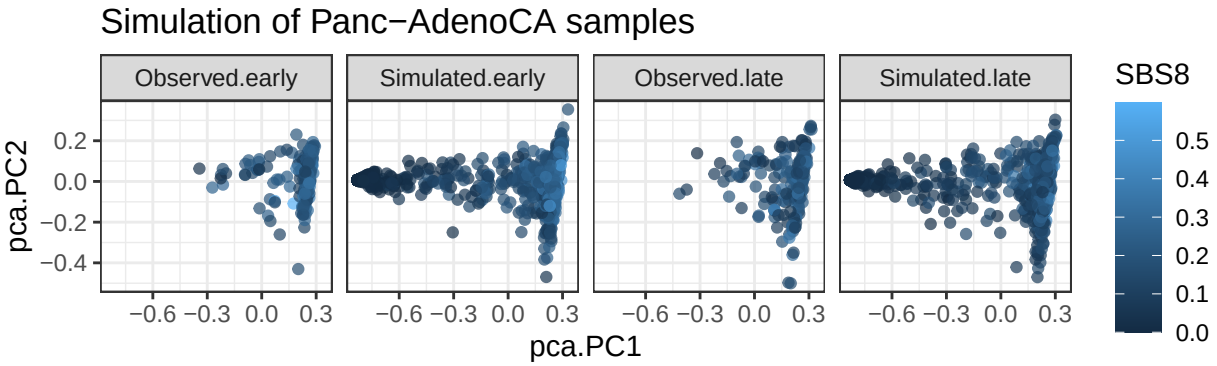
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```



Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

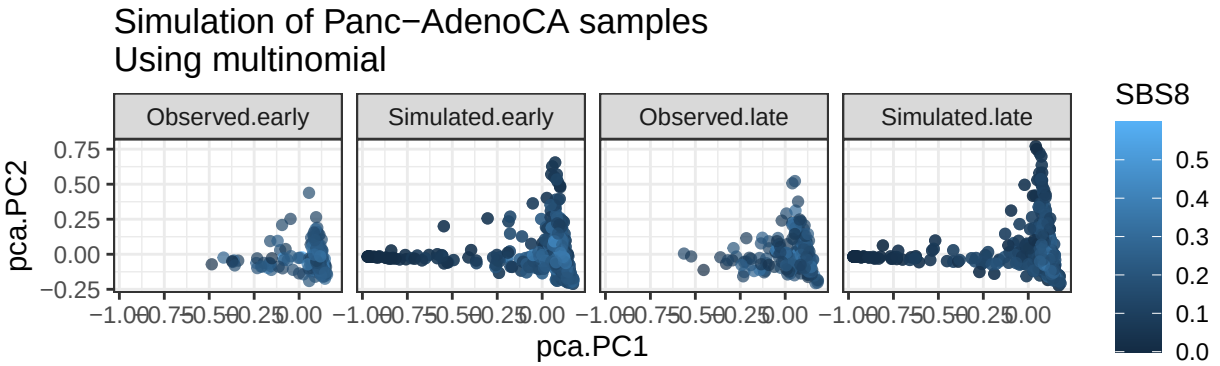


Multinomial:

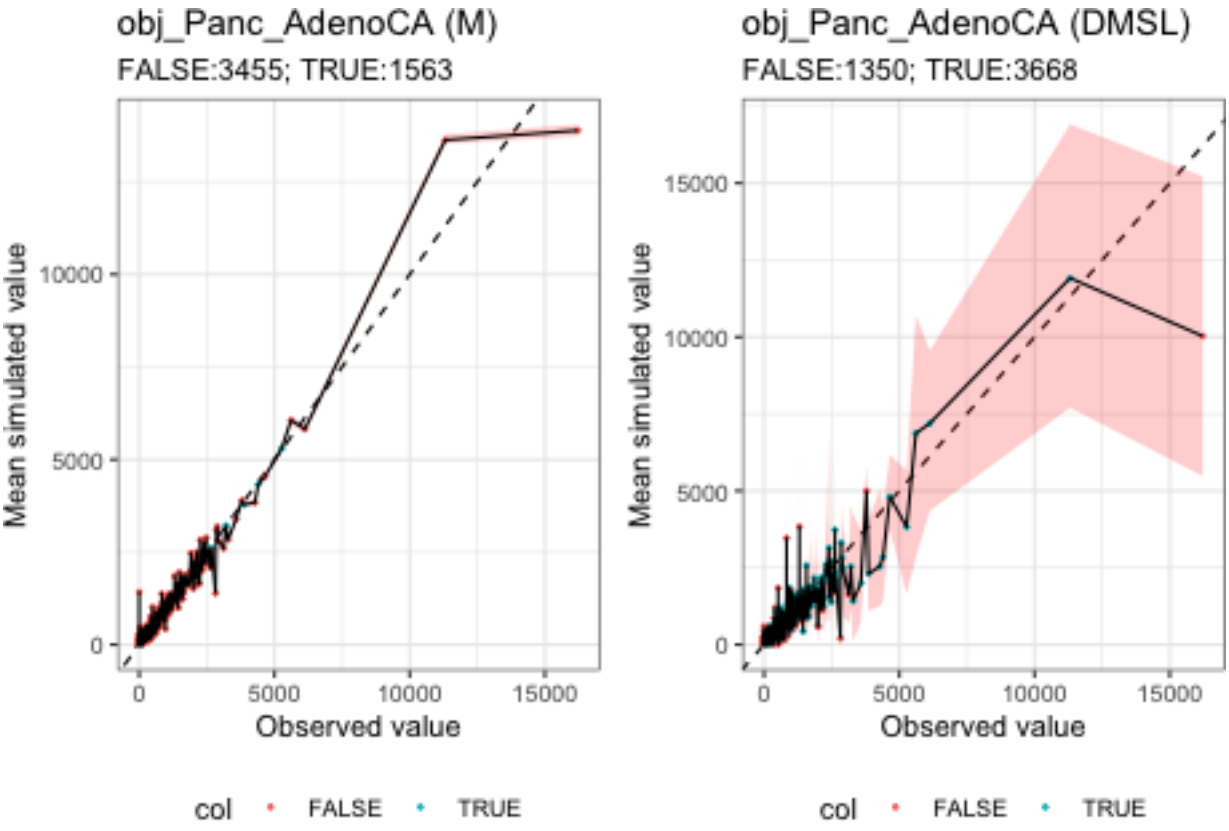
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in mvtnorm::rmvnorm(n = n_sim, mean = rep(0, dmin1), sigma = cov_mat):
## sigma is numerically not positive semidefinite
```



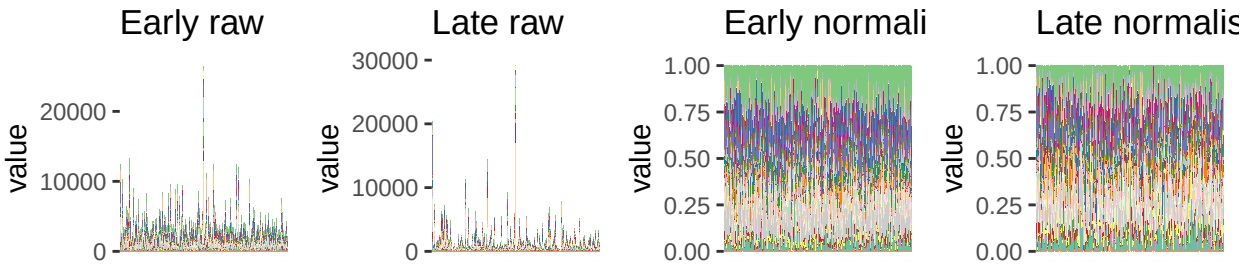
Ranked plot for coverage



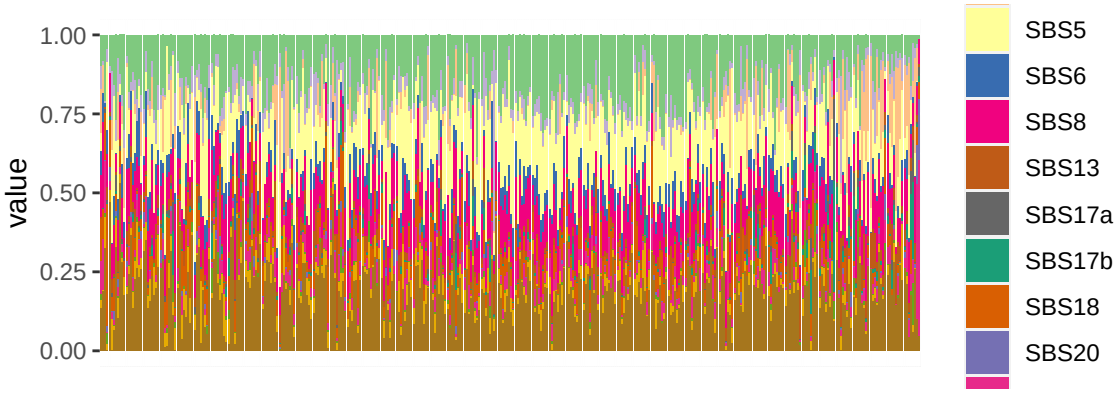
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 193
```



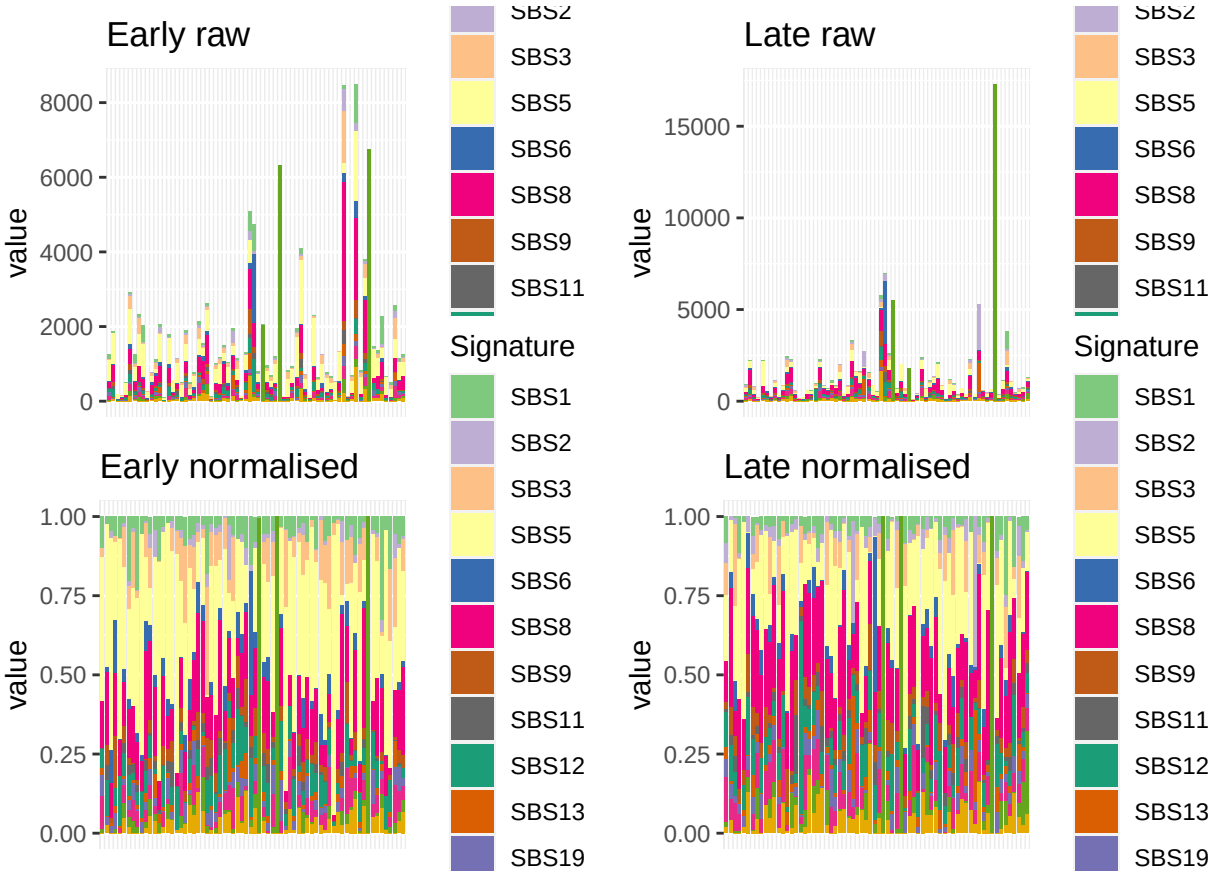
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Panc-Endocrine

Barplot and general statistics

```
## [1] 70
```



The number of samples and signatures is:

```
## [1] 140  14
```

The signatures are:

```
##  [1] "SBS1"  "SBS2"  "SBS3"  "SBS5"  "SBS6"  "SBS8"  "SBS9"  "SBS11" "SBS12"
## [10] "SBS13" "SBS19" "SBS30" "SBS36" "SBS39"
```

Convergence table

These are the results for the convergence of models fits. fullRE_DMSL and fullRE_DMSL_nonexo haven't.

```
##             value                                L2                   L1
## 1  Panc-Endocrine    hessian_positivedefinite_bool          diagRE_M
## 2  Panc-Endocrine hessian_nonpositivedefinite_bool          fullRE_M
## 3  Panc-Endocrine hessian_nonpositivedefinite_bool       diagRE_DMDL
## 4  Panc-Endocrine                          Timeout     fullRE_halfDM
## 5  Panc-Endocrine hessian_nonpositivedefinite_bool       fullRE_DMDL
## 6  Panc-Endocrine    hessian_positivedefinite_bool       diagRE_DMSL
## 7  Panc-Endocrine    hessian_positivedefinite_bool     sparseRE_DMSL
```

```
## 8  Panc-Endocrine hessian_nonpositivedefinite_bool             fullRE_DMSL
## 9  Panc-Endocrine hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Panc-Endocrine    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Panc-Endocrine    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Panc-Endocrine    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Panc-Endocrine hessian_nonpositivedefinite_bool       fullRE_DMSL_nonexo
## 14 Panc-Endocrine hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Panc-Endocrine                          Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo doesn't converge:

```
## [1] FALSE
```

Potentially problematic signatures

We explore whether there are problematic signatures:

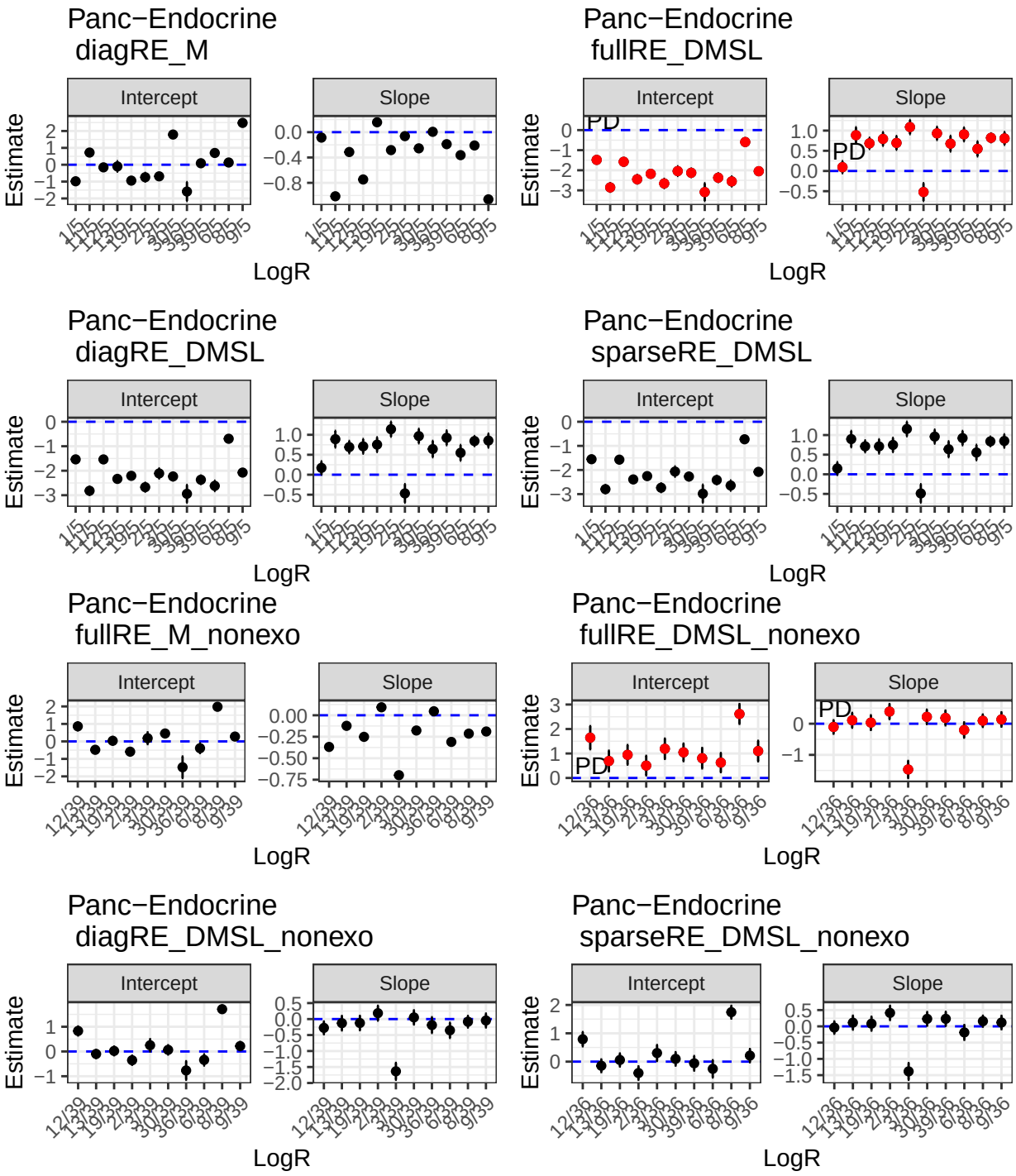
```
colSums(obj_Panc_Endocrine$Y == 0)/nrow(obj_Panc_Endocrine$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5       SBS6       SBS8       SBS9
## 0.08571429 0.20714286 0.43571429 0.09285714 0.35714286 0.05000000 0.14285714
##      SBS11      SBS12      SBS13      SBS19      SBS30      SBS36      SBS39
## 0.32142857 0.12142857 0.13571429 0.18571429 0.25000000 0.35000000 0.26428571
```

```
colSums(obj_Panc_Endocrine$Y)/sum(obj_Panc_Endocrine$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5       SBS6       SBS8       SBS9
## 0.04497473 0.03735807 0.05104230 0.19581878 0.04501764 0.17414887 0.03565881
##      SBS11      SBS12      SBS13      SBS19      SBS30      SBS36      SBS39
## 0.01924975 0.05892071 0.03124330 0.02551471 0.04475588 0.19146763 0.04482883
```

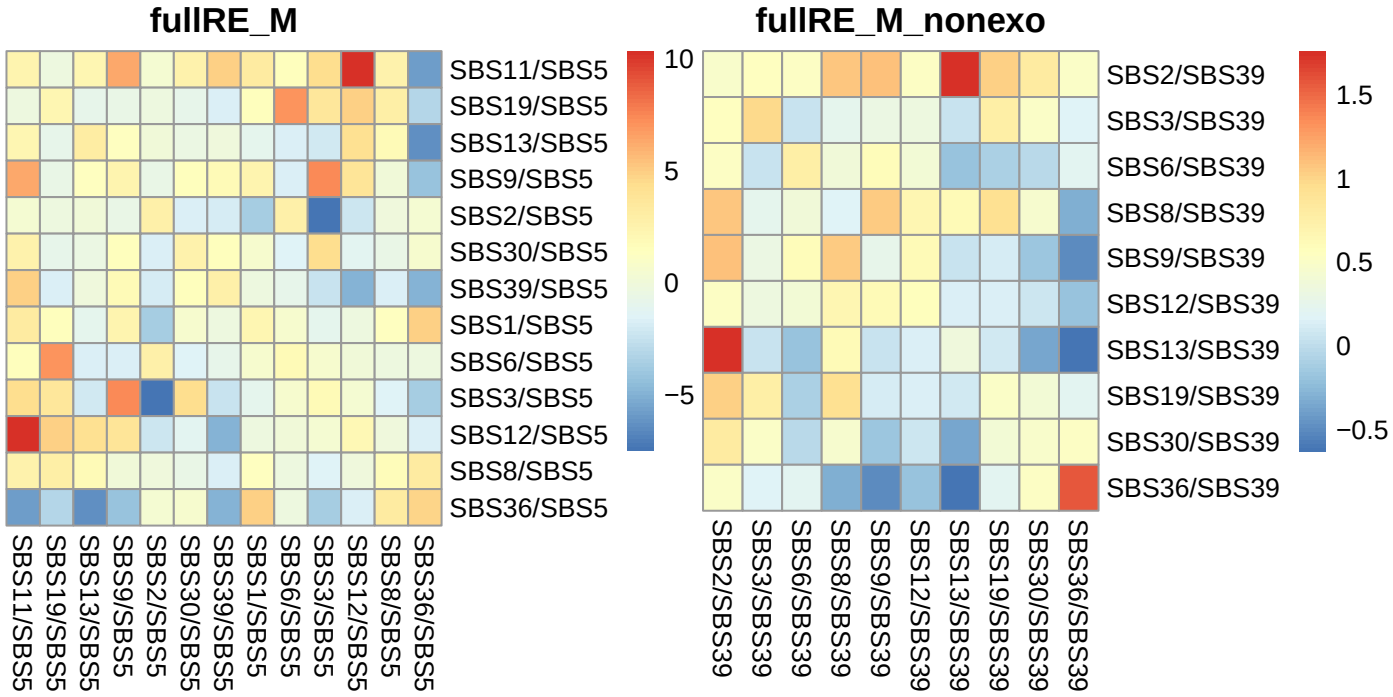
Betas



We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 1.9645842×10^{-9}.

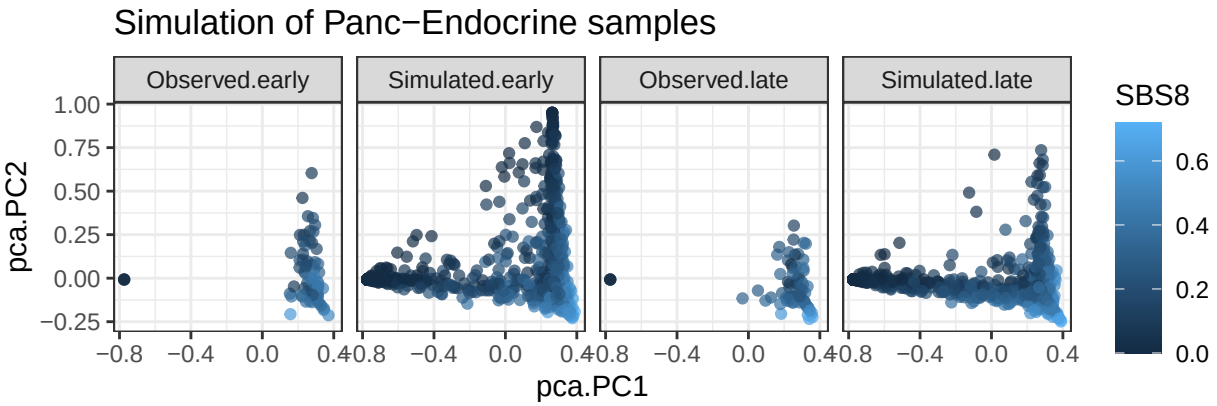
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

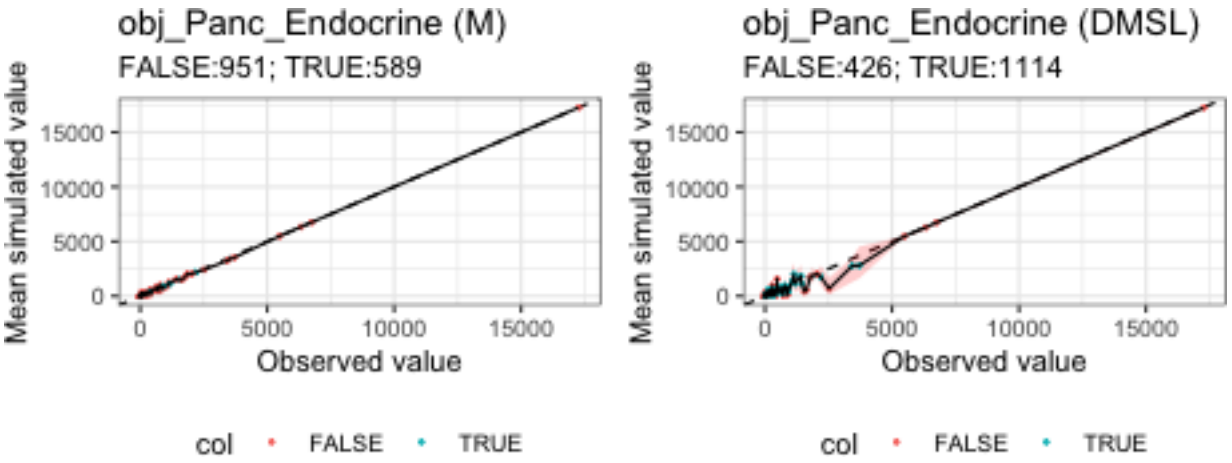


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



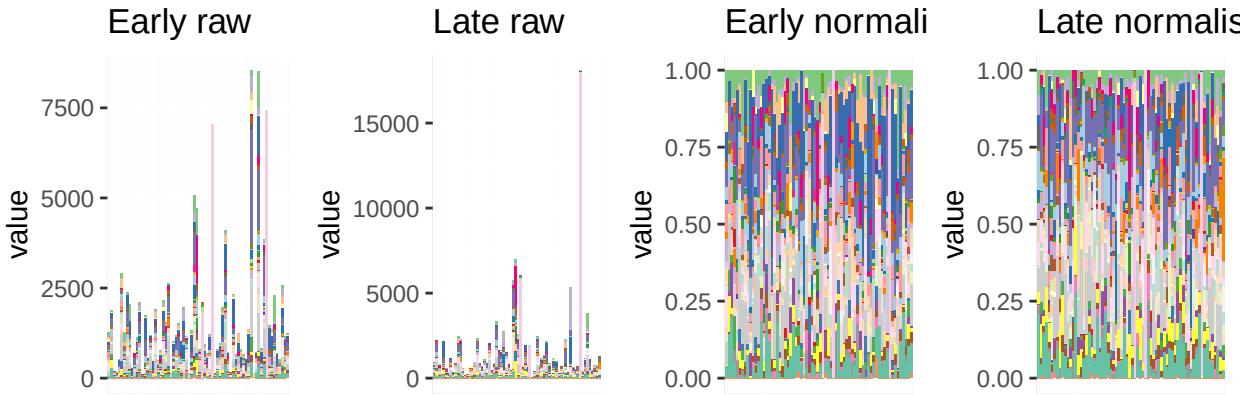
Ranked plot for coverage



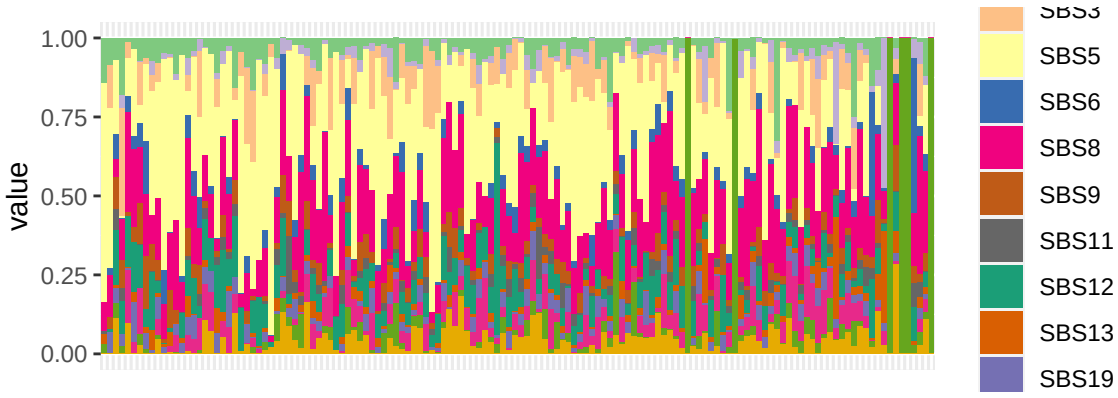
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 70
```



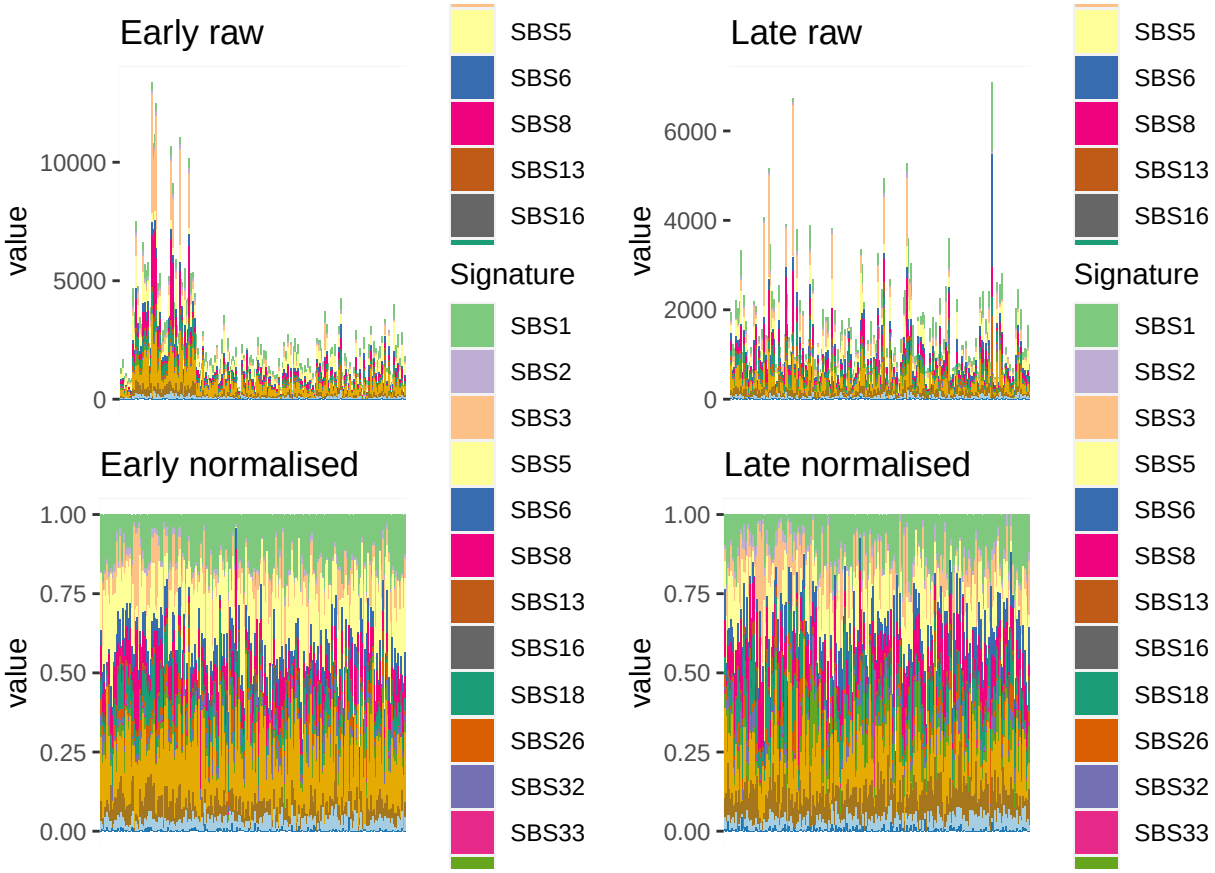
Exposures sorted by increasing number of mutations: there is a trend with one signature only being present, and in very large amounts, in hyeprmutated samples.



Prost-AdenoCA

Barplot and general statistics

```
## [1] 208
```



The number of samples and signatures is:

```
## [1] 416  17
```

The signatures are:

```
##  [1] "SBS1"  "SBS2"  "SBS3"  "SBS5"  "SBS6"  "SBS8"  "SBS13" "SBS16" "SBS18"
## [10] "SBS26" "SBS32" "SBS33" "SBS37" "SBS40" "SBS41" "SBS50" "SBS52"
```

Convergence table

These are the results for the convergence of models fits. Most have converged. fullRE_DMSL_nonexo hasn't run and needs to be re-run.

```
##            value                                L2                  L1
## 1  Prost-AdenoCA    hessian_positivedefinite_bool          diagRE_M
## 2  Prost-AdenoCA hessian_nonpositivedefinite_bool          fullRE_M
## 3  Prost-AdenoCA hessian_nonpositivedefinite_bool        diagRE_DMDL
## 4  Prost-AdenoCA                           Timeout       fullRE_halfDM
## 5  Prost-AdenoCA                           Timeout        fullRE_DMDL
## 6  Prost-AdenoCA    hessian_positivedefinite_bool        diagRE_DMSL
```

```
## 7  Prost-AdenoCA    hessian_positivedefinite_bool            sparseRE_DMSL
## 8  Prost-AdenoCA hessian_nonpositivedefinite_bool              fullRE_DMSL
## 9  Prost-AdenoCA hessian_nonpositivedefinite_bool         fullRE_DMSL_SBS1
## 10 Prost-AdenoCA    hessian_positivedefinite_bool          fullRE_M_nonexo
## 11 Prost-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Prost-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Prost-AdenoCA                          Timeout       fullRE_DMSL_nonexo
## 14 Prost-AdenoCA    hessian_positivedefinite_bool       fullRE_DMDL_nonexo
## 15 Prost-AdenoCA hessian_nonpositivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

But DMSL hasn't:

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

```
## [1] FALSE
```

Potentially problematic signatures

We explore whether there are problematic signatures. None seem to be, although SBS33 is absent in 60% of samples.

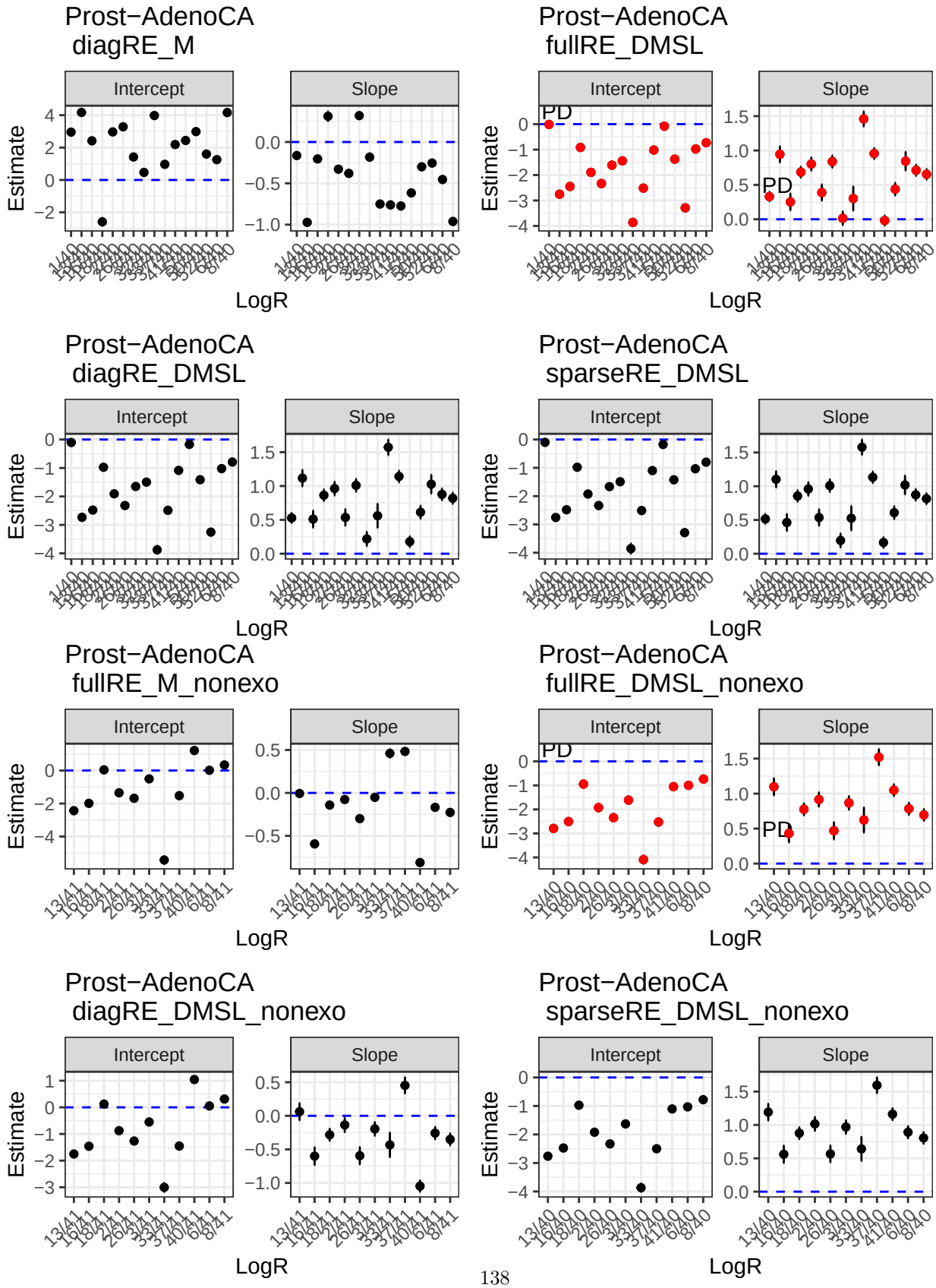
```
colSums(obj_Prost_AdenoCA$Y == 0)/nrow(obj_Prost_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5        SBS6        SBS8
## 0.007211538 0.040865385 0.225961538 0.108173077 0.033653846 0.052884615
##       SBS13       SBS16       SBS18       SBS26       SBS32       SBS33
## 0.259615385 0.331730769 0.043269231 0.317307692 0.100961538 0.665865385
##       SBS37       SBS40       SBS41       SBS50       SBS52
## 0.257211538 0.086538462 0.026442308 0.057692308 0.427884615
```

```
colSums(obj_Prost_AdenoCA$Y)/sum(obj_Prost_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5        SBS6        SBS8
## 0.123798623 0.018138349 0.101489206 0.145492714 0.063352195 0.110930563
##       SBS13       SBS16       SBS18       SBS26       SBS32       SBS33
## 0.011721046 0.011380527 0.071927318 0.022167235 0.022482663 0.003387275
##       SBS37       SBS40       SBS41       SBS50       SBS52
## 0.028615602 0.166680208 0.065069131 0.029925110 0.003442236
```

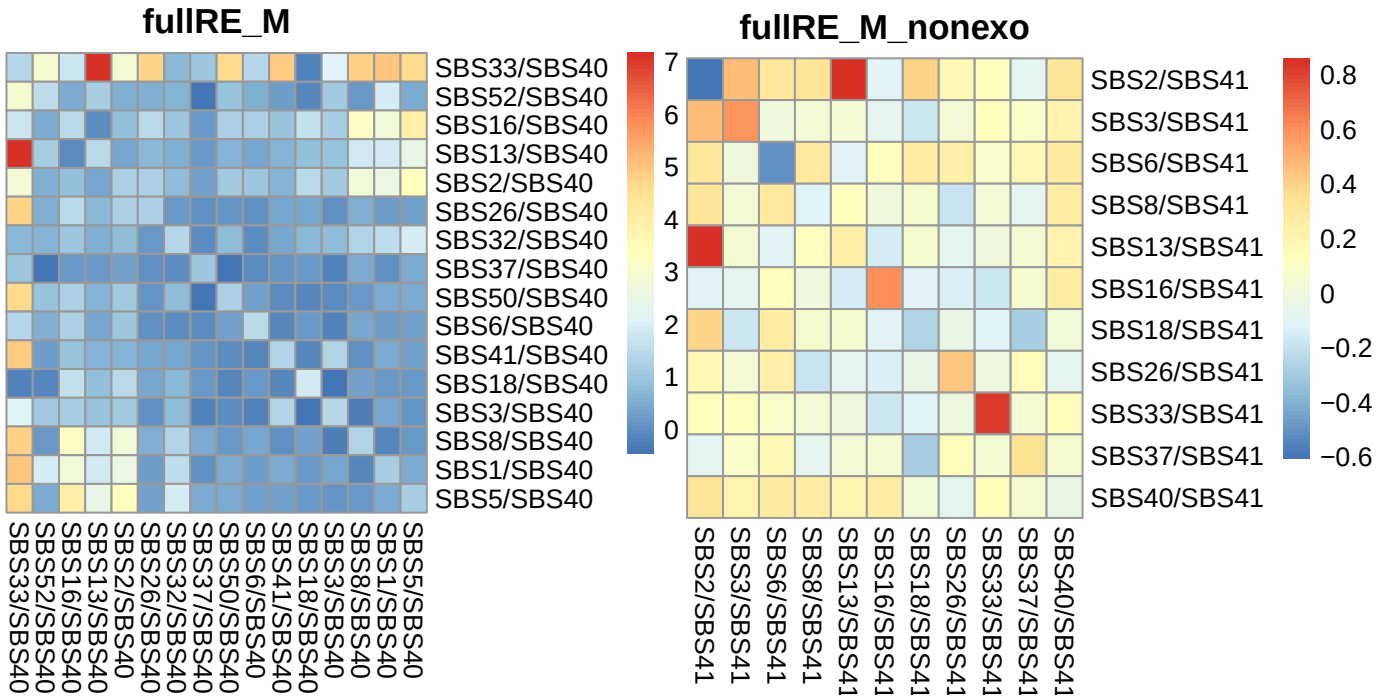
Betas


Prost−AdenoCA
diagRE_M
Intercept
Slope
Estimate
LogR
Prost−AdenoCA
fullRE_DMSL
PD
Prost−AdenoCA
diagRE_DMSL
Prost−AdenoCA
sparseRE_DMSL
Prost−AdenoCA
fullRE_M_nonexo
Prost−AdenoCA
fullRE_DMSL_nonexo
Prost−AdenoCA
diagRE_DMSL_nonexo
Prost−AdenoCA
sparseRE_DMSL_nonexo

We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 5.142907×10^{-58}.

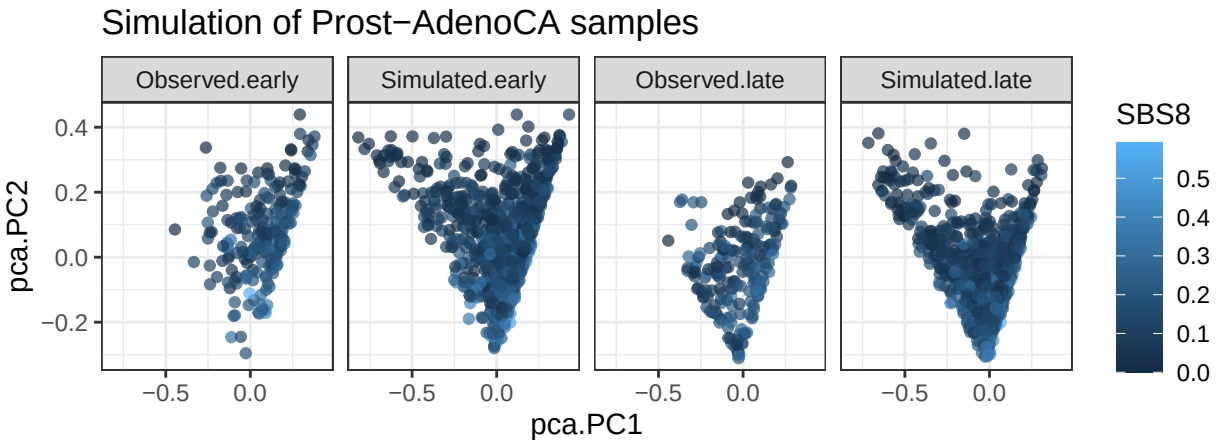
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

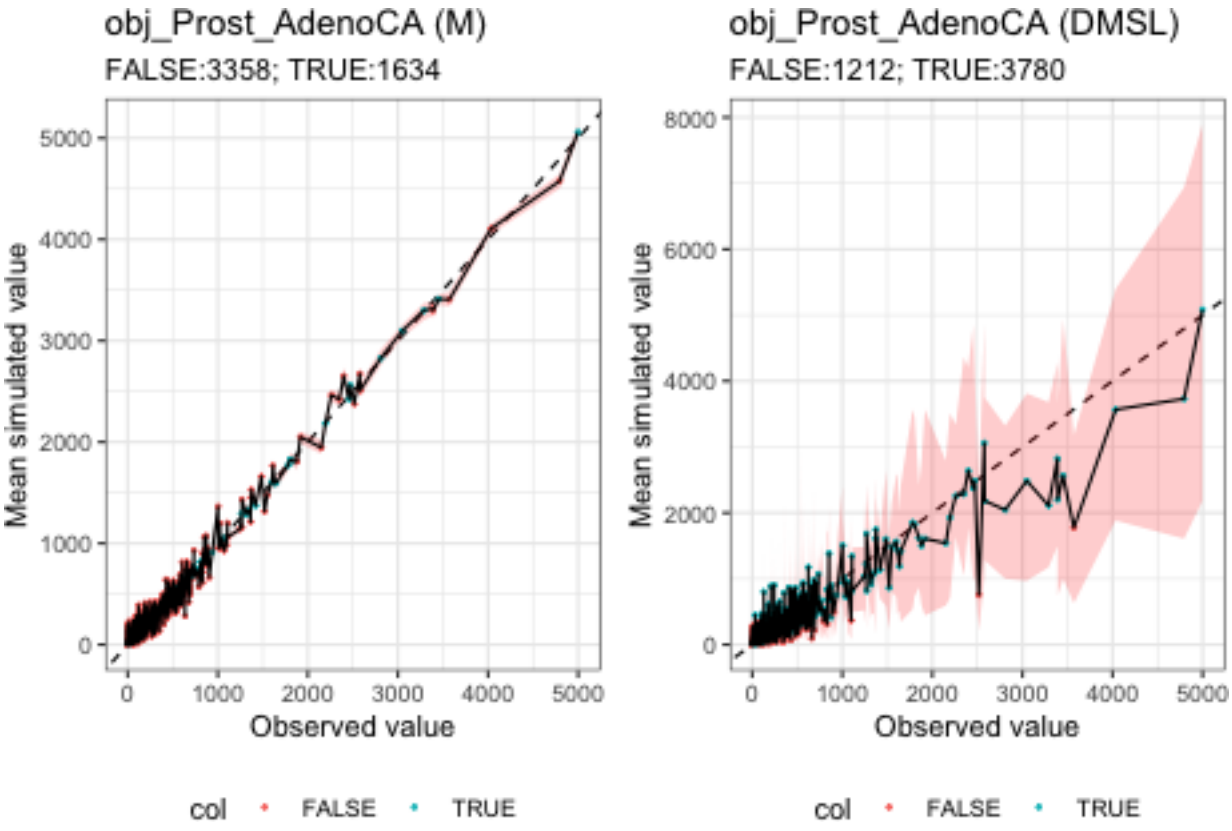


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



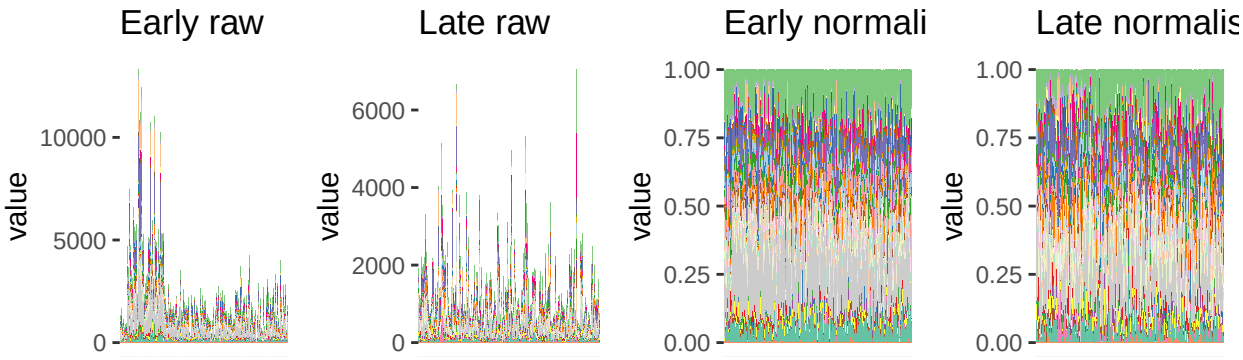
Ranked plot for coverage



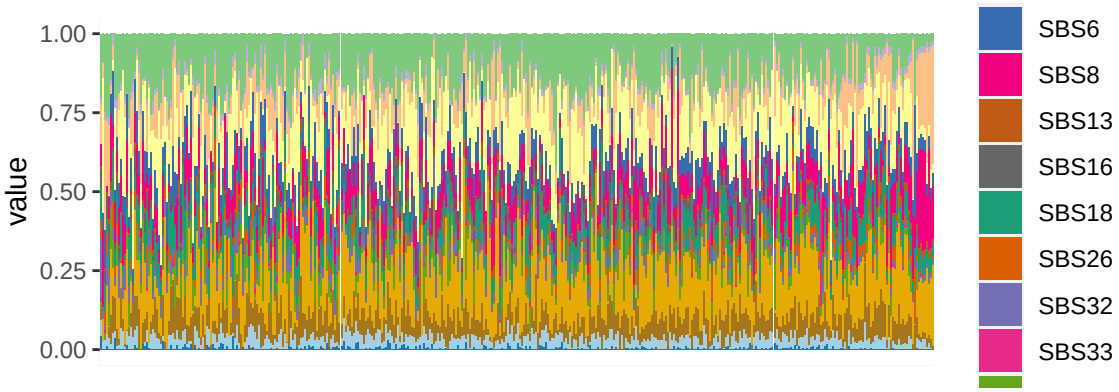
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 208
```



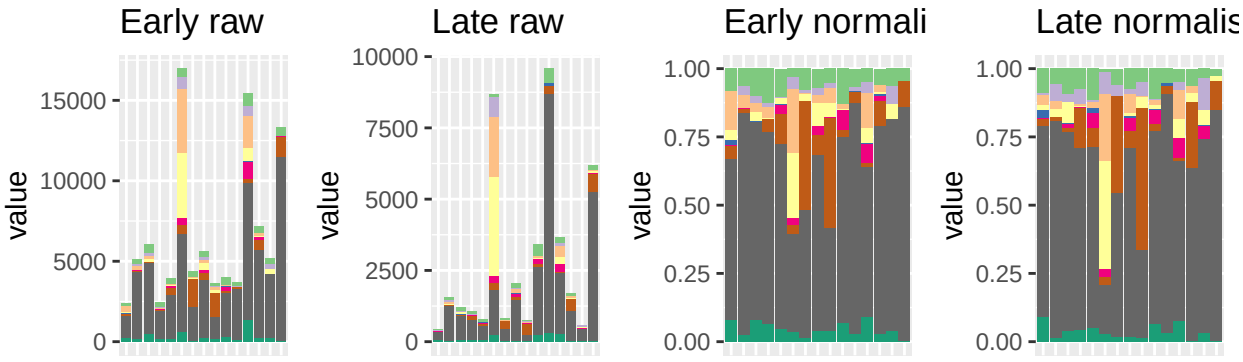
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations, except perhaps SBS3 in the hypermutated ones.



Skin-Melanoma.acral

Barplot and general statistics

```
## [1] 15
```



The number of samples and signatures is:

```
## [1] 30  9
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS7a" "SBS7b" "SBS7c" "SBS31" "SBS38" "SBS40" "SBS58"
```

Convergence table

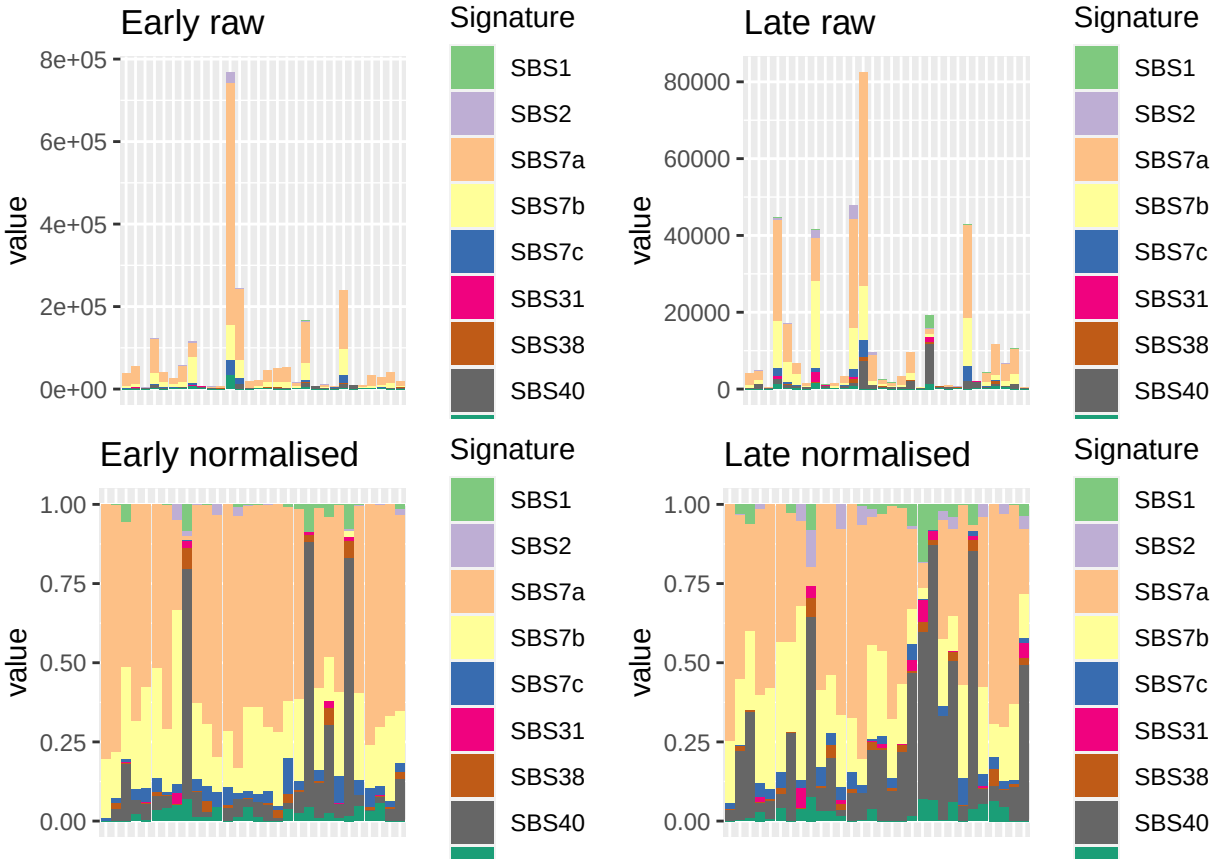
These are the results for the convergence of models fits. They have converged even though very clearly we have very few observations and too many parameters. I thought I would have excluded this cancer type? In CT_sufficient_samples.txt it does appear but shouldn't - I don't continue the analyses for this cancer type.

```
## [1] value L2    L1
## <0 rows> (or 0-length row.names)
```

Skin-Melanoma.cutaneous

Barplot and general statistics

```
## [1] 30
```



The number of samples and signatures is:

```
## [1] 60  9
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS7a" "SBS7b" "SBS7c" "SBS31" "SBS38" "SBS40" "SBS58"
```

Convergence table

These are the results for the convergence of models fits. fullRE_DMSL have not converged.

```
##                      value                                 L2
```

```
## 1  Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 2  Skin-Melanoma.cutaneous hessian_nonpositivedefinite_bool
## 3  Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 4  Skin-Melanoma.cutaneous                          Timeout
## 5  Skin-Melanoma.cutaneous hessian_nonpositivedefinite_bool
## 6  Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 7  Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 8  Skin-Melanoma.cutaneous hessian_nonpositivedefinite_bool
## 9  Skin-Melanoma.cutaneous hessian_nonpositivedefinite_bool
## 10 Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 11 Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
## 12 Skin-Melanoma.cutaneous                          Timeout
## 13 Skin-Melanoma.cutaneous                          Timeout
## 14 Skin-Melanoma.cutaneous hessian_nonpositivedefinite_bool
## 15 Skin-Melanoma.cutaneous    hessian_positivedefinite_bool
##                          L1
## 1                 diagRE_M
## 2                 fullRE_M
## 3              diagRE_DMDL
## 4             fullRE_halfDM
## 5              fullRE_DMDL
## 6              diagRE_DMSL
## 7             sparseRE_DMSL
## 8              fullRE_DMSL
## 9          fullRE_DMSL_SBS1
## 10          fullRE_M_nonexo
## 11       diagRE_DMSL_nonexo
## 12     sparseRE_DMSL_nonexo
## 13       fullRE_DMSL_nonexo
## 14       fullRE_DMDL_nonexo
## 15 fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do converge:

```
## [1] TRUE
```

It has converged.

Potentially problematic signatures

We explore whether there are problematic signatures; there are none.

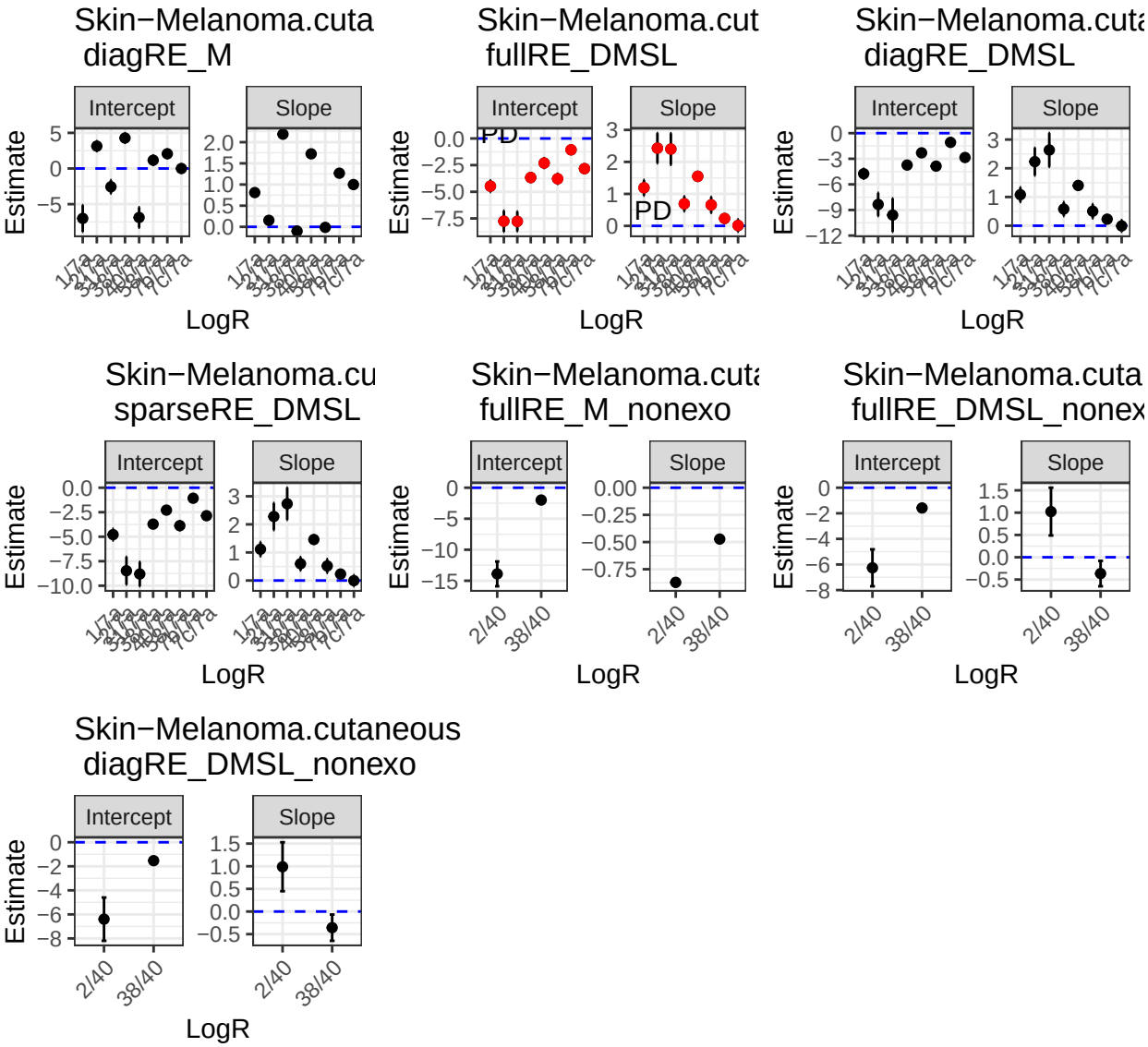
```
colSums(obj_Skin_Melanomacutaneous$Y == 0)/nrow(obj_Skin_Melanomacutaneous$Y)
```

```
##       SBS1       SBS2      SBS7a      SBS7b      SBS7c      SBS31      SBS38
## 0.31666667 0.68333333 0.03333333 0.06666667 0.13333333 0.71666667 0.11666667
##      SBS40      SBS58
## 0.05000000 0.25000000
```

```
colSums(obj_Skin_Melanomacutaneous$Y)/sum(obj_Skin_Melanomacutaneous$Y)
```

```
##         SBS1         SBS2        SBS7a        SBS7b        SBS7c        SBS31
## 0.004433478 0.015369819 0.652197003 0.207691993 0.044244244 0.003864520
##        SBS38        SBS40        SBS58
## 0.005701701 0.042679042 0.023818200
```

Betas

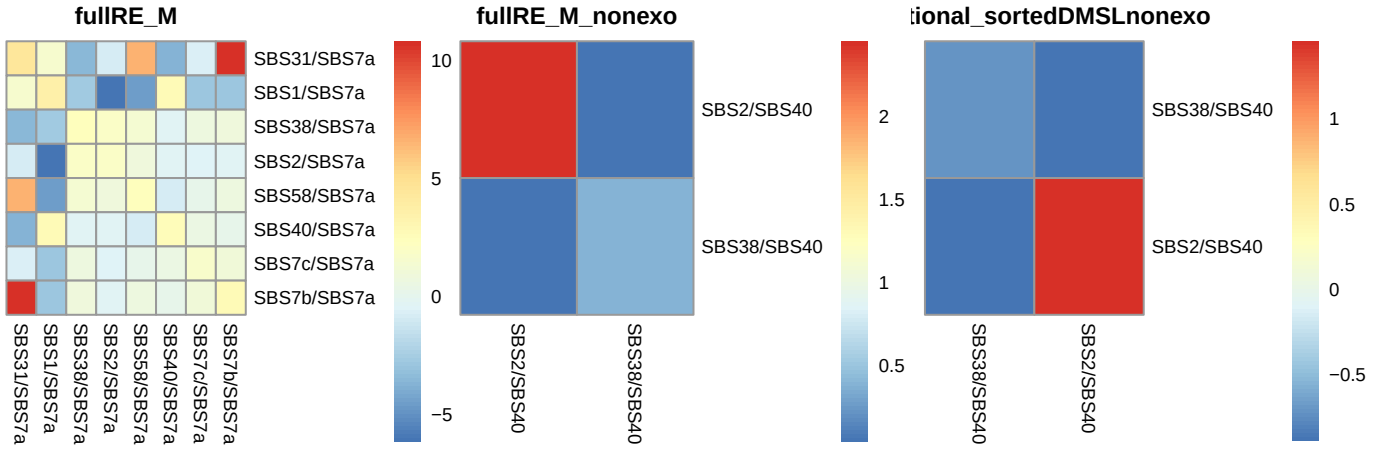


We use the results from the full RE single lambda DM to test for differential abundance, giving a p-value of 0.0628799.

Covariance matrices

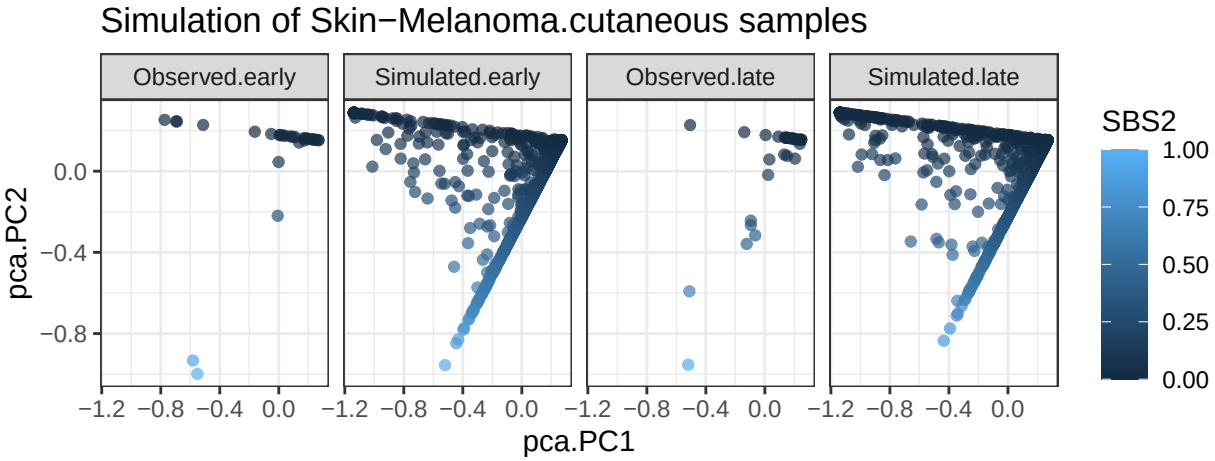
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
```

```
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

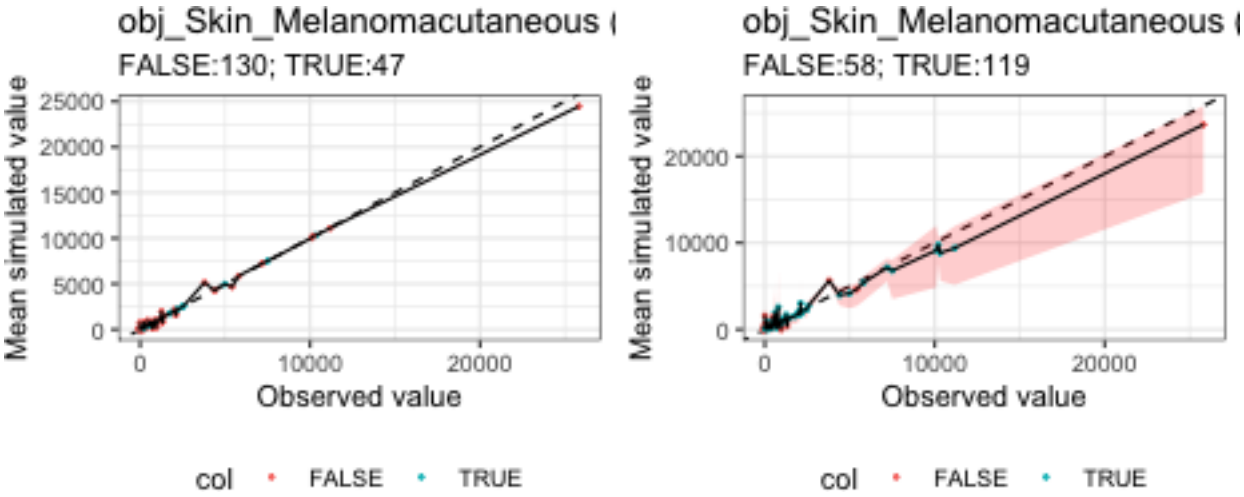


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



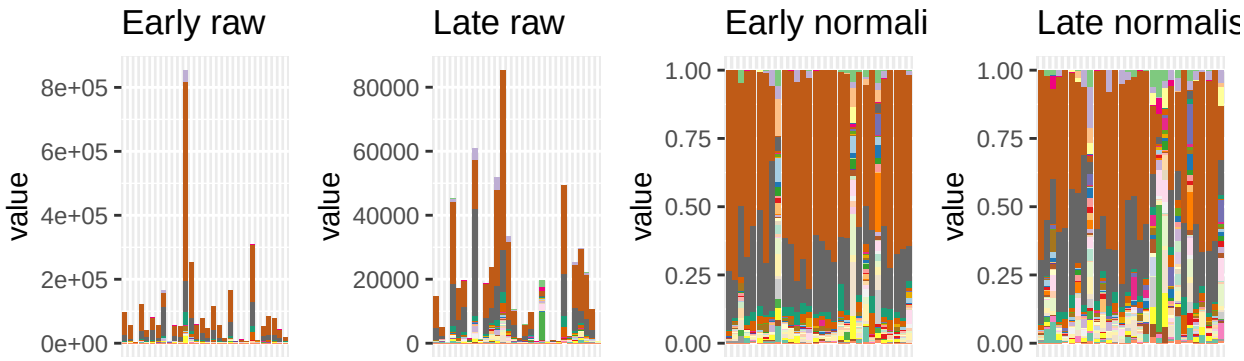
Ranked plot for coverage



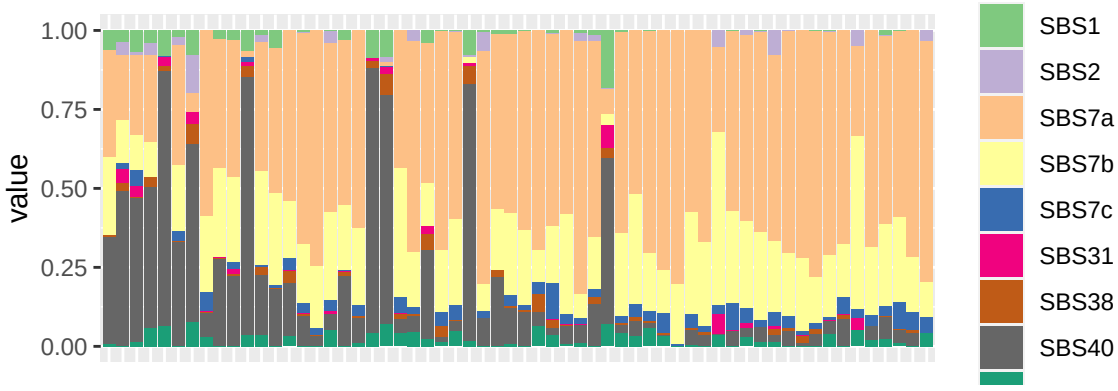
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 30
```



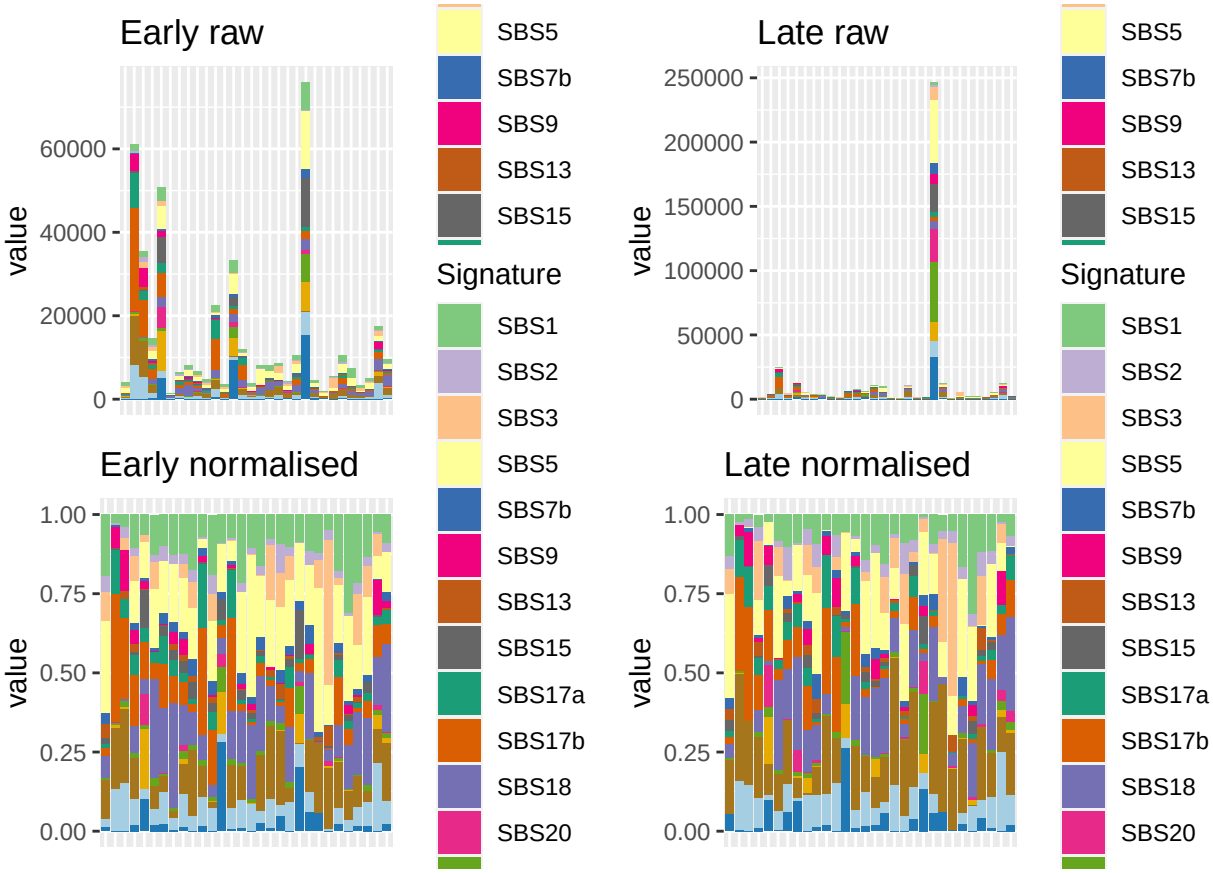
Exposures sorted by increasing number of mutations: SBS40 is crearly prevalent in samples with few mutations.



Stomach-AdenoCA

Barplot and general statistics

```
## [1] 30
```



The number of samples and signatures is:

```
## [1] 60 17
```

The signatures are:

```
##  [1] "SBS1"   "SBS2"   "SBS3"   "SBS5"   "SBS7b"  "SBS9"   "SBS13"  "SBS15"
##  [9] "SBS17a" "SBS17b" "SBS18"  "SBS20"  "SBS21"  "SBS26"  "SBS40"  "SBS41"
## [17] "SBS44"
```

Convergence table

These are the results for the convergence of models fits. Besides fullRE_DMSL_nonexo, we have convergence with almost everything.

```
##              value                                L2                 L1
## 1  Stomach-AdenoCA    hessian_positivedefinite_bool            diagRE_M
## 2  Stomach-AdenoCA hessian_nonpositivedefinite_bool            fullRE_M
## 3  Stomach-AdenoCA hessian_nonpositivedefinite_bool          diagRE_DMDL
## 4  Stomach-AdenoCA                         Timeout         fullRE_halfDM
## 5  Stomach-AdenoCA                         Timeout           fullRE_DMDL
```

```
## 6  Stomach-AdenoCA    hessian_positivedefinite_bool             diagRE_DMSL
## 7  Stomach-AdenoCA    hessian_positivedefinite_bool           sparseRE_DMSL
## 8  Stomach-AdenoCA                          Timeout             fullRE_DMSL
## 9  Stomach-AdenoCA hessian_nonpositivedefinite_bool        fullRE_DMSL_SBS1
## 10 Stomach-AdenoCA hessian_nonpositivedefinite_bool         fullRE_M_nonexo
## 11 Stomach-AdenoCA    hessian_positivedefinite_bool      diagRE_DMSL_nonexo
## 12 Stomach-AdenoCA    hessian_positivedefinite_bool    sparseRE_DMSL_nonexo
## 13 Stomach-AdenoCA                          Timeout       fullRE_DMSL_nonexo
## 14 Stomach-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Stomach-AdenoCA hessian_nonpositivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo, but M hasn't converged. We should include fewer signatures.

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Potentially problematic signatures

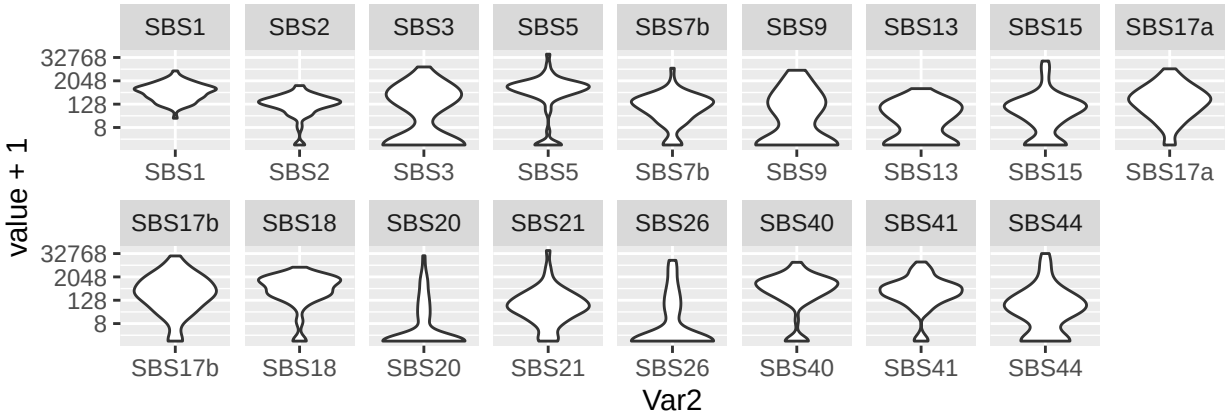
We explore whether there are problematic signatures:

```
colSums(obj_Stomach_AdenoCA$Y == 0)/nrow(obj_Stomach_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS3       SBS5      SBS7b       SBS9      SBS13
## 0.00000000 0.05000000 0.45000000 0.13333333 0.08333333 0.41666667 0.30000000
##      SBS15     SBS17a     SBS17b      SBS18      SBS20      SBS21      SBS26
## 0.21666667 0.05000000 0.06666667 0.06666667 0.70000000 0.06666667 0.68333333
##      SBS40      SBS41      SBS44
## 0.11666667 0.05000000 0.26666667
```

```
colSums(obj_Stomach_AdenoCA$Y)/sum(obj_Stomach_AdenoCA$Y)
```

```
##        SBS1        SBS2        SBS3        SBS5       SBS7b        SBS9
## 0.060565293 0.014095732 0.036577106 0.141513807 0.023236523 0.033484479
##       SBS13       SBS15      SBS17a      SBS17b       SBS18       SBS20
## 0.005394089 0.054607746 0.047754477 0.106132160 0.070925941 0.041749297
##       SBS21       SBS26       SBS40       SBS41       SBS44
## 0.074860849 0.047217740 0.084139891 0.076236383 0.081508486
```

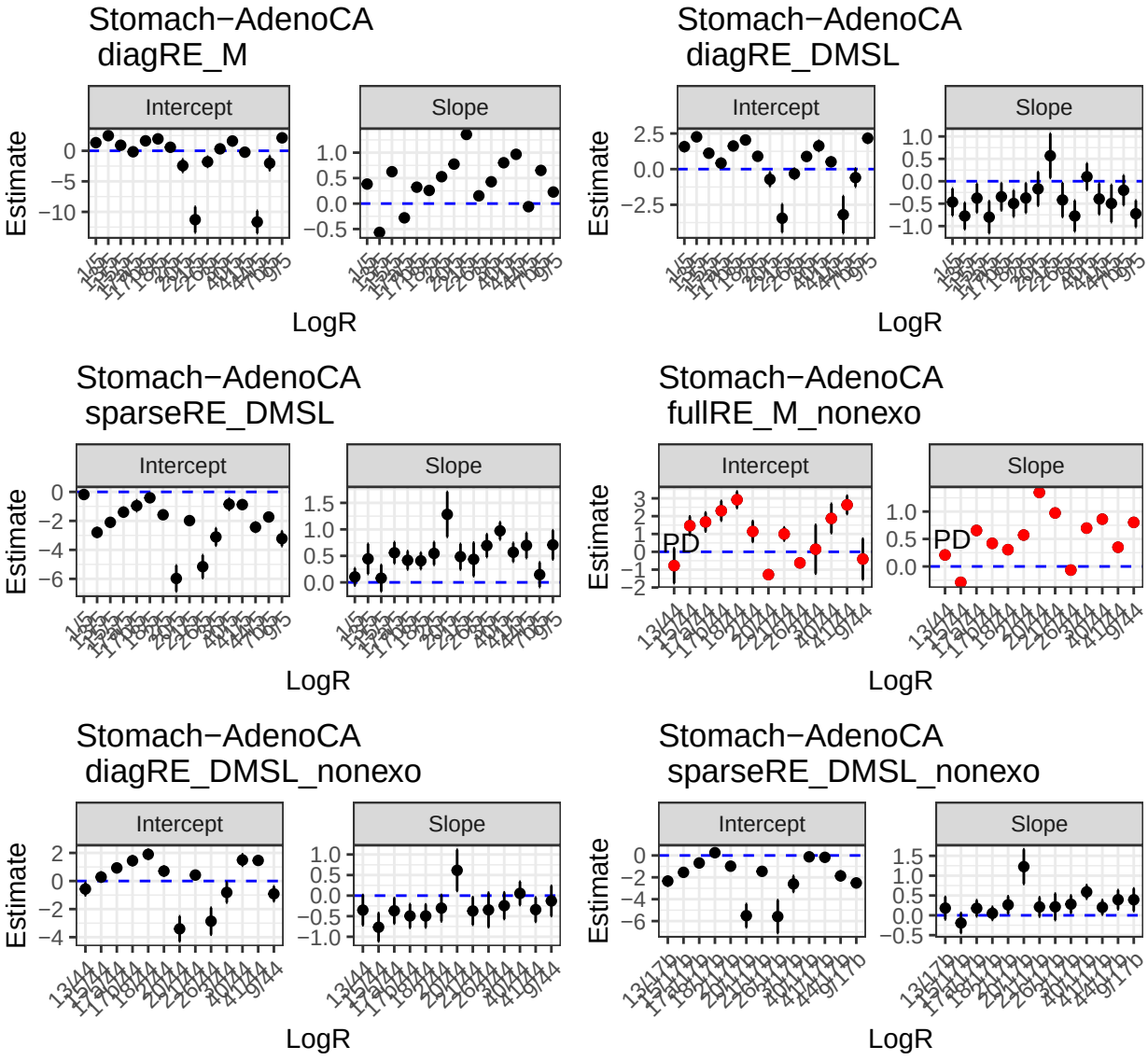


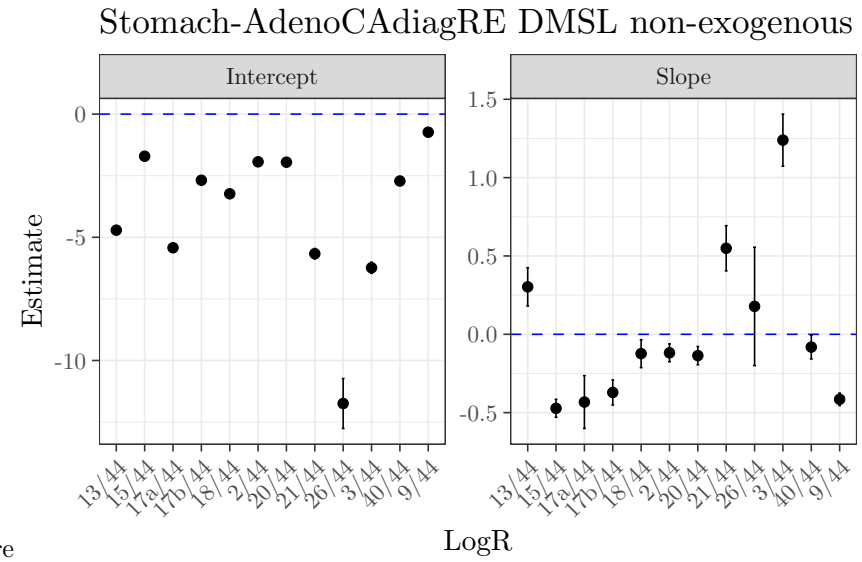
- Removing SBS20, SBS26, fullM still hasn't converged

- Removing SBS20, SBS26, SBS9 fullM still hasn't converged
- Removing SBS20, SBS26, SBS9, SBS13 fullM still hasn't converged
- Removing SBS20, SBS26, SBS9, SBS13, SBS44 fullM still hasn't converged

```
## Warning in sqrt(diag(object$cov.fixed)): NaNs produced
```

Betas





Difficult to see the labels, so I replot it here

We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 0.047603.

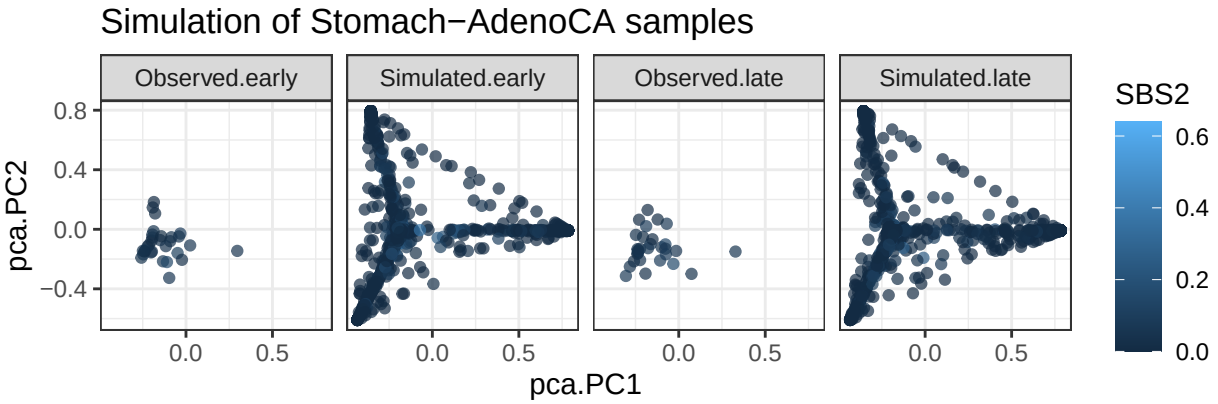
Covariance matrices

I do not include this section as I have had to use ony diagonal matrices.

Simulation under inferred data

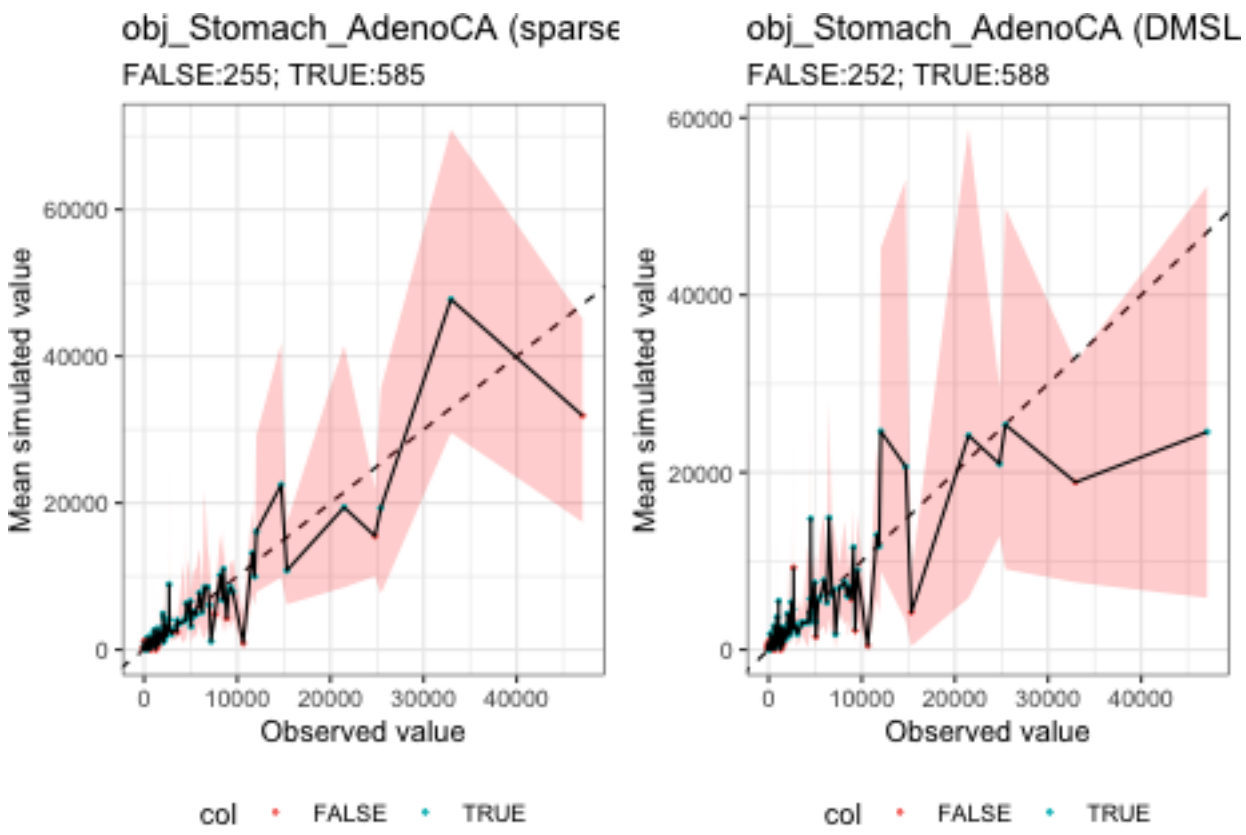
```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



Ranked plot for coverage

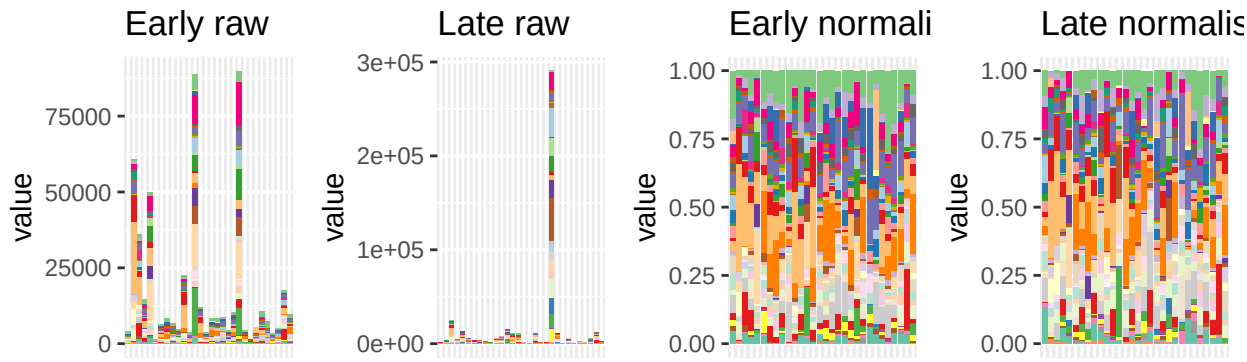
Comparing for now only diagRE_DMSL_nonexo and sparse for nonexo. The blue/red dots seem to behave incorrectly.



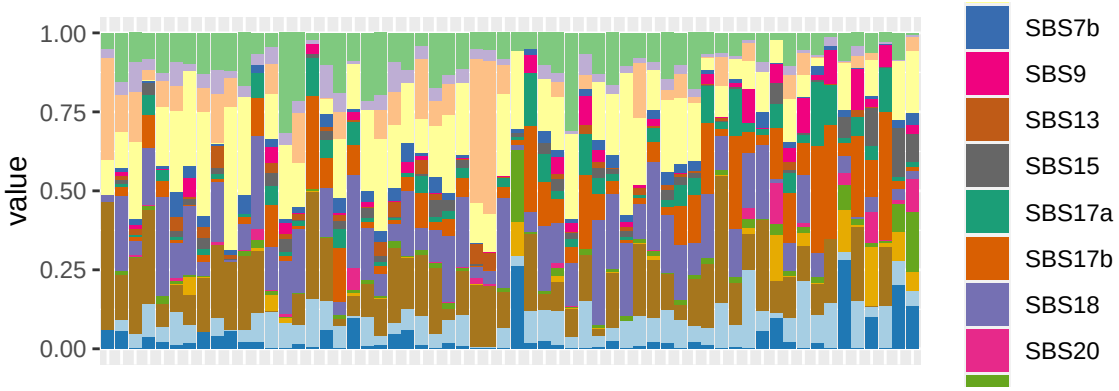
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 30
```



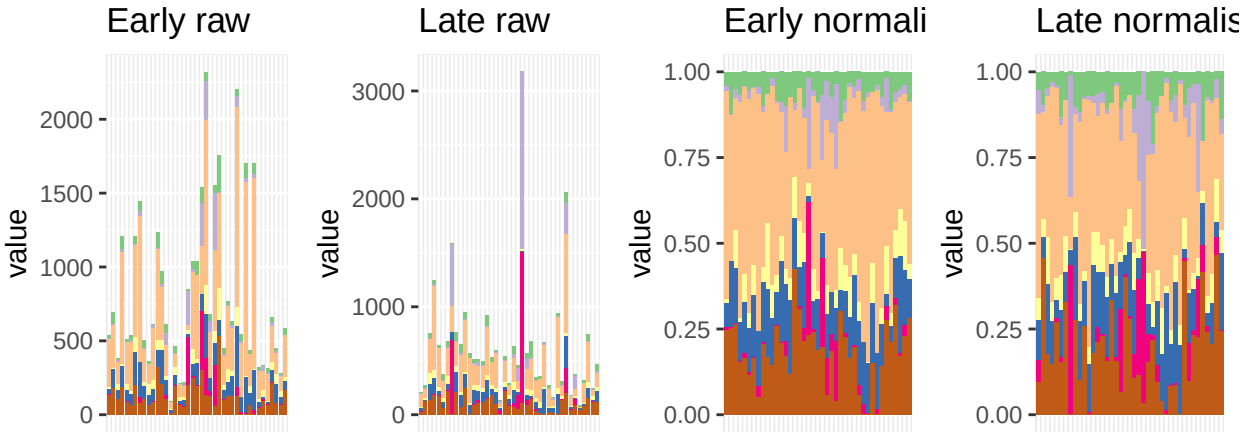
Exposures sorted by increasing number of mutations: there is no clear trend of signatures being associated with the number of mutations.



Thy-AdenoCA

Barplot and general statistics

```
## [1] 41
```



The number of samples and signatures is:

```
## [1] 82  7
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS5"  "SBS6"  "SBS8"  "SBS13" "SBS40"
```

Convergence table

These are the results for the convergence of models fits. Practically everything timed-out.

```
##          value                                L2                 L1
## 1  Thy-AdenoCA    hessian_positivedefinite_bool          diagRE_M
## 2  Thy-AdenoCA    hessian_positivedefinite_bool          fullRE_M
## 3  Thy-AdenoCA    hessian_positivedefinite_bool        diagRE_DMDL
## 4  Thy-AdenoCA                         Timeout       fullRE_halfDM
## 5  Thy-AdenoCA hessian_nonpositivedefinite_bool        fullRE_DMDL
## 6  Thy-AdenoCA    hessian_positivedefinite_bool        diagRE_DMSL
## 7  Thy-AdenoCA    hessian_positivedefinite_bool      sparseRE_DMSL
## 8  Thy-AdenoCA                         Timeout        fullRE_DMSL
## 9  Thy-AdenoCA                         Timeout  fullRE_DMSL_SBS1
```

```
## 10 Thy-AdenoCA                          Timeout          fullRE_M_nonexo
## 11 Thy-AdenoCA    hessian_positivedefinite_bool       diagRE_DMSL_nonexo
## 12 Thy-AdenoCA    hessian_positivedefinite_bool     sparseRE_DMSL_nonexo
## 13 Thy-AdenoCA                          Timeout       fullRE_DMSL_nonexo
## 14 Thy-AdenoCA                          Timeout       fullRE_DMDL_nonexo
## 15 Thy-AdenoCA                          Timeout fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo do not converge, even though there aren't many signatures:

```
## [1] FALSE
```

Potentially problematic signatures

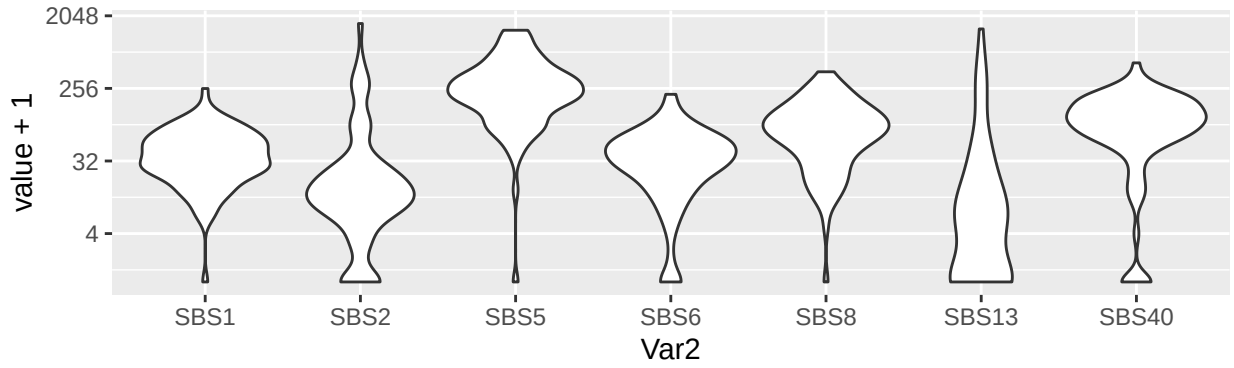
We explore whether there are problematic signatures:

```
colSums(obj_Thy_AdenoCA$Y == 0)/nrow(obj_Thy_AdenoCA$Y)
```

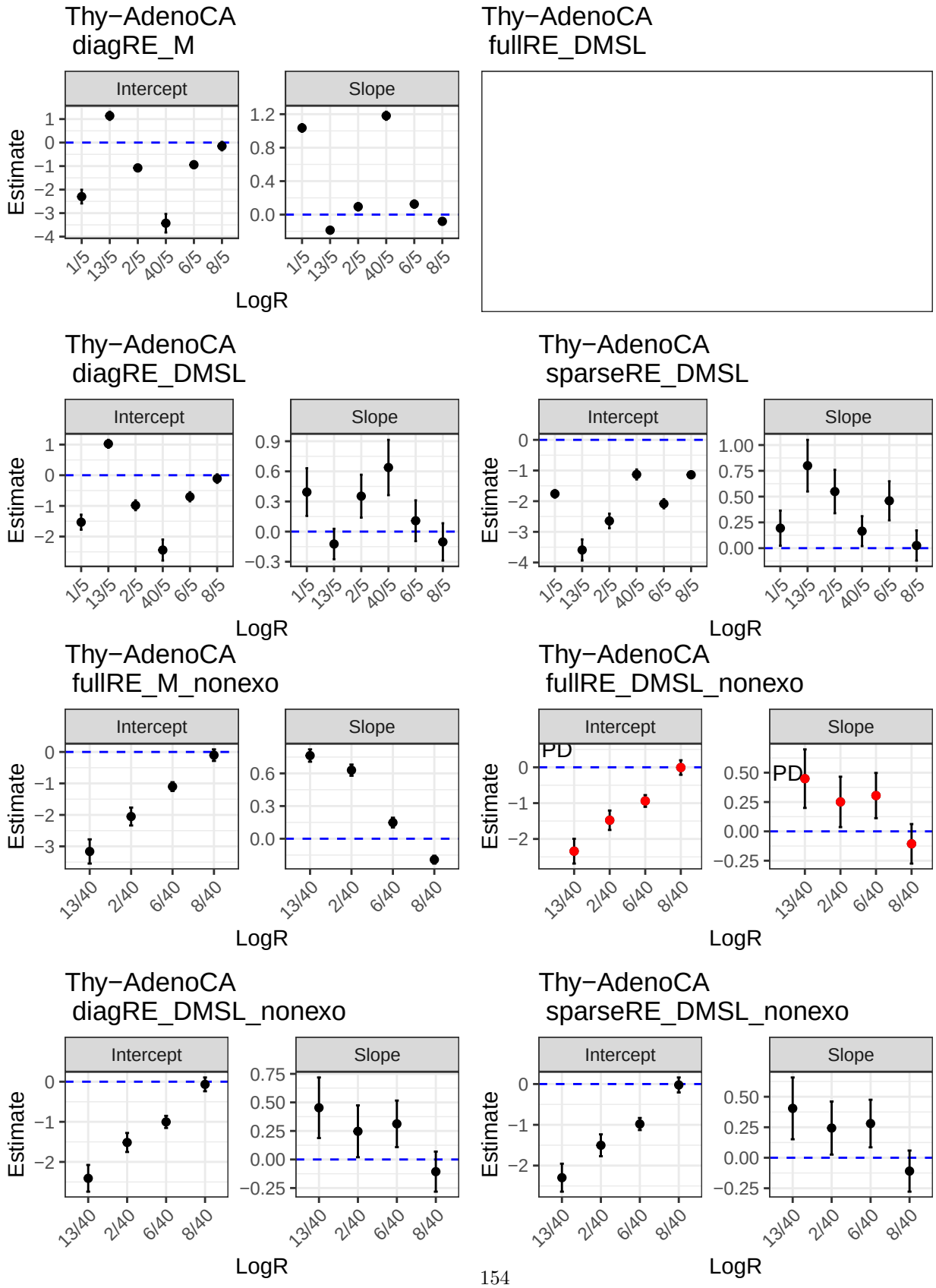
```
##       SBS1       SBS2       SBS5       SBS6       SBS8      SBS13      SBS40
## 0.01219512 0.12195122 0.01219512 0.06097561 0.01219512 0.35365854 0.07317073
```

```
colSums(obj_Thy_AdenoCA$Y)/sum(obj_Thy_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS5       SBS6       SBS8      SBS13      SBS40
## 0.06120959 0.08303459 0.44489809 0.05581557 0.12930691 0.07239594 0.15333931
```



Betas


Thy−AdenoCA
diagRE_M
Intercept
Slope
Estimate
LogR
1/5
13/5
2/5
40/5
6/5
8/5
Thy−AdenoCA
fullRE_DMSL
Thy−AdenoCA
diagRE_DMSL
Thy−AdenoCA
sparseRE_DMSL
Thy−AdenoCA
fullRE_M_nonexo
13/40
2/40
6/40
8/40
Thy−AdenoCA
fullRE_DMSL_nonexo
PD
Thy−AdenoCA
diagRE_DMSL_nonexo
Thy−AdenoCA
sparseRE_DMSL_nonexo

We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 0.1064332.

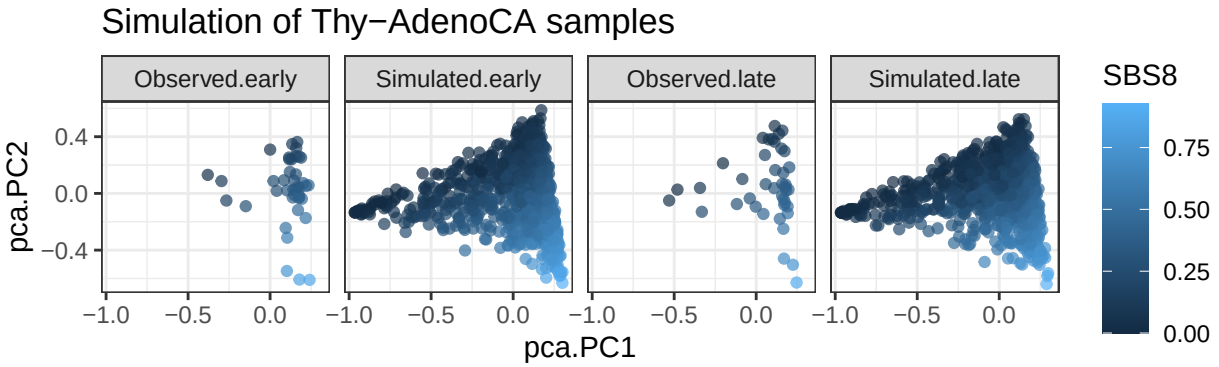
Covariance matrices

I do not include those.

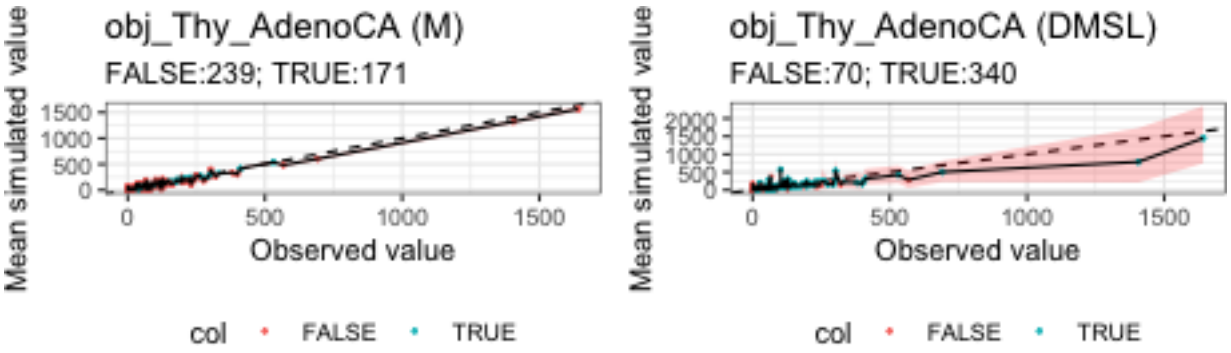
Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



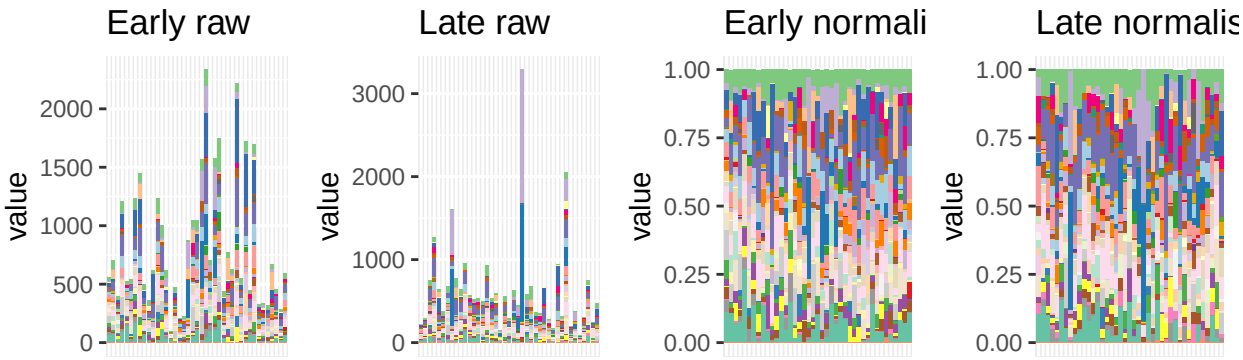
Ranked plot for coverage



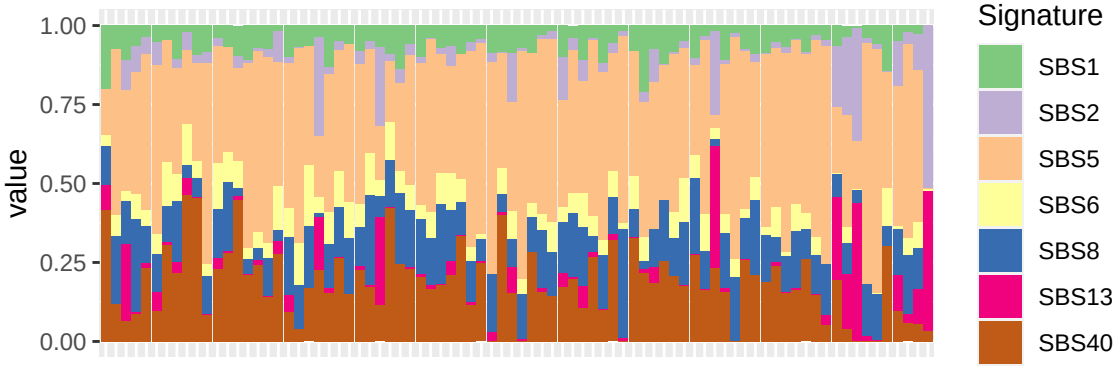
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 41
```



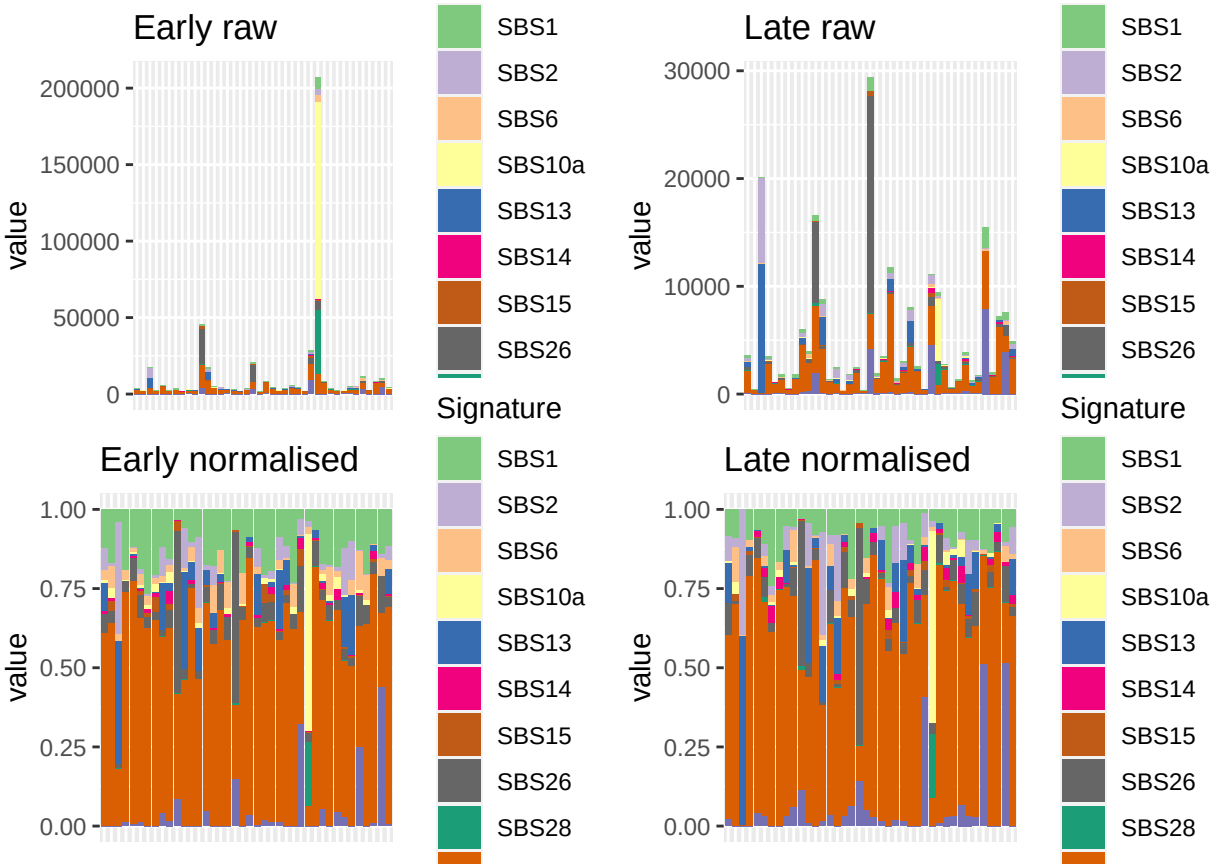
Exposures sorted by increasing number of mutations: there is no trend of signatures being associated with the number of mutations.



Uterus-AdenoCA

Barplot and general statistics

```
## [1] 40
```



The number of samples and signatures is:

```
## [1] 82  7
```

The signatures are:

```
## [1] "SBS1"  "SBS2"  "SBS5"  "SBS6"  "SBS8"  "SBS13" "SBS40"
```

Convergence table

These are the results for the convergence of models fits. Almost everything has converged.

```
##             value                                L2                   L1
## 1  Uterus-AdenoCA hessian_nonpositivedefinite_bool              diagRE_M
## 2  Uterus-AdenoCA hessian_nonpositivedefinite_bool              fullRE_M
## 3  Uterus-AdenoCA hessian_nonpositivedefinite_bool            diagRE_DMDL
## 4  Uterus-AdenoCA                          Timeout          fullRE_halfDM
## 5  Uterus-AdenoCA hessian_nonpositivedefinite_bool            fullRE_DMDL
## 6  Uterus-AdenoCA    hessian_positivedefinite_bool            diagRE_DMSL
## 7  Uterus-AdenoCA    hessian_positivedefinite_bool          sparseRE_DMSL
## 8  Uterus-AdenoCA                          Timeout            fullRE_DMSL
## 9  Uterus-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMSL_SBS1
## 10 Uterus-AdenoCA    hessian_positivedefinite_bool        fullRE_M_nonexo
## 11 Uterus-AdenoCA    hessian_positivedefinite_bool      diagRE_DMSL_nonexo
## 12 Uterus-AdenoCA    hessian_positivedefinite_bool    sparseRE_DMSL_nonexo
## 13 Uterus-AdenoCA                          Timeout      fullRE_DMSL_nonexo
```

```
## 14 Uterus-AdenoCA hessian_nonpositivedefinite_bool       fullRE_DMDL_nonexo
## 15 Uterus-AdenoCA    hessian_positivedefinite_bool fullRE_DMDL_sortednonexo
```

Re-running of fitting

Using fullRE_M_nonexo to fit fullRE_DMSL_nonexo.

If we use the values of the fullRE M exo as initial values for the fullRE DMSL exo doesn't converge:

```
## [1] FALSE
```

Potentially problematic signatures

We explore whether there are problematic signatures:

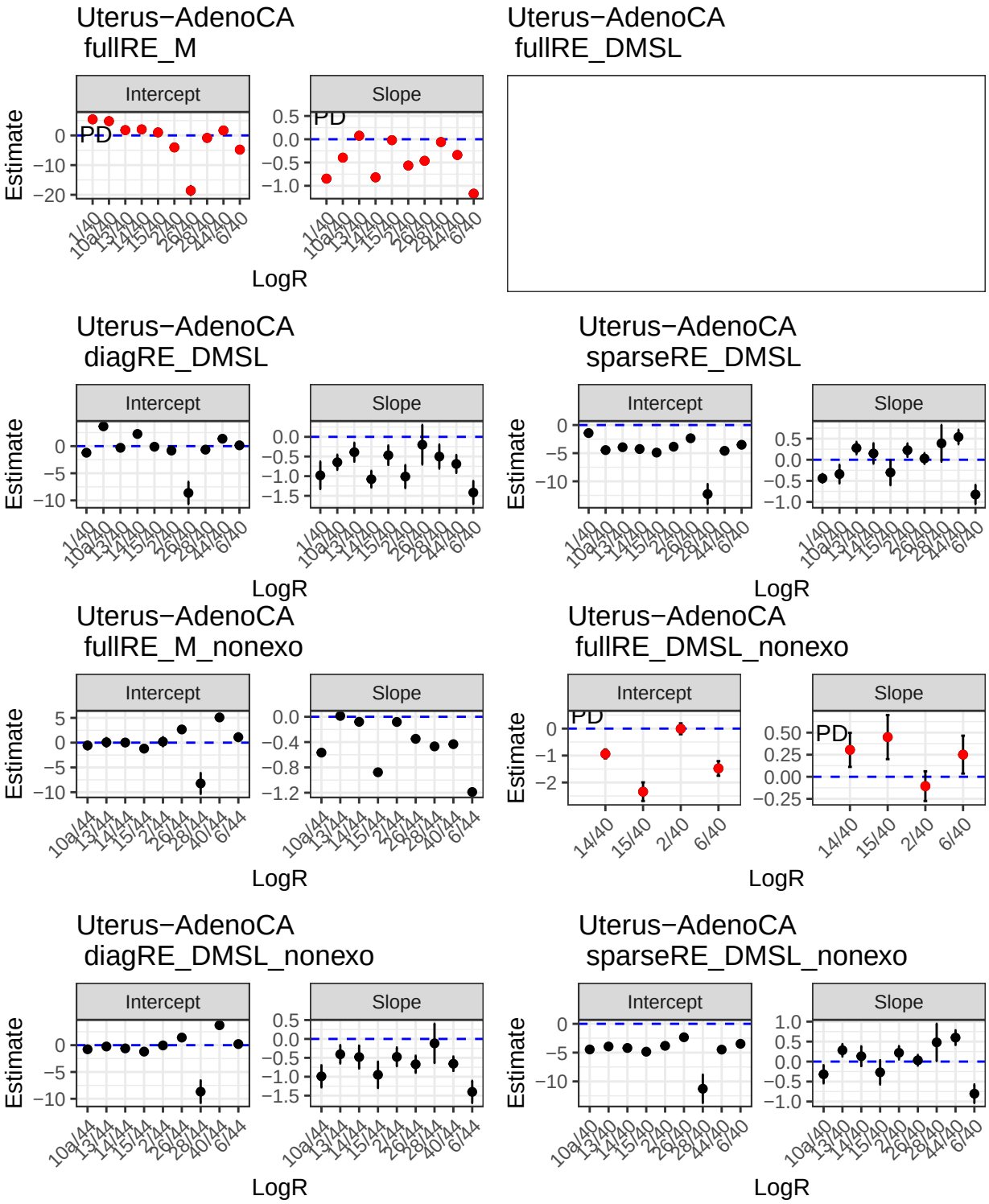
```
colSums(obj_Uterus_AdenoCA$Y == 0)/nrow(obj_Uterus_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS5       SBS6       SBS8      SBS13      SBS40
## 0.01219512 0.12195122 0.01219512 0.06097561 0.01219512 0.35365854 0.07317073
```

```
colSums(obj_Uterus_AdenoCA$Y)/sum(obj_Uterus_AdenoCA$Y)
```

```
##       SBS1       SBS2       SBS5       SBS6       SBS8      SBS13      SBS40
## 0.06120959 0.08303459 0.44489809 0.05581557 0.12930691 0.07239594 0.15333931
```

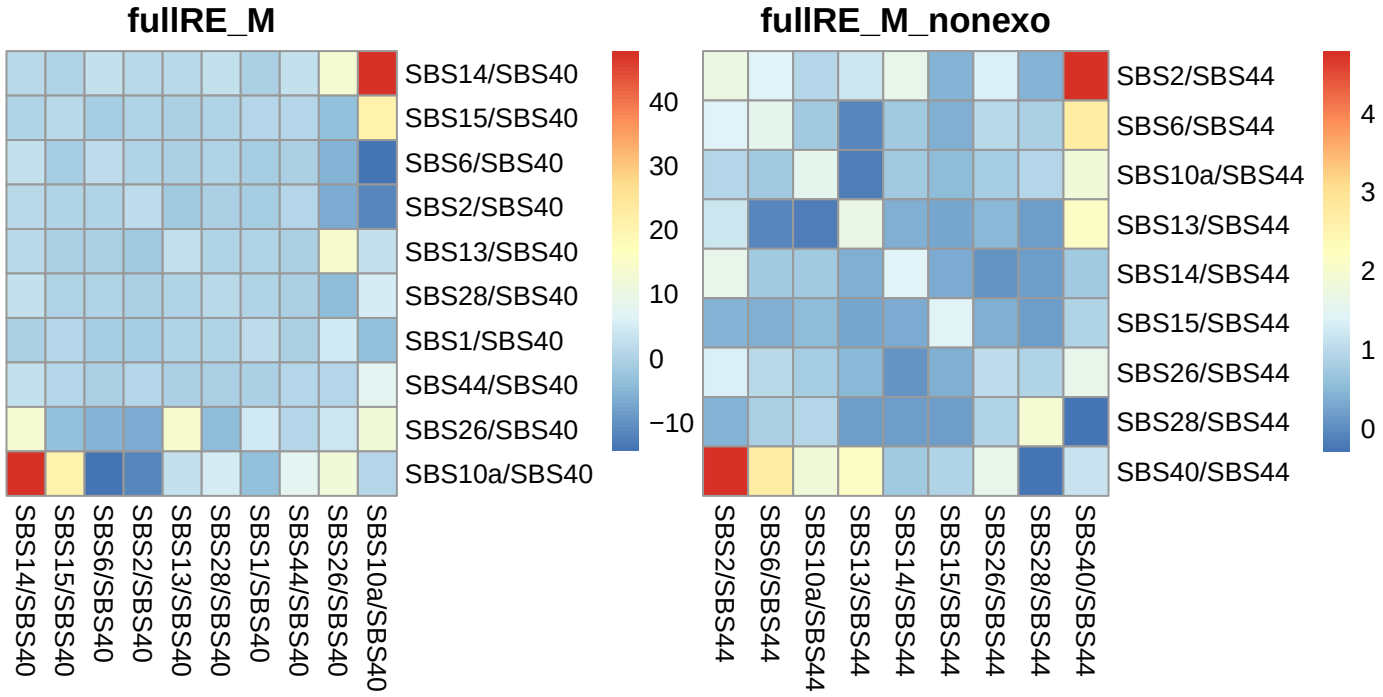
Betas



We use the results from the diag RE single lambda DM to test for differential abundance, giving a p-value of 1.4837739×10^{-4}.

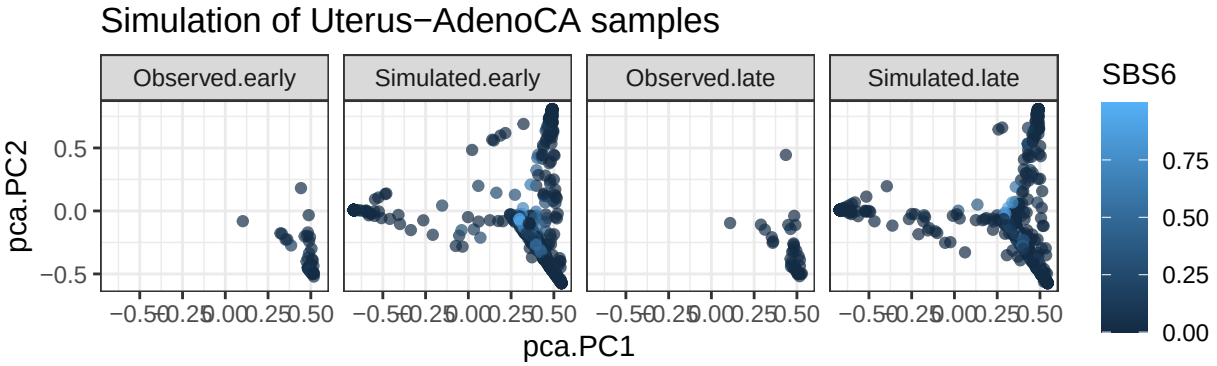
Covariance matrices

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

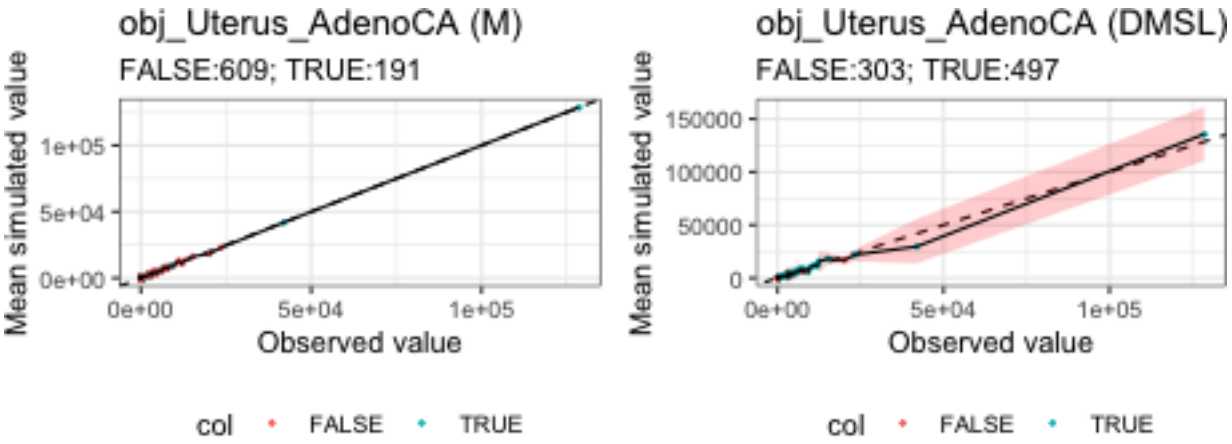


Simulation under inferred data

```
## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec, :
## This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = dmin1, arg_entries_var = var_vec_v2, :
## This function had been incorrect until now (30 july 2021)
```



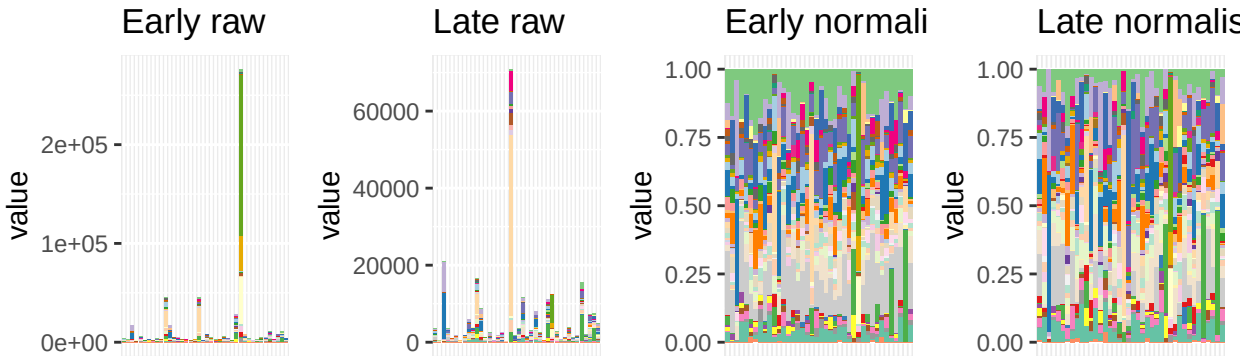
Ranked plot for coverage



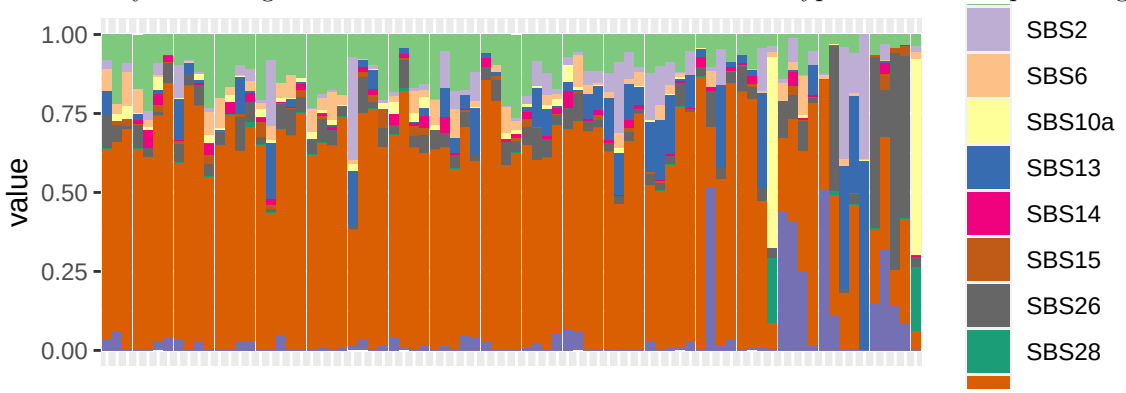
Signatures from mutSigExtractor

The signatures from mutSigExtractor are as follows:

```
## [1] 40
```



Exposures sorted by increasing number of mutations: there is a trend of hypermutated samples being very chaotic.



All p-values for non-exogenous signatures

% latex table generated in R 4.0.3 by xtable 1.8-4 package % Thu Jul 15 14:31:19 2021

	ct	pvalue	model
1	Bone-Osteosarc	0.00	diagRE_DMSL_nonexo
2	Breast-AdenoCA	0.00	diagRE_DMSL_nonexo
3	Cervix-SCC	0.00	fullRE_DMSL_nonexo
4	CNS-GBM	0.00	fullRE_DMSL_nonexo
5	CNS-Medullo	0.07	fullRE_DMSL_nonexo
6	CNS-Oligo	0.52	fullRE_DMSL_nonexo
7	CNS-PiloAstro	0.26	fullRE_DMSL_nonexo
8	ColoRect-AdenoCA	0.00	diagRE_DMSL_nonexo
9	Eso-AdenoCA	0.00	diagRE_DMSL_nonexo
10	Head-SCC	0.00	fullRE_DMSL_nonexo
11	Kidney-ChRCC	0.27	fullRE_DMSL_nonexo
12	Kidney-RCC.clearcell	0.00	fullRE_DMSL_nonexo
13	Kidney-RCC.papillary	0.00	fullRE_DMSL_nonexo
14	Liver-HCC	0.00	fullRE_DMSL_nonexo
15	Lung-AdenoCA	0.00	diagRE_DMSL_nonexo
16	Lung-SCC	0.03	diagRE_DMSL_nonexo
17	Lymph-BNHL	0.00	fullRE_DMSL_nonexo
18	Lymph-CLL	0.00	fullRE_DMSL_nonexo
19	Myeloid-MPN	0.00	fullRE_DMSL_nonexo
20	Ovary-AdenoCA	0.00	diagRE_DMSL_nonexo
21	Panc-AdenoCA	0.00	diagRE_DMSL_nonexo
22	Panc-Endocrine	0.00	diagRE_DMSL_nonexo
23	Prost-AdenoCA	0.00	diagRE_DMSL_nonexo
24	Skin-Melanoma.acral		
25	Skin-Melanoma.cutaneous	0.06	fullRE_DMSL_nonexo
26	Stomach-AdenoCA	0.05	diagRE_DMSL_nonexo
27	Thy-AdenoCA	0.11	diagRE_DMSL_nonexo
28	Uterus-AdenoCA	0.00	diagRE_DMSL_nonexo

All p-values

% latex table generated in R 4.0.3 by xtable 1.8-4 package % Thu Jul 15 14:13:08 2021

Correlation between p-values and number of samples

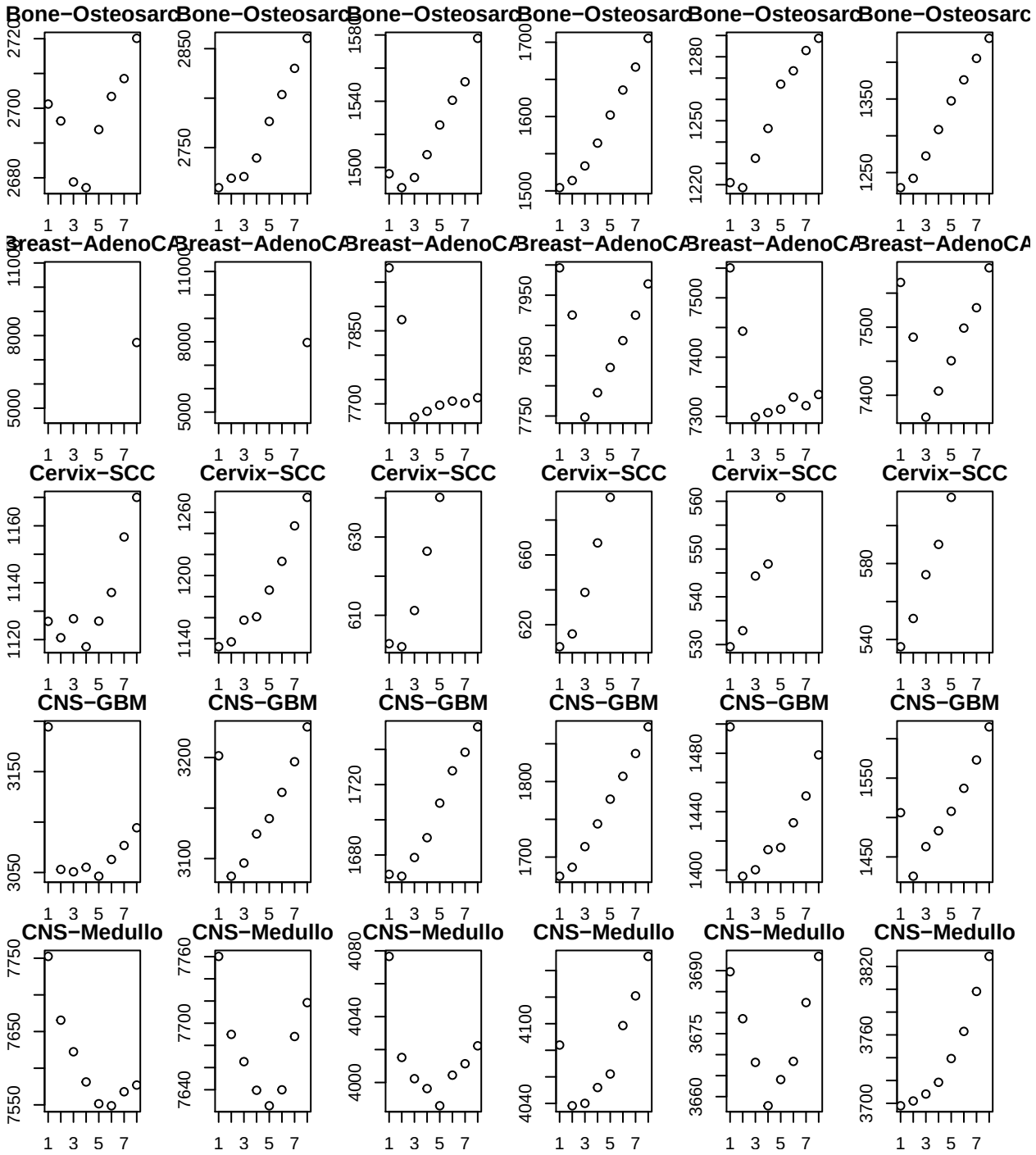
	pvals_fullRE_M	pvals_diagRE_DM	pvals_DM	pvals_DMnonexo
Bone-Osteosarc	0.00	0.00	0.00	
Breast-AdenoCA	0.00	0.00	0.00	0.00
Cervix-SCC	0.00	0.11	0.12	0.01
CNS-GBM	0.00	0.00	0.03	0.00
CNS-Medullo	0.00	0.02	0.02	0.47
CNS-Oligo	0.00	0.03	0.03	
CNS-PiloAstro	0.00	0.03	0.03	0.64
ColoRect-AdenoCA	0.00	0.00	0.00	0.00
Eso-AdenoCA	0.00	0.00	0.00	0.00
Head-SCC	0.00	0.00	0.00	0.00
Kidney-ChRCC	0.00	0.00	0.00	0.64
Kidney-RCC.clearcell	0.00	0.00	0.00	0.00
Kidney-RCC.papillary	0.00	0.00	0.00	0.00
Liver-HCC	0.00	0.00	0.00	0.00
Lung-AdenoCA	0.00	0.01	0.00	0.01
Lung-SCC	0.00	0.00	0.00	0.17
Lymph-BNHL	0.00	0.00	0.00	0.00
Lymph-CLL	0.00	0.00	0.00	
Myeloid-MPN	0.00	0.00	0.00	
Ovary-AdenoCA	0.00	0.00	0.00	0.00
Panc-AdenoCA	0.00	0.00	0.00	0.00
Panc-Endocrine	0.00	0.00	0.00	0.00
Prost-AdenoCA	0.00	0.00	0.00	0.00
Skin-Melanoma.acral	0.00	0.18	0.15	
Skin-Melanoma.cutaneous	0.00	0.00	0.00	
Stomach-AdenoCA	0.00	0.00	0.00	0.08
Thy-AdenoCA	0.00	0.03	0.02	0.47
Uterus-AdenoCA	0.00	0.00	0.00	0.00

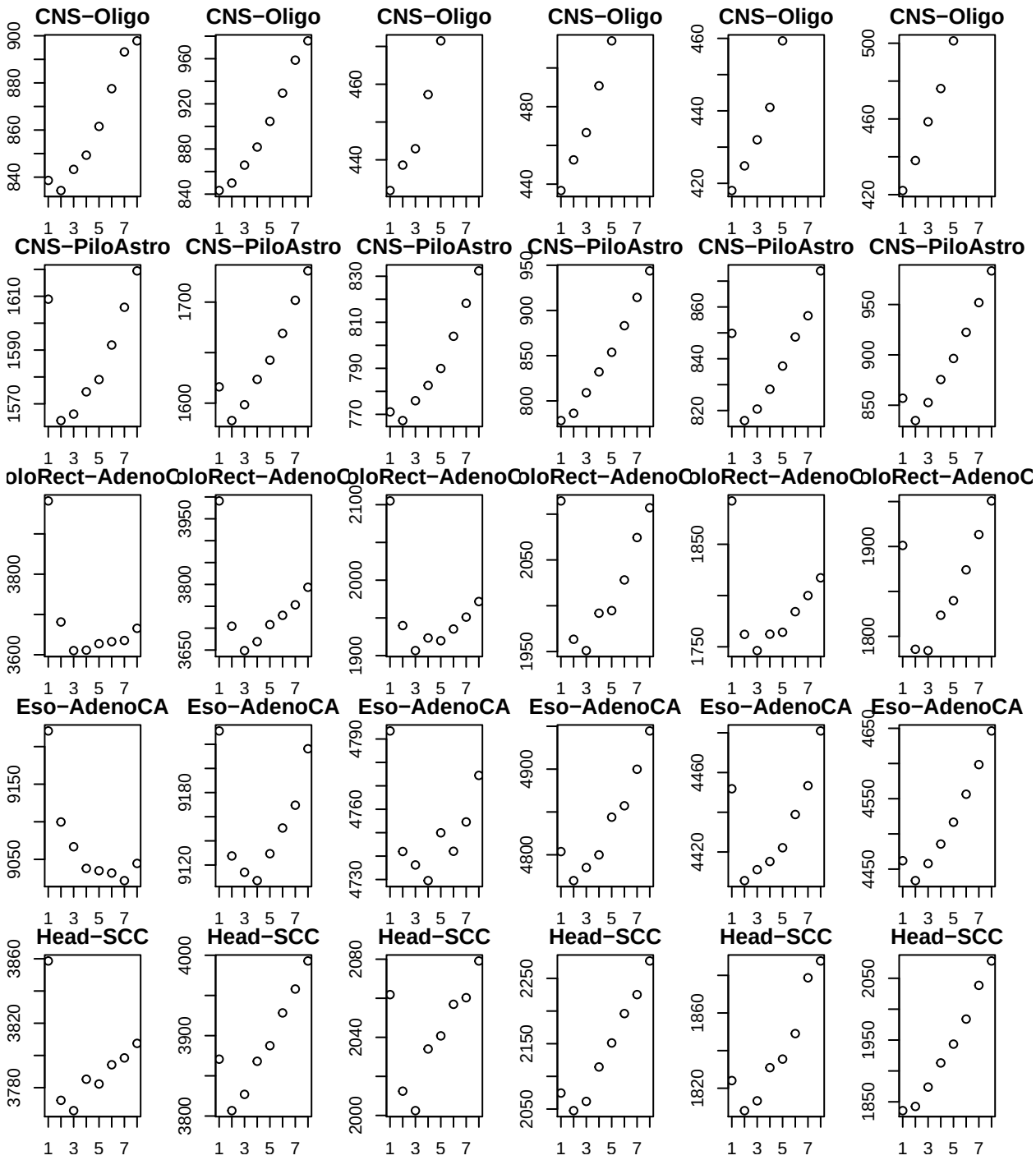
Dirichlet-Multinomial Mixtures

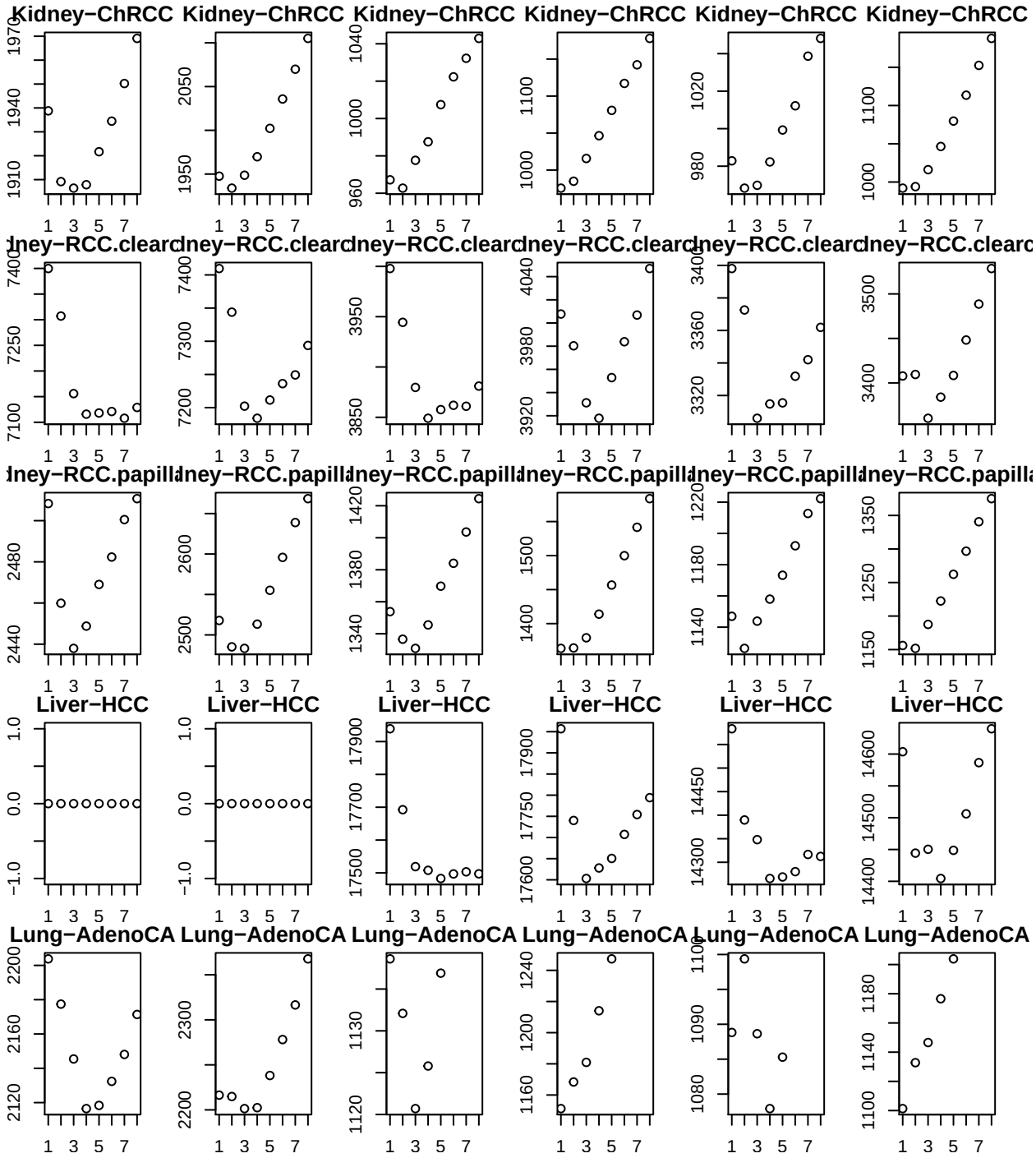
We run the software `MicrobeDMMv1.0` to determine whether we are facing DMM mixtures or not.

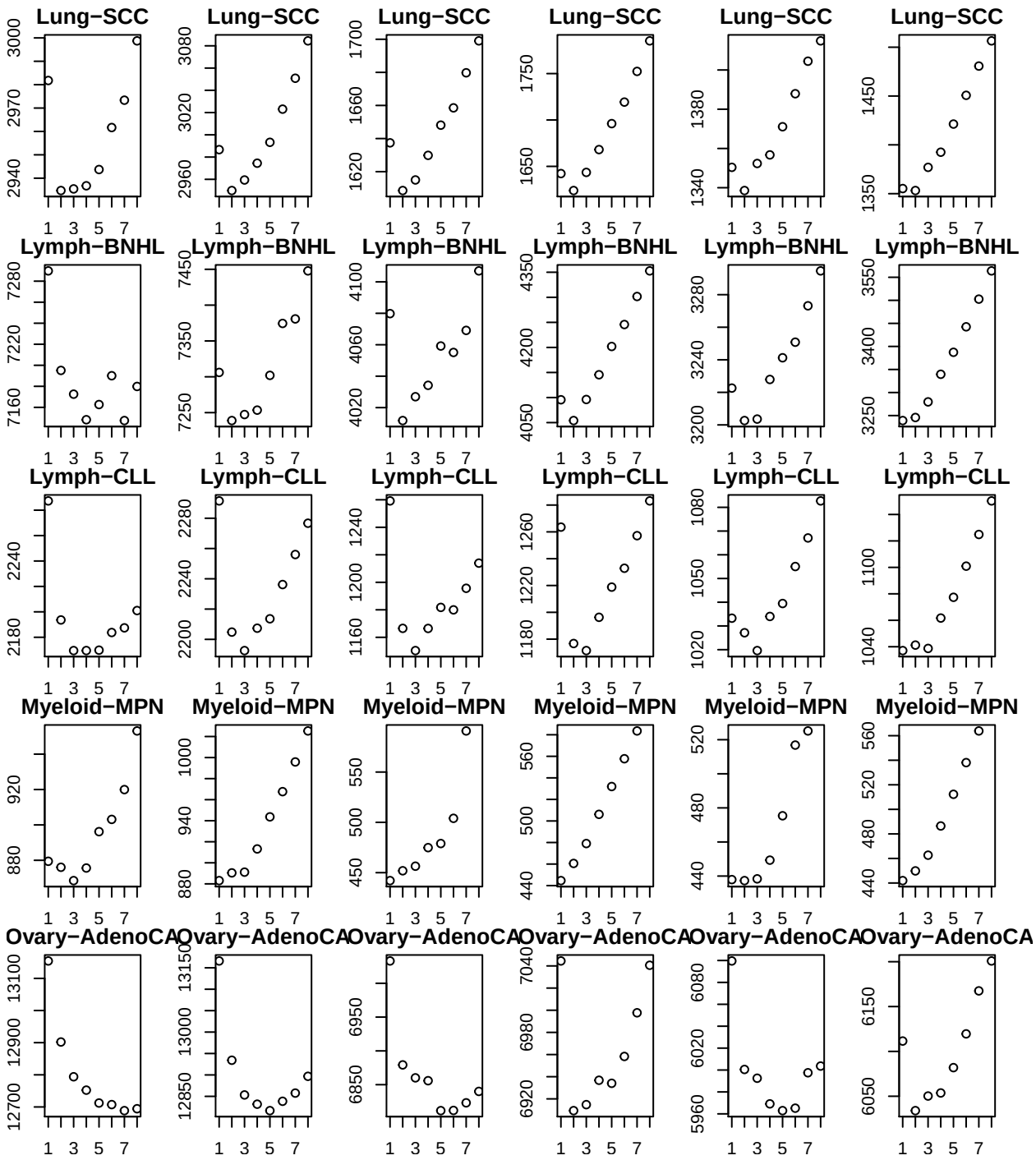
We save the files in two ways: all of the samples - early or not - together, and separately.

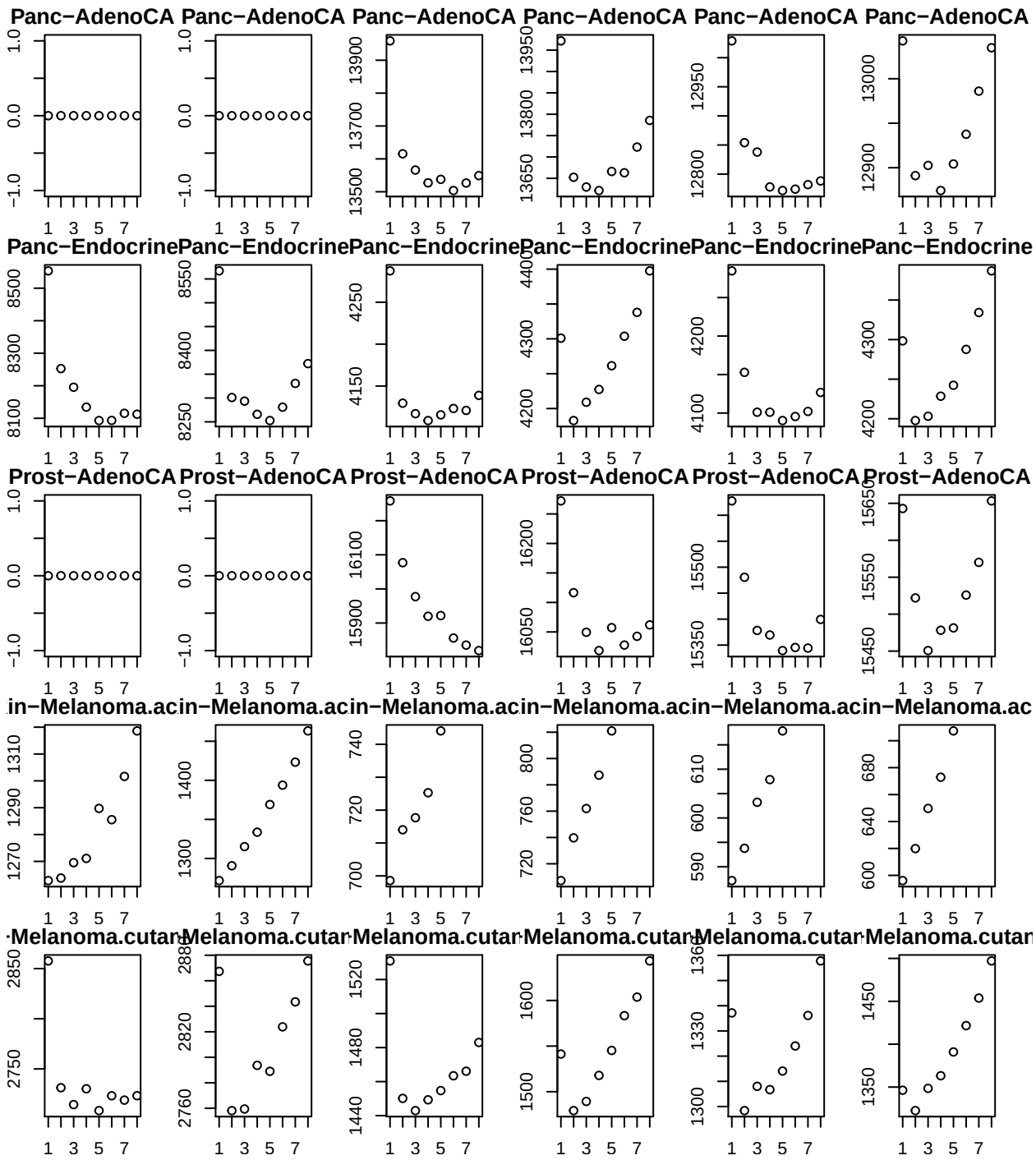
In some cases DMM says that there is an error with the input file - in this case the AIC or BIC is not plotted. If all of them are missing, all BIC and AIC are set to zero.

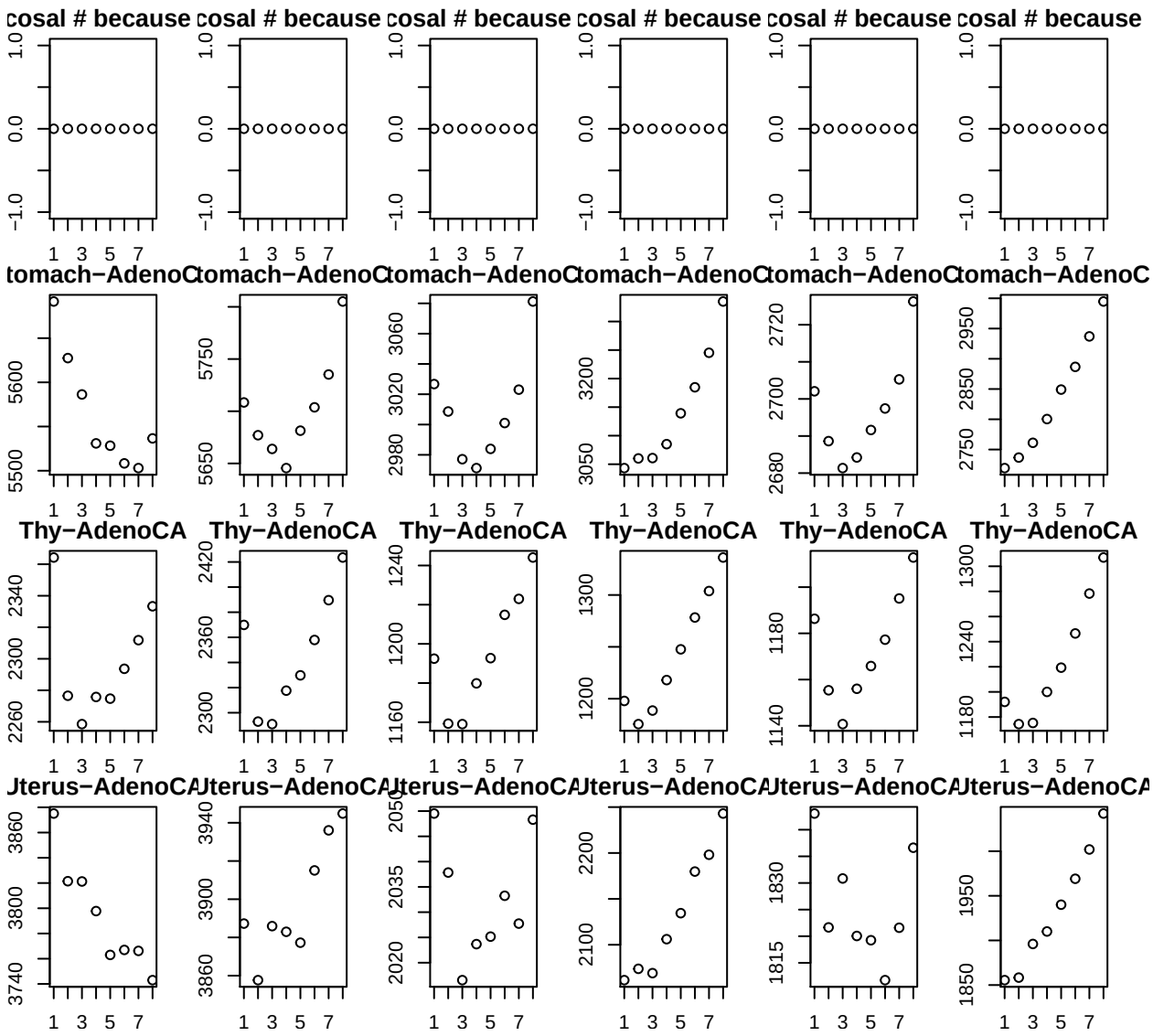



CNS−Oligo
CNS−Oligo
CNS−Oligo
CNS−Oligo
CNS−Oligo
CNS−Oligo
CNS−PiloAstro
CNS−PiloAstro
CNS−PiloAstro
CNS−PiloAstro
CNS−PiloAstro
CNS−PiloAstro
oloRect−AdenoC
oloRect−AdenoC
oloRect−AdenoC
oloRect−AdenoC
oloRect−AdenoC
oloRect−AdenoC
Eso−AdenoCA
Eso−AdenoCA
Eso−AdenoCA
Eso−AdenoCA
Eso−AdenoCA
Eso−AdenoCA
Head−SCC
Head−SCC
Head−SCC
Head−SCC
Head−SCC
Head−SCC

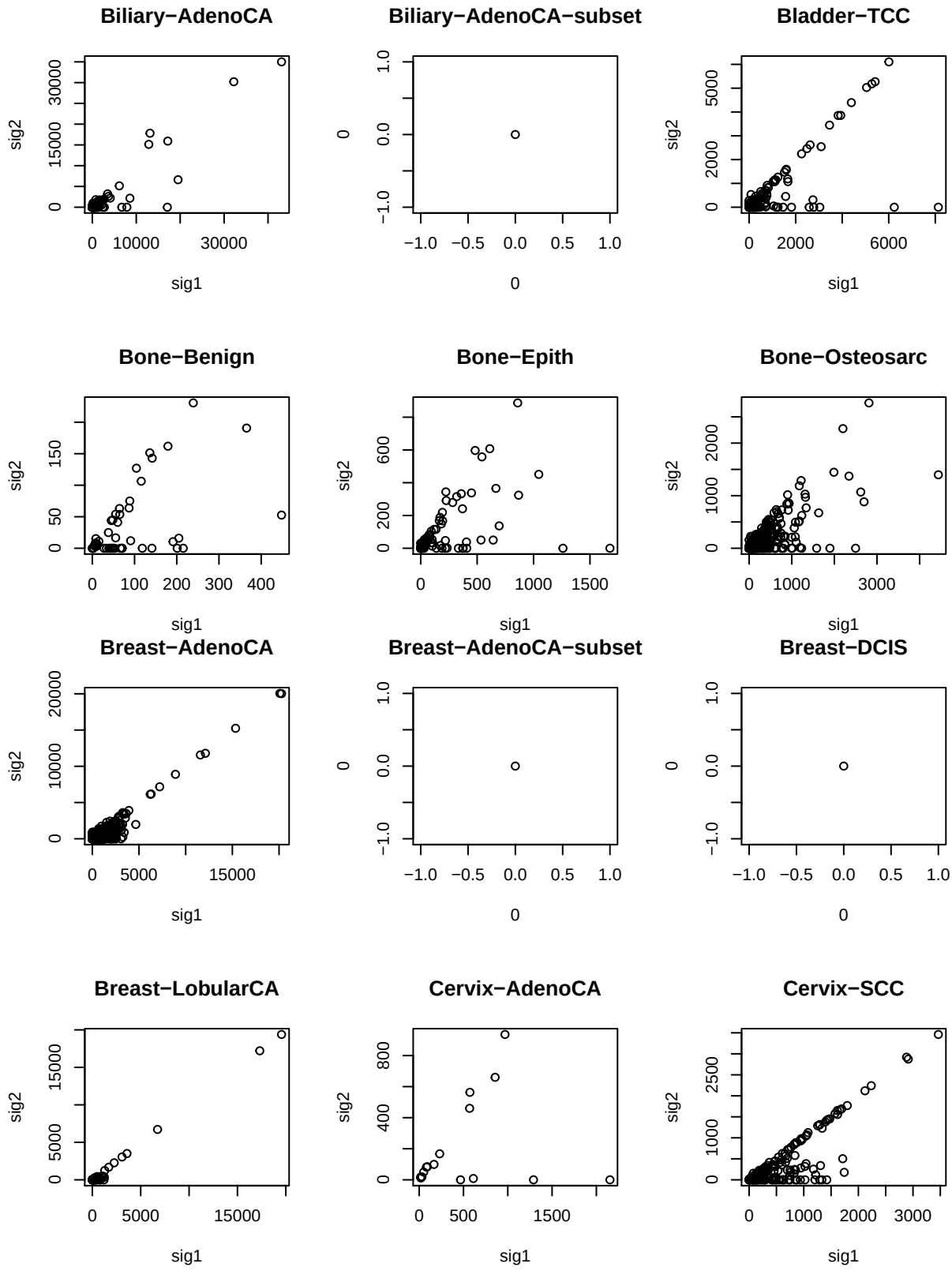

Kidney−ChRCC
Kidney−ChRCC
Kidney−ChRCC
Kidney−ChRCC
Kidney−ChRCC
Kidney−ChRCC
ney−RCC.clearc
ney−RCC.clearc
ney−RCC.clearc
ney−RCC.clearc
ney−RCC.clearc
ney−RCC.clearc
ney−RCC.papilla
ney−RCC.papilla
ney−RCC.papilla
ney−RCC.papilla
ney−RCC.papilla
ney−RCC.papilla
Liver−HCC
Liver−HCC
Liver−HCC
Liver−HCC
Liver−HCC
Liver−HCC
Lung−AdenoCA
Lung−AdenoCA
Lung−AdenoCA
Lung−AdenoCA
Lung−AdenoCA
Lung−AdenoCA

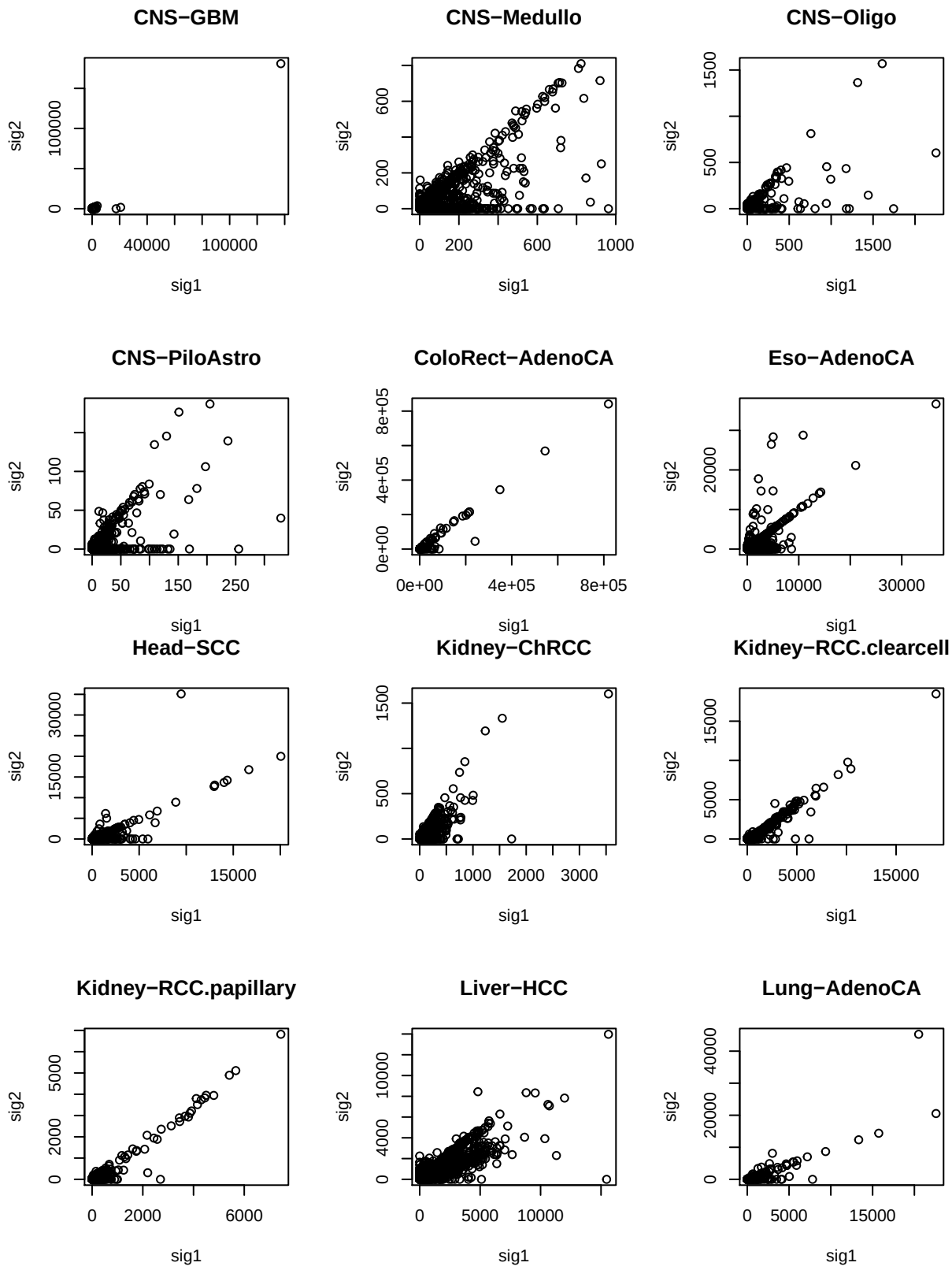

Lung−SCC
Lung−SCC
Lung−SCC
Lung−SCC
Lung−SCC
Lung−SCC
Lymph−BNHL
Lymph−BNHL
Lymph−BNHL
Lymph−BNHL
Lymph−BNHL
Lymph−BNHL
Lymph−CLL
Lymph−CLL
Lymph−CLL
Lymph−CLL
Lymph−CLL
Lymph−CLL
Myeloid−MPN
Myeloid−MPN
Myeloid−MPN
Myeloid−MPN
Myeloid−MPN
Myeloid−MPN
Ovary−AdenoCA
Ovary−AdenoCA
Ovary−AdenoCA
Ovary−AdenoCA
Ovary−AdenoCA
Ovary−AdenoCA

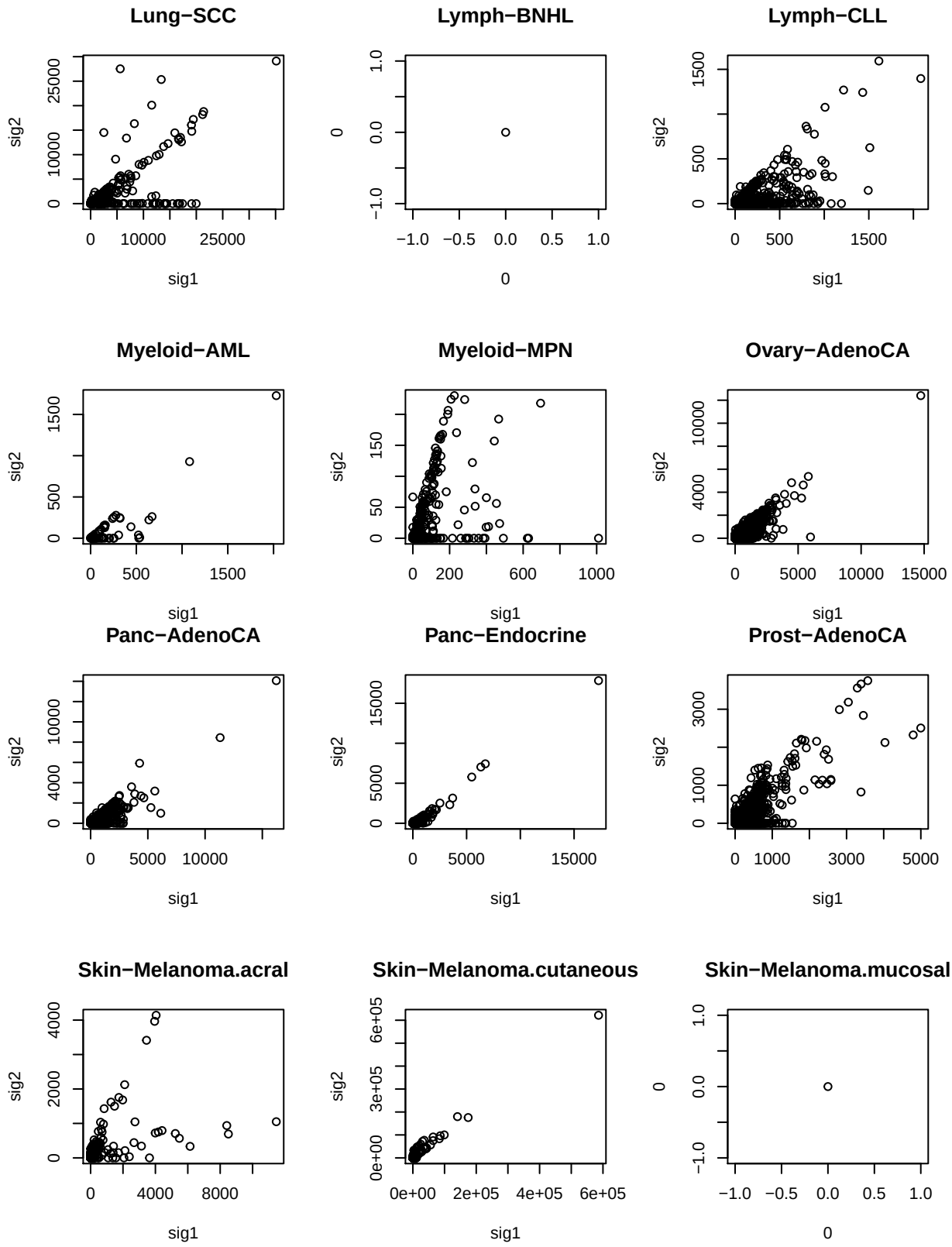

Panc−AdenoCA
Panc−Endocrine
Prost−AdenoCA
in−Melanoma.acin−Melanoma.ac
·Melanoma.cutan

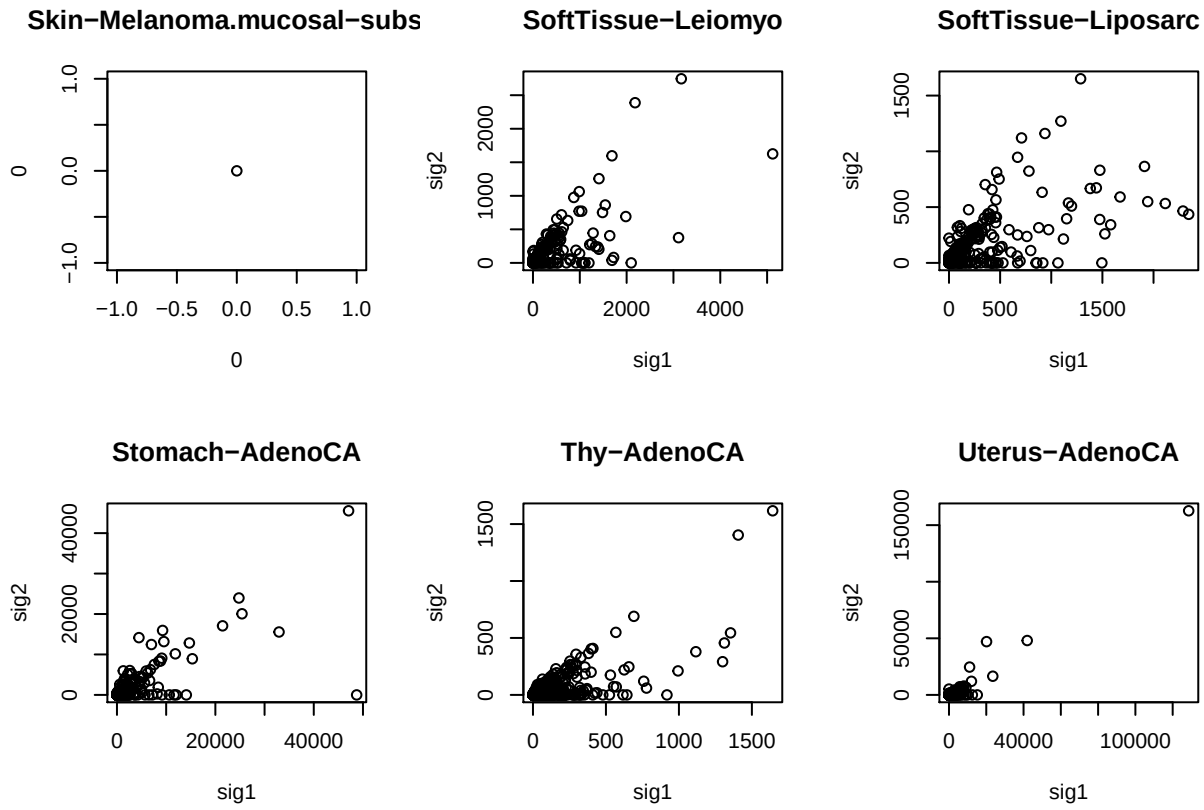

cosal # because
tomach−AdenoC
Thy−AdenoCA
Jterus−AdenoCA

Comparison of signature exposures with QP and mutsigextractor




CNS−GBM
CNS−Medullo
CNS−Oligo
CNS−PiloAstro
ColoRect−AdenoCA
Eso−AdenoCA
Head−SCC
Kidney−ChRCC
Kidney−RCC.clearcell
Kidney−RCC.papillary
Liver−HCC
Lung−AdenoCA
sig1
sig2


Lung−SCC
sig2
sig1
0
10000
25000
Lymph−BNHL
0
−1.0
−0.5
0.0
0.5
1.0
Lymph−CLL
500
1500
Myeloid−AML
Myeloid−MPN
50
150
200
600
1000
Ovary−AdenoCA
4000
5000
10000
15000
Panc−AdenoCA
Panc−Endocrine
Prost−AdenoCA
1000
3000
Skin−Melanoma.acral
2000
4000
8000
Skin−Melanoma.cutaneous
0e+00
2e+05
3e+05
4e+05
6e+05
Skin−Melanoma.mucosal



Correlation of p-values and number of samples

```
pcawg_palette <- pcawg.colour.palette(gsub("\\..*", "", all_pvals$ct),
                                      scheme = "tumour.subtype")

names(pcawg_palette) <- all_pvals$ct

ggplot(data.frame(num_samples=num_samples[match(all_pvals$ct, names(num_samples))],
                  pvalue=all_pvals$pvalue,
                  ct=all_pvals$ct), aes(x=num_samples, y=pvalue, label=ct, col=ct, group=1))+
  geom_point()+scale_y_continuous(trans = "log2")+
  geom_smooth(method = lm)+theme_bw()+
  geom_label_repel()+geom_hline(yintercept = log2(0.05), lty='dashed')+
  labs(x='Number of samples', y = 'p-value (log2')+
  scale_color_manual(values = pcawg_palette)+
  guides(col=FALSE)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

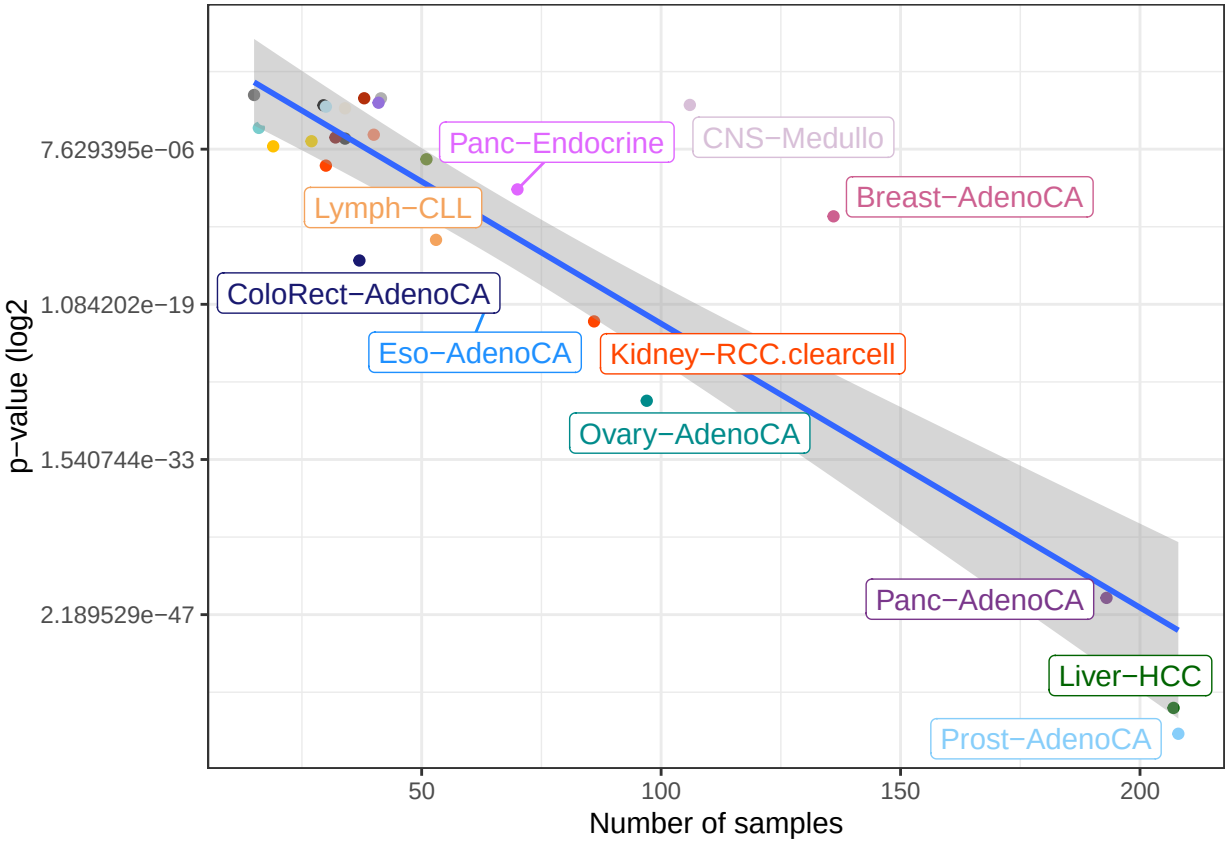
```
## Warning in log(x, base): NaNs produced
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
## Warning: Removed 1 rows containing missing values (geom_label_repel).
## Warning: Removed 1 rows containing missing values (geom_hline).
## Warning: ggrepel: 16 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Effect size

We are dealing with multivariate data, and it is not clear how we could come up with a metric to show the change between two sets of paired mutational signatures.

- From the Aitchison school, the 'perturbation' is what is used, and is a $d-1$-dimensional vector. We must turn this value into a single value.

$$e_1 = \frac{\sum_{i=1}^{n} (\bar{W}_i^l - \bar{W}_i^e)^2}{n}$$

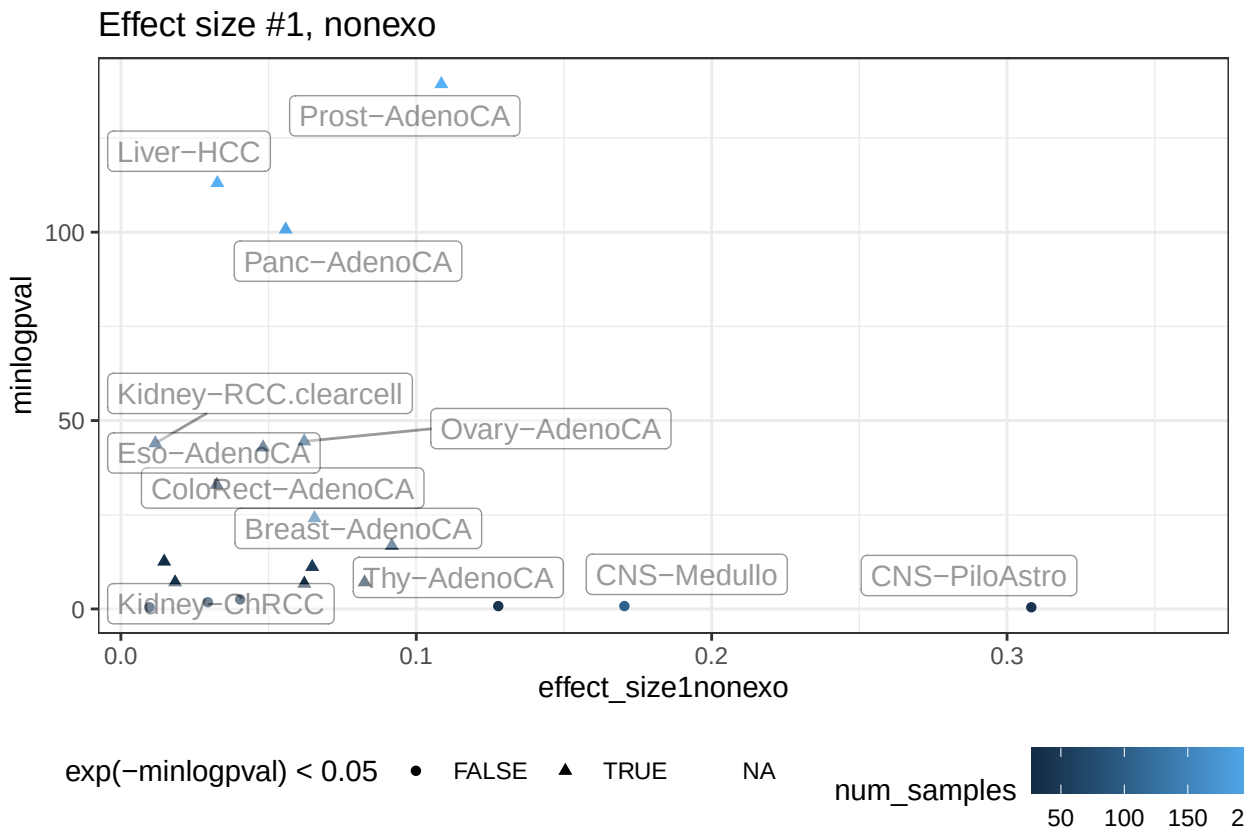
```
## [1] 27  8
## [1] 27  8
## [1] 136  12
## [1] 136  12
## [1] 34  5
## [1] 34  5
```

```
## [1] 106   6
## [1] 106   6
## [1] 41  5
## [1] 41  5
## [1] 37  8
## [1] 37  8
## [1] 65  9
## [1] 65  9
## [1] 32  8
## [1] 32  8
## [1] 38  5
## [1] 38  5
## [1] 86  7
## [1] 86  7
## [1] 30  7
## [1] 30  7
## [1] 207  13
## [1] 207  13
## [1] 34  4
## [1] 34  4
## [1] 51 12
## [1] 51 12
## [1] 53  3
## [1] 53  3
## [1] 97 10
## [1] 97 10
## [1] 193  13
## [1] 193  13
## [1] 70 11
## [1] 70 11
## [1] 208  12
## [1] 208  12
## [1] 29  3
## [1] 29  3
## [1] 30 14
## [1] 30 14
## [1] 41  5
## [1] 41  5
## [1] 40 10
## [1] 40 10
```

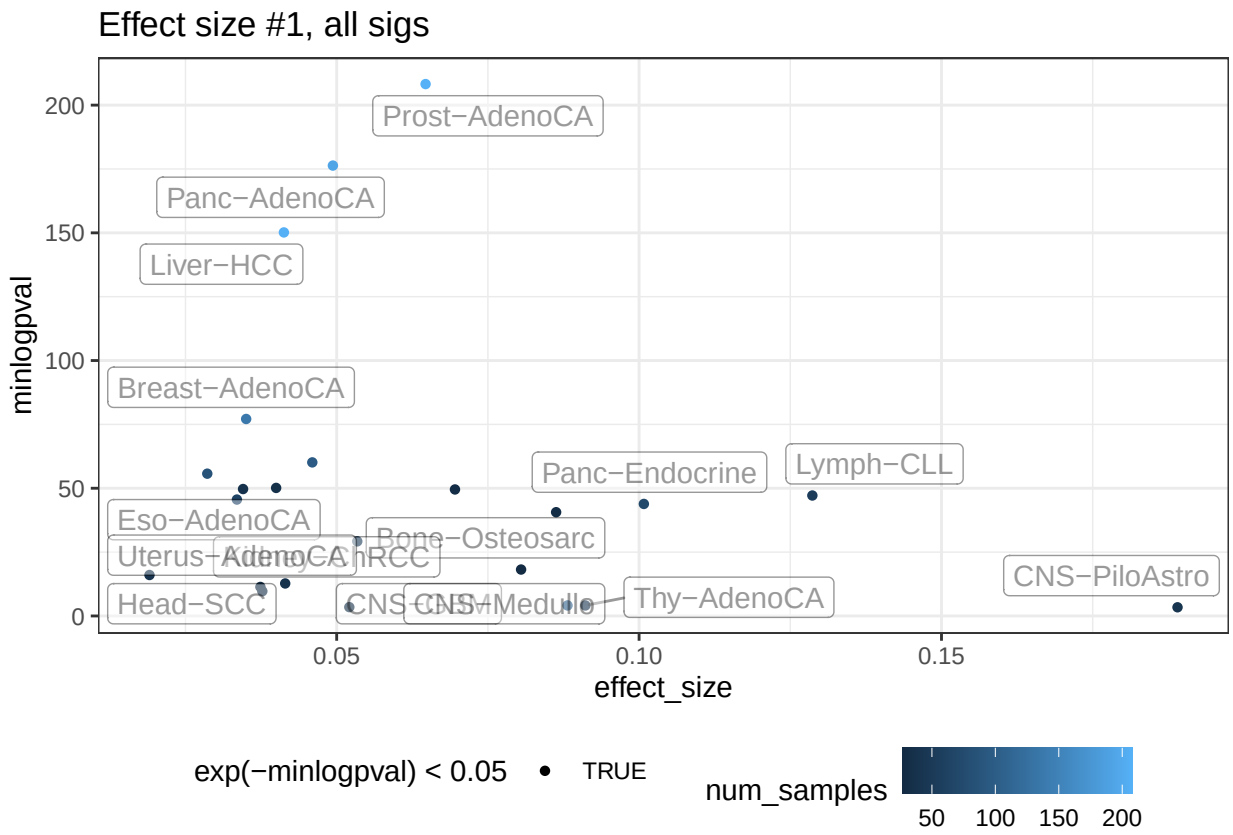
```
## Warning: Removed 3 rows containing missing values (geom_point).
```

```
## Warning: Removed 3 rows containing missing values (geom_label_repel).
```

```
## Warning: ggrepel: 8 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

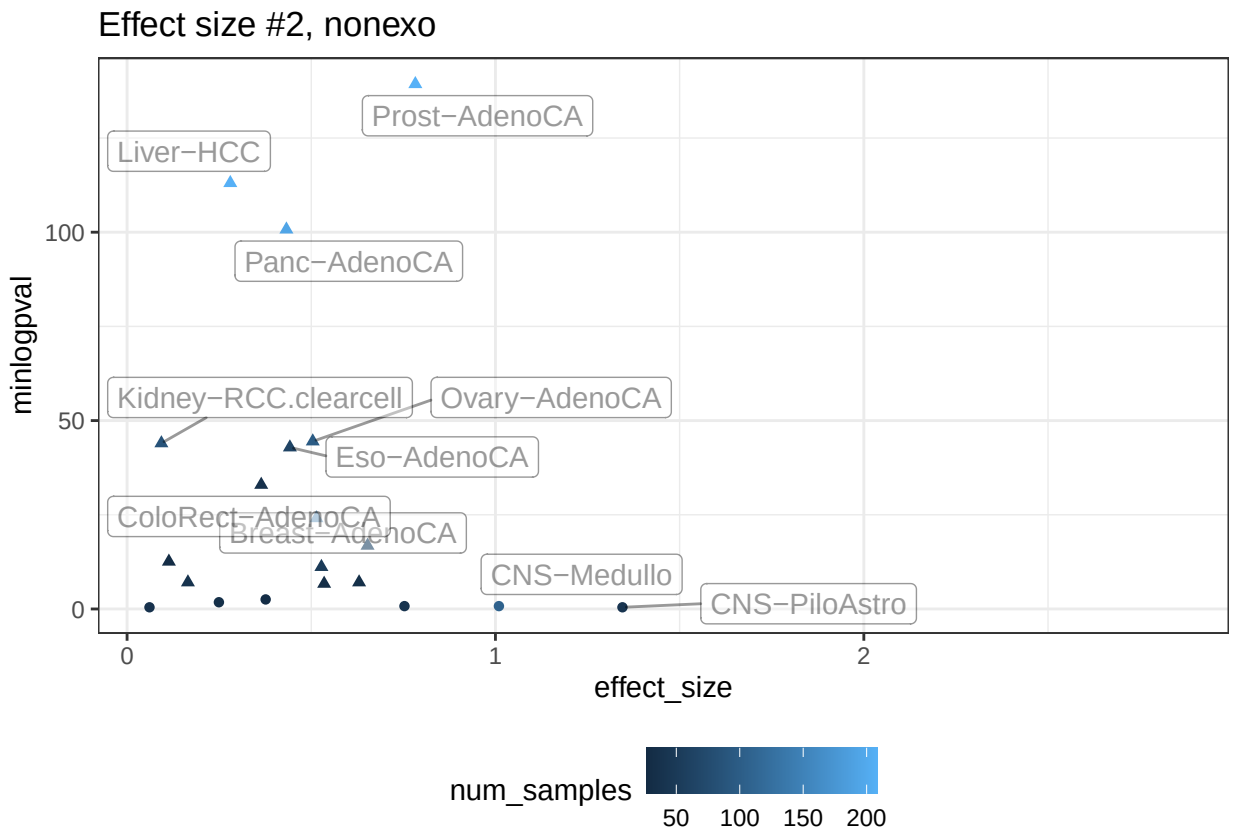



```
## Warning: ggrepel: 8 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



$$e_2 = \frac{\sum_{i=1}^{n}(\bar{W}_i^l - \bar{W}_i^e)^2}{n} \cdot \log\left(\frac{\sum_{i=1}^{n}\sum_{j=1}^{k} W_{ij}}{n}\right)$$

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: Removed 3 rows containing missing values (geom_point).

## Warning: Removed 3 rows containing missing values (geom_label_repel).

## Warning: ggrepel: 10 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Signature-averaged perturbation

```
## Error in apply(pert, 1, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
## Error in apply(pert, 1, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
```

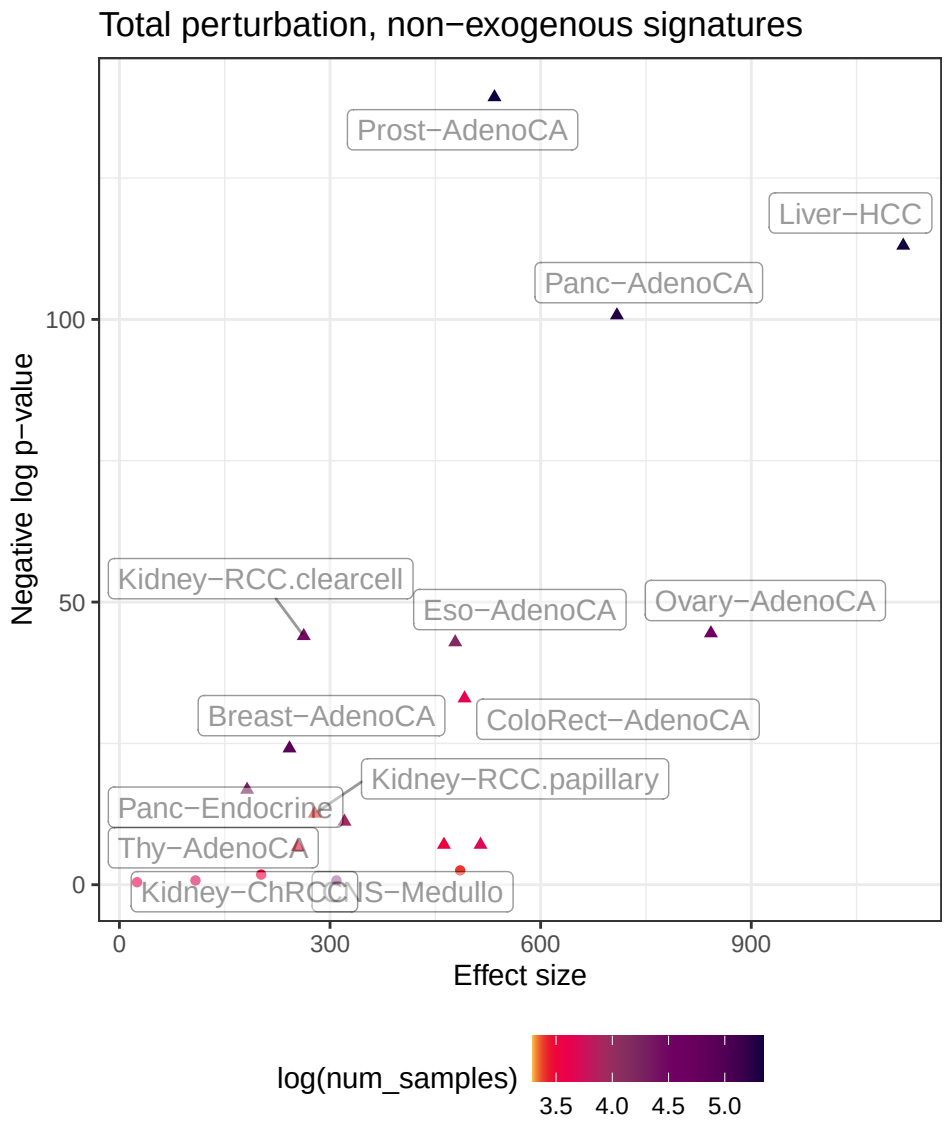
```
## Warning: NAs introduced by coercion
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```

```
## Warning: Removed 4 rows containing missing values (geom_label_repel).
```

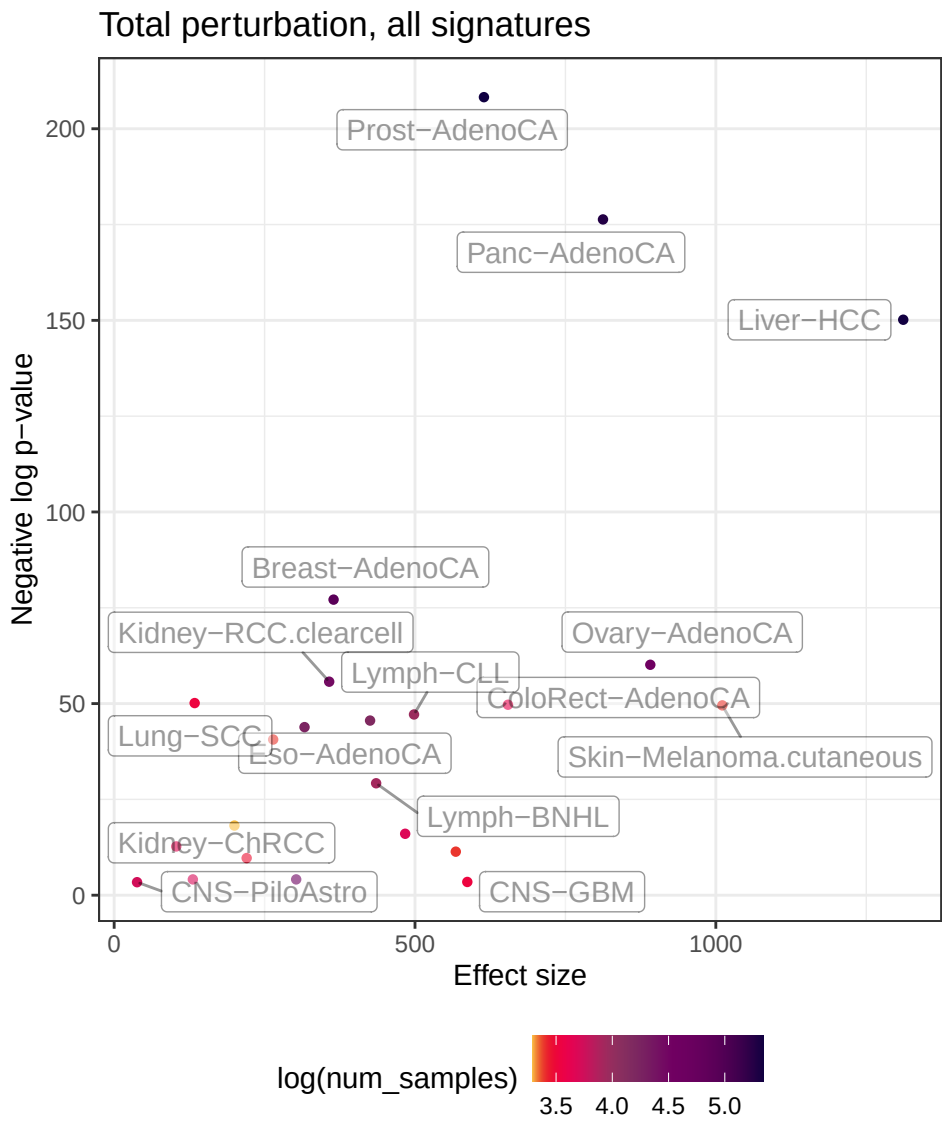
```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Signature-averaged perturbation (all sigs)

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: ggrepel: 8 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Total perturbation (sample-averaged)

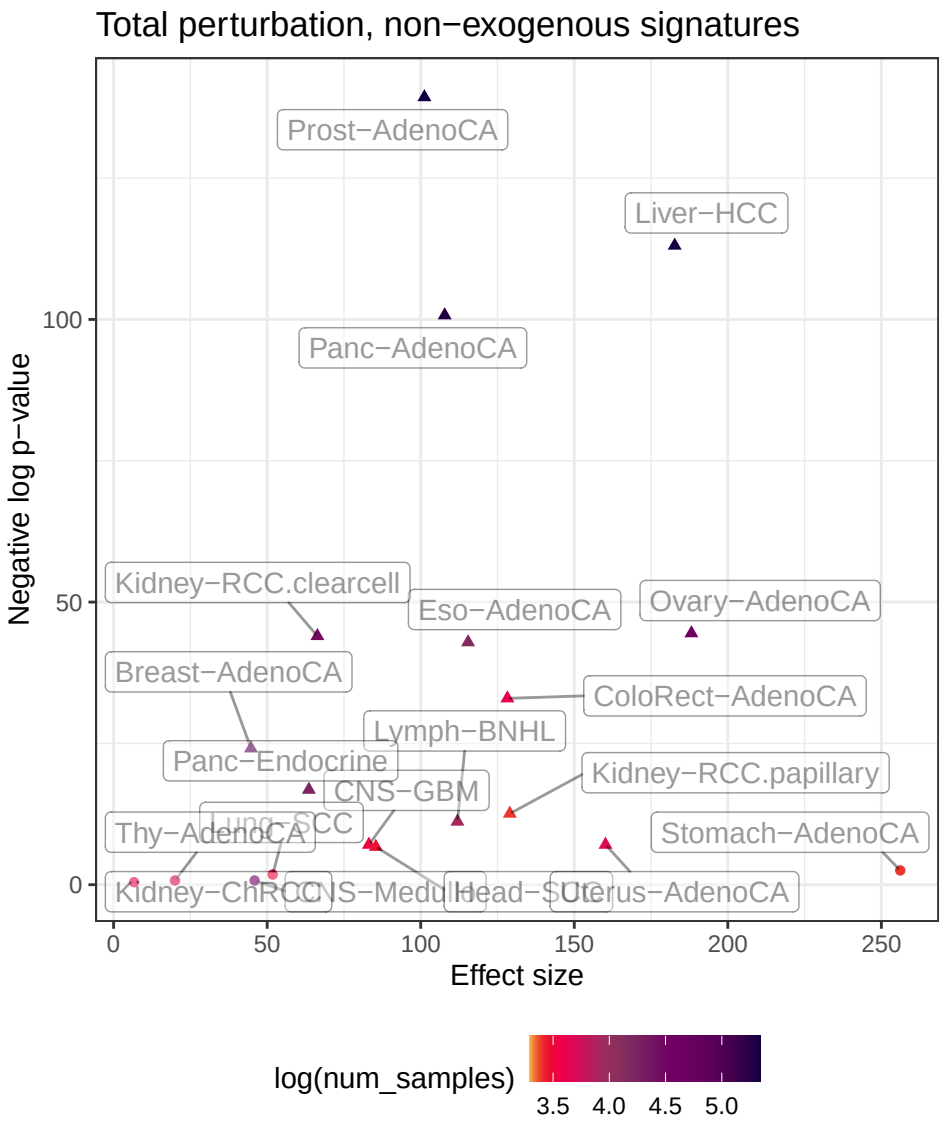
```
## Error in apply(pert, 2, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
## Error in apply(pert, 2, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
```

```
## Warning: NAs introduced by coercion
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```

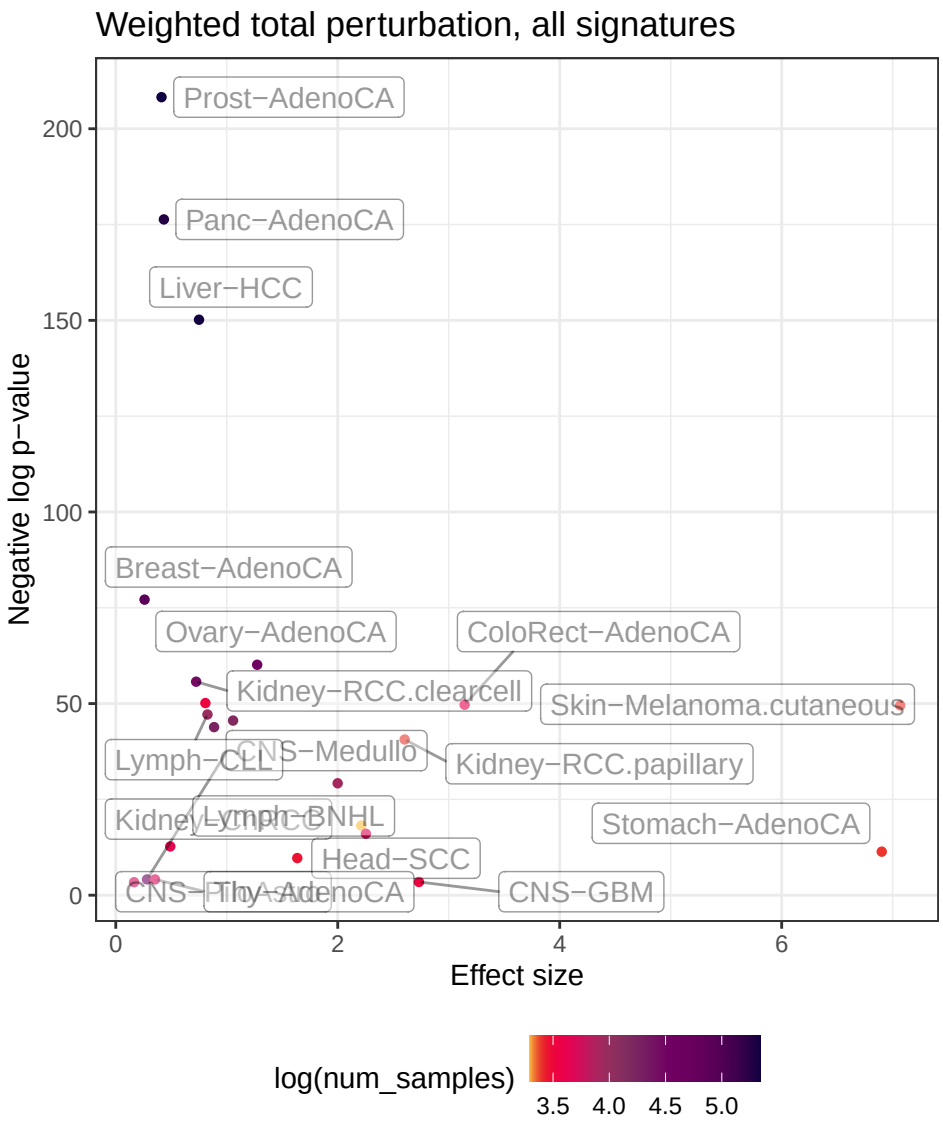
```
## Warning: Removed 4 rows containing missing values (geom_label_repel).
```



Weighted total perturbation

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: ggrepel: 5 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Which inferred covariance matrices are not positive semi-definite?

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
```

```
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
```

```
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
```

```
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
```

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
```

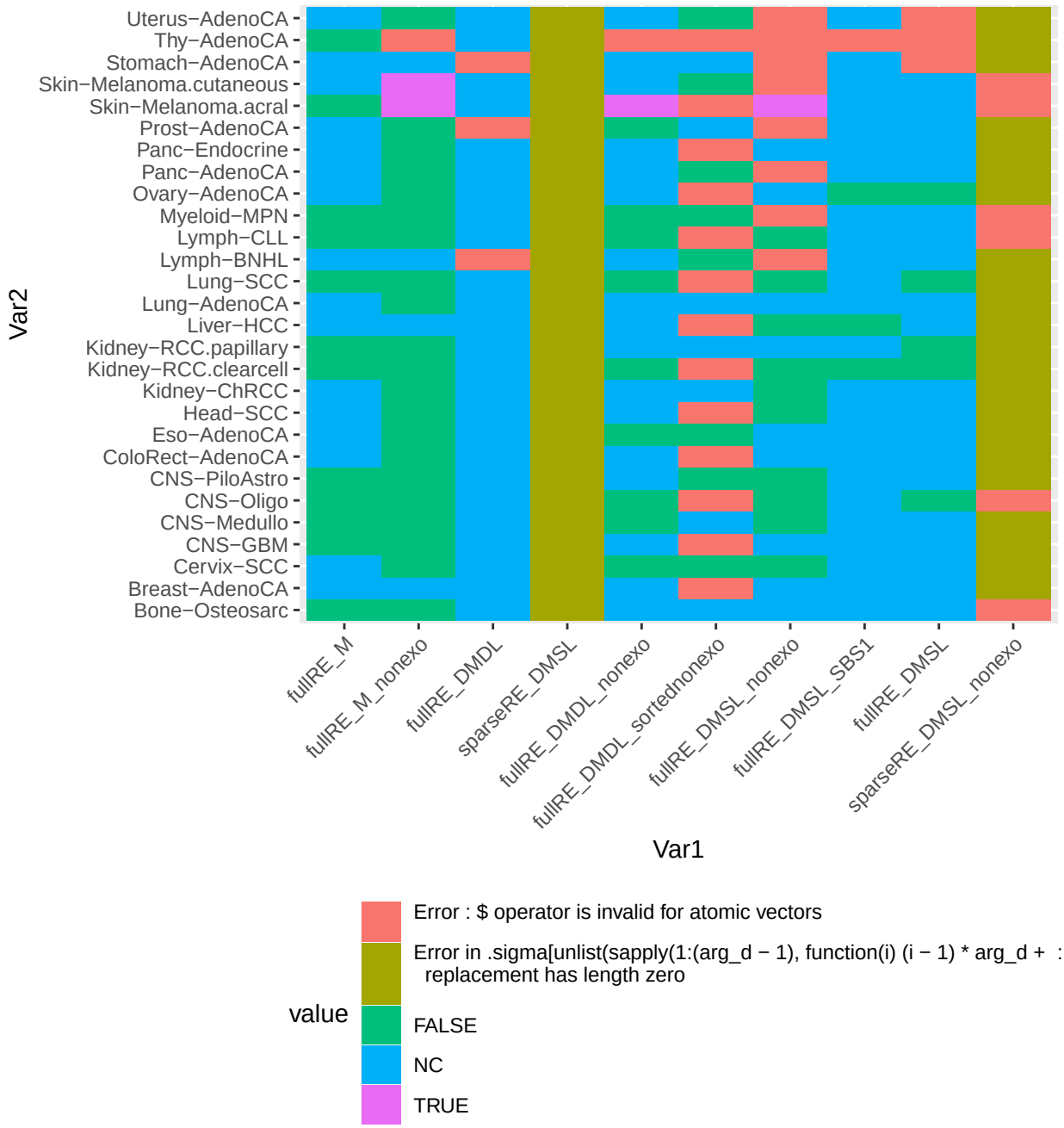
```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors
```

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
## Error : $ operator is invalid for atomic vectors
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

```
## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

```
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero
## Error : $ operator is invalid for atomic vectors

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)
```

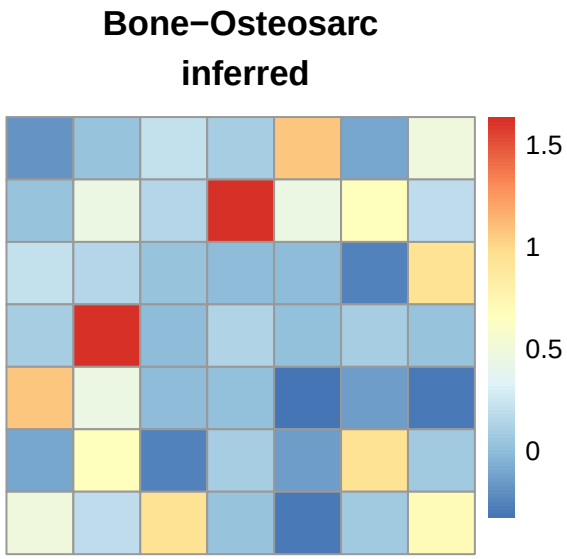
```
## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## Warning in fill_covariance_matrix(arg_d = length(python_like_select_name(get(i)
## [[ct]]$par.fixed, : This function had been incorrect until now (30 july 2021)

## Error in m[unlist(sapply(2:nrow(m), function(rw) seq(from = rw, length.out = (rw -  :
##   replacement has length zero

## [1] 11 28

## [1] 308   3

## [1] "Var1"  "Var2"  "value"
```



Comparison of the inferred covariance matrices and the matrices from the fitted intercepts

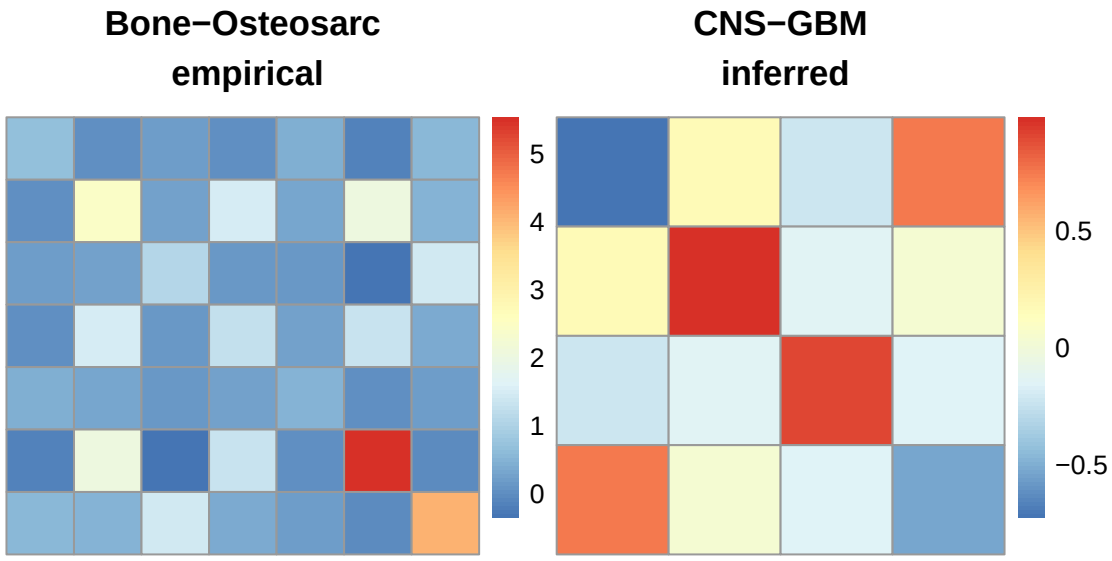
```
## [1] "Bone-Osteosarc"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



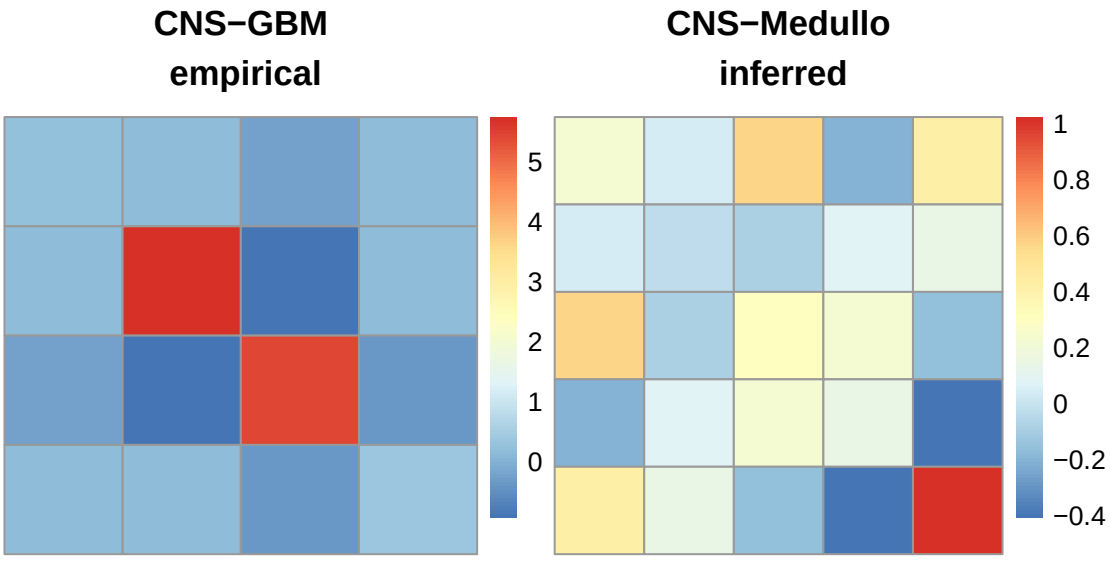

```
## [1] "CNS-GBM"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



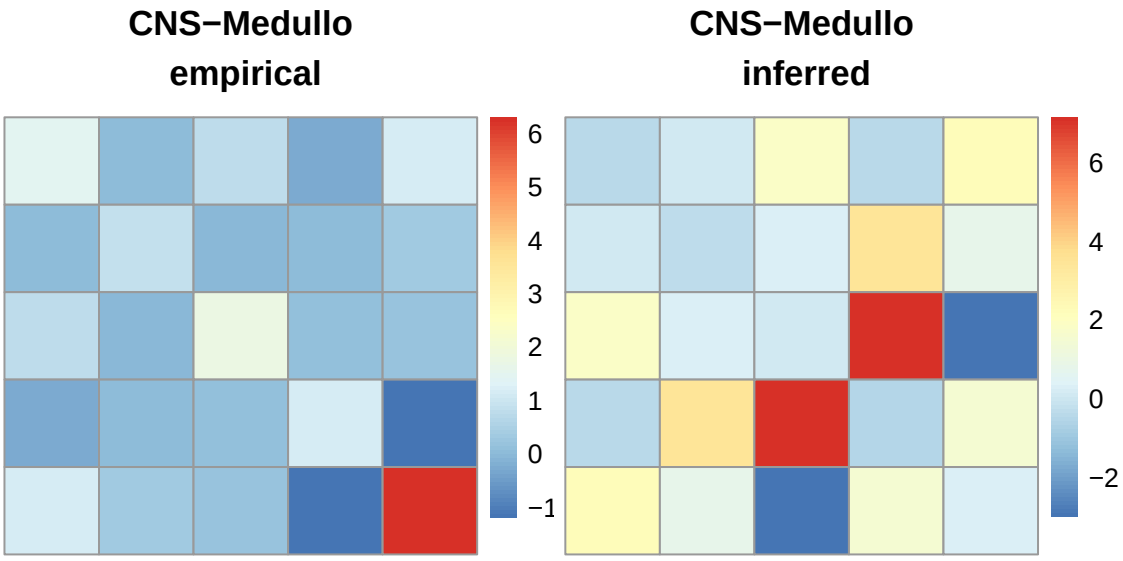

```
## [1] "CNS-Medullo"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



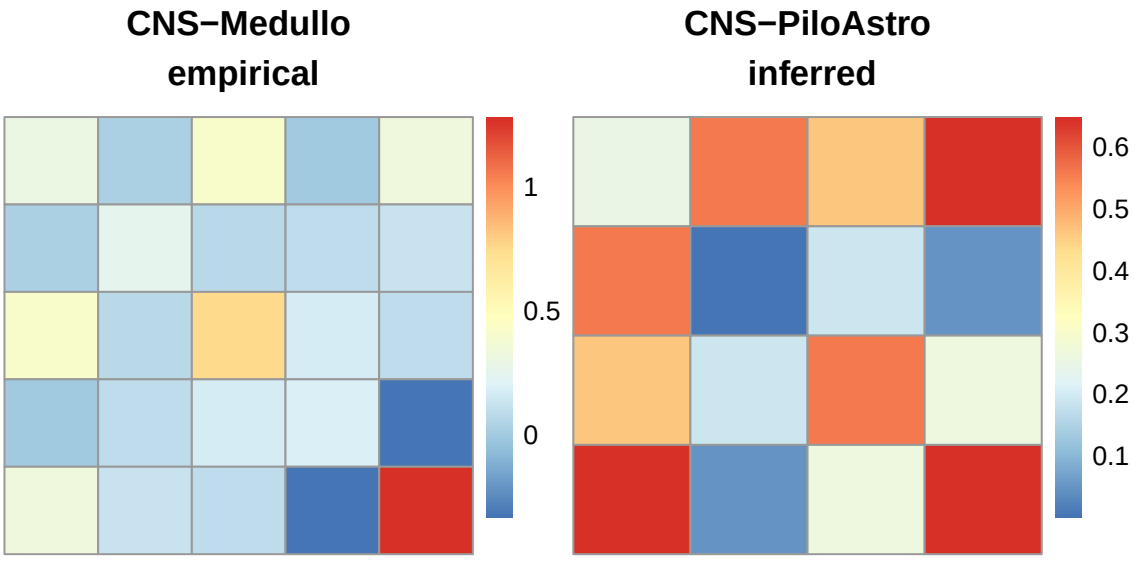

```
## [1] "CNS-Medullo"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



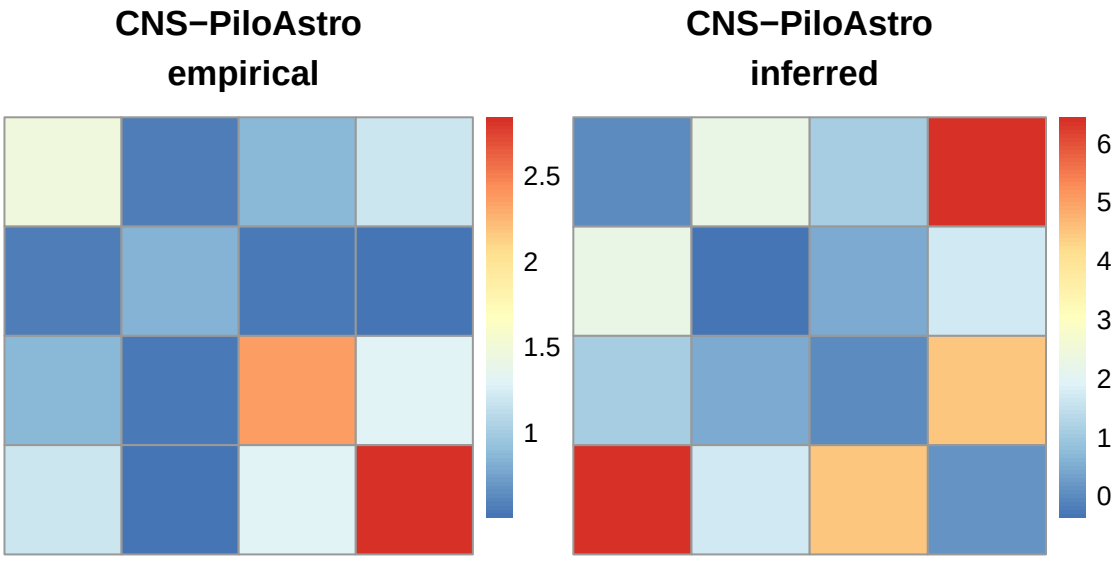

```
## [1] "CNS-PiloAstro"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



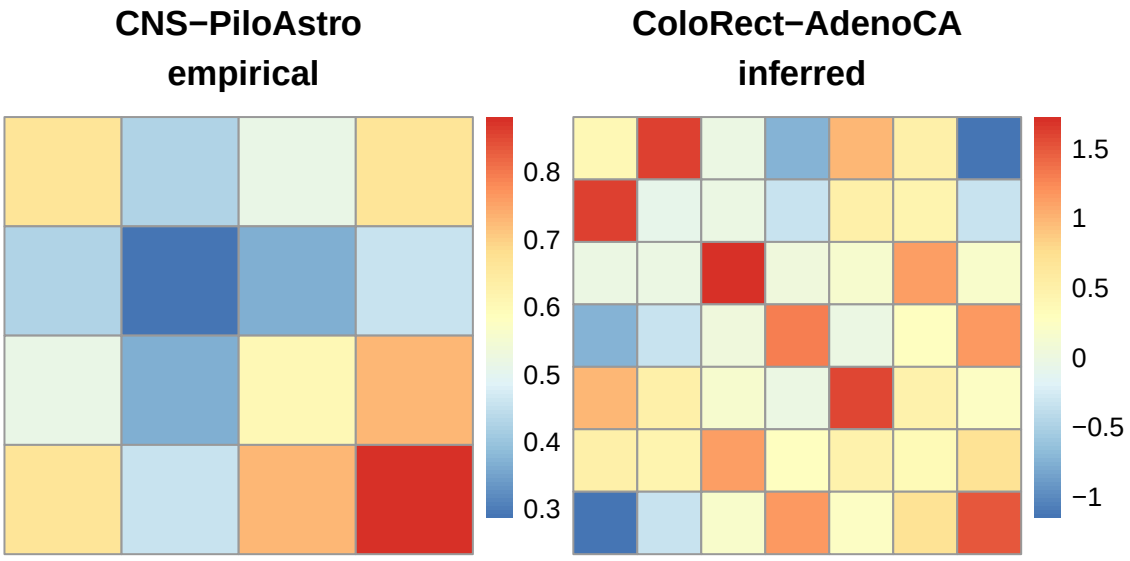

```
## [1] "CNS-PiloAstro"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



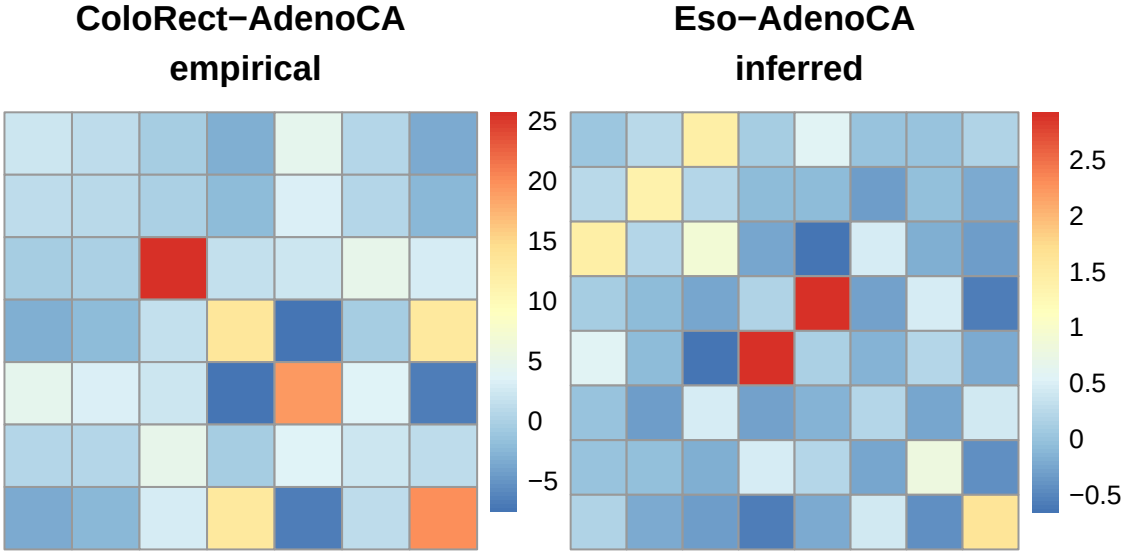

```
## [1] "ColoRect-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



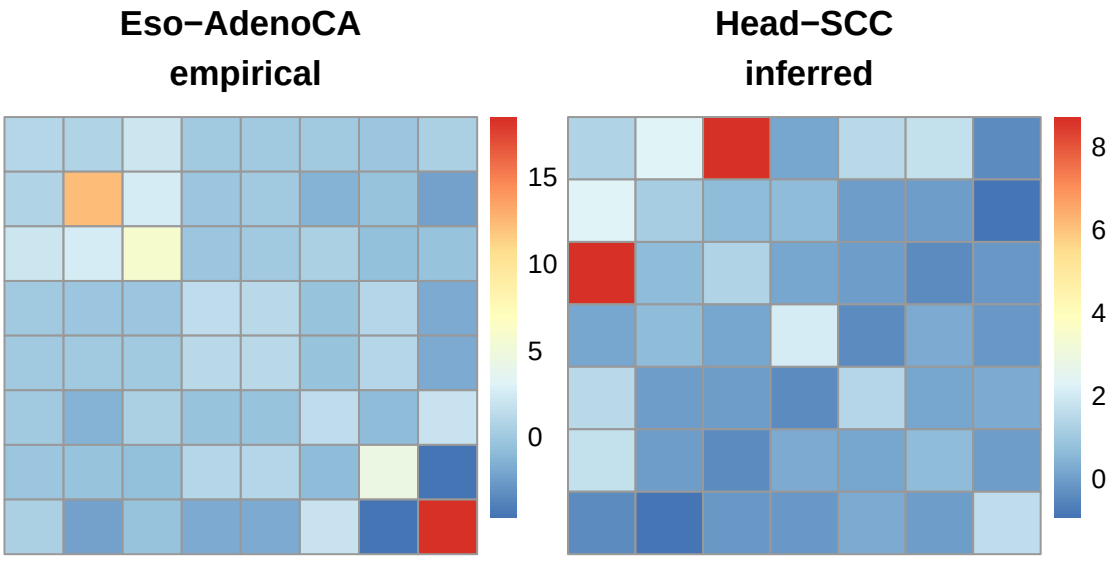

```
## [1] "Eso-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



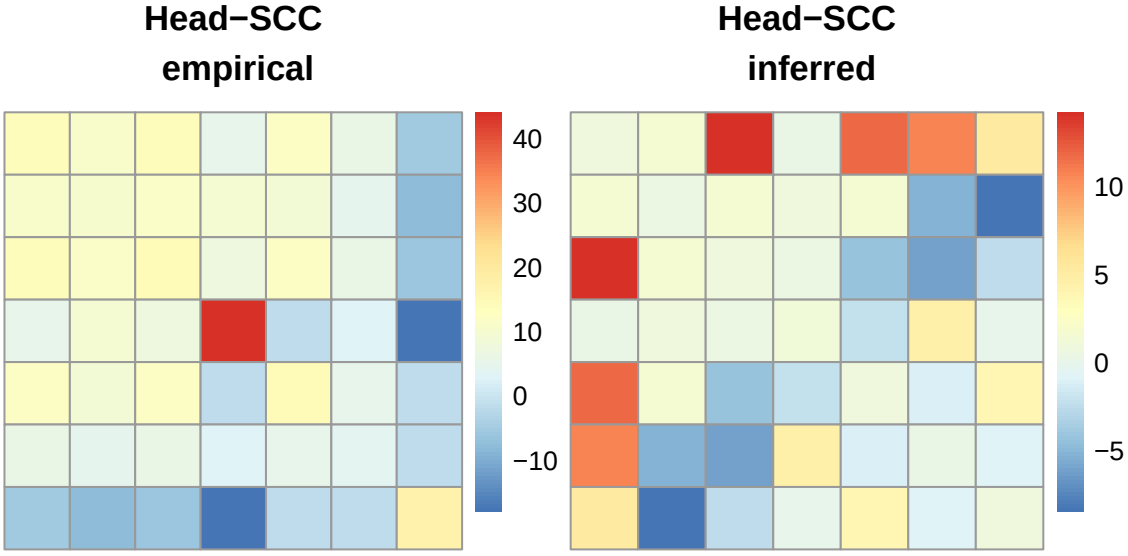

```
## [1] "Head-SCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



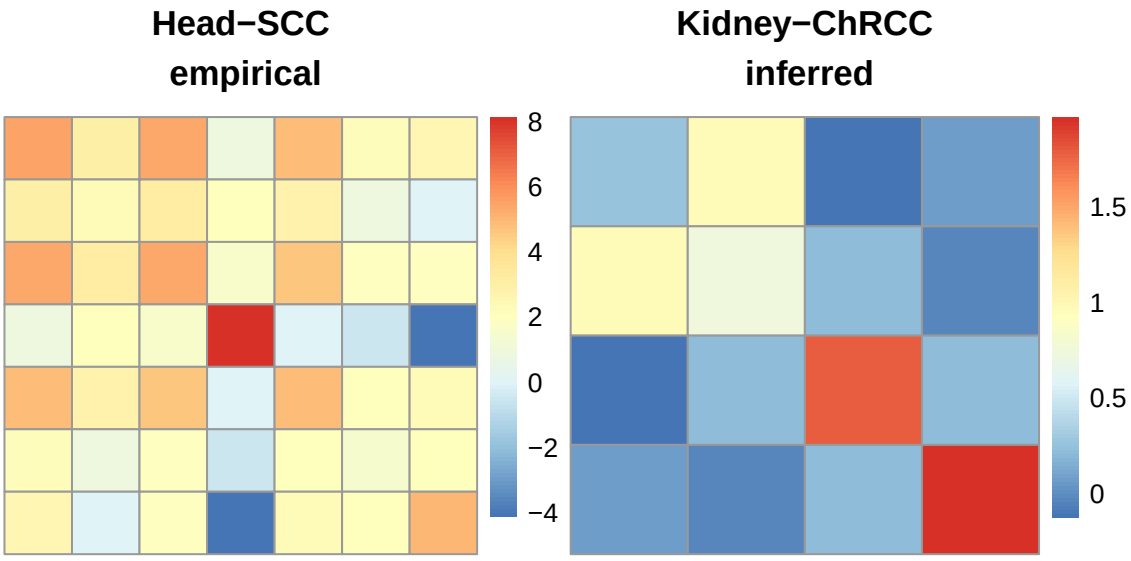

```
## [1] "Head-SCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



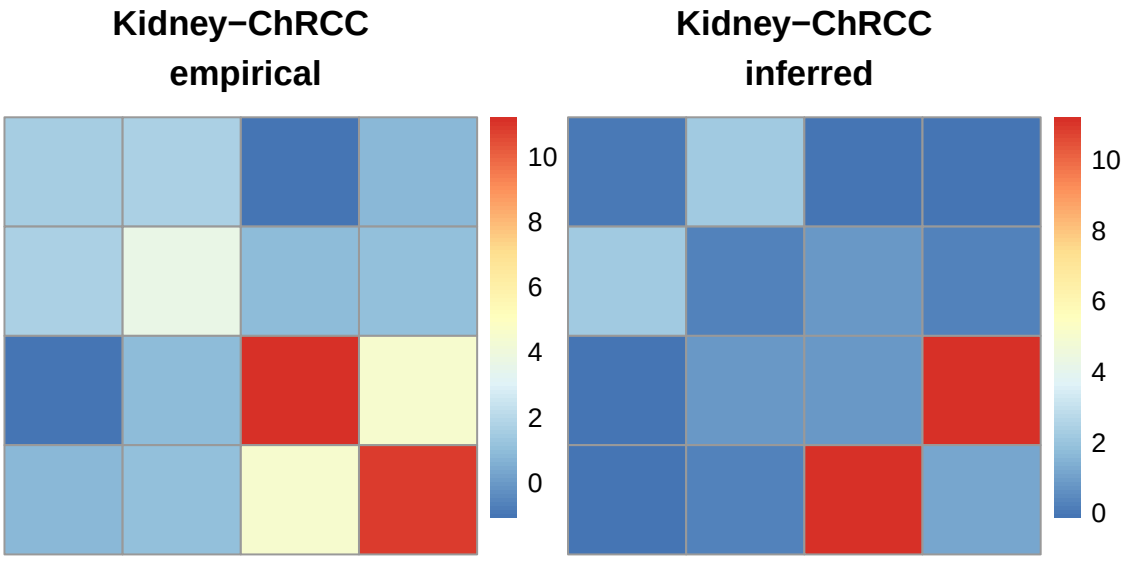

```
## [1] "Kidney-ChRCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



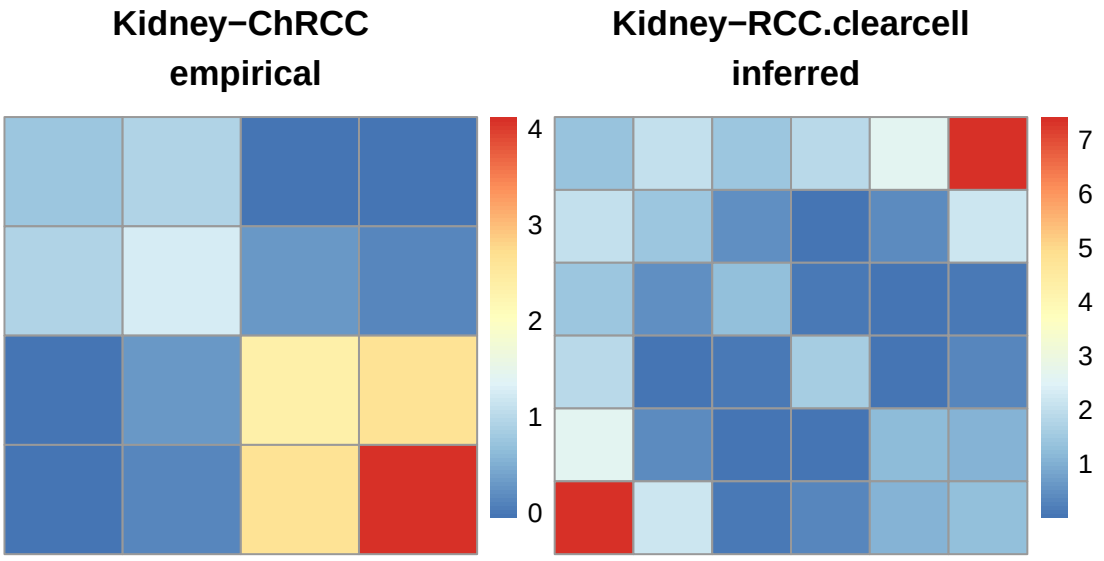

```
## [1] "Kidney-ChRCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



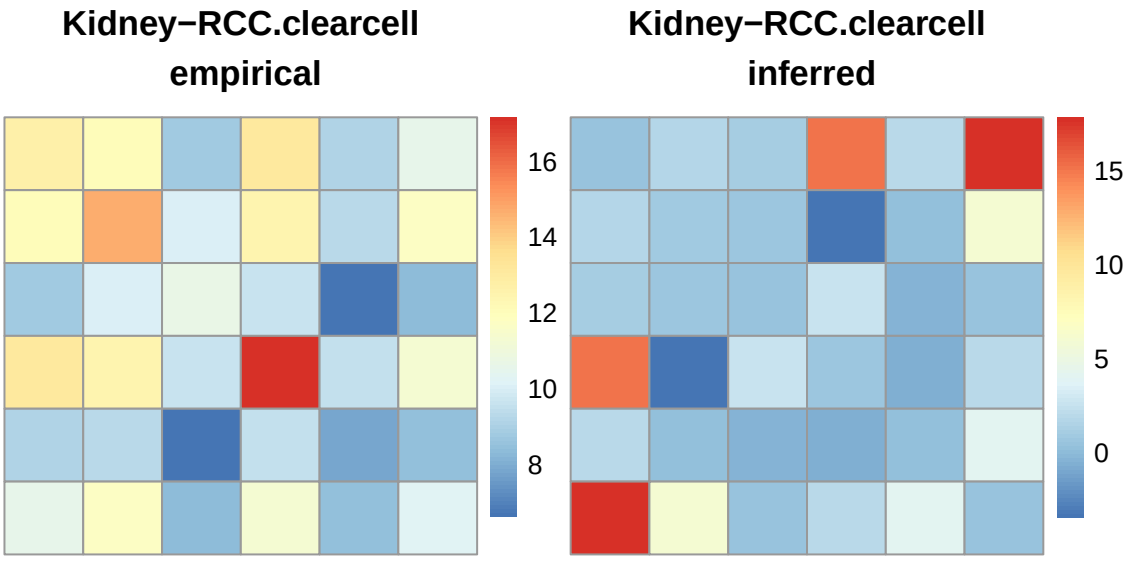

```
## [1] "Kidney-RCC.clearcell"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



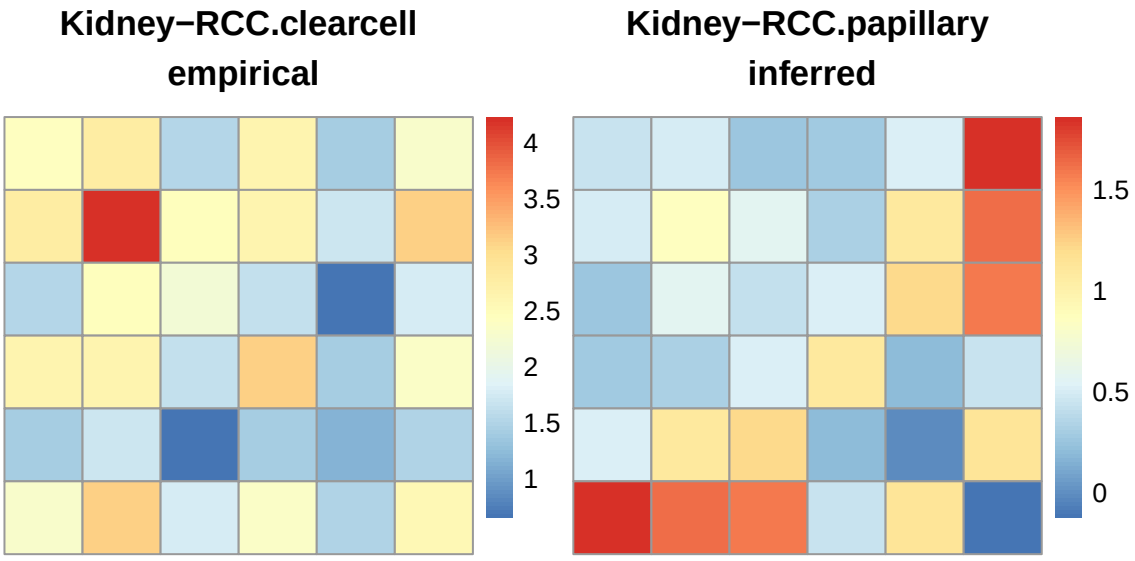

```
## [1] "Kidney-RCC.clearcell"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



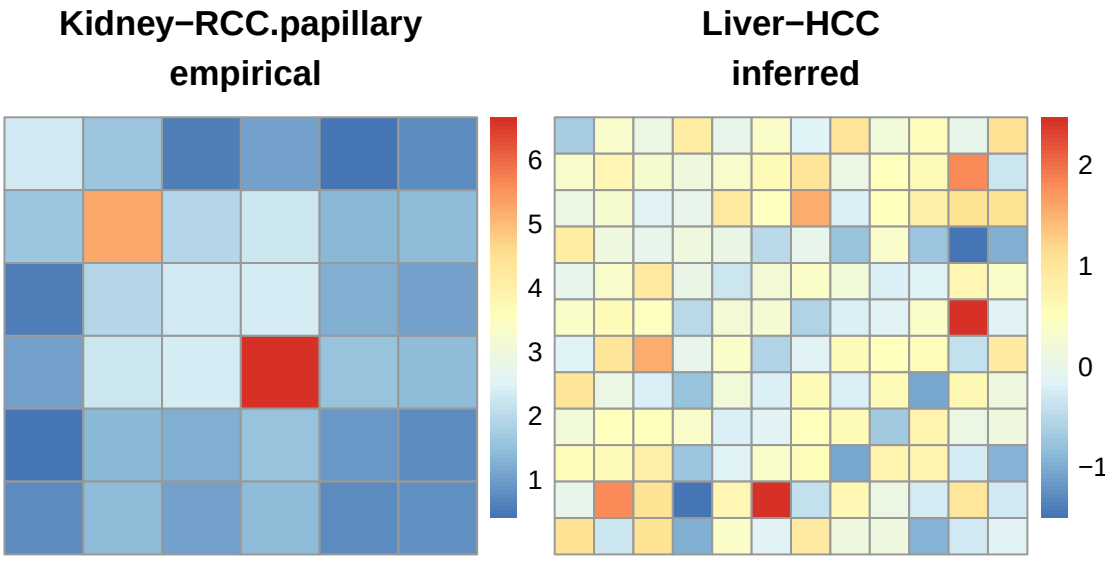

```
## [1] "Kidney-RCC.papillary"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



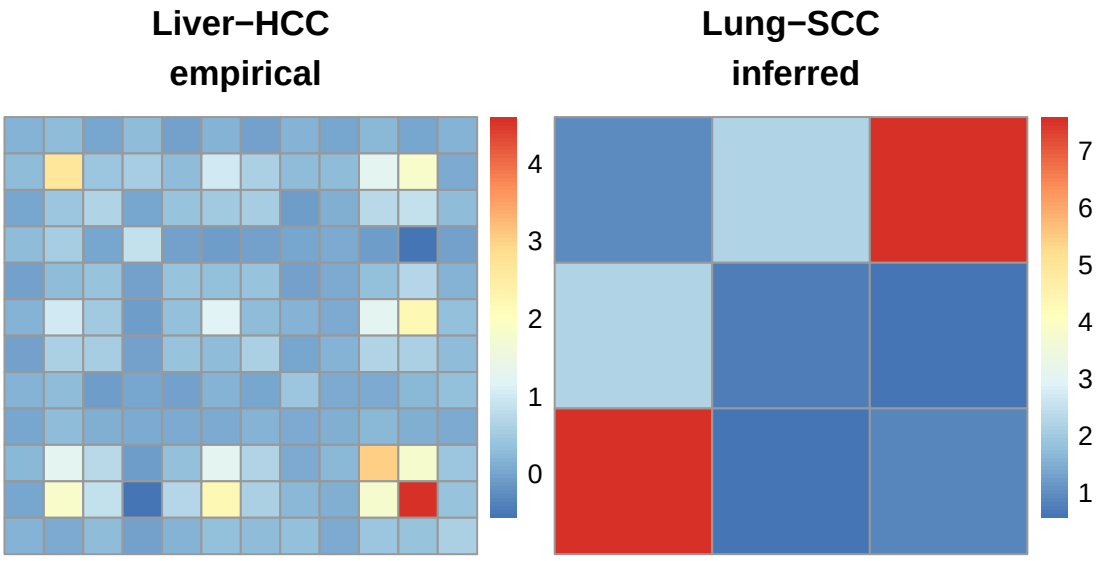

```
## [1] "Liver-HCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



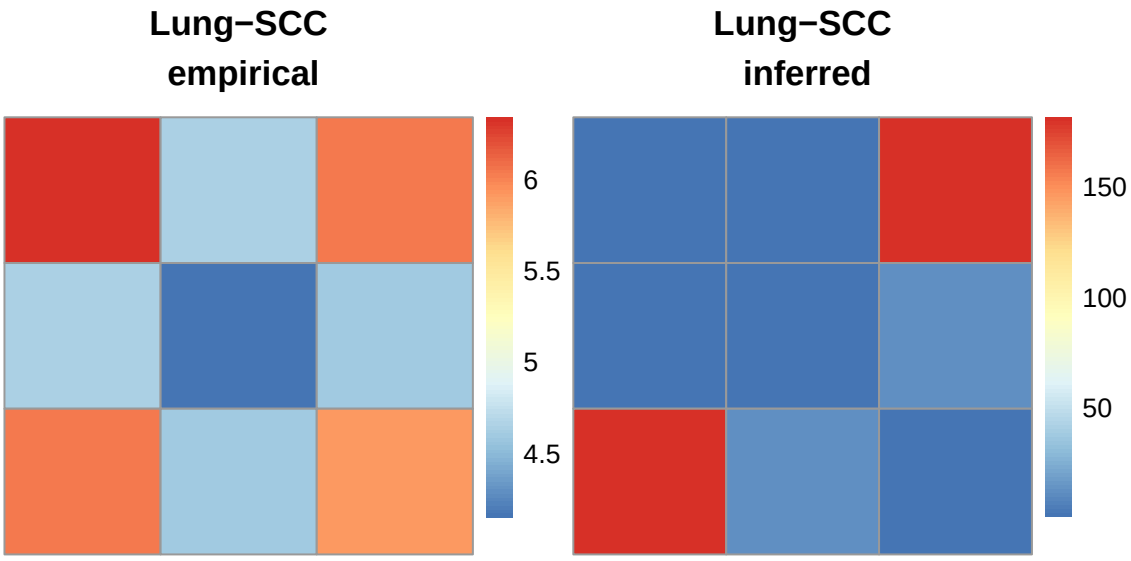

```
## [1] "Lung-SCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



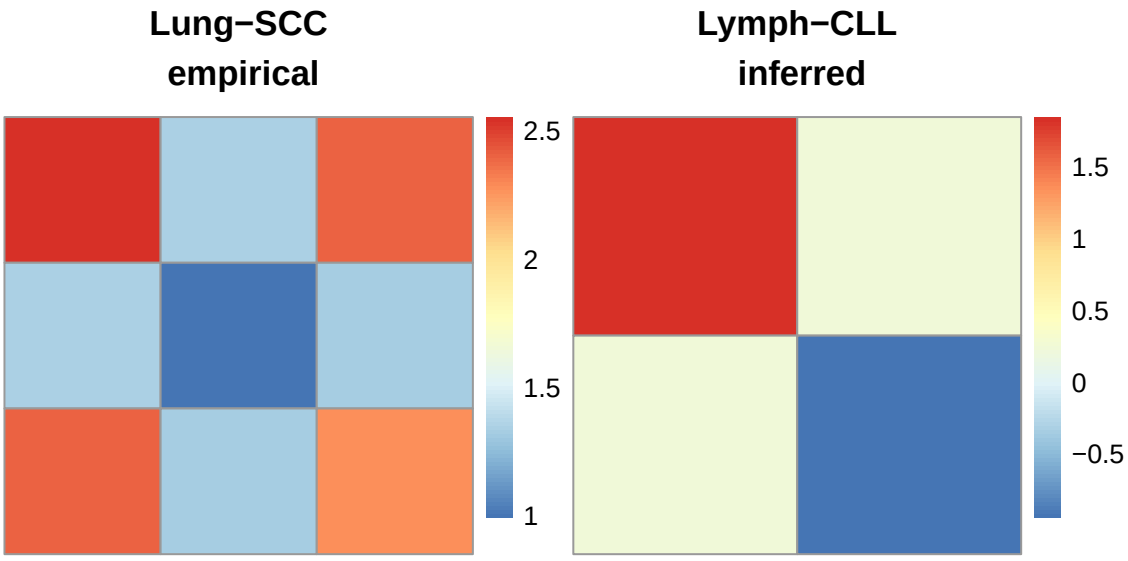

```
## [1] "Lung-SCC"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



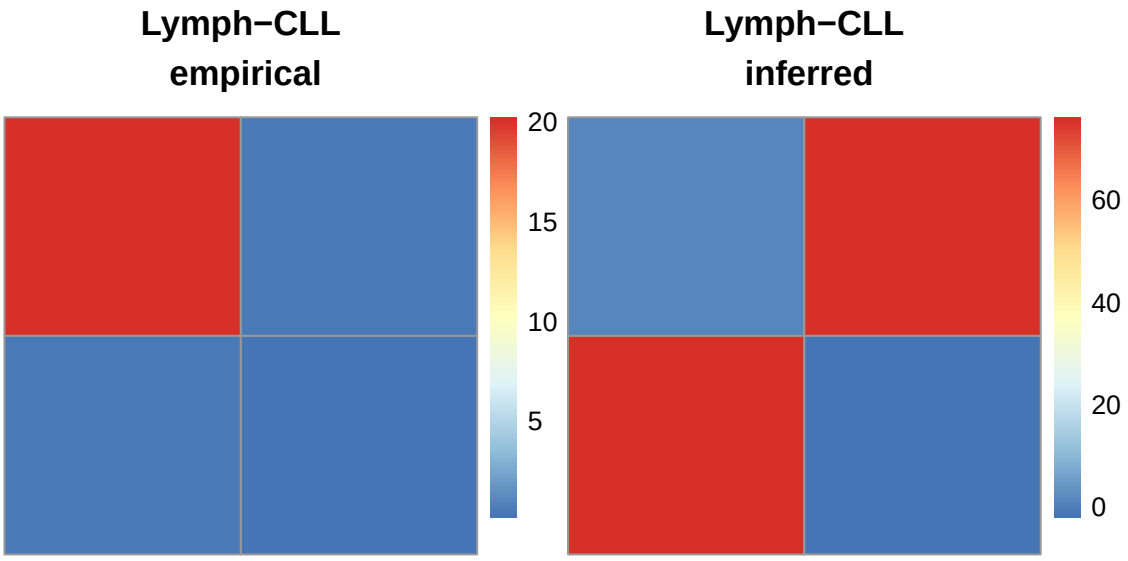

```
## Error : $ operator is invalid for atomic vectors
## [1] "Lymph-CLL"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



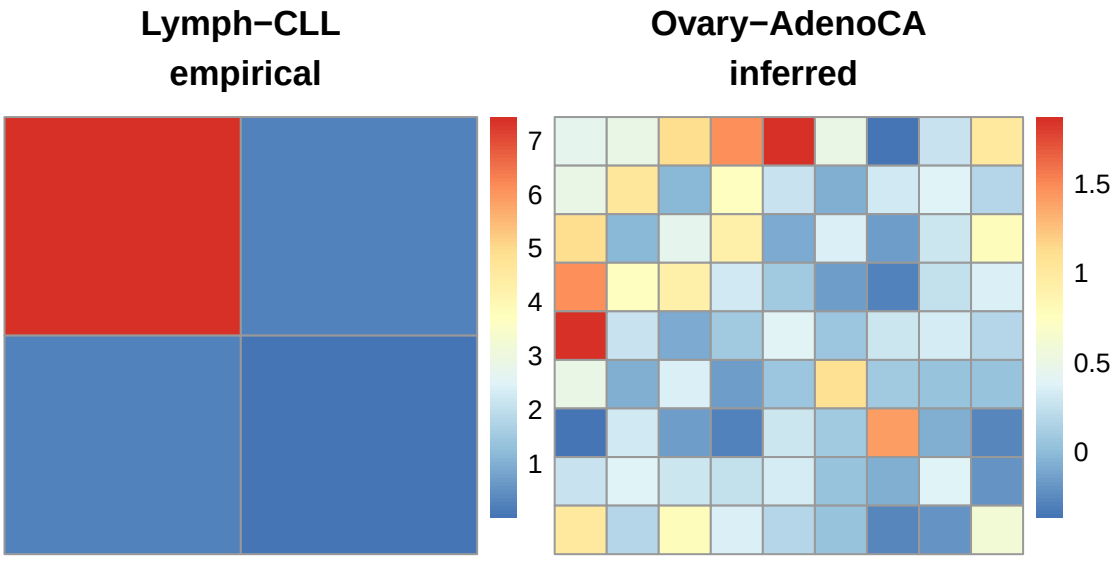

```
## [1] "Lymph-CLL"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



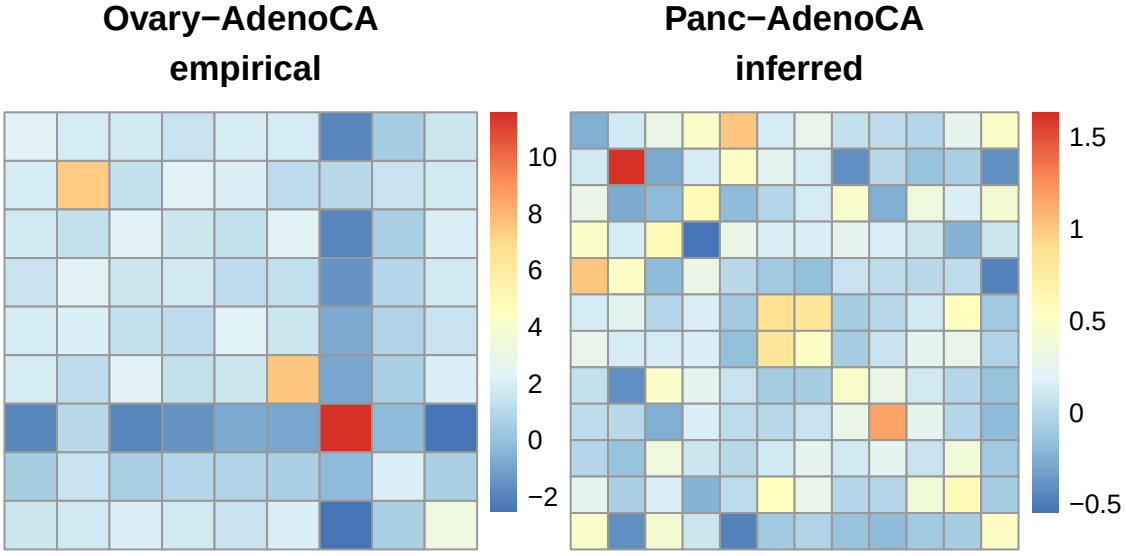

```
## [1] "Ovary-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



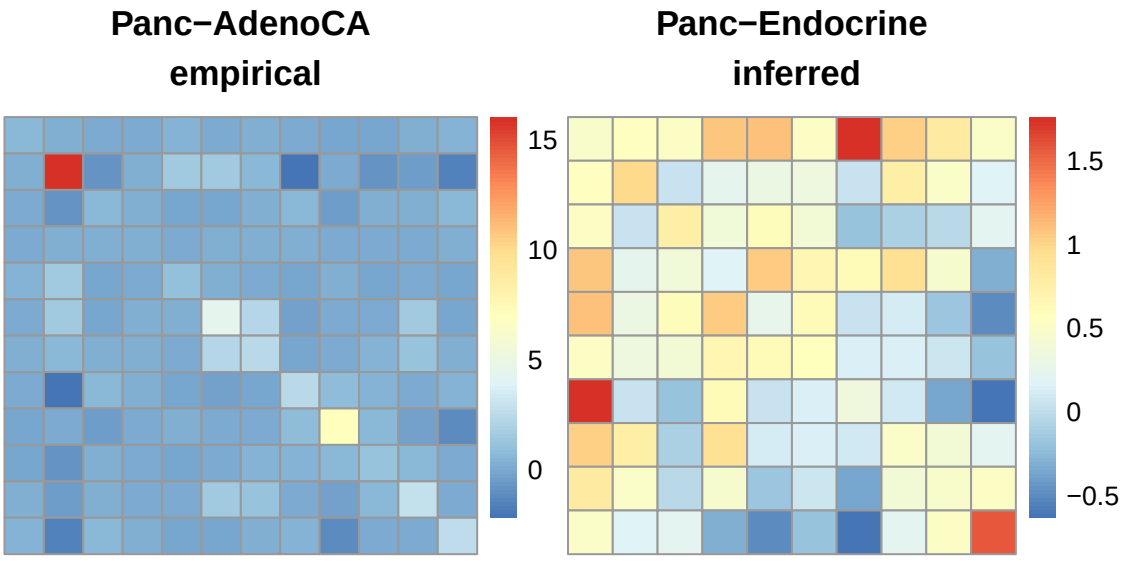

```
## [1] "Panc-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



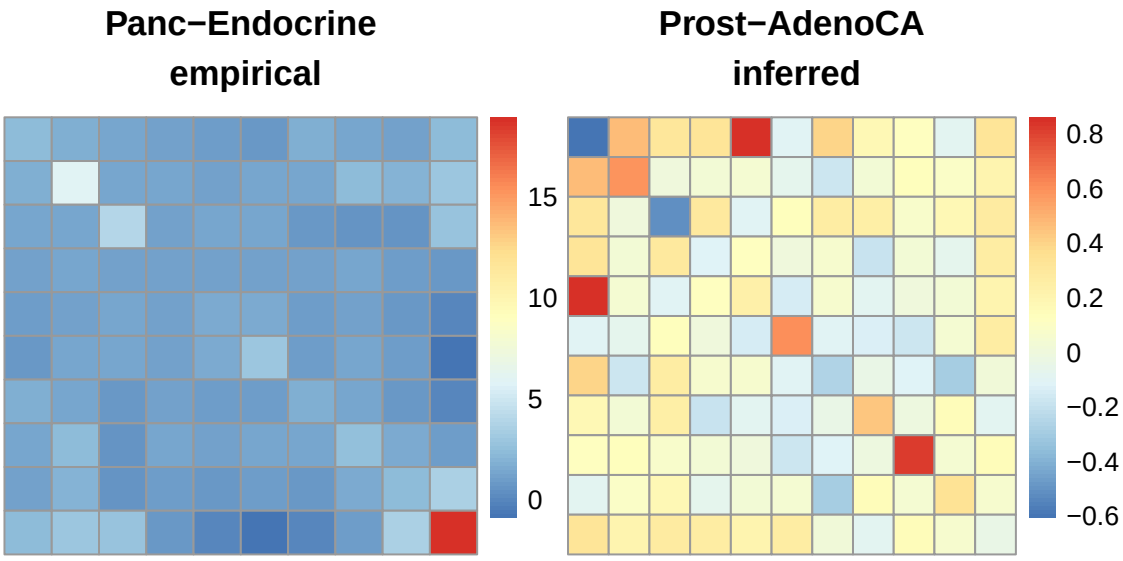

```
## Error : $ operator is invalid for atomic vectors
## [1] "Panc-Endocrine"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



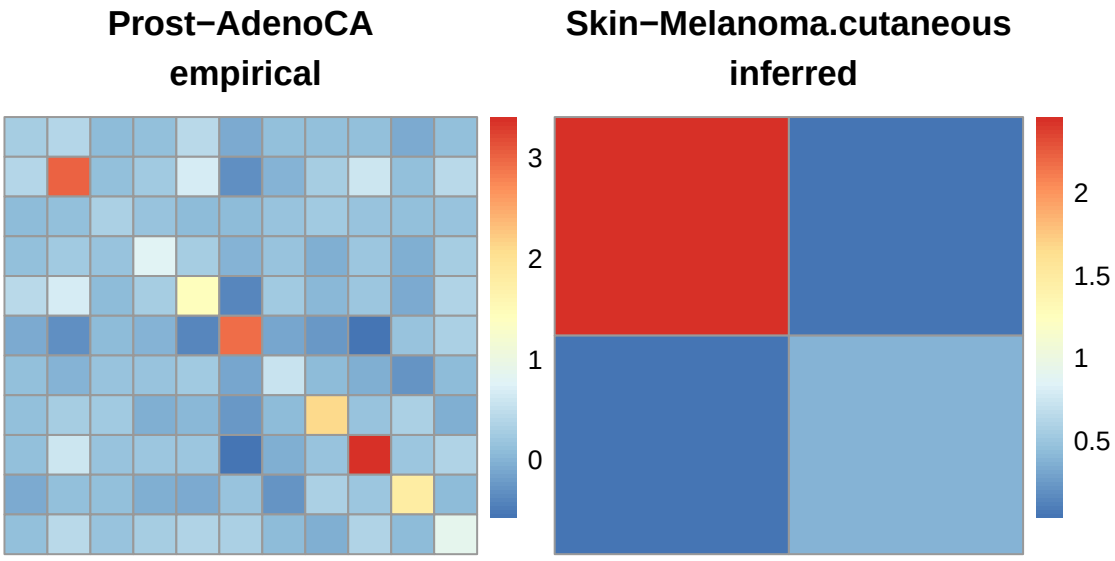

```
## [1] "Prost-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```



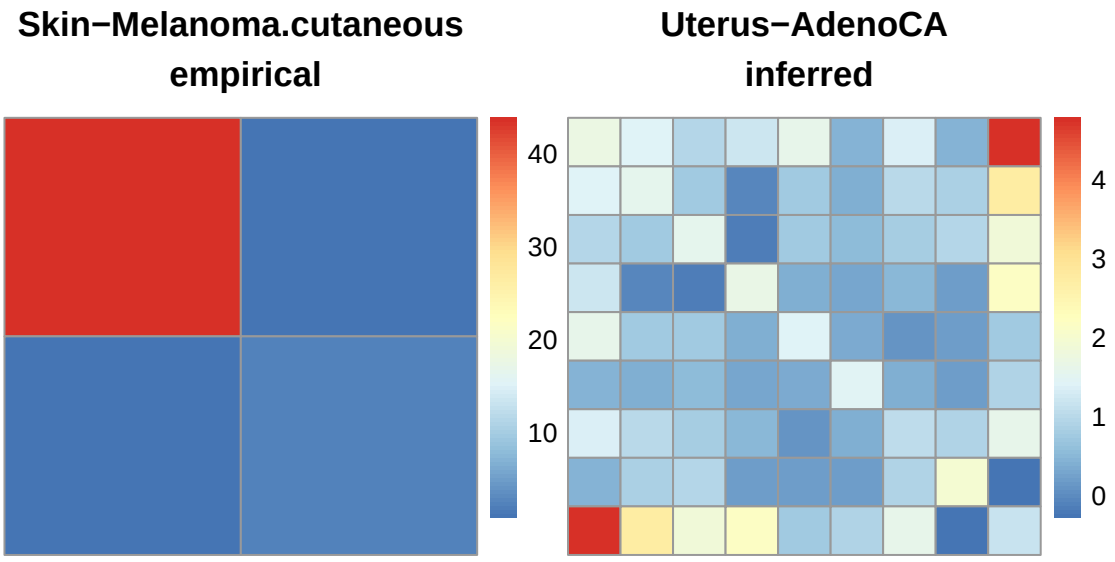

```
## Error : $ operator is invalid for atomic vectors
## [1] "Skin-Melanoma.cutaneous"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```

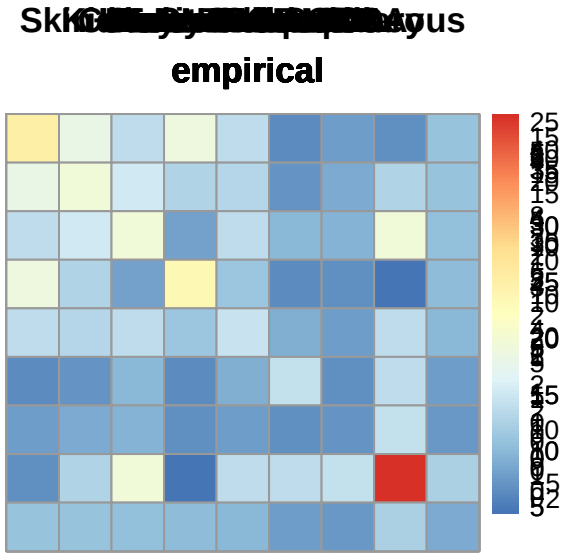



```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## [1] "Uterus-AdenoCA"
```

```
## Warning in fill_covariance_matrix(arg_d = .d_logR, arg_entries_var =
## python_like_select_name(get(i)[[ct]]$par.fixed, : This function had been
## incorrect until now (30 july 2021)
```




```
## Error : $ operator is invalid for atomic vectors
```




```
## $`Bone-Osteosarc`
## $`Bone-Osteosarc`$fullRE_M_nonexo
##
## $`Bone-Osteosarc`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Breast-AdenoCA`
##    fullRE_M_nonexo fullRE_DMSL_nonexo
##   "No convergence"   "No convergence"
##
## $`CNS-GBM`
## $`CNS-GBM`$fullRE_M_nonexo
##
## $`CNS-GBM`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`CNS-Medullo`
##           fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA               NA
## tree_col NA               NA
## kmeans   NA               NA
## gtable   List,12          List,12
##
## $`CNS-PiloAstro`
##           fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA               NA
## tree_col NA               NA
## kmeans   NA               NA
## gtable   List,12          List,12
##
## $`ColoRect-AdenoCA`
## $`ColoRect-AdenoCA`$fullRE_M_nonexo
```

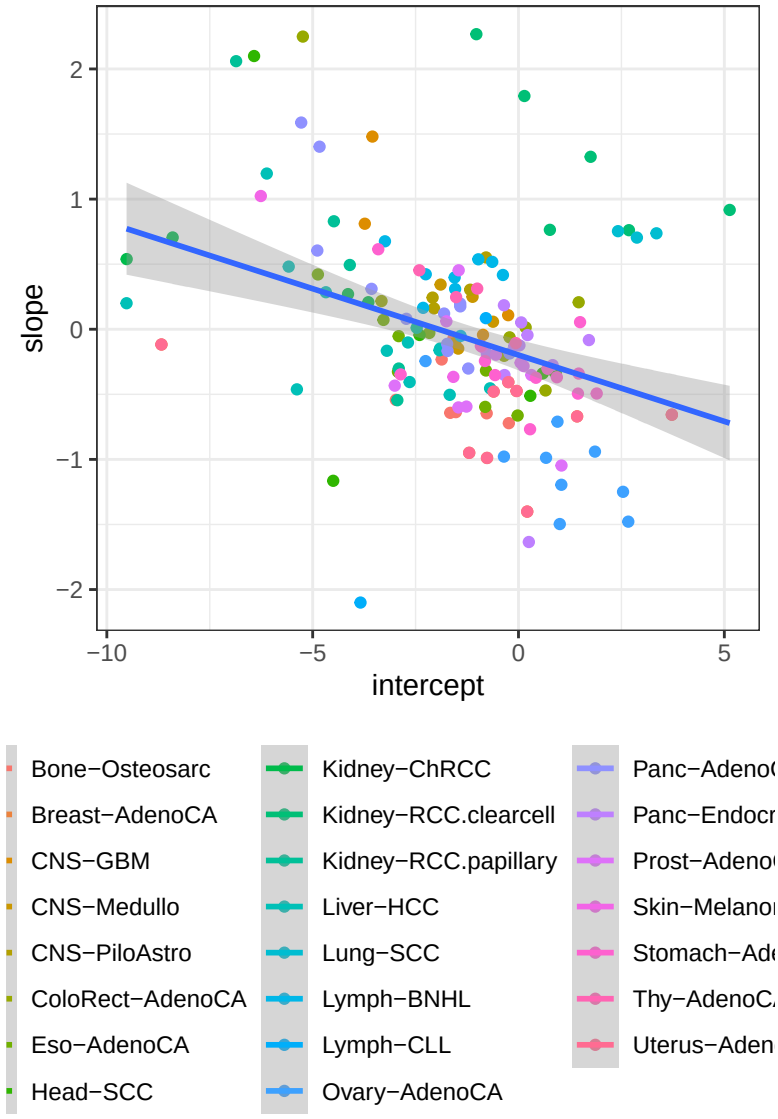
```
##
## $`ColoRect-AdenoCA`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Eso-AdenoCA`
## $`Eso-AdenoCA`$fullRE_M_nonexo
##
## $`Eso-AdenoCA`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Head-SCC`
##          fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA              NA
## tree_col NA              NA
## kmeans   NA              NA
## gtable   List,12         List,12
##
## $`Kidney-ChRCC`
##          fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA              NA
## tree_col NA              NA
## kmeans   NA              NA
## gtable   List,12         List,12
##
## $`Kidney-RCC.clearcell`
##          fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA              NA
## tree_col NA              NA
## kmeans   NA              NA
## gtable   List,12         List,12
##
## $`Kidney-RCC.papillary`
## $`Kidney-RCC.papillary`$fullRE_M_nonexo
##
## $`Kidney-RCC.papillary`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Liver-HCC`
## $`Liver-HCC`$fullRE_M_nonexo
## [1] "No convergence"
##
## $`Liver-HCC`$fullRE_DMSL_nonexo
##
##
## $`Lung-SCC`
##          fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA              NA
## tree_col NA              NA
```

```
## kmeans   NA              NA
## gtable   List,12         List,12
##
## $`Lymph-BNHL`
##                                      fullRE_M_nonexo
##                                     "No convergence"
##                                   fullRE_DMSL_nonexo
## "Error : $ operator is invalid for atomic vectors\n"
##
## $`Lymph-CLL`
##          fullRE_M_nonexo fullRE_DMSL_nonexo
## tree_row NA              NA
## tree_col NA              NA
## kmeans   NA              NA
## gtable   List,12         List,12
##
## $`Ovary-AdenoCA`
## $`Ovary-AdenoCA`$fullRE_M_nonexo
##
## $`Ovary-AdenoCA`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Panc-AdenoCA`
## $`Panc-AdenoCA`$fullRE_M_nonexo
##
## $`Panc-AdenoCA`$fullRE_DMSL_nonexo
## [1] "Error : $ operator is invalid for atomic vectors\n"
## attr(,"class")
## [1] "try-error"
## attr(,"condition")
## <simpleError: $ operator is invalid for atomic vectors>
##
##
## $`Panc-Endocrine`
## $`Panc-Endocrine`$fullRE_M_nonexo
##
## $`Panc-Endocrine`$fullRE_DMSL_nonexo
## [1] "No convergence"
##
##
## $`Prost-AdenoCA`
## $`Prost-AdenoCA`$fullRE_M_nonexo
##
## $`Prost-AdenoCA`$fullRE_DMSL_nonexo
## [1] "Error : $ operator is invalid for atomic vectors\n"
## attr(,"class")
## [1] "try-error"
## attr(,"condition")
## <simpleError: $ operator is invalid for atomic vectors>
##
```

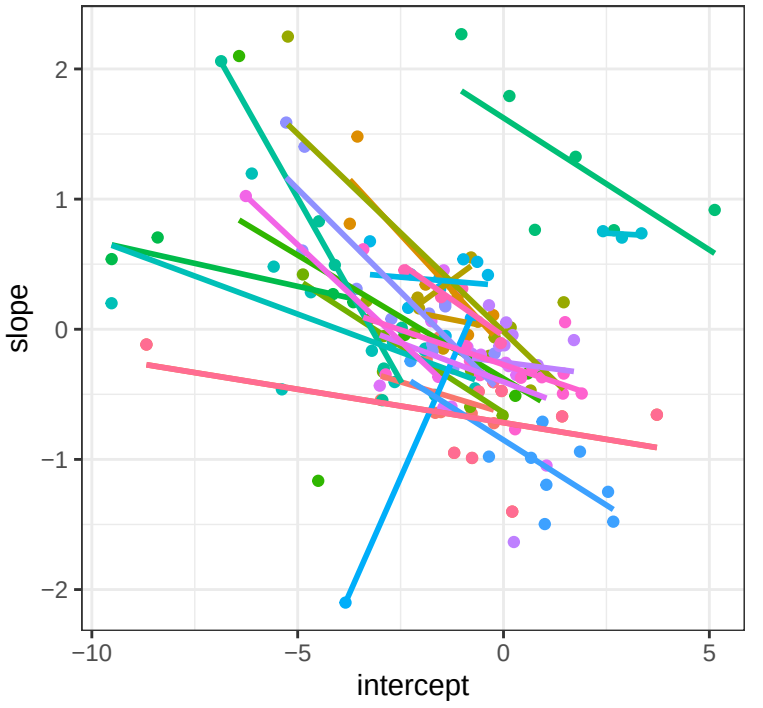
```
##
## $`Skin-Melanoma.cutaneous`
## $`Skin-Melanoma.cutaneous`$fullRE_M_nonexo
##
## $`Skin-Melanoma.cutaneous`$fullRE_DMSL_nonexo
## [1] "Error : $ operator is invalid for atomic vectors\n"
## attr(,"class")
## [1] "try-error"
## attr(,"condition")
## <simpleError: $ operator is invalid for atomic vectors>
##
##
## $`Stomach-AdenoCA`
##                                      fullRE_M_nonexo
##                                    "No convergence"
##                                  fullRE_DMSL_nonexo
## "Error : $ operator is invalid for atomic vectors\n"
##
## $`Thy-AdenoCA`
##                                      fullRE_M_nonexo
## "Error : $ operator is invalid for atomic vectors\n"
##                                  fullRE_DMSL_nonexo
## "Error : $ operator is invalid for atomic vectors\n"
##
## $`Uterus-AdenoCA`
## $`Uterus-AdenoCA`$fullRE_M_nonexo
##
## $`Uterus-AdenoCA`$fullRE_DMSL_nonexo
## [1] "Error : $ operator is invalid for atomic vectors\n"
## attr(,"class")
## [1] "try-error"
## attr(,"condition")
## <simpleError: $ operator is invalid for atomic vectors>
```

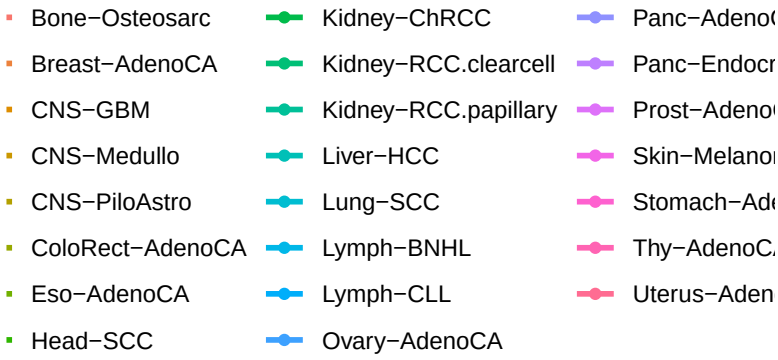
Correlation of intercept and slope values

```
## `geom_smooth()` using formula 'y ~ x'
```

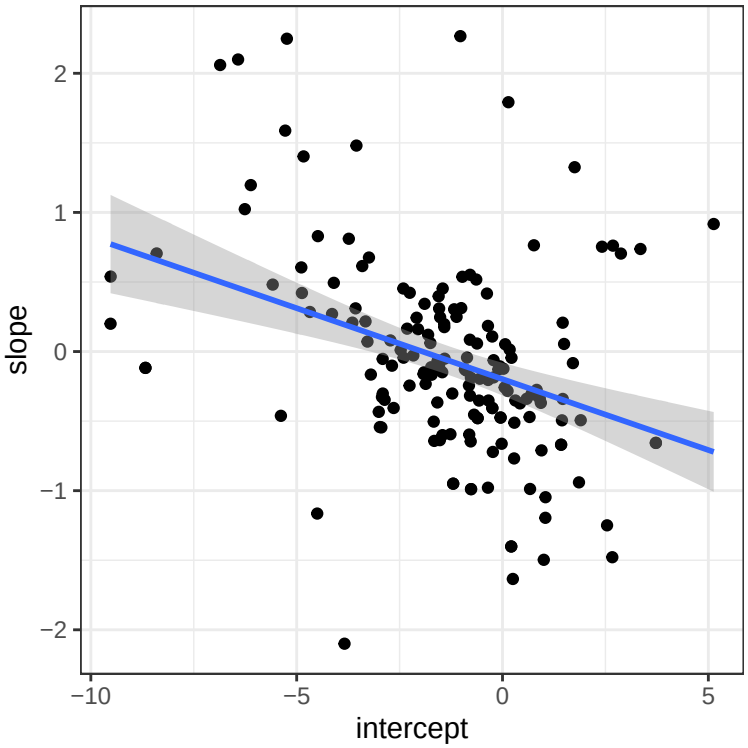



```
## `geom_smooth()` using formula 'y ~ x'
```

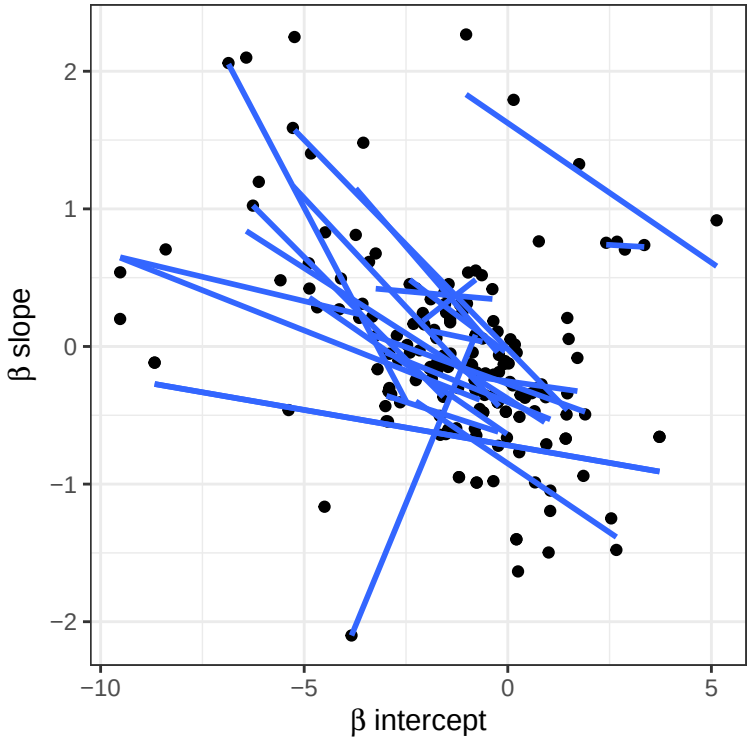





```
## `geom_smooth()` using formula 'y ~ x'
```




```
## `geom_smooth()` using formula 'y ~ x'
```



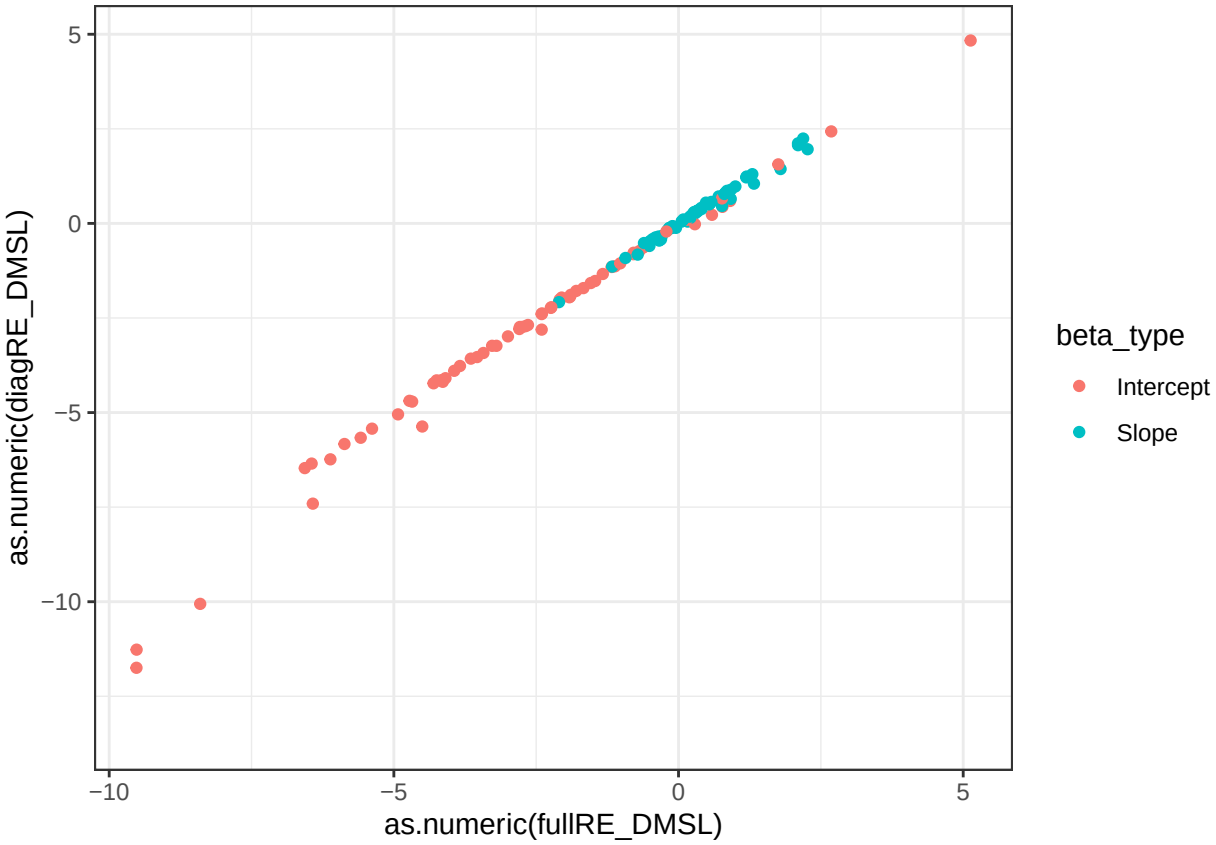
For the cancer types for which we have both fullRE and diagRE, see how the beta coefficients change in these two models, in a scatteplot where we show beta intercepts and slopes.

Note that not all fullRE M or diagM have converged - I have removed those which haven't. In some cases I ran DMSL with a subset of signatures, so that diagRE DMSL (with more signatures) cannot be compared to fullRE DMSL.

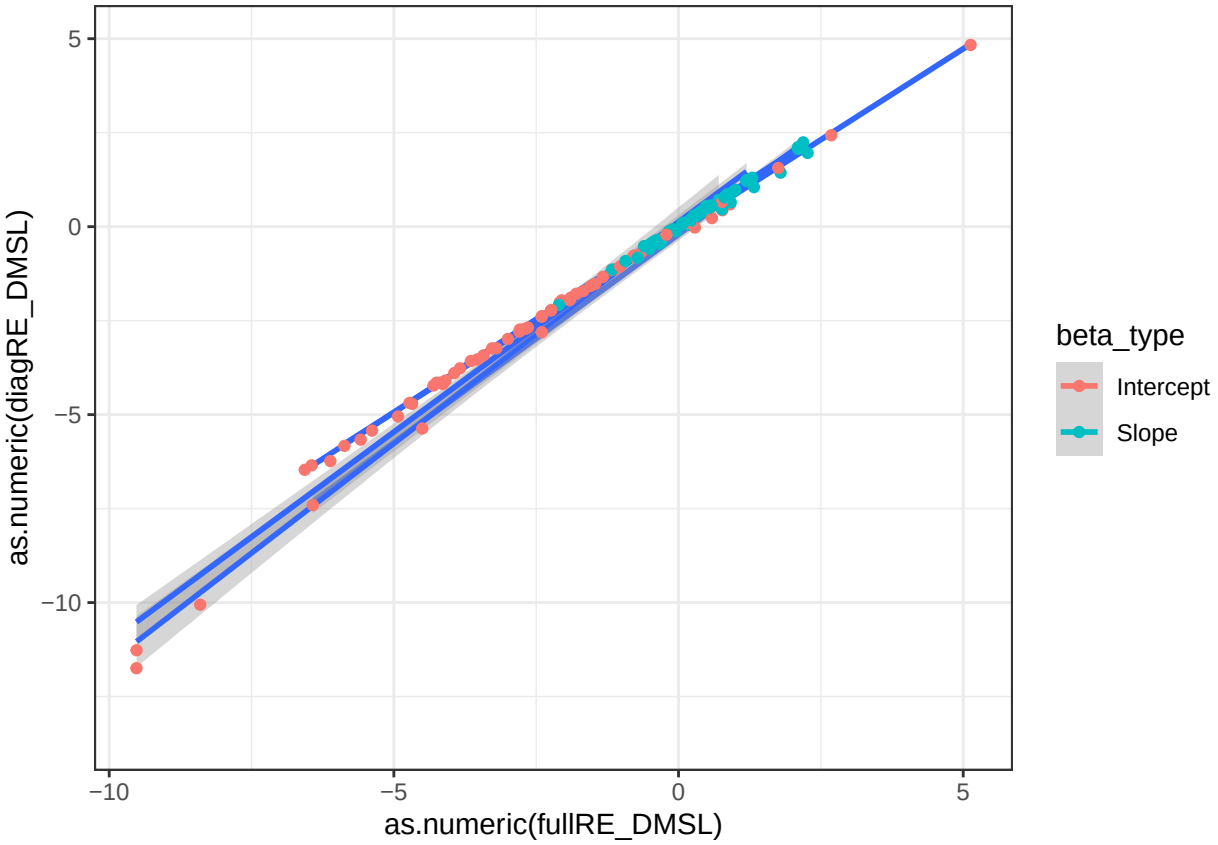
```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

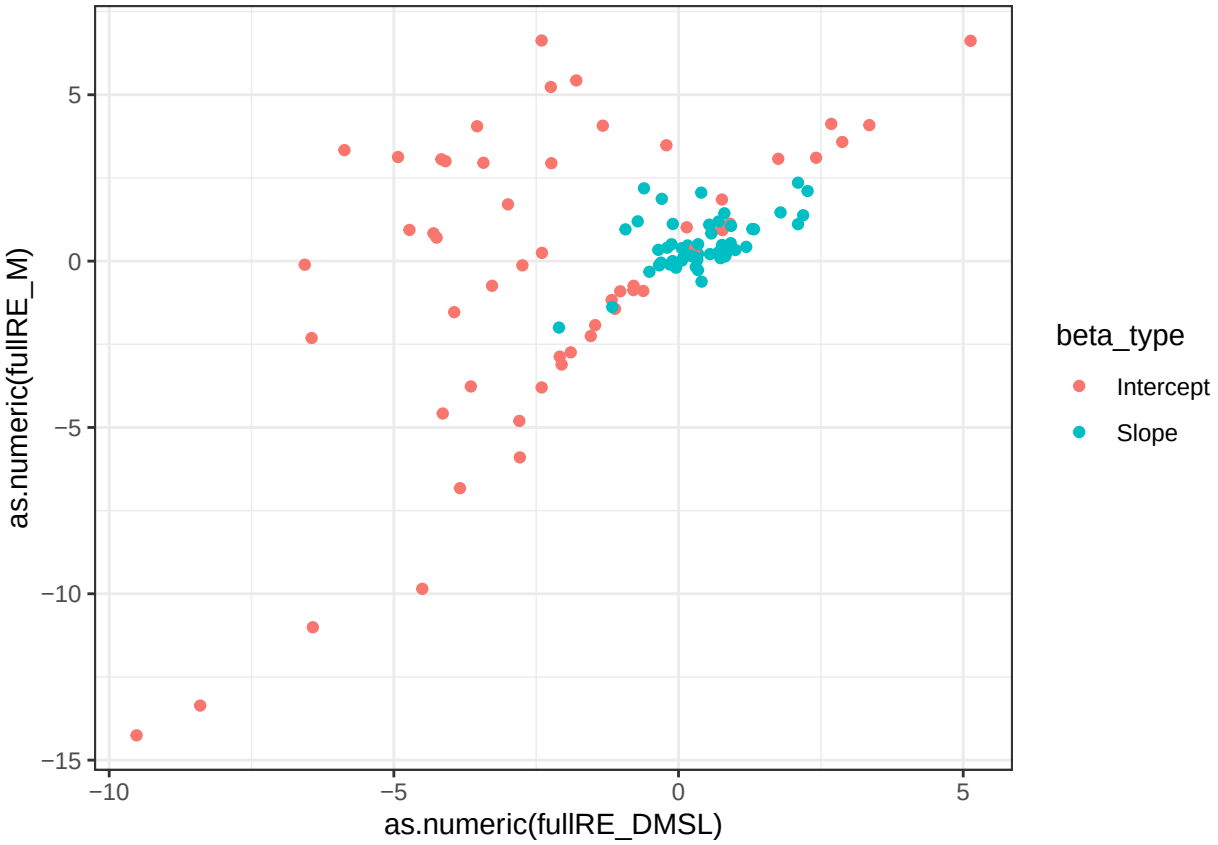
```
## Warning: Removed 678 rows containing missing values (geom_point).
```



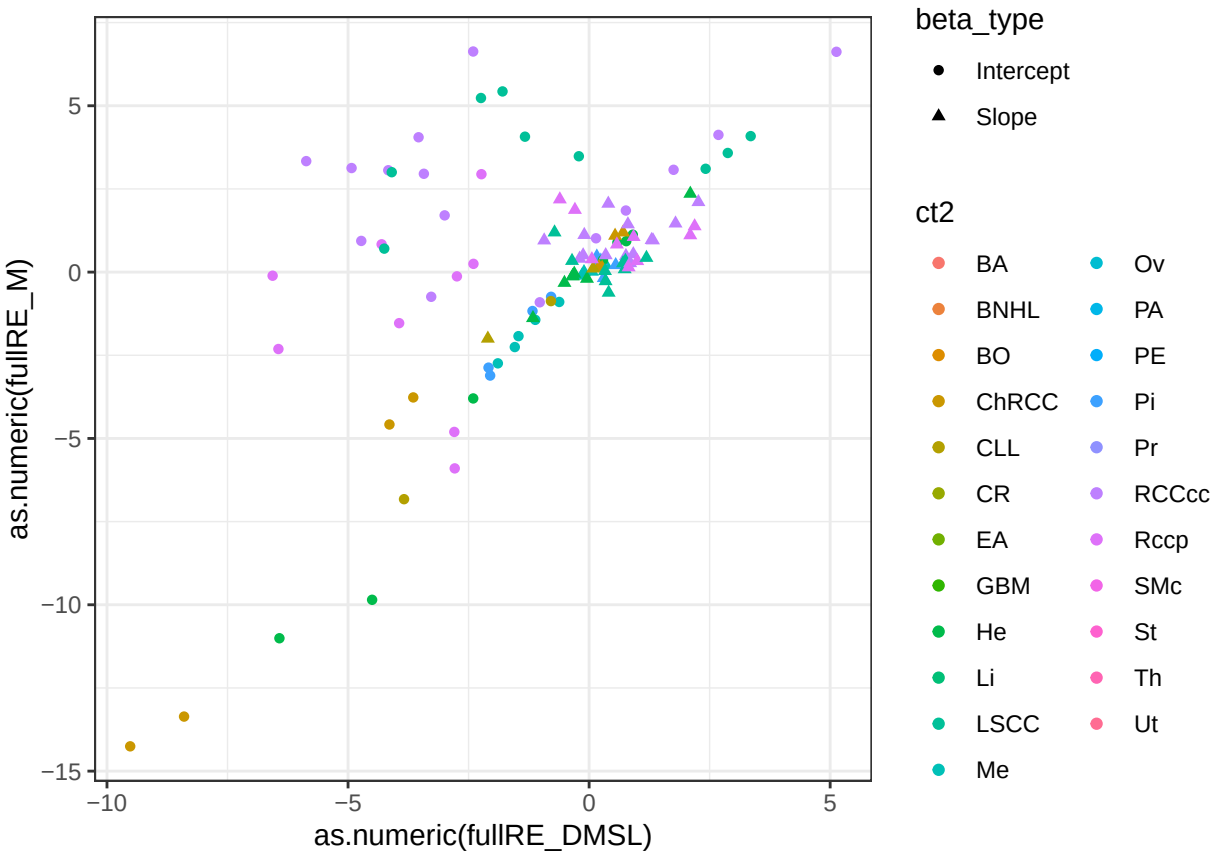

```
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 678 rows containing non-finite values (stat_smooth).
## Warning: Removed 678 rows containing missing values (geom_point).
```



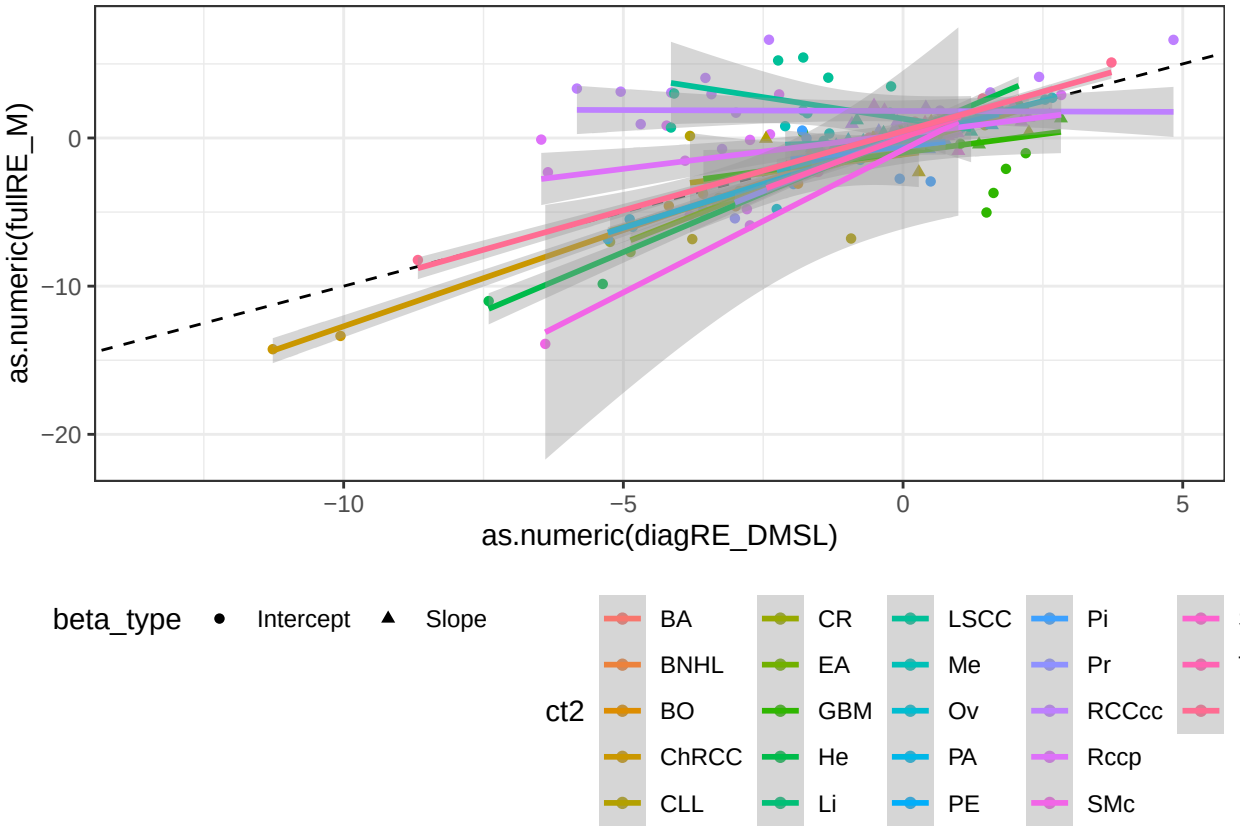

```
## Warning: Removed 696 rows containing missing values (geom_point).
```




```
## Warning: Removed 696 rows containing missing values (geom_point).
```



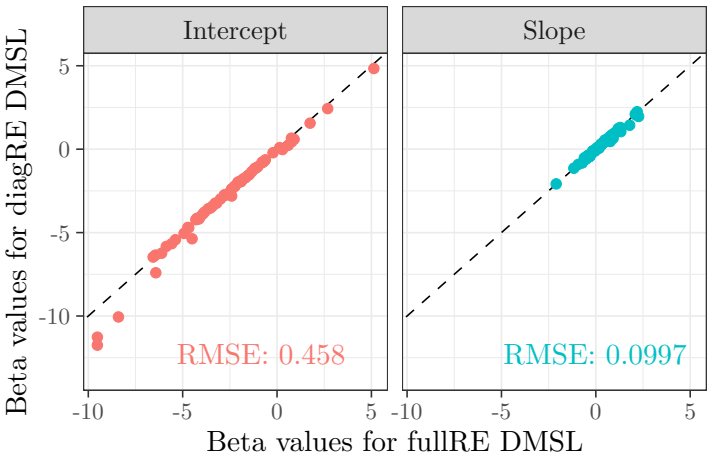

```
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 450 rows containing non-finite values (stat_smooth).
## Warning: Removed 450 rows containing missing values (geom_point).
```




```
##   rmse_diag_full_DMSL rmse_fullDMSL_fullM beta_type
## 1          0.09972866           0.7838289     Slope
## 2          0.45761210           4.1215070 Intercept
```

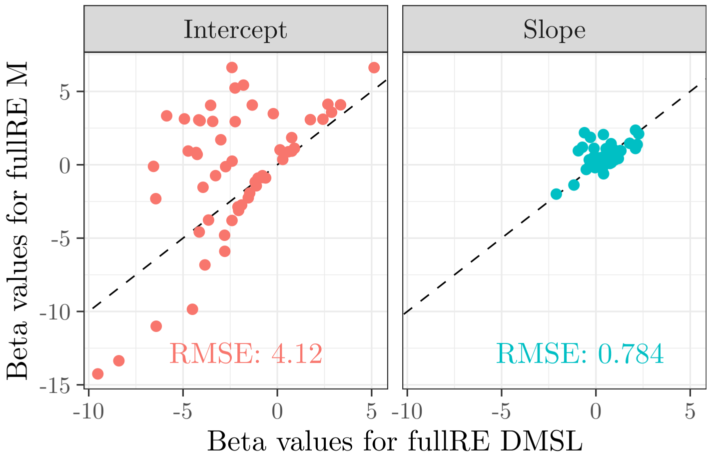
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 678 rows containing missing values (geom_point).
```




```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 696 rows containing missing values (geom_point).
```




```
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
```

```
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
```

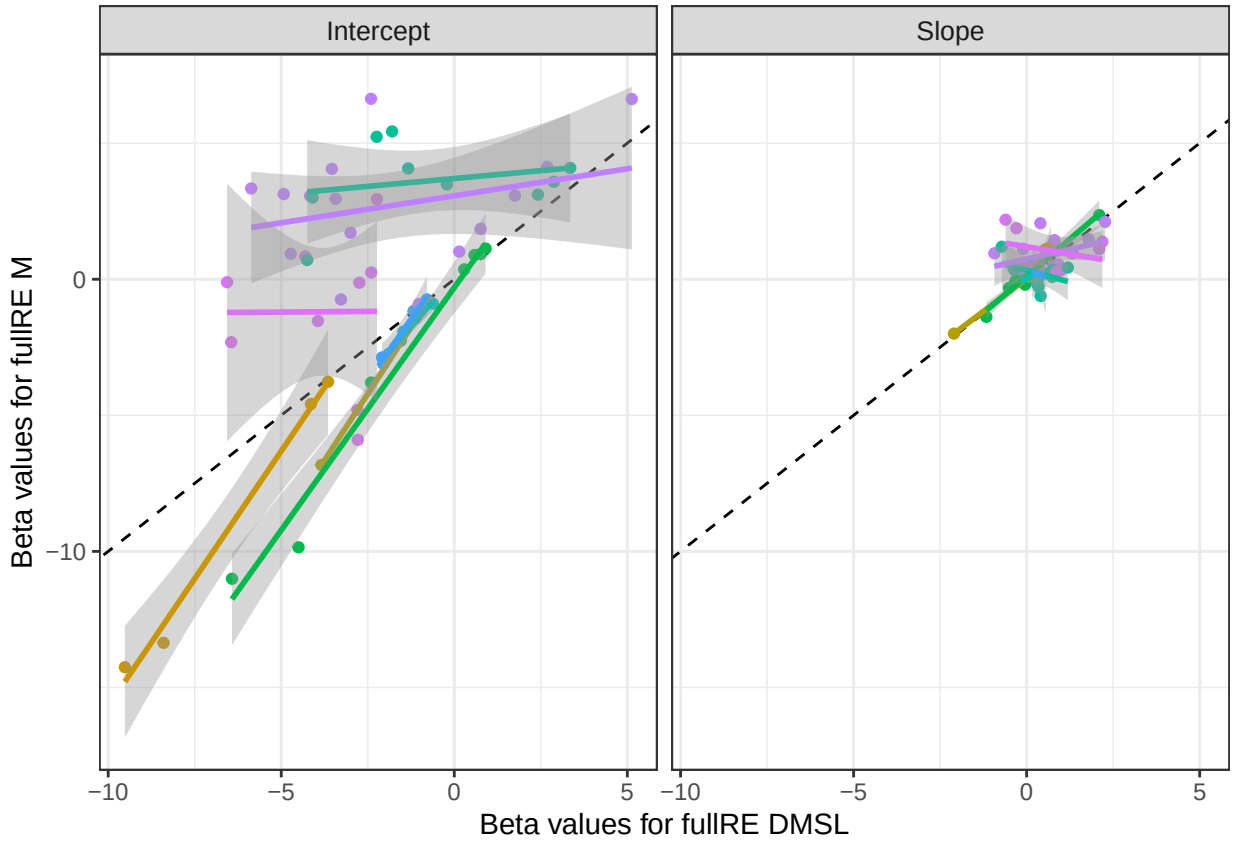
```
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases

## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
```

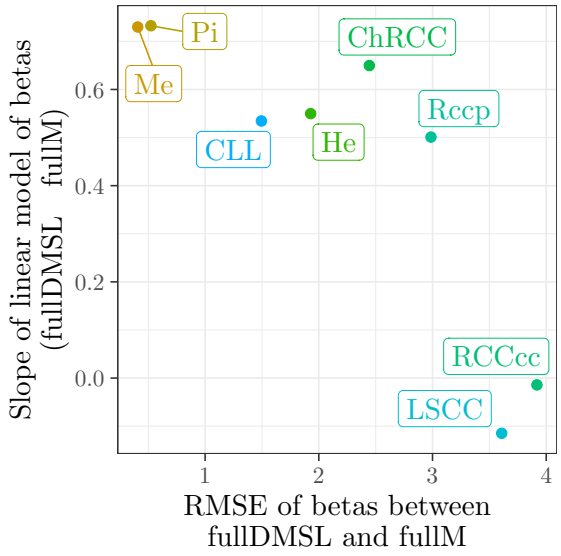
```
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
## Error in lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##   0 (non-NA) cases
```

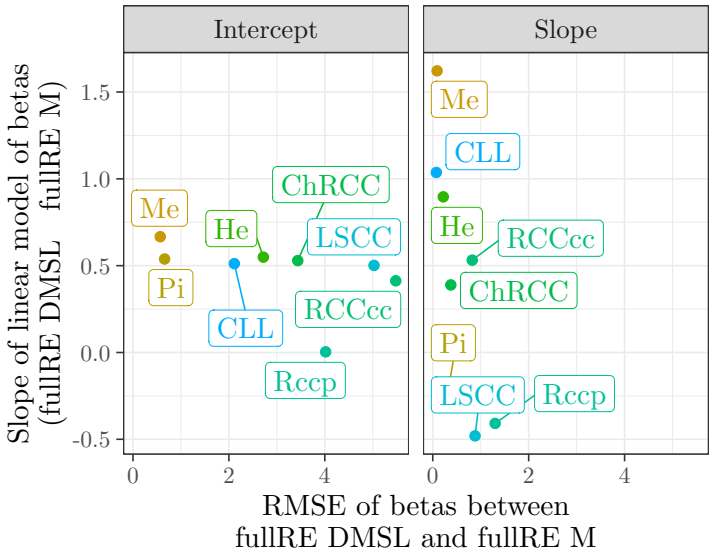
```
##                        ct rmse_diag_full_DMSL slope_diag_full_DMSL
## 1          Bone-Osteosarc                 NaN                   NA
## 2          Breast-AdenoCA                 NaN                   NA
## 3                 CNS-GBM                 NaN                   NA
## 4             CNS-Medullo          0.01960951            1.0817072
## 5           CNS-PiloAstro          0.06634326            0.8340930
## 6        ColoRect-AdenoCA                 NaN                   NA
## 7             Eso-AdenoCA                 NaN                   NA
## 8                Head-SCC          0.08445690            0.9948229
## 9            Kidney-ChRCC          0.03673637            0.9343117
## 10   Kidney-RCC.clearcell          0.18583165            1.1204075
## 11   Kidney-RCC.papillary          0.03958342            0.9962834
## 12              Liver-HCC          0.03602693            0.9933884
## 13               Lung-SCC          0.04704876            0.9478835
```

```
## 14              Lymph-BNHL                NaN                  NA
## 15               Lymph-CLL          0.01890391            0.9998600
## 16           Ovary-AdenoCA                NaN                  NA
## 17            Panc-AdenoCA                NaN                  NA
## 18          Panc-Endocrine                NaN                  NA
## 19           Prost-AdenoCA                NaN                  NA
## 20 Skin-Melanoma.cutaneous                NaN                  NA
## 21         Stomach-AdenoCA                NaN                  NA
## 22             Thy-AdenoCA                NaN                  NA
## 23          Uterus-AdenoCA                NaN                  NA
## 24          Bone-Osteosarc                NaN                  NA
## 25          Breast-AdenoCA                NaN                  NA
## 26                 CNS-GBM                NaN                  NA
## 27              CNS-Medullo          0.03219255            1.0027564
## 28            CNS-PiloAstro          0.06126289            1.0620811
## 29        ColoRect-AdenoCA                NaN                  NA
## 30             Eso-AdenoCA                NaN                  NA
## 31                Head-SCC          0.57577998            0.9145442
## 32            Kidney-ChRCC          1.20493056            0.7500070
## 33    Kidney-RCC.clearcell          0.12124177            1.0300353
## 34    Kidney-RCC.papillary          0.05713314            1.0202147
## 35               Liver-HCC          0.64474403            0.8268624
## 36                Lung-SCC          0.04132926            1.0154875
## 37              Lymph-BNHL                NaN                  NA
## 38               Lymph-CLL          0.05207686            1.0313870
## 39           Ovary-AdenoCA                NaN                  NA
## 40            Panc-AdenoCA                NaN                  NA
## 41          Panc-Endocrine                NaN                  NA
## 42           Prost-AdenoCA                NaN                  NA
## 43 Skin-Melanoma.cutaneous                NaN                  NA
## 44         Stomach-AdenoCA                NaN                  NA
## 45             Thy-AdenoCA                NaN                  NA
## 46          Uterus-AdenoCA                NaN                  NA
##    rmse_fullDMSL_fullM slope_fullDMSL_fullM beta_type   ct2
## 1                  NaN                   NA     Slope    BO
## 2                  NaN                   NA     Slope    BA
## 3                  NaN                   NA     Slope   GBM
## 4           0.09059152          1.621940968     Slope    Me
## 5           0.33664739         -0.191799809     Slope    Pi
## 6                  NaN                   NA     Slope    CR
## 7                  NaN                   NA     Slope    EA
## 8           0.21918157          0.896372091     Slope    He
## 9           0.37670836          0.389119063     Slope ChRCC
## 10          0.82278832          0.531725788     Slope RCCcc
## 11          1.30070772         -0.408535086     Slope  Rccp
## 12                 NaN                   NA     Slope    Li
## 13          0.88196759         -0.480178978     Slope  LSCC
## 14                 NaN                   NA     Slope  BNHL
## 15          0.07406513          1.036692829     Slope   CLL
## 16                 NaN                   NA     Slope    Ov
## 17                 NaN                   NA     Slope    PA
```

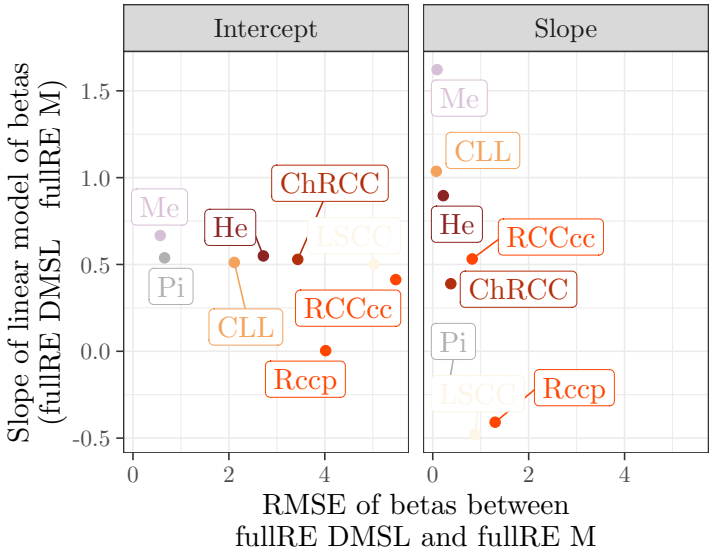
```
## 18                NaN                  NA     Slope    PE
## 19                NaN                  NA     Slope    Pr
## 20                NaN                  NA     Slope   SMc
## 21                NaN                  NA     Slope    St
## 22                NaN                  NA     Slope    Th
## 23                NaN                  NA     Slope    Ut
## 24                NaN                  NA Intercept    BO
## 25                NaN                  NA Intercept    BA
## 26                NaN                  NA Intercept   GBM
## 27         0.56916461         0.666803710 Intercept    Me
## 28         0.65905520         0.538204984 Intercept    Pi
## 29                NaN                  NA Intercept    CR
## 30                NaN                  NA Intercept    EA
## 31         2.71848733         0.549152181 Intercept    He
## 32         3.43614717         0.529239927 Intercept ChRCC
## 33         5.47957627         0.412829456 Intercept RCCcc
## 34         4.01789367         0.003543566 Intercept  Rccp
## 35                NaN                  NA Intercept    Li
## 36         5.02604140         0.501340467 Intercept  LSCC
## 37                NaN                  NA Intercept  BNHL
## 38         2.11315976         0.511657714 Intercept   CLL
## 39                NaN                  NA Intercept    Ov
## 40                NaN                  NA Intercept    PA
## 41                NaN                  NA Intercept    PE
## 42                NaN                  NA Intercept    Pr
## 43                NaN                  NA Intercept   SMc
## 44                NaN                  NA Intercept    St
## 45                NaN                  NA Intercept    Th
## 46                NaN                  NA Intercept    Ut
```


Intercept
Slope
Beta values for fullRE M
0
−10
−10
−5
0
5
Beta values for fullRE DMSL


Pi
Me
ChRCC
Rccp
He
CLL
RCCcc
LSCC
Slope of linear model of betas
(fullDMSL fullM)
0.0
0.2
0.4
0.6
1
2
3
4
RMSE of betas between
fullDMSL and fullM




```
##                 ct rmse_diag_full_DMSL slope_diag_full_DMSL rmse_fullDMSL_fullM
## 1   Bone-Osteosarc                 NaN                   NA                 NaN
## 2   Breast-AdenoCA                 NaN                   NA                 NaN
## 3         CNS-GBM                  NaN                   NA                 NaN
## 4      CNS-Medullo           0.01960951             1.081707          0.09059152
## 5    CNS-PiloAstro           0.06634326             0.834093          0.33664739
## 6 ColoRect-AdenoCA                 NaN                   NA                 NaN
##   slope_fullDMSL_fullM beta_type ct2
## 1                   NA     Slope  BO
## 2                   NA     Slope  BA
## 3                   NA     Slope GBM
## 4            1.6219410     Slope  Me
## 5           -0.1917998     Slope  Pi
## 6                   NA     Slope  CR
```



Compare the random effects intercepts. We cannot compare across cancer types because the number of observation

is different. For each cancer type, compare the intercept fullRE DMSL of each patient to the intercept of fullRE M and diagRE DMSL. I.e. for each cancer type we have a boxplot.

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
```

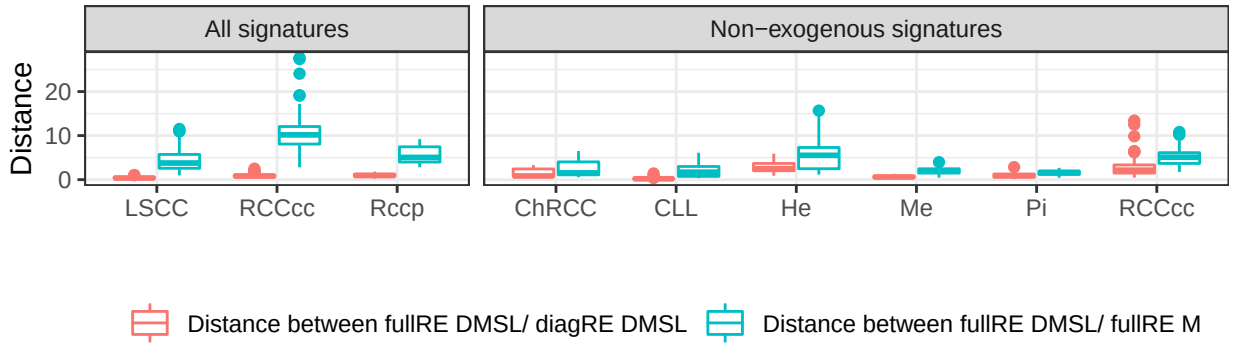
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Error : $ operator is invalid for atomic vectors
## Error : $ operator is invalid for atomic vectors
```

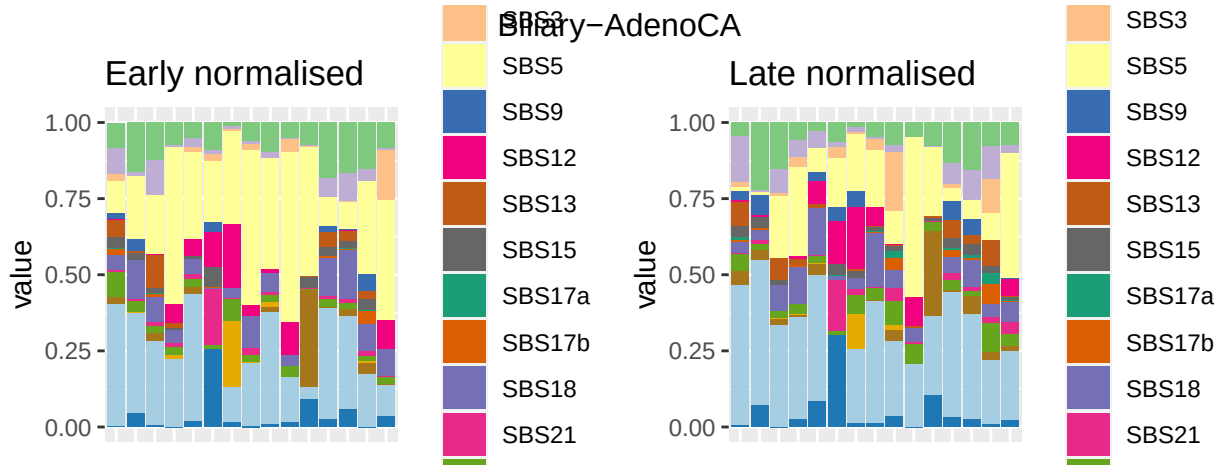
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

Would like to have the picture below in computer modern, but I cannot make boxplot work with tikz

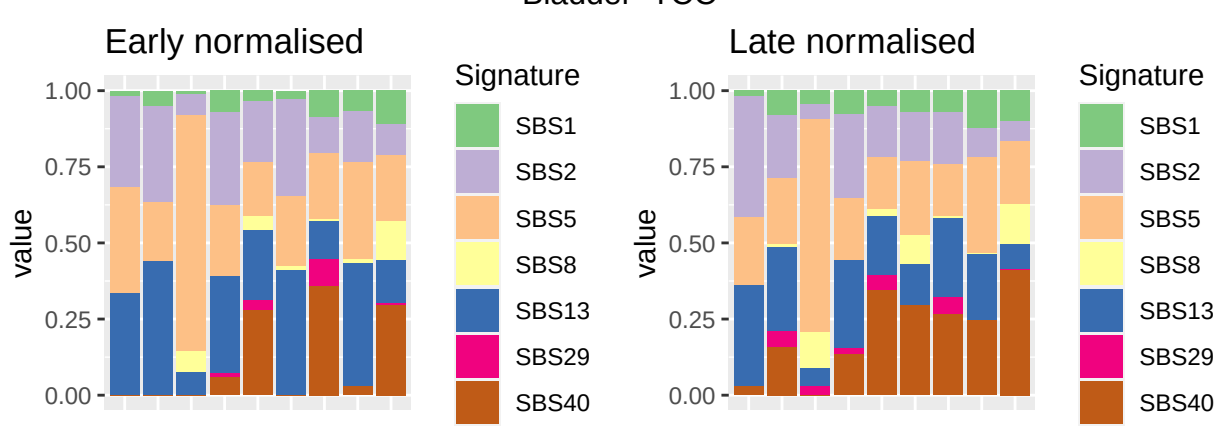
```
## `summarise()` has grouped output by 'ct'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'ct'. You can override using the `.groups` argument.
```



Using subset of active signatures from the PCAWG paper

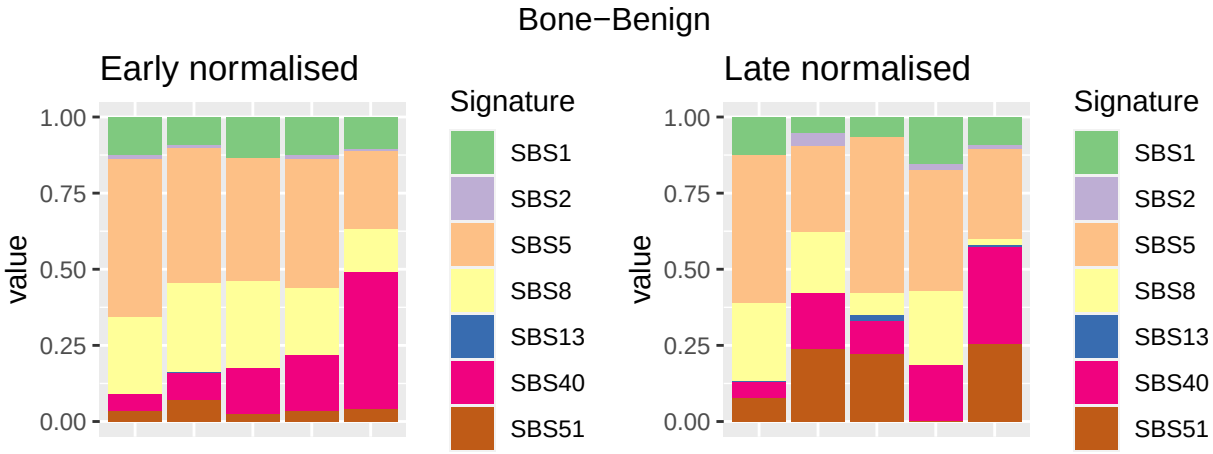



```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.328]
```

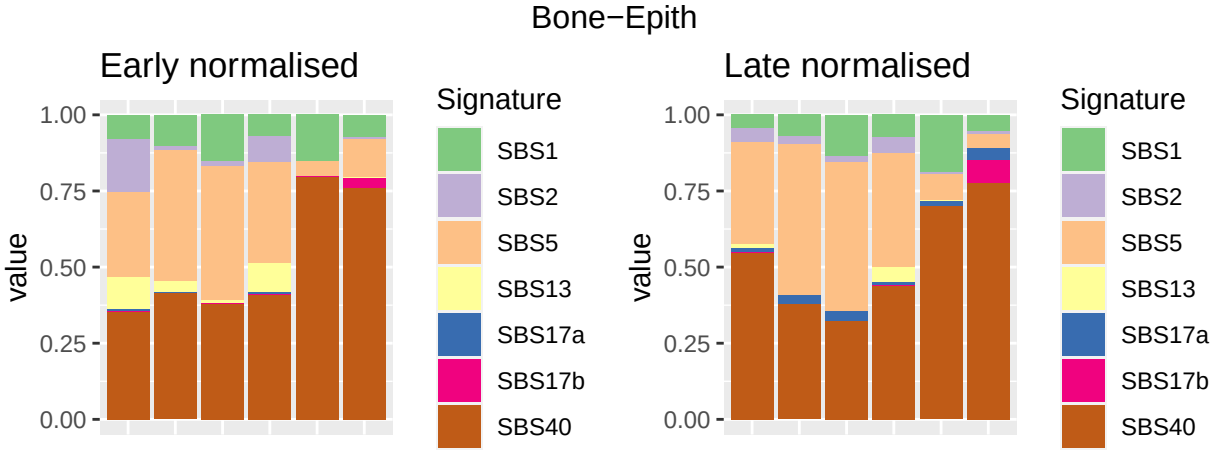



```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
```

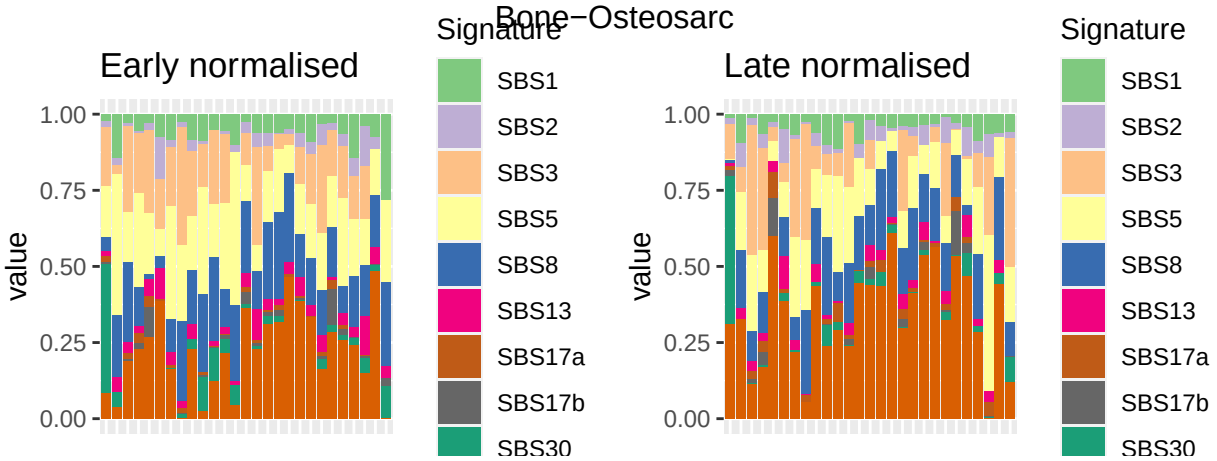
```
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.503]
```



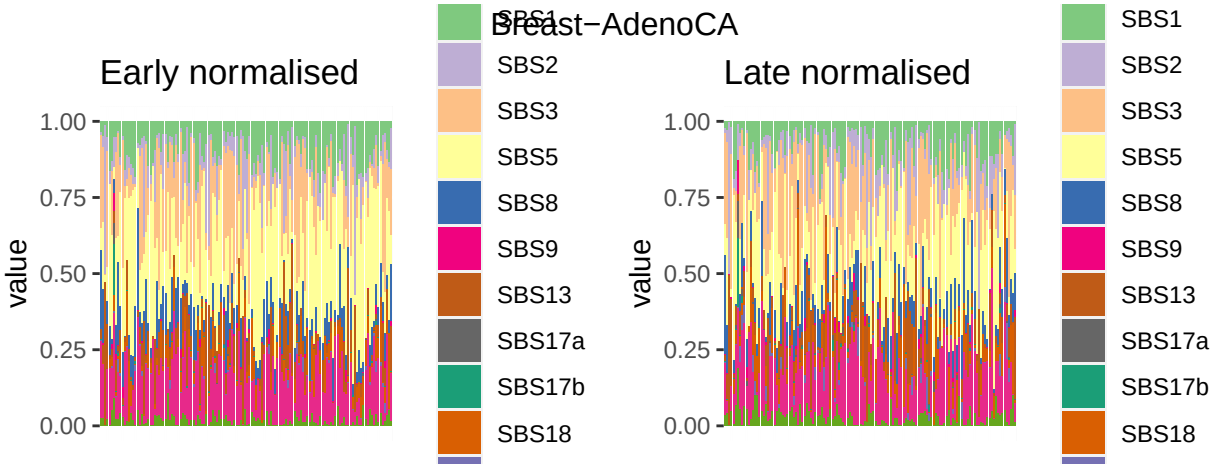

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.678]
```




```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.853]
```

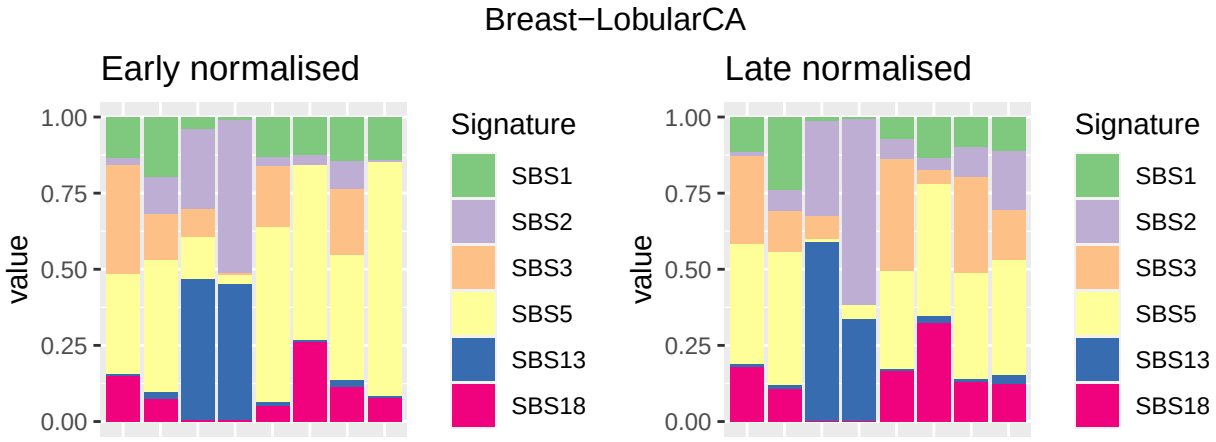



```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1070]
```

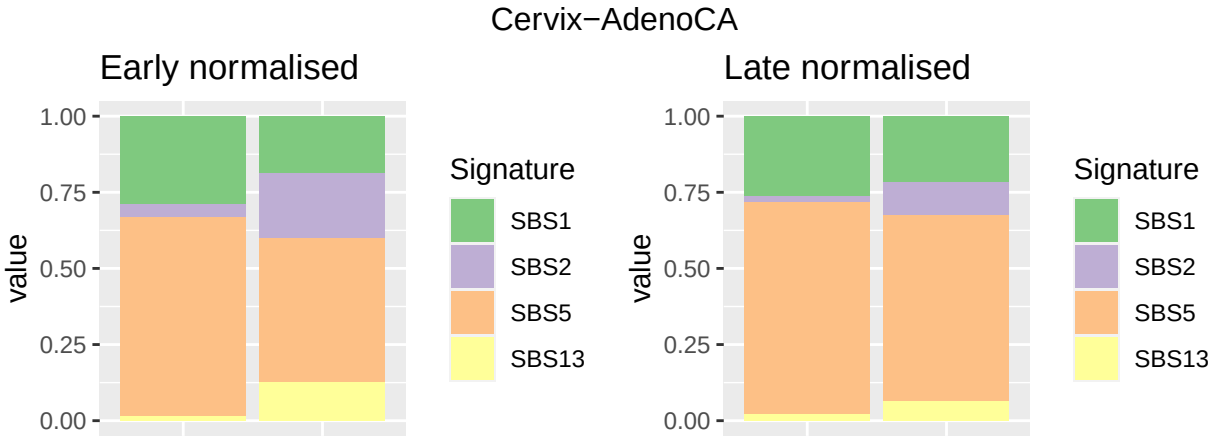



```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1329]
```

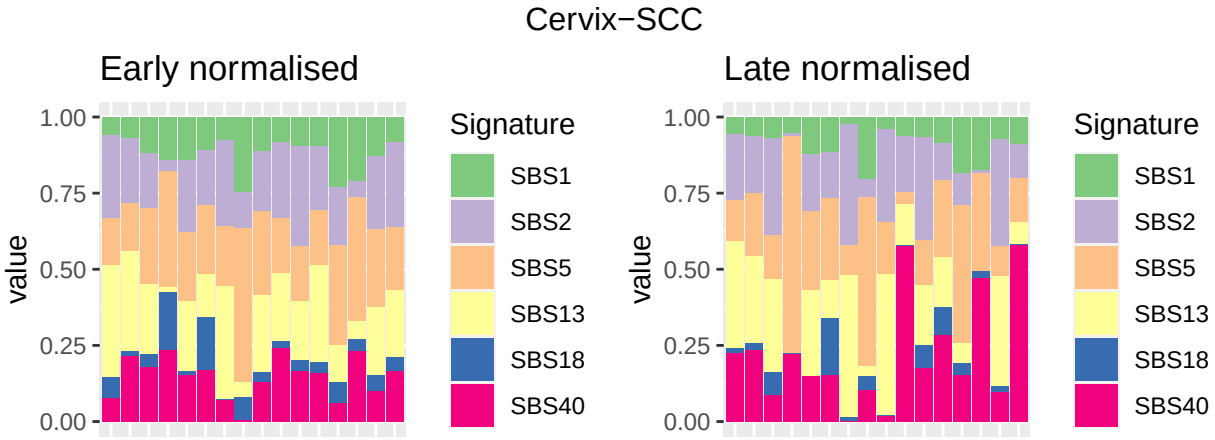

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1448]
```



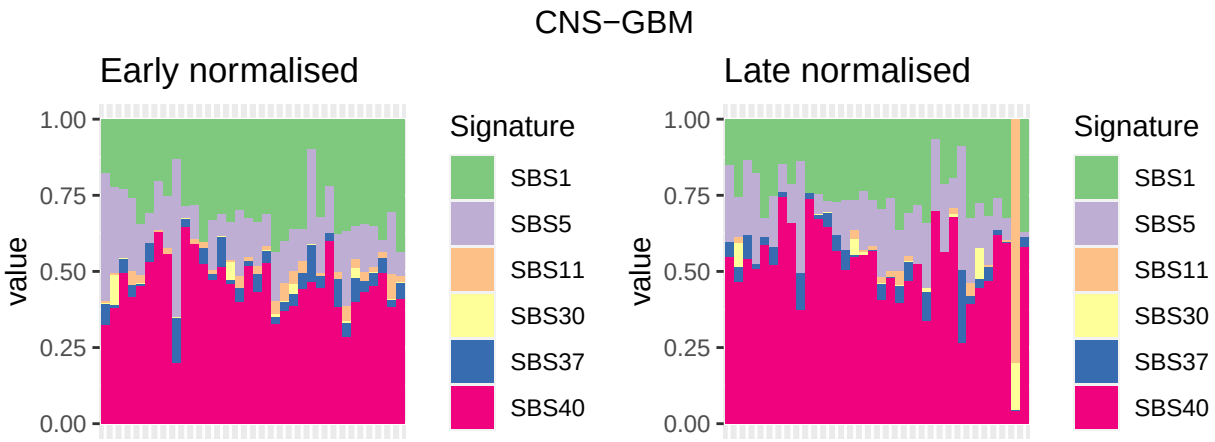

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1609]
```



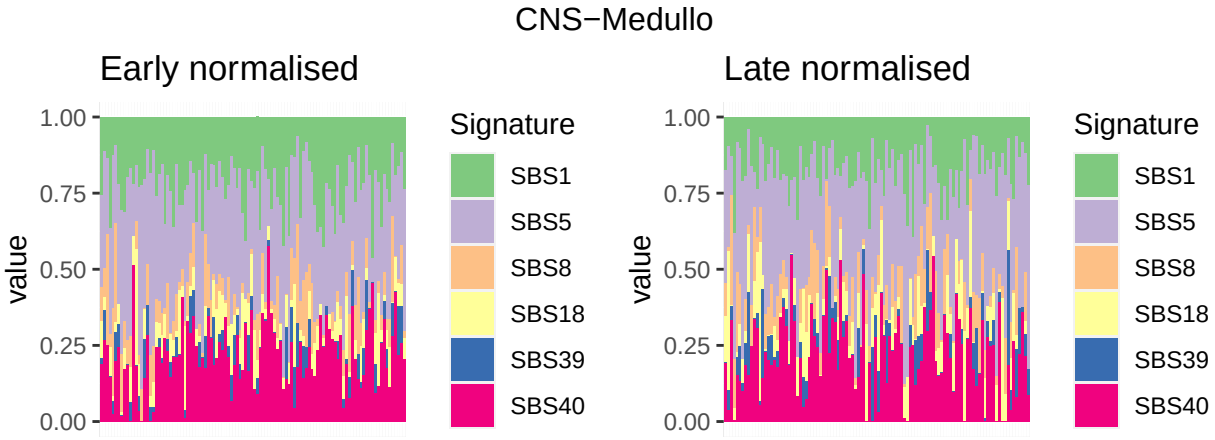

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1742]
```



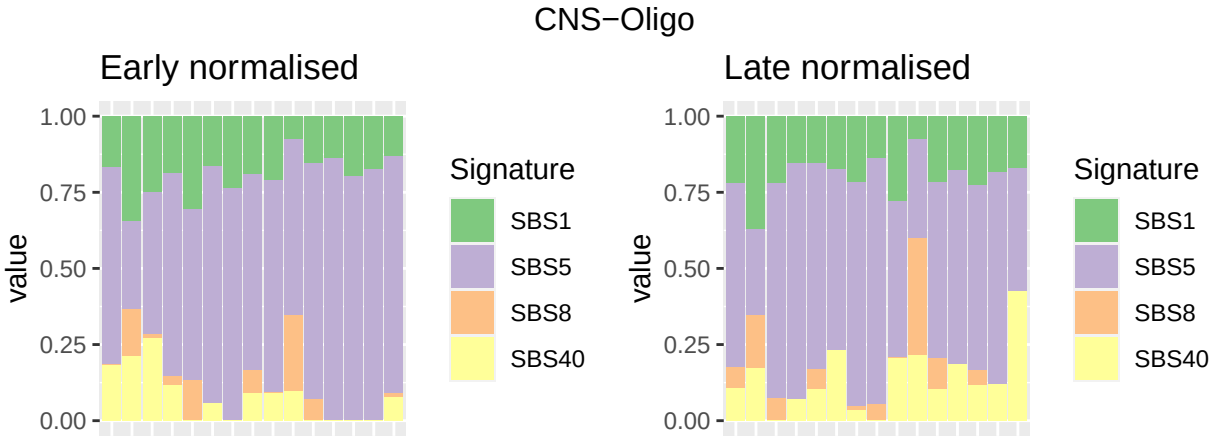

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1903]
```



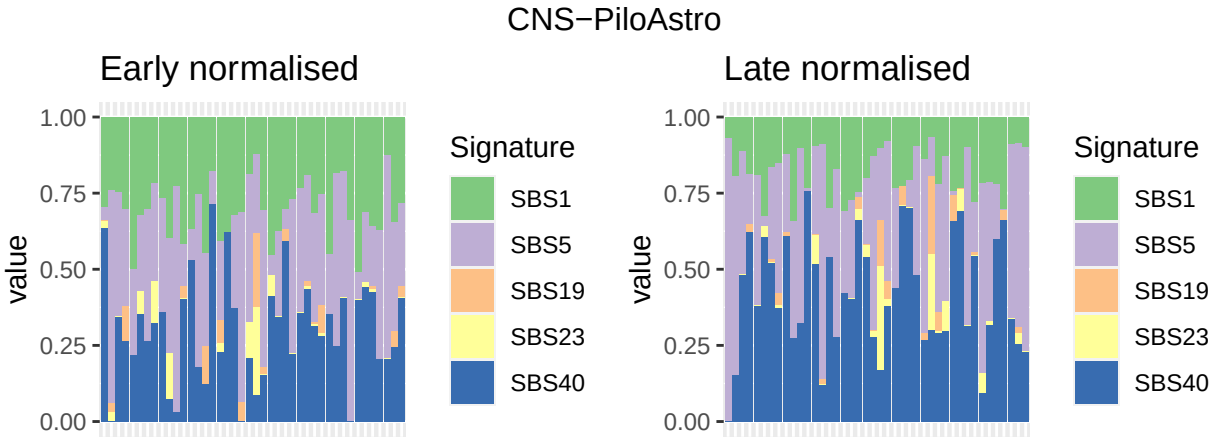

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2064]
```



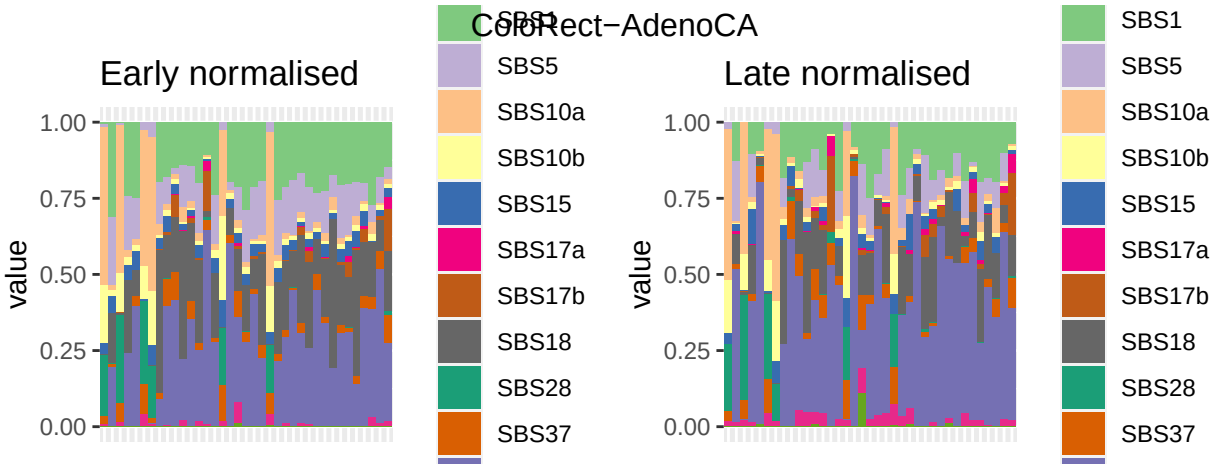

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2225]
```



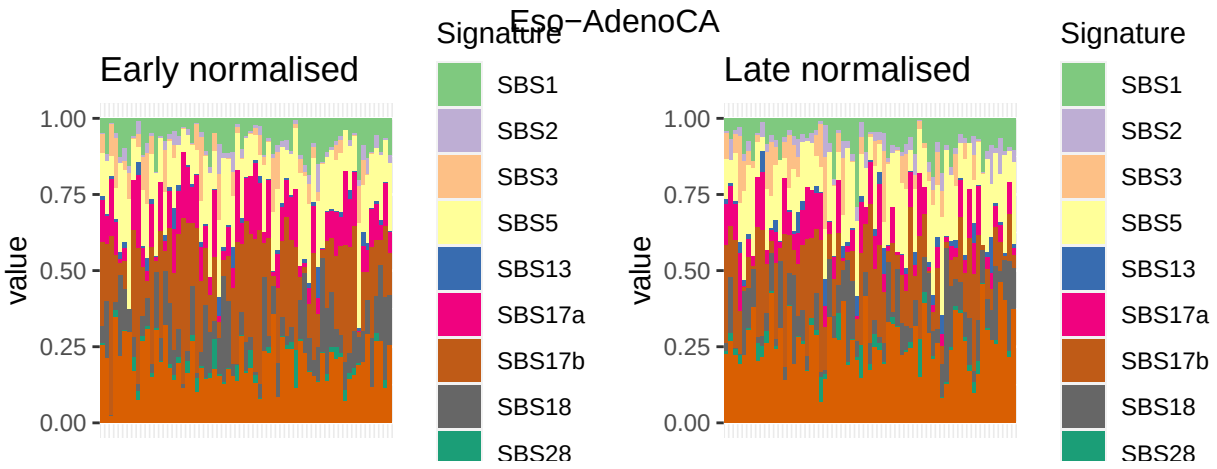

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2358]
```



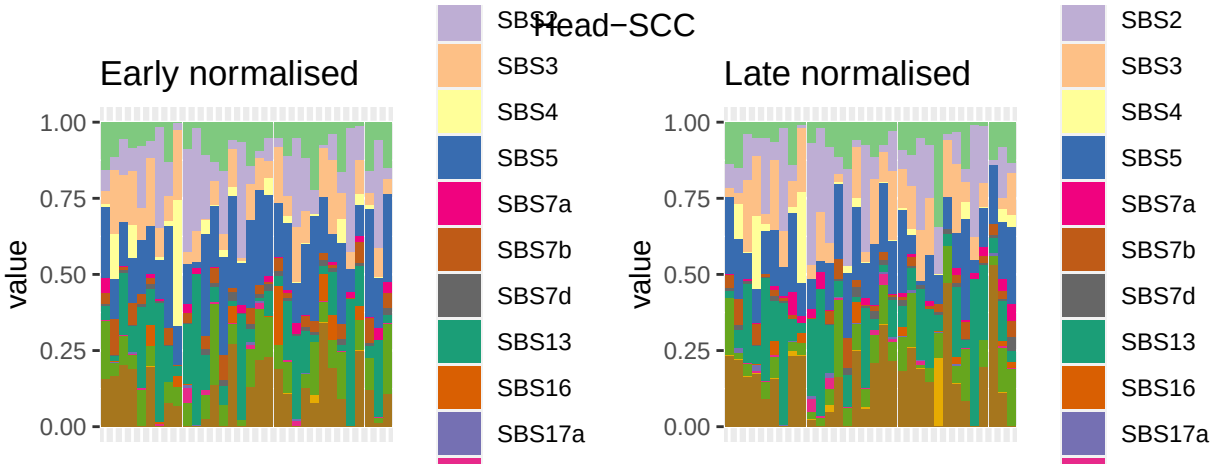

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2505]
```



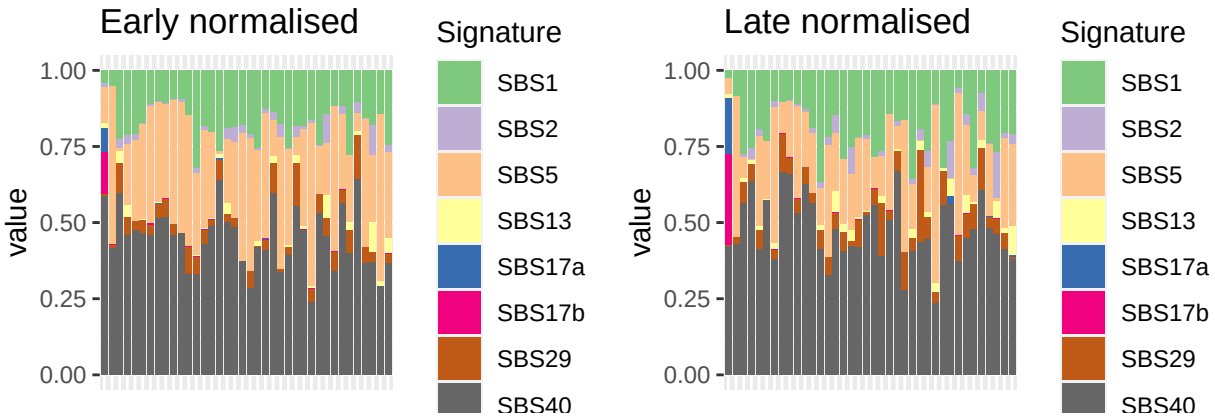

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2764]
```



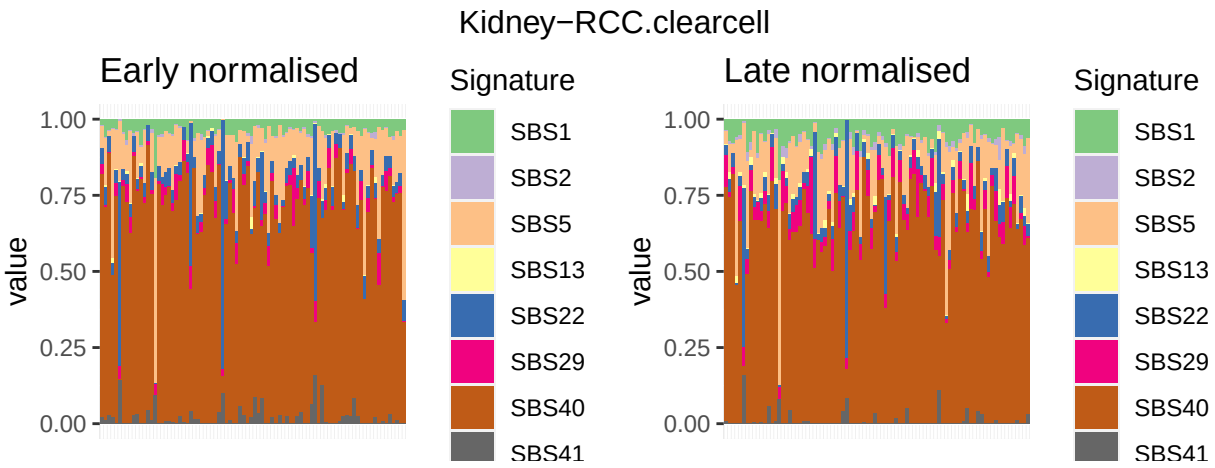

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2981]
```



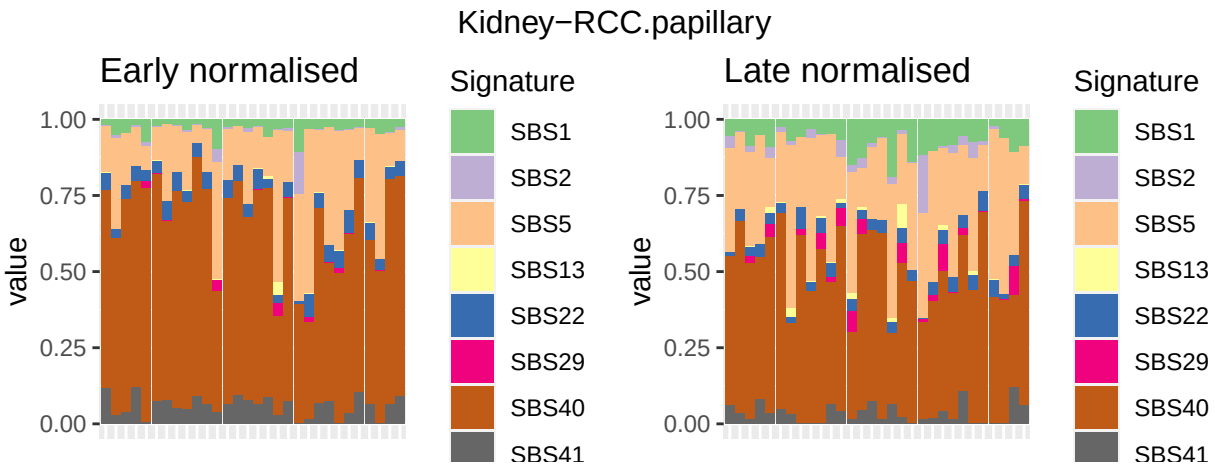

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3268]
```



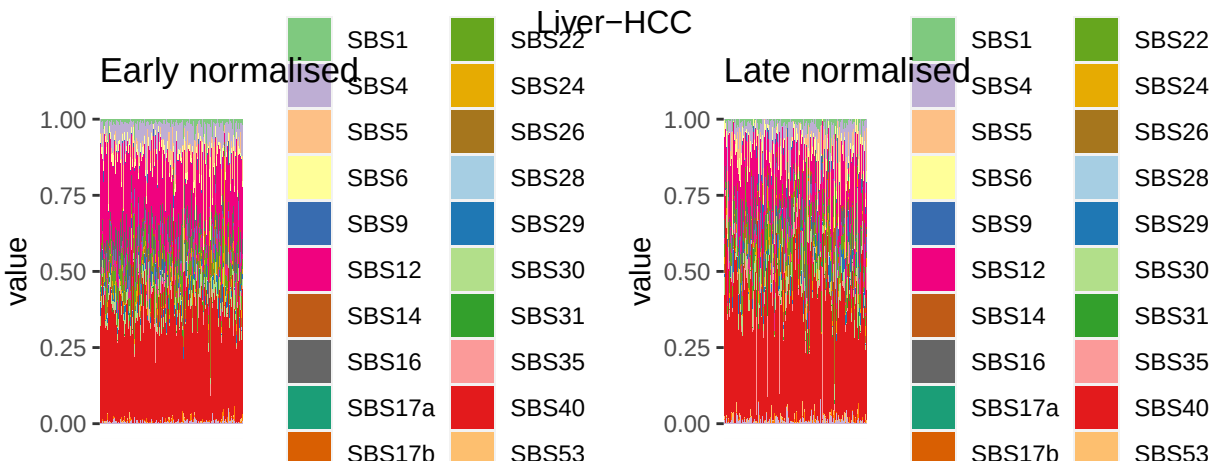

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3457]
```



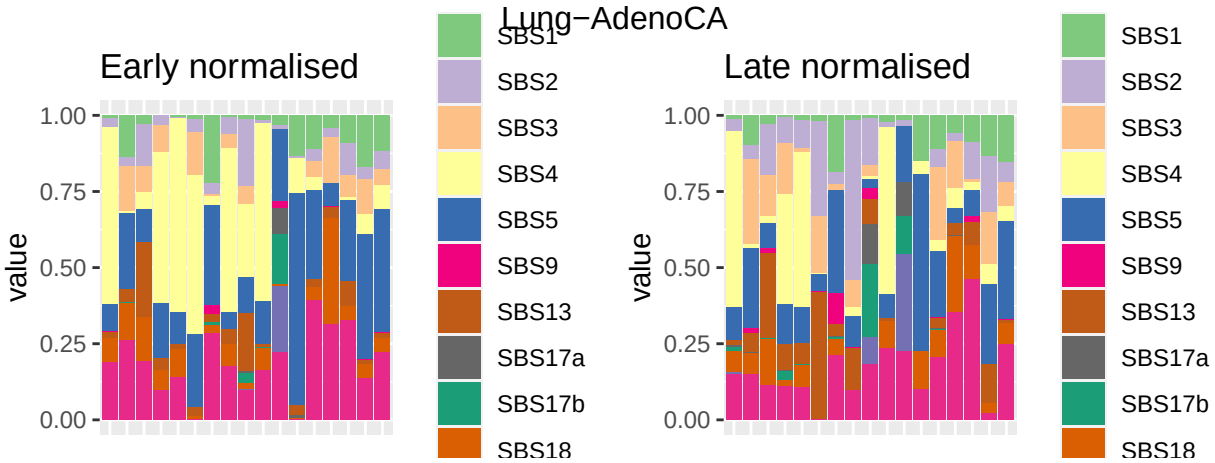

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3646]
```



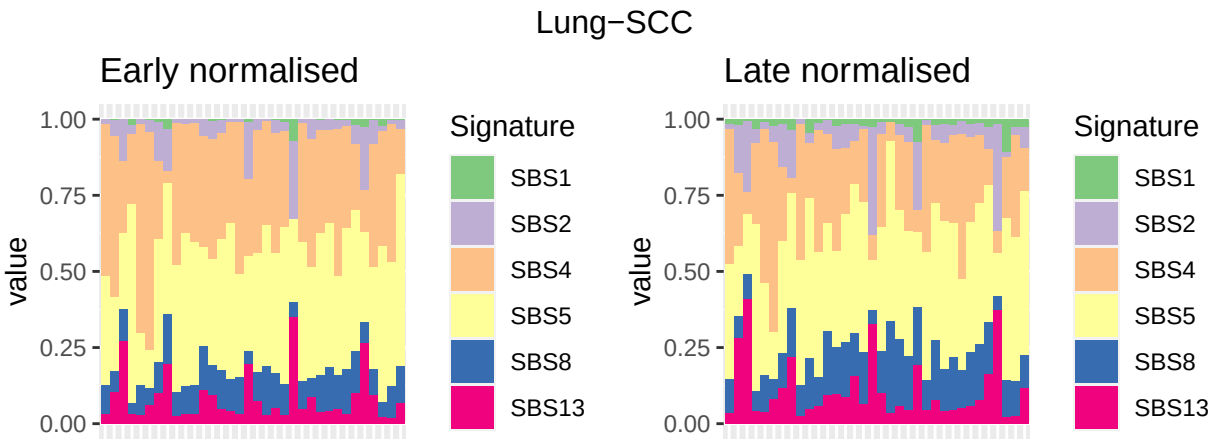

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3835]
```



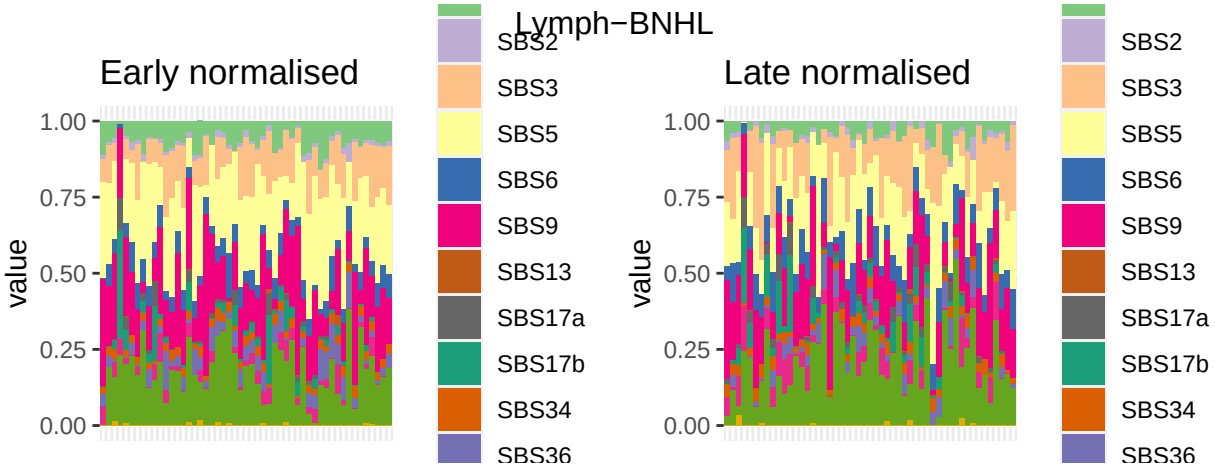

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4248]
```



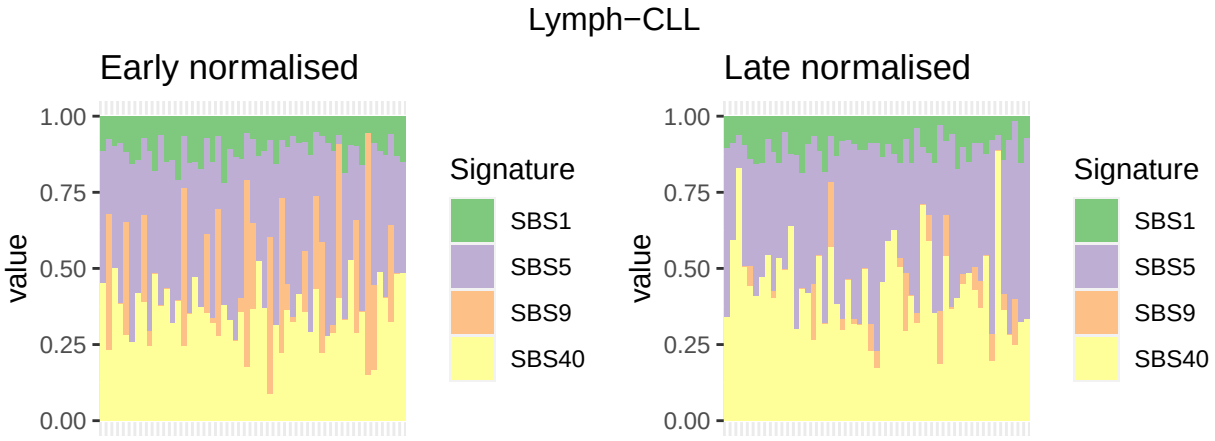

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4493]
```



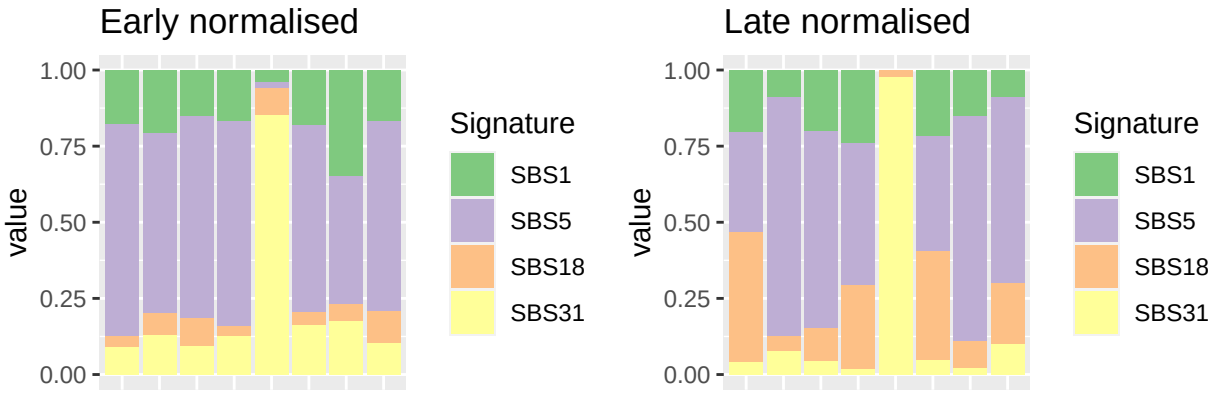

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4654]
```



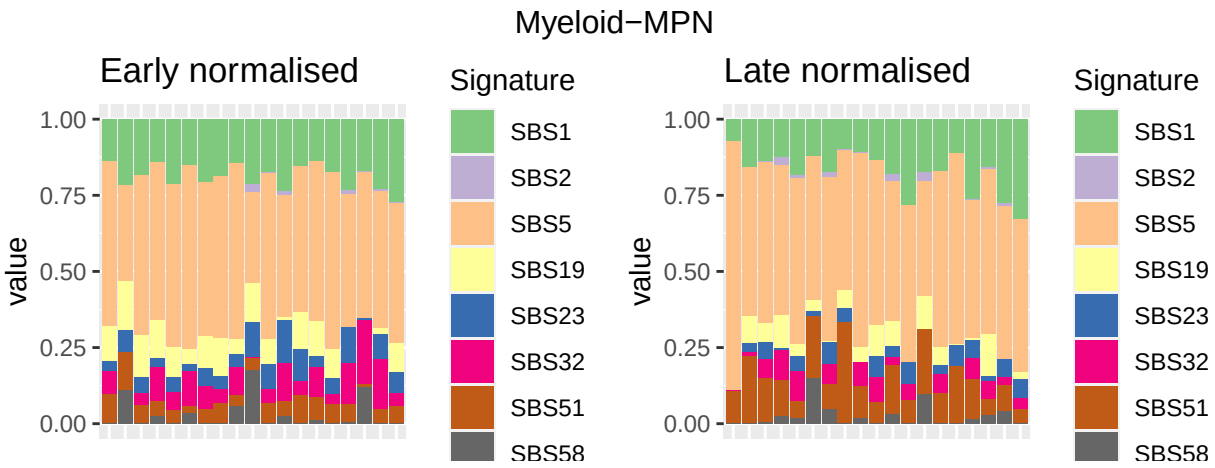

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4927]
```



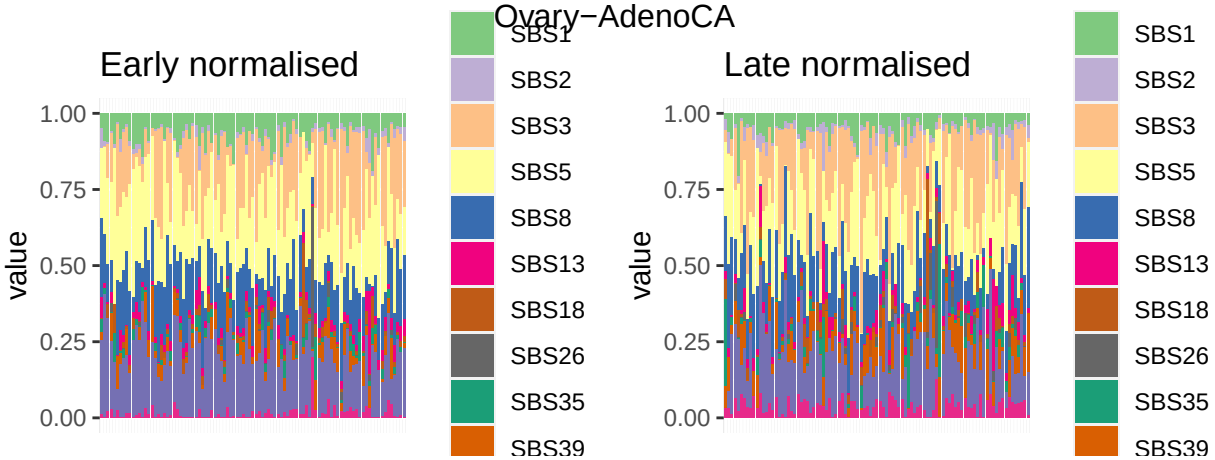

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5060]
```



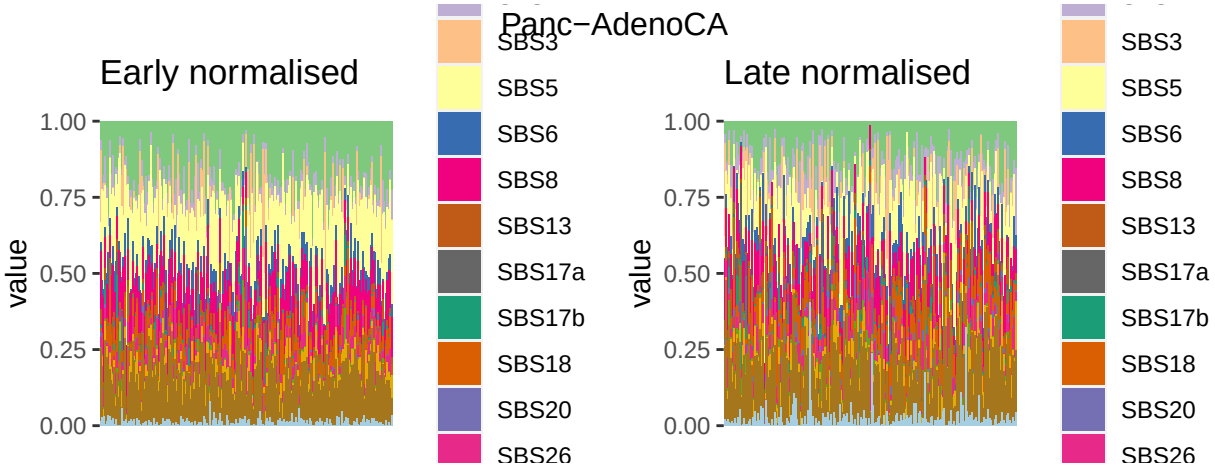

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5193]
```



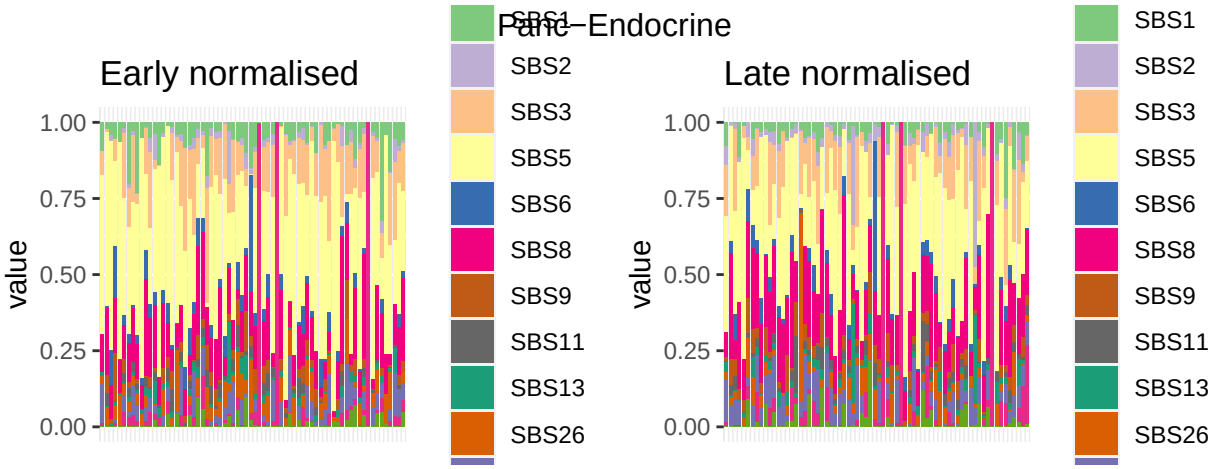

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5382]
```



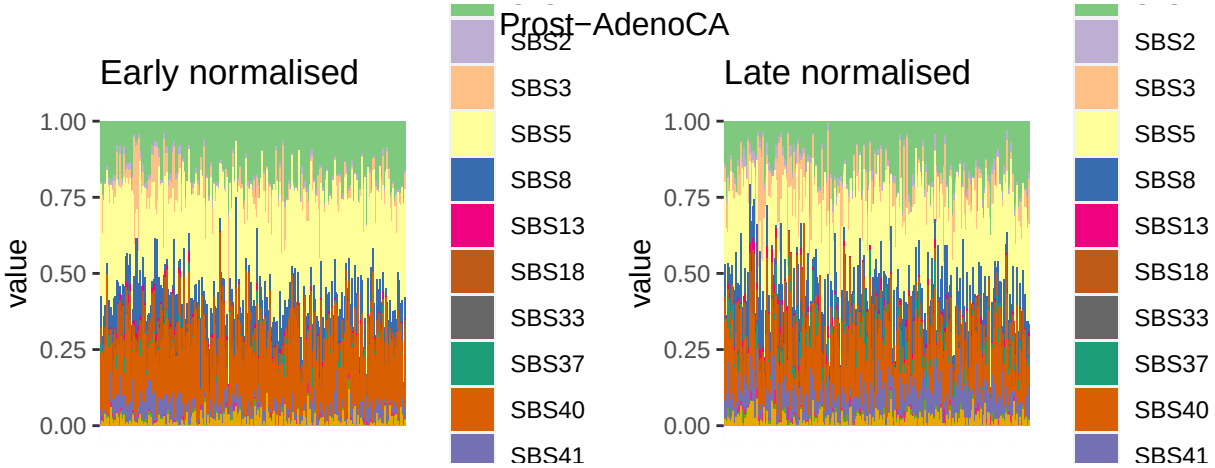

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5627]
```



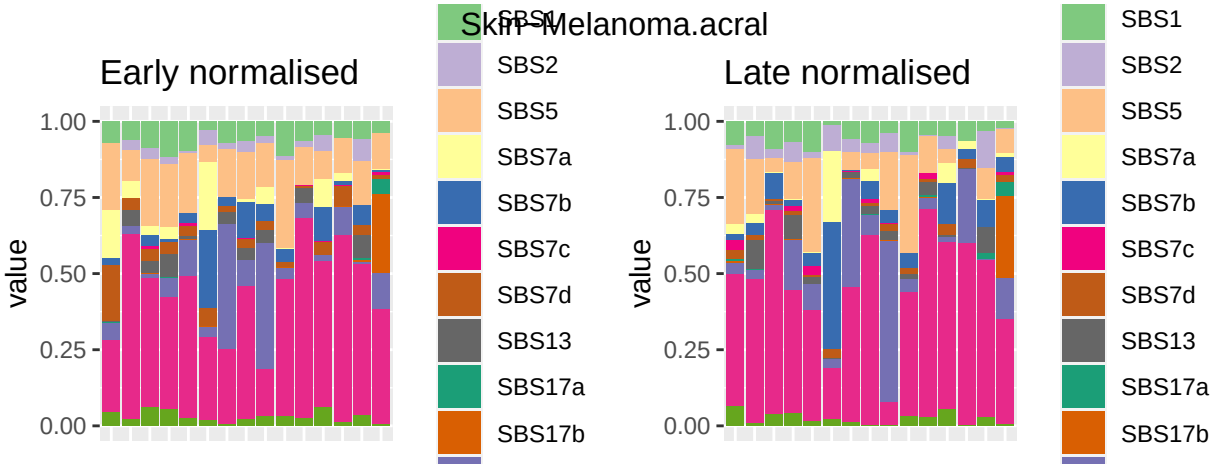

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5928]
```



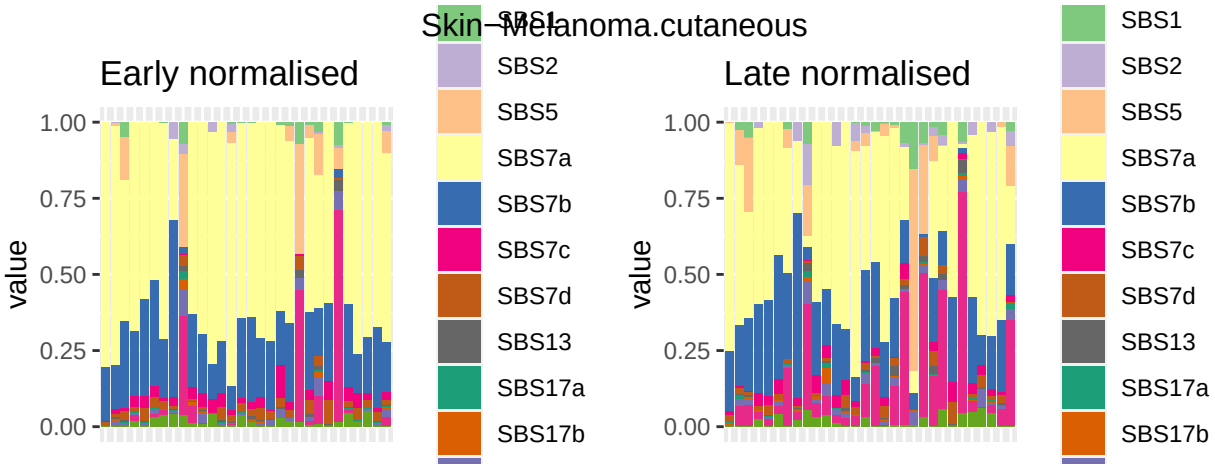

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6187]
```



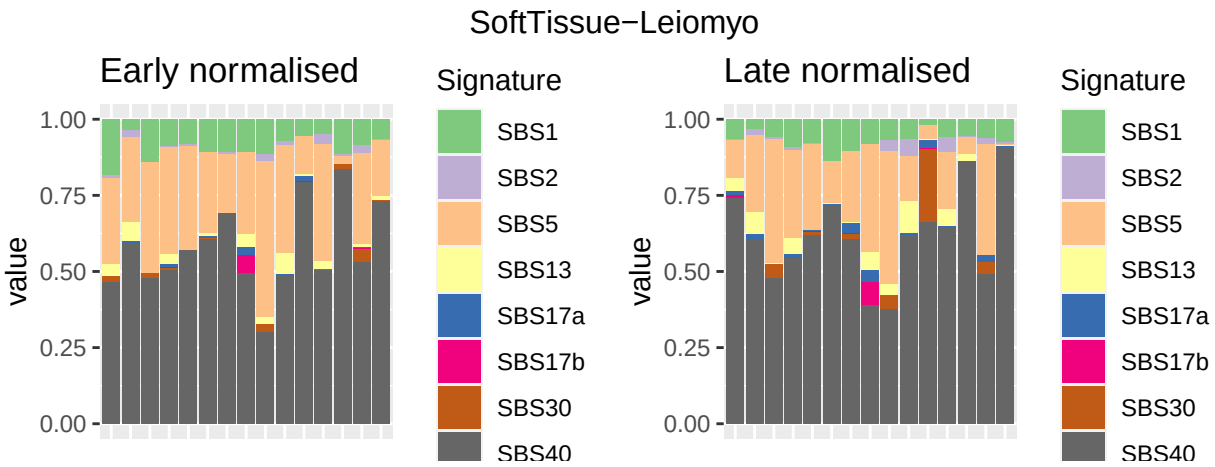

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6460]
```



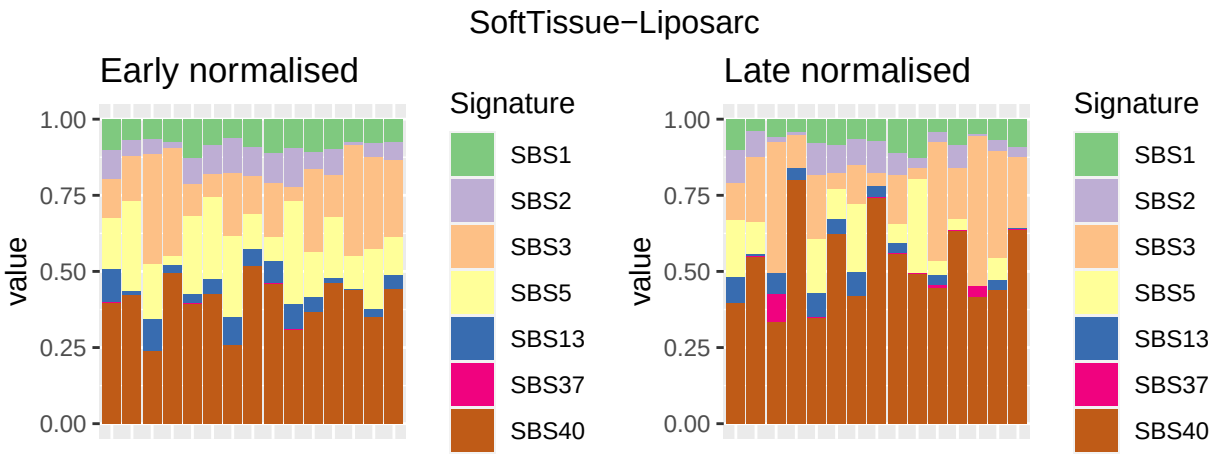

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6719]
```



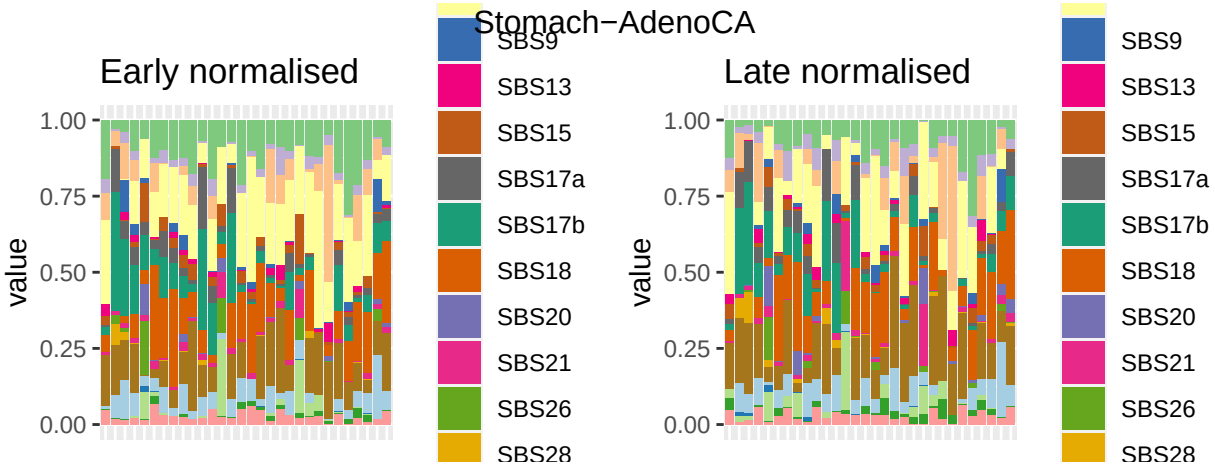

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6978]
## Error : $ operator is invalid for atomic vectors
```



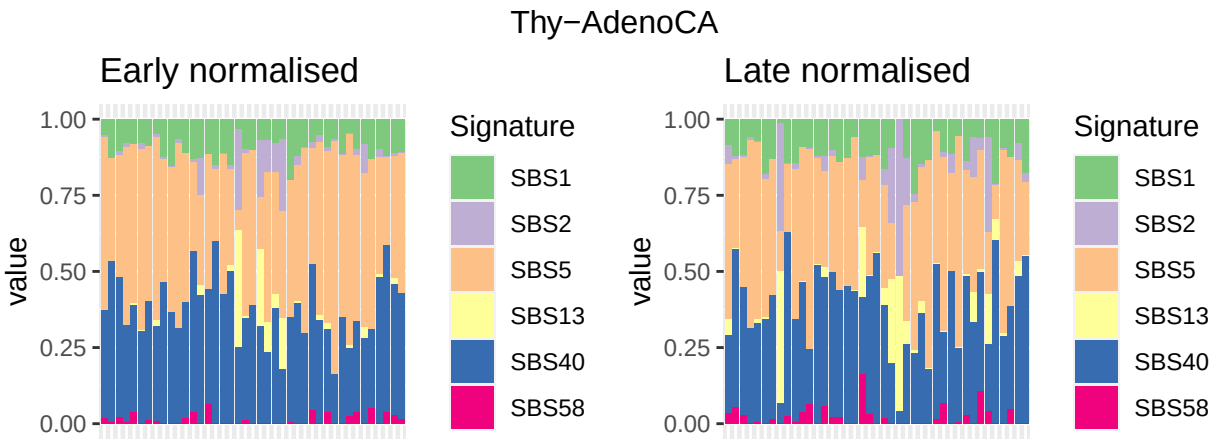

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7167]
```



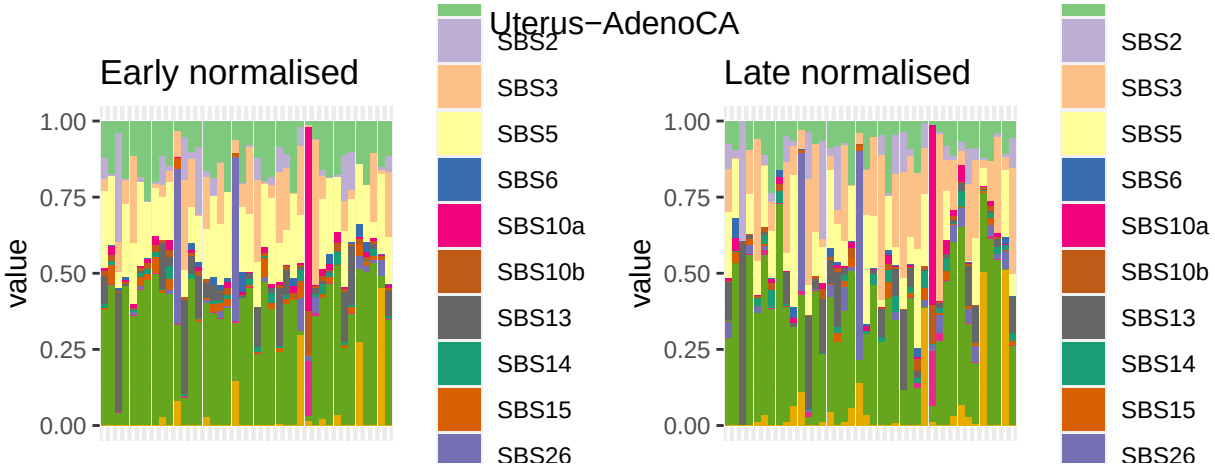

```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7342]
```




```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7699]
```




```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7860]
```




```
## TableGrob (2 x 2) "arrange": 3 grobs
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.8133]
```

```
## $`Biliary-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.328]
##
## $`Bladder-TCC`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.503]
```

```
##
## $`Bone-Benign`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.678]
##
## $`Bone-Epith`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange      gtable[layout]
## 2 2 (2-2,2-2) arrange      gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.853]
##
## $`Bone-Osteosarc`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1070]
##
## $`Breast-AdenoCA`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1329]
##
## $`Breast-DCIS`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1448]
##
## $`Breast-LobularCA`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1609]
##
## $`Cervix-AdenoCA`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1742]
##
## $`Cervix-SCC`
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.1903]
##
## $`CNS-GBM`
##   z     cells    name                 grob
```

```
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2064]
##
## $`CNS-Medullo`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2225]
##
## $`CNS-Oligo`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2358]
##
## $`CNS-PiloAstro`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2505]
##
## $`ColoRect-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2764]
##
## $`Eso-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.2981]
##
## $`Head-SCC`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3268]
##
## $`Kidney-ChRCC`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3457]
##
## $`Kidney-RCC.clearcell`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3646]
```

```
##
## $`Kidney-RCC.papillary`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.3835]
##
## $`Liver-HCC`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4248]
##
## $`Lung-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4493]
##
## $`Lung-SCC`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4654]
##
## $`Lymph-BNHL`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.4927]
##
## $`Lymph-CLL`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5060]
##
## $`Myeloid-AML`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5193]
##
## $`Myeloid-MPN`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5382]
##
## $`Ovary-AdenoCA`
##   z     cells    name                grob
```

```
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5627]
##
## $`Panc-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.5928]
##
## $`Panc-Endocrine`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6187]
##
## $`Prost-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6460]
##
## $`Skin-Melanoma.acral`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6719]
##
## $`Skin-Melanoma.cutaneous`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.6978]
##
## $`Skin-Melanoma.mucosal`
## [1] "Error : $ operator is invalid for atomic vectors\n"
## attr(,"class")
## [1] "try-error"
## attr(,"condition")
## <simpleError: $ operator is invalid for atomic vectors>
##
## $`SoftTissue-Leiomyo`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7167]
##
## $`SoftTissue-Liposarc`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
```

```
## 3 3 (1-1,1-2) arrange text[GRID.text.7342]
##
## $`Stomach-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7699]
##
## $`Thy-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.7860]
##
## $`Uterus-AdenoCA`
##   z     cells    name                grob
## 1 1 (2-2,1-1) arrange       gtable[layout]
## 2 2 (2-2,2-2) arrange       gtable[layout]
## 3 3 (1-1,1-2) arrange text[GRID.text.8133]
```

```
pvals_diagRE_DMDL_SP <- sapply(diagRE_DMDL_SP, function(i) try(wald_TMB_wrapper(i)))
pvals_diagRE_DMDL_nonexo_SP <- sapply(diagRE_DMDL_nonexo_SP, function(i) try(wald_TMB_wrapper(i)))
```

```
## Check data - slope appears to be of lentgh one (binomial)
```

```
pvals_fullRE_M_nonexo_SP <- sapply(fullRE_M_nonexo_SP, function(i) try(wald_TMB_wrapper(i)))
```

```
## Check data - slope appears to be of lentgh one (binomial)
```

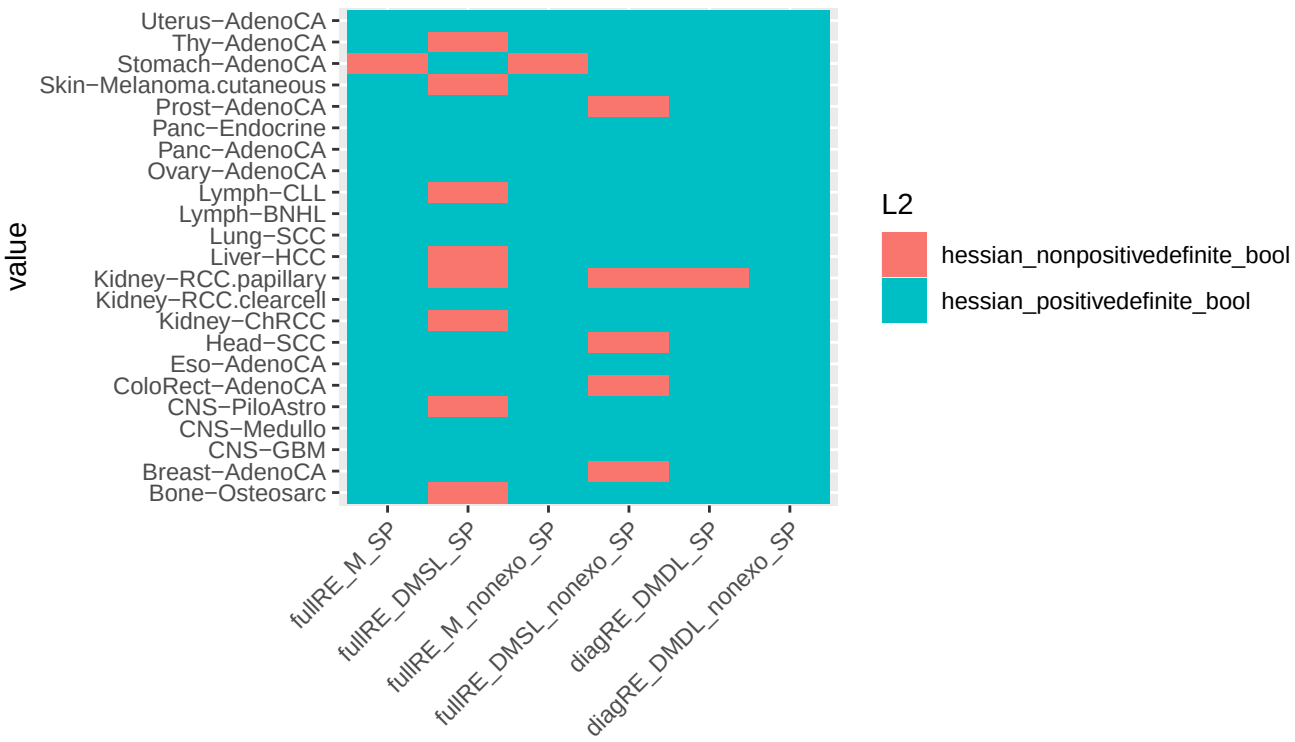
```
pvals_fullRE_DMSL_nonexo_SP <- sapply(fullRE_DMSL_nonexo_SP, function(i) try(wald_TMB_wrapper(i)))
```

```
## Check data - slope appears to be of lentgh one (binomial)
```

```
names(pvals_diagRE_DMDL_SP) <- names(pvals_diagRE_DMDL_nonexo_SP) <- names(pvals_fullRE_M_nonexo_SP) <-
  names(pvals_fullRE_DMSL_nonexo_SP) <- enough_samples
```

```
pvals_diagRE_DMDL_SP
```

```
##          Bone-Osteosarc          Breast-AdenoCA                CNS-GBM
##            1.080828e-04            2.239756e-28            3.390137e-03
##             CNS-Medullo           CNS-PiloAstro        ColoRect-AdenoCA
##            8.431463e-03            5.615238e-04            6.356131e-26
##             Eso-AdenoCA               Head-SCC           Kidney-ChRCC
##            5.329093e-21            4.975610e-05            1.562125e-09
##    Kidney-RCC.clearcell    Kidney-RCC.papillary               Liver-HCC
##            4.027485e-18                      NA           4.747822e-107
##                Lung-SCC             Lymph-BNHL                Lymph-CLL
##            7.747310e-22            3.908637e-19            6.611927e-20
##           Ovary-AdenoCA            Panc-AdenoCA          Panc-Endocrine
##            8.965185e-38           4.096402e-119            3.987099e-10
##           Prost-AdenoCA Skin-Melanoma.cutaneous         Stomach-AdenoCA
##            6.474116e-99            9.272113e-25            1.715150e-06
##             Thy-AdenoCA          Uterus-AdenoCA
##            8.821583e-06            4.819867e-10
```



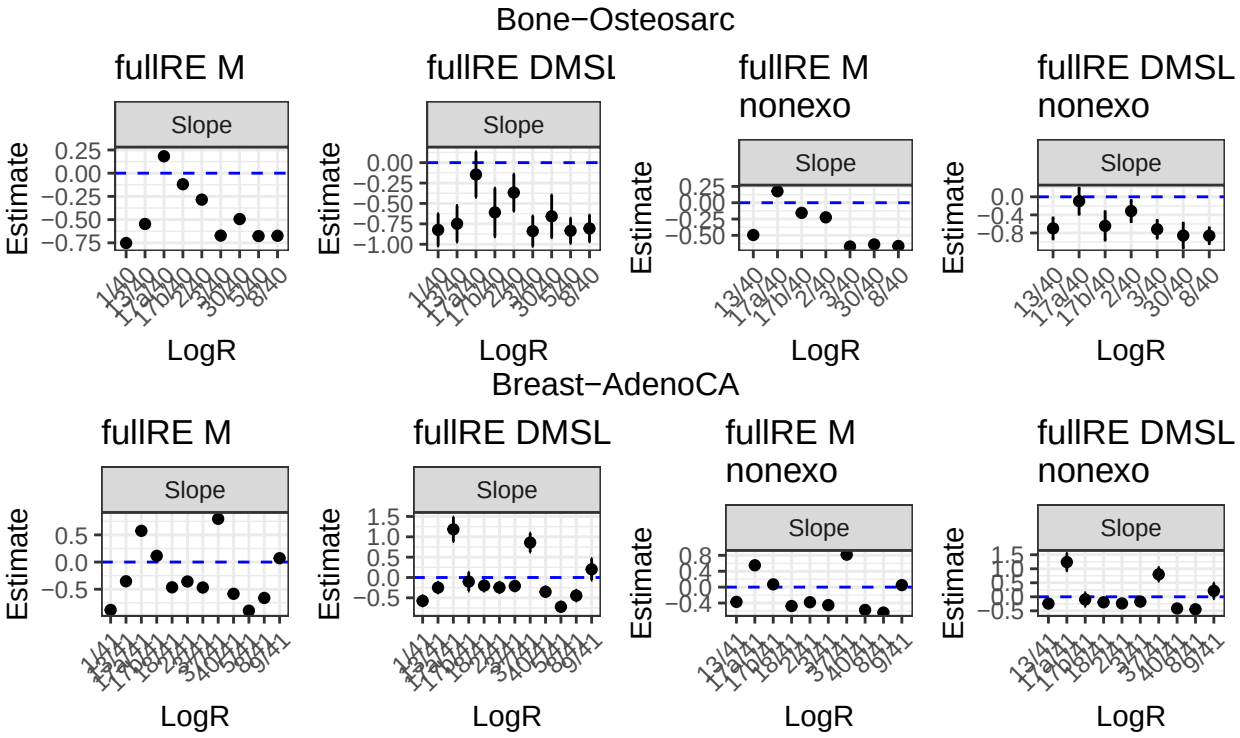
differences between signatures from sigprofiler from the paper, and the ones I get

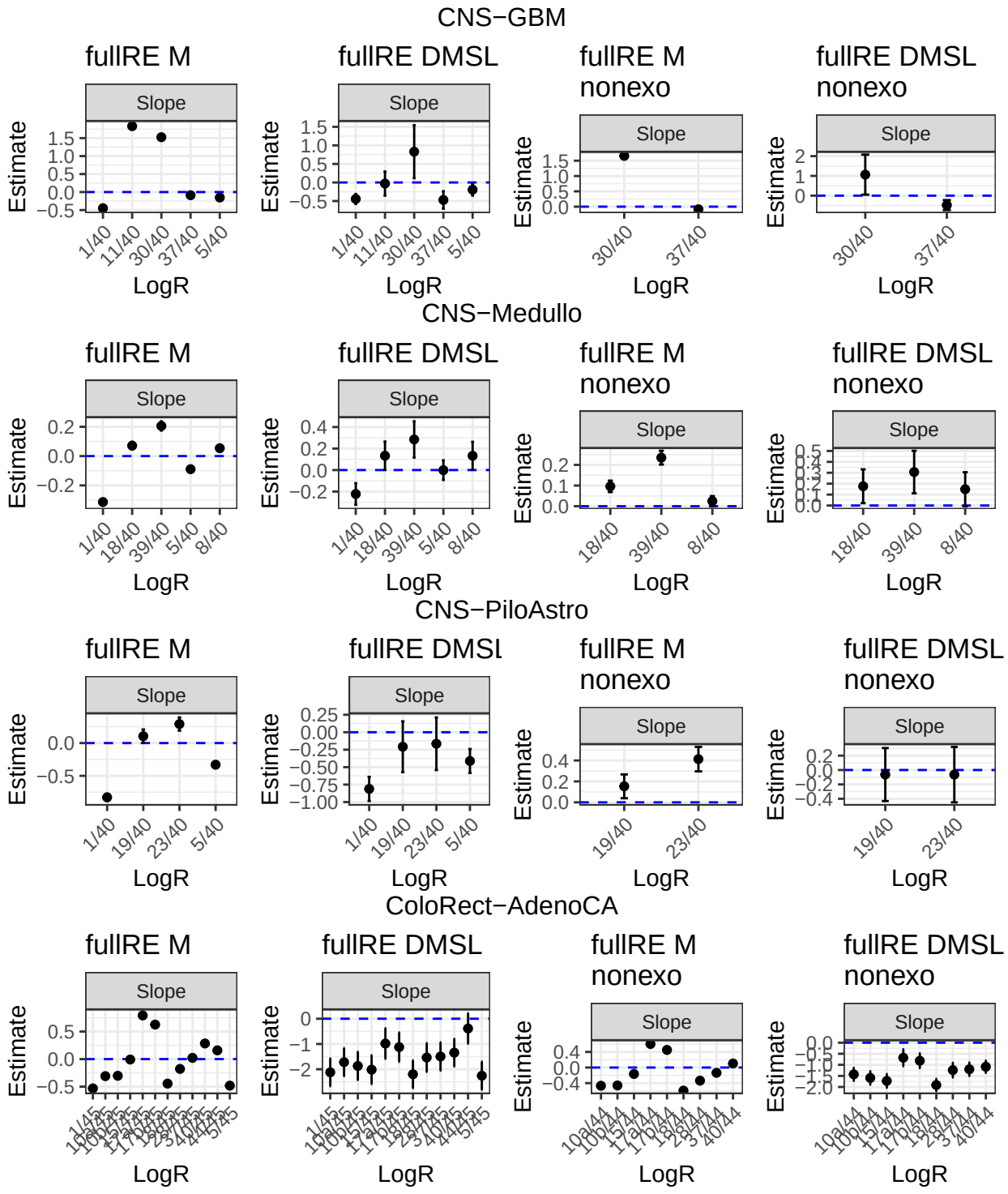
- biliary adenoca similar, though not exact
- bladder tcc: very similar
- bone benign: similar
- bone epith: extremely similar
- bone osteosarc: I have a lot of SBS8, which they don't. Other than that, similar
- breast adenicaL I have a lot of SBS9, which they don't. Other than that, similar
- breast DCIS: I have more SBS40 than they do
- breast lobularca: very similar
- cervix adenoca: very similar
- cervix SCC: very similar, although I have more SBS40
- CNS GBM: very similar, mine seem to be more homogeneous
- CNS medullo: very similar, mine seem to be more homogeneous
- CNS oligo: very similar, mine seem to be more homogeneous
- CNS piloastro: very similar, mine seem to be more homogeneous
- Colorect adenoca: quite similar
- eso adenoca: very similar
- head scc: very similar, mine seem to be more homogeneous
- kidney chrcc: very similar
- kidney rcc.clearcell: very similar, although I have more SBS29
- kidney papillary: only I have it
- liver hcc: different. I have a lot of SBS40 and SBS12, they have mostly SBS5 ***
- lung adenoca: very similar
- lung SCC: very similar, though I have more SBS8
- lymph BNHL: very similar
- Lymph CLL: very similar, althpugh theirs are much more sparse
- myeloid AML: very similar, although I don't have any SBS60 and they seem to have

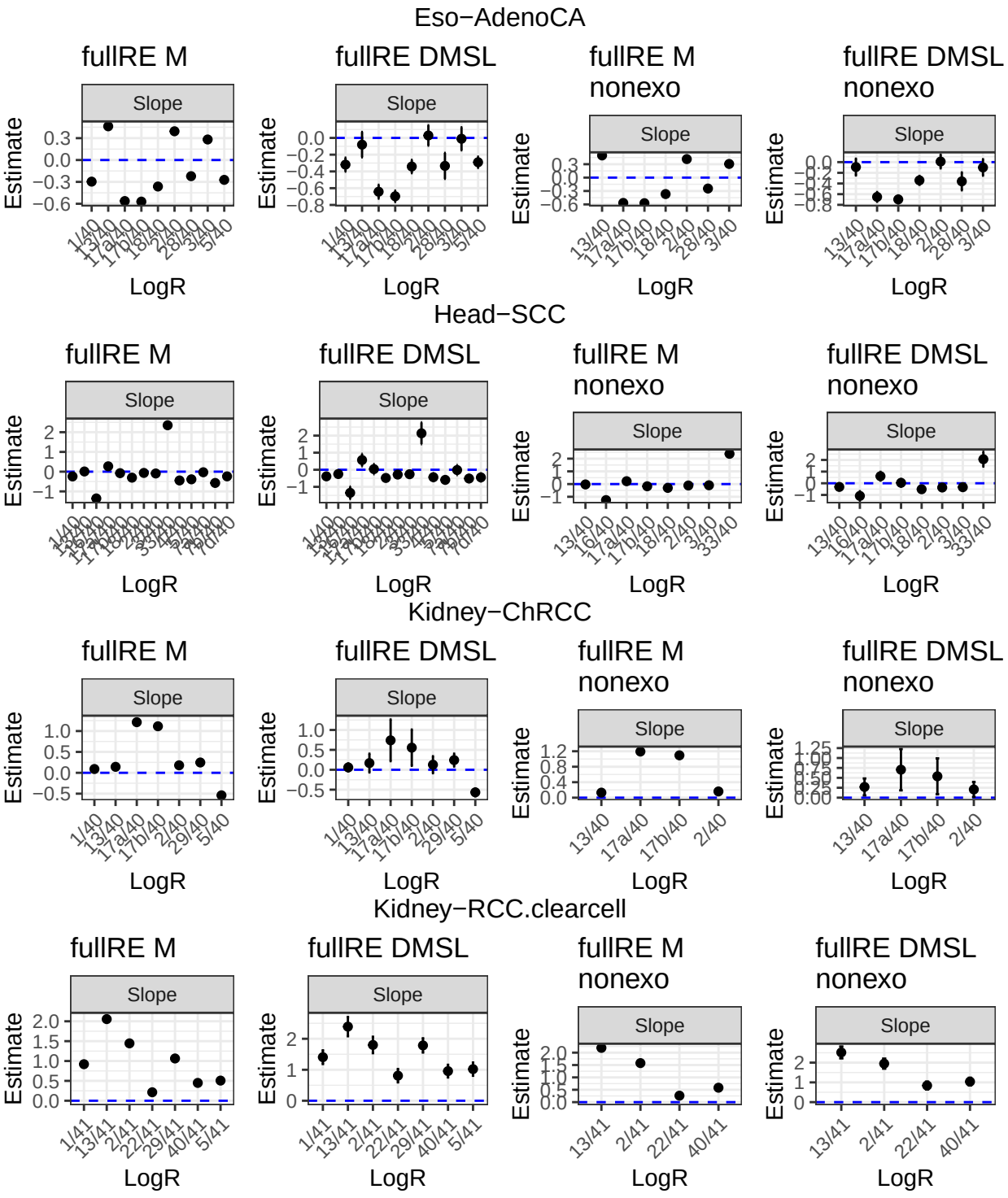
- myeloid MDS: I don't have it
- myeloid MPN: similar, although mine is much more sparse
- ovary adenoca: different. I have a lot of SBS40, which in their case is rare, and they have much more of SBS3 than I do ***
- panc-adenoca: different. I have a lot of SBS8 that they don't have. ***
- panc-endocrine: sort of similar. I have more SBS8 than they and they have more SBS5
- Prost-adenoca: sort of similar, I have more SBS8
- skin-melanoma.acral: they don't have this category. They have "skin-melanoma", which might be both together? (!!!) Similar exposures...
- softtissue-leiomyo: very similar exposures
- softtissue-liposarc: very similar exposures
- stomach adenoca: very similar, mine seem to be more homogeneous
- thy-adenoca: very similar
- uterus-adenoca: very similar

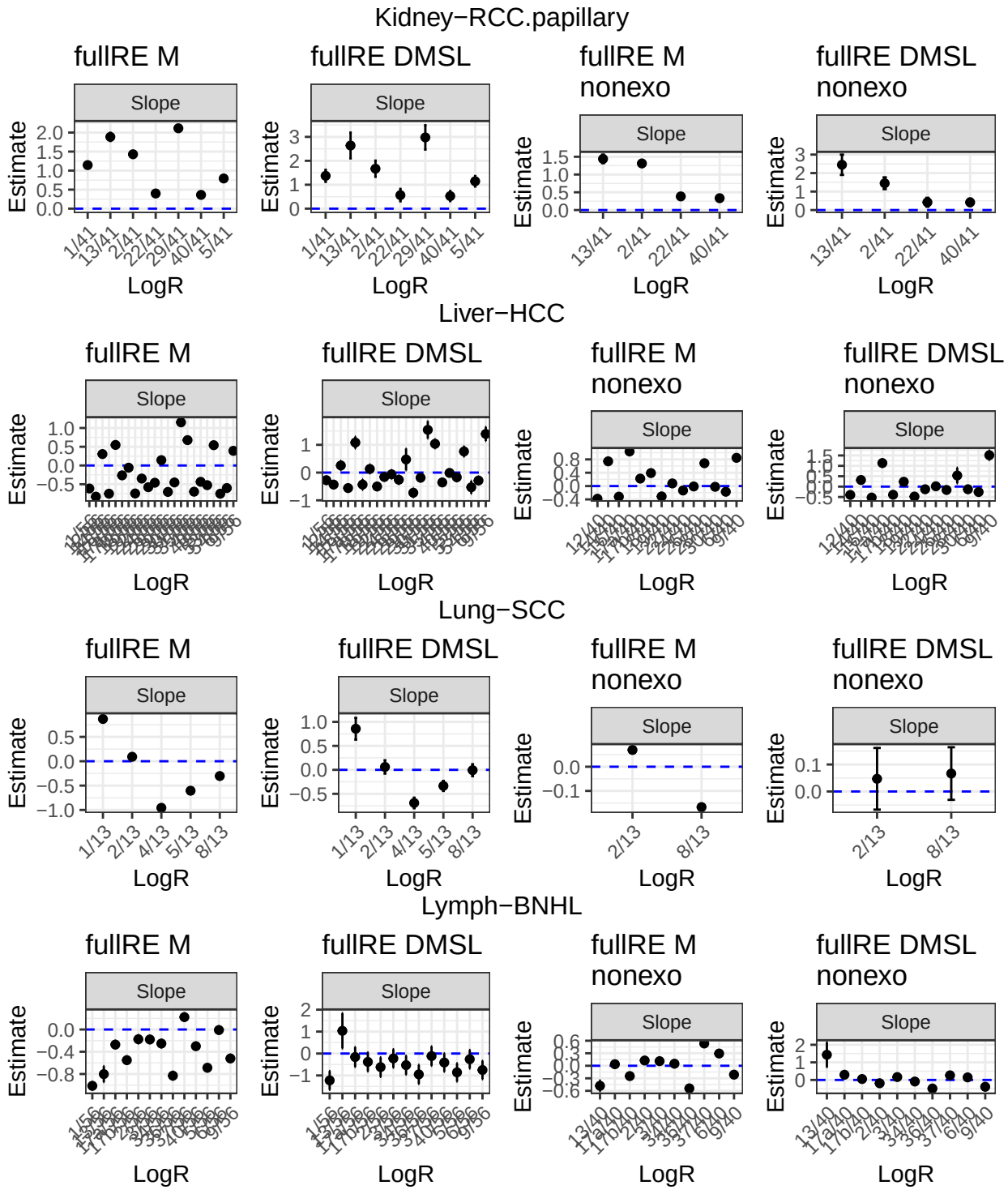
Betas from PCAWG subset of signatures

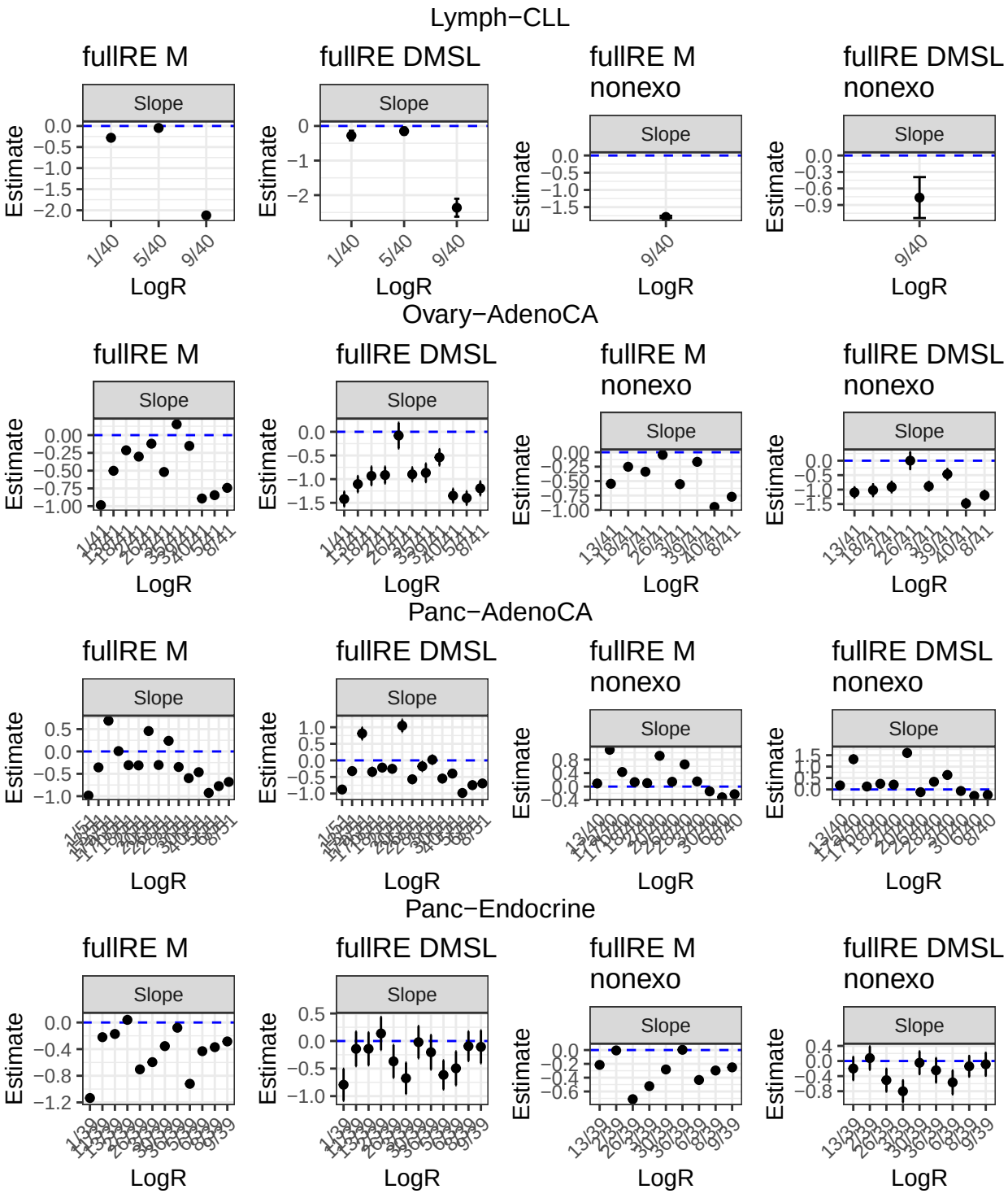
Reminder that the clonesig signatures were a subset of WES data.

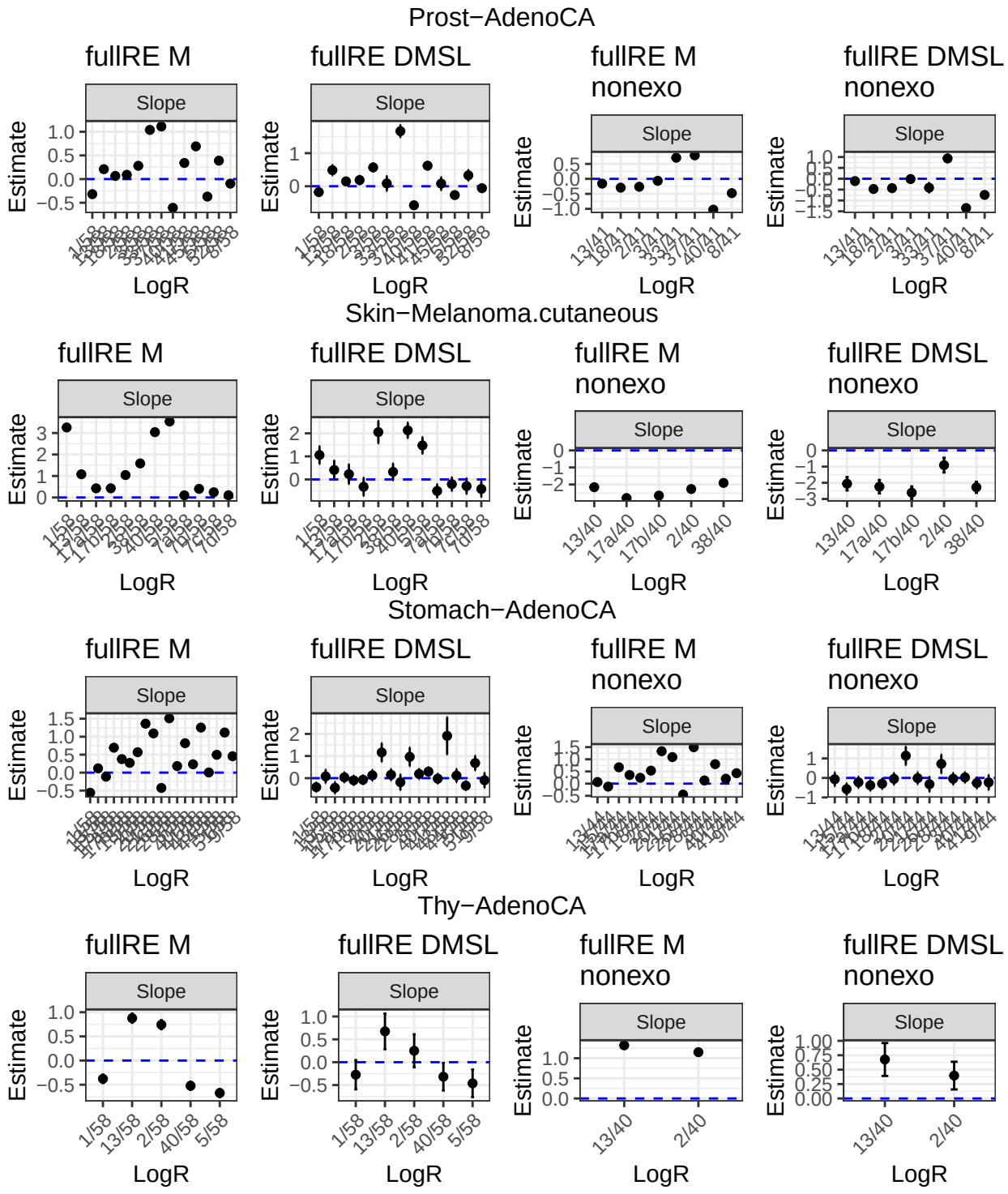


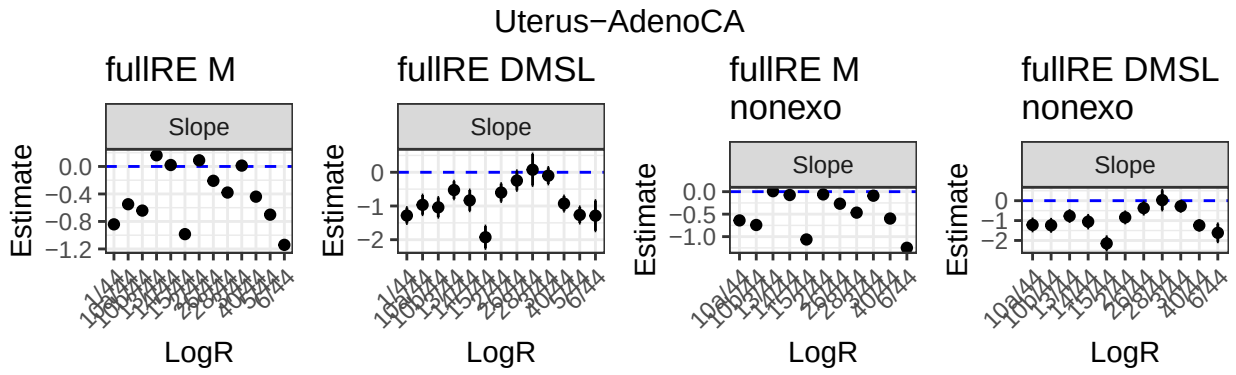

CNS−GBM
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
CNS−Medullo
CNS−PiloAstro
ColoRect−AdenoCA


Eso−AdenoCA
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
Head−SCC
Kidney−ChRCC
Kidney−RCC.clearcell


Kidney−RCC.papillary
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Liver−HCC
Lung−SCC
Lymph−BNHL
Slope
Estimate
LogR

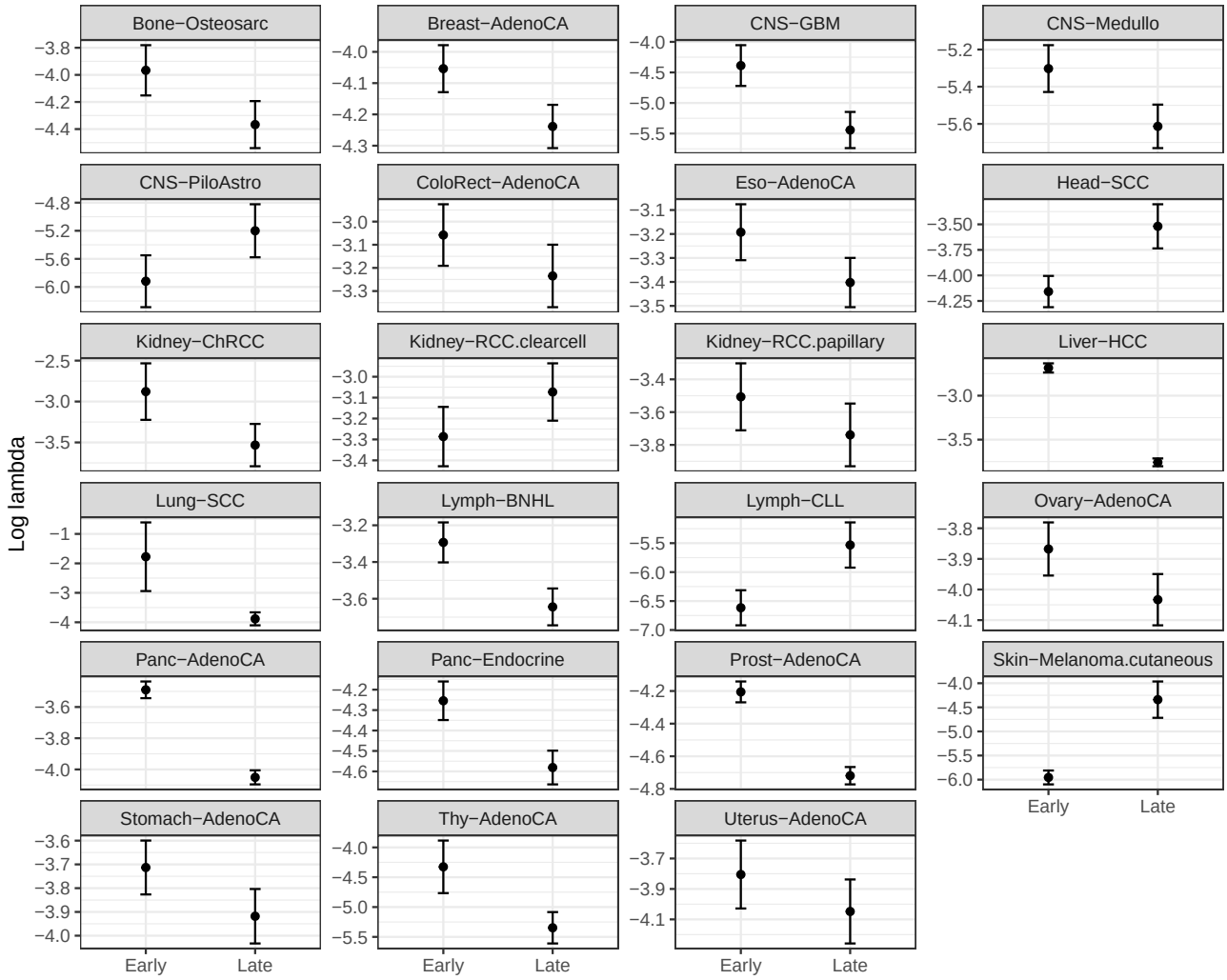

Lymph−CLL
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
Ovary−AdenoCA
Panc−AdenoCA
Panc−Endocrine


Prost−AdenoCA
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
Skin−Melanoma.cutaneous
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
13/40
17a/40
17b/40
2/40
38/40
Stomach−AdenoCA
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
Thy−AdenoCA
fullRE M
fullRE DMSL
fullRE M nonexo
fullRE DMSL nonexo
Slope
Estimate
LogR
1/58
13/58
2/58
40/58
5/58
13/40
2/40

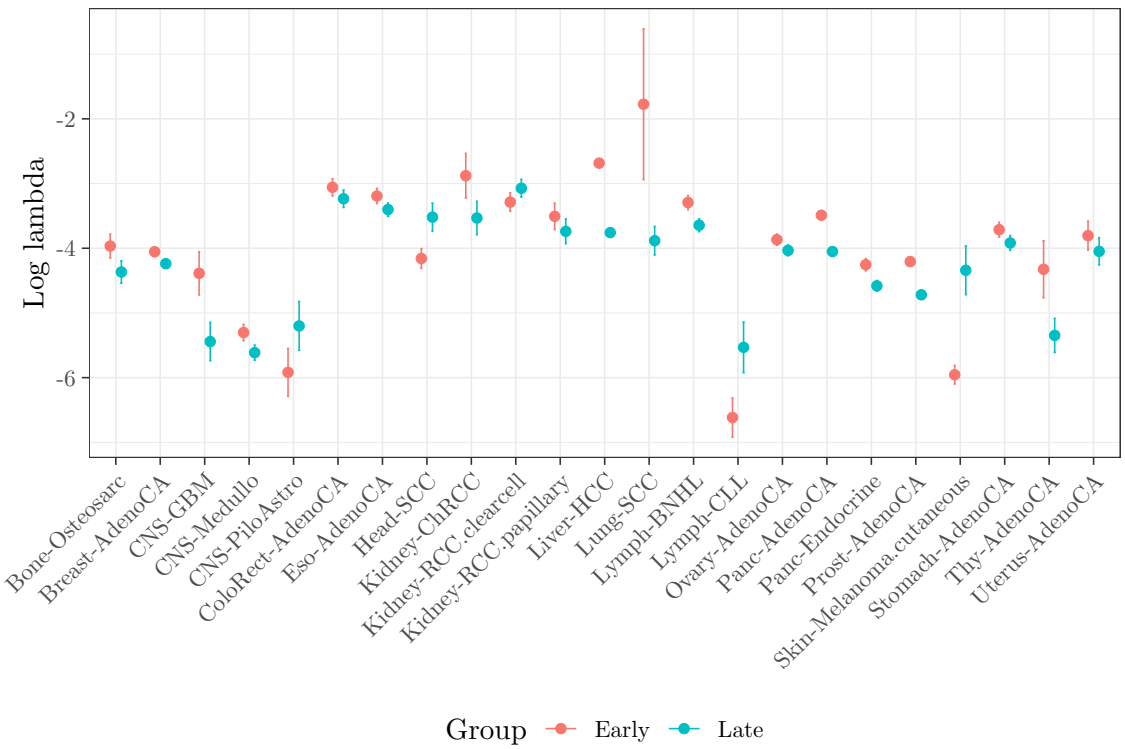


Overdispersion parameters in double-lambda DM

```
ovrdisp <- do.call('rbind.data.frame', lapply(1:length(diagRE_DMDL_nonexo_SP), try(function(idx){
  if(diagRE_DMDL_nonexo_SP[[idx]]$pdHess){
    cbind.data.frame( plot_lambdas(diagRE_DMDL_nonexo_SP[[idx]], return_df=T, plot=F), ct=names(diagRE_DM
  }else{
    c(NA, NA)
  }
})))
ovrdisp[ovrdisp$name == 'Lambda 1','name'] = 'Early'
ovrdisp[ovrdisp$name == 'Lambda 2','name'] = 'Late'

ggplot(ovrdisp, aes(x=name,  y=`Estimate`))+
  geom_point()+
  geom_errorbar(aes(ymin=`Estimate`-`Std..Error`,
                    ymax=`Estimate`+`Std..Error`), width=.1)+
  theme_bw()+
  facet_wrap(.~ct, scales = "free_y", nrow=6)+
  labs(x='', y='Log lambda')
```




```
ggplot(ovrdisp, aes(x=ct,  y=`Estimate`, group=name, col=name))+
  geom_point(position=position_dodge(width=0.5))+
  geom_errorbar(aes(ymin=`Estimate`-`Std..Error`,
                    ymax=`Estimate`+`Std..Error`), width=.1, position=position_dodge(width=0.5))+
  theme_bw()+
  theme(axis.text.x=element_text(angle = 45, hjust = 1, vjust=1))+
  labs(x='', y='Log lambda', col='Group')+theme(legend.position = "bottom")+
    theme(         legend.margin=margin(0,0,0,0),
                   legend.box.margin=margin(-10,-10,-10,-10),
                   plot.margin = unit(c(1,1,1,1), "cm"))
```



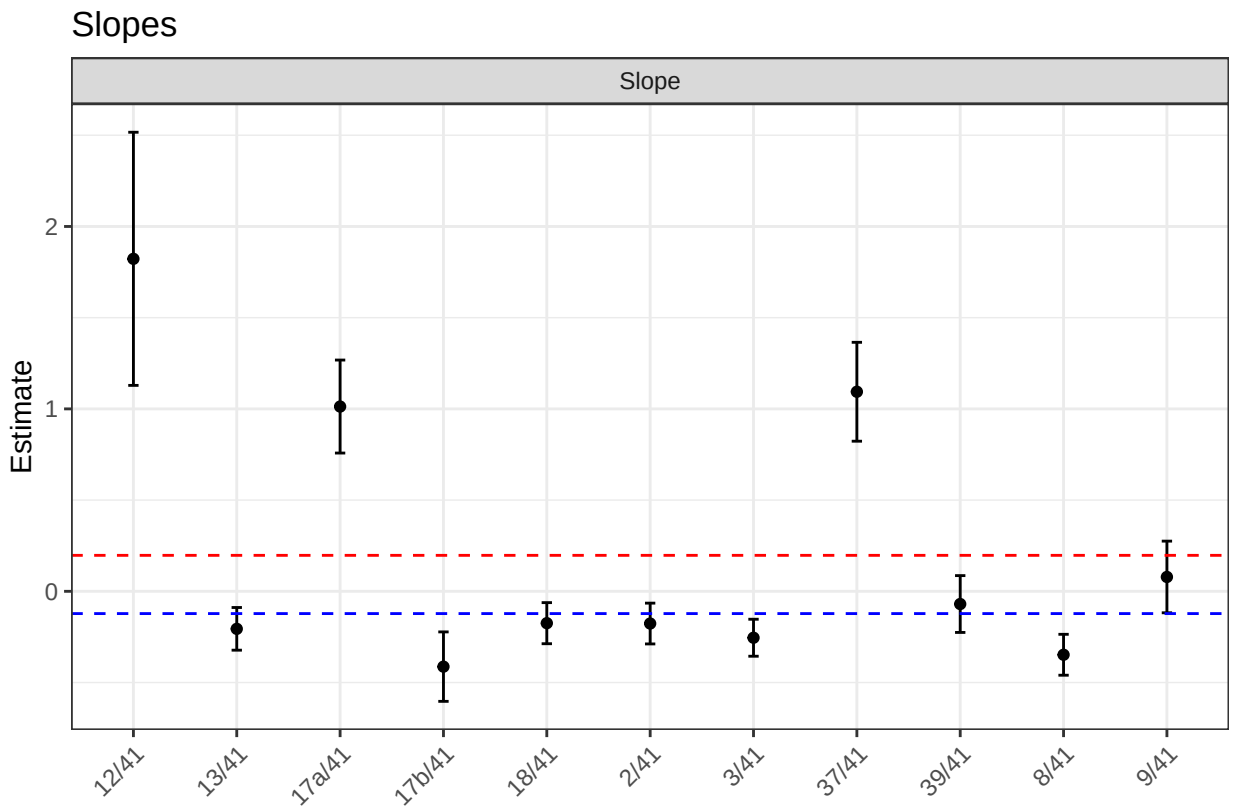
Minimal perturbation

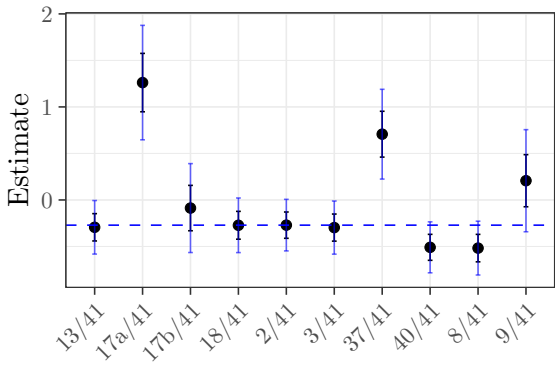
Note: this is using the original, non SP, signatures (hence not the best version).

```
minimalpert_L2 <- function(i){
  sum(i)/sum(i^2)
}

betas_breast <- data.frame(plot_betas(diagRE_DMSL_nonexo[["Breast-AdenoCA"]], names_cats= logR_nonexo_not
                           return_df=T, plot=F))

.slopes_minpert <- betas_breast %>% dplyr::filter(type_beta == "Slope") %>% dplyr::select(Estimate) %>% u

# minimalpert_L2(softmax(c(.slopes_minpert, 0)))
median(c(.slopes_minpert, 0))
```

```
## [1] -0.1222111
```

```
aa <- plot_betas(diagRE_DMSL_nonexo[["Breast-AdenoCA"]], names_cats= logR_nonexo_notsorted[["Breast-Adeno
                           return_df=F, plot=F, only_slope = T, line_zero=F)
# aa <- geom_hline(yintercept = 0)+geom_vline(xintercept = 1)+geom_hline(yintercept = median(c(.slopes_m
aa + geom_hline(yintercept = median(c(.slopes_minpert, 0)), lty='dashed', col='blue')+
  geom_hline(yintercept = mean(c(.slopes_minpert, 0)), lty='dashed', col='red')
```





Log ratio of signatures

For the thesis:

```
pdf("../../results/results_TMB/pcawg/minimalperturbation_SP_all.pdf", width = 4, height = 3)
for(ct in names(diagRE_DMDL_nonexo_SP)){
  .betas_ct_it <- data.frame(plot_betas(TMB_obj = diagRE_DMDL_nonexo_SP[[ct]],
                                        names_cats= logR_nonexo_notsorted_SP[[ct]],
                                return_df=T, plot=F))
  .slopes_minpert <- .betas_ct_it %>% dplyr::filter(type_beta == "Slope") %>% dplyr::select(Estimate) %>%
  print(aaa <- plot_betas(TMB_obj = diagRE_DMDL_nonexo_SP[[ct]], names_cats= logR_nonexo_notsorted_SP[[ct
                                return_df=F, plot=F, only_slope = T, line_zero=F, add_confint = T)+
          geom_hline(yintercept = median(c(.slopes_minpert, 0)), lty='dashed', col='blue')+ggtitle(ct))
}
dev.off()
```

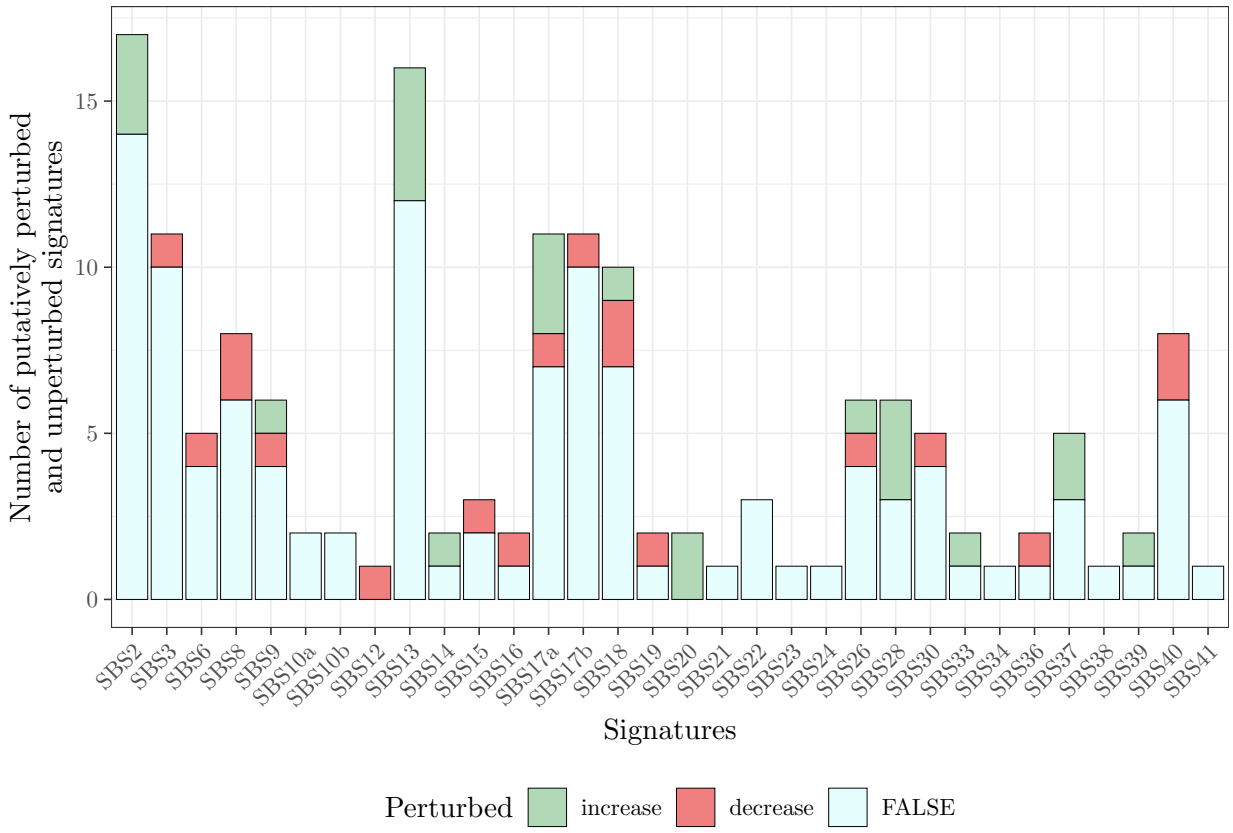
```
## pdf
##   2
```

\subsection{Minimal perturbation in diagRE_DMDL_nonexo_S}

```
perturbed_betas_diagRE_DMDL_nonexo_SP <- lapply(names(diagRE_DMDL_nonexo_SP), try(function(idx_sp){
  .betas_SP <- data.frame(plot_betas(diagRE_DMDL_nonexo_SP[[idx_sp]], names_cats= logR_nonexo_notsorted_S
                             return_df=T, plot=F))

  .slopes_minpert_SP <- .betas_SP %>% dplyr::filter(type_beta == "Slope") %>% dplyr::select(Estimate) %>%
  # print(.slopes_minpert_SP)
  ## check if the CI of the betas touches this median value
  .summary_betas_slope_SP <- python_like_select_rownames(summary(diagRE_DMDL_nonexo_SP[[idx_sp]]), 'beta'
  nrow(.summary_betas_slope_SP)

  minimal_change_baseline <- median(c(.slopes_minpert_SP, 0))
  # print(.summary_betas_slope_SP)
  # print(logR_nonexo_notsorted_SP[[idx_sp]])
  # print(dim(.summary_betas_slope_SP))
  if(!is.null(dim(.summary_betas_slope_SP))){
    .params_in_ci <- give_params_in_CI(vec_est=.summary_betas_slope_SP[,1],
                      vec_stderr=.summary_betas_slope_SP[,2],
                      vec_true=rep(minimal_change_baseline, nrow(.summary_betas_slope_SP)))
  }else{
    .params_in_ci <- give_params_in_CI(vec_est=.summary_betas_slope_SP[1],
                      vec_stderr=.summary_betas_slope_SP[2],
                      vec_true=minimal_change_baseline)
  }
  .params_in_ci <- sapply(1:length(.params_in_ci), function(i){
      ## for the ones in which there is a change, say whether it's up- or down-regulated
      if(!.params_in_ci[i]){
        ## if there is a change: not in confidence interval
        if(is.null(dim(.summary_betas_slope_SP))){
          ## one-dim
          if(.summary_betas_slope_SP[1] > minimal_change_baseline){
            'increase'
          }else{
            'decrease'
          }
        }else{
          ## multi-dim
          if(.summary_betas_slope_SP[i,1] > minimal_change_baseline){
            'increase'
          }else{
            'decrease'
          }
        }
      }else{
        'FALSE'
      }
    })
  names(.params_in_ci) <- sapply(logR_nonexo_notsorted_SP[[idx_sp]], function(i) strsplit(i, '/')[[1]][1]
```

```
  .baseline <- strsplit(logR_nonexo_notsorted_SP[[idx_sp]][[1]], '/')[[1]][2]
  return(list(betas_perturbed=.params_in_ci, baseline=.baseline))
}))
```

```
perturbed_betas_diagRE_DMDL_nonexo_SP_vec <- do.call('c', sapply(perturbed_betas_diagRE_DMDL_nonexo_SP, f

perturbed_betas_diagRE_DMDL_nonexo_SP_df <- cbind.data.frame(sig=gsub("betas_perturbed.", "",
                                                                      names(perturbed_betas_diagRE_DMDL_r
                                                                    perturbed=perturbed_betas_diagRE_DMDL_nonexo
ggplot(perturbed_betas_diagRE_DMDL_nonexo_SP_df, aes(x=factor(sig, levels=gtools::mixedsort(unique(sig)))
                                                     fill=factor(perturbed, levels=c('increase', 'decreas
  geom_bar(col='black', size=0.001)+theme_bw()+
  scale_fill_manual(values=c( '#b1d8b7', '#f08080', '#e7feff'))+
  theme(axis.text.x=element_text(angle = 45, hjust = 1, vjust=1), legend.position = "bottom")+
  labs(x='Signatures', y='Number of putatively perturbed\n and unperturbed signatures', fill='Perturbed')
```



Minimal perturbation per signature

```
names(perturbed_betas_diagRE_DMDL_nonexo_SP) <- names(diagRE_DMDL_nonexo_SP)
ggplot(reshape2::melt(lapply(perturbed_betas_diagRE_DMDL_nonexo_SP, `[`, 'betas_perturbed')),
       aes(x=L1, fill=factor(value, levels=c('increase', 'decrease', 'FALSE'))))+geom_bar(col='black', si
        scale_fill_manual(values=c('#b1d8b7', '#f08080', '#e7feff'))+theme(legend.position = "bottom")+
    theme(axis.text.x=element_text(angle = 45, hjust = 1, vjust=1))+labs(x='Cancer type',
                                                                         y='Number of putatively perturbe
```

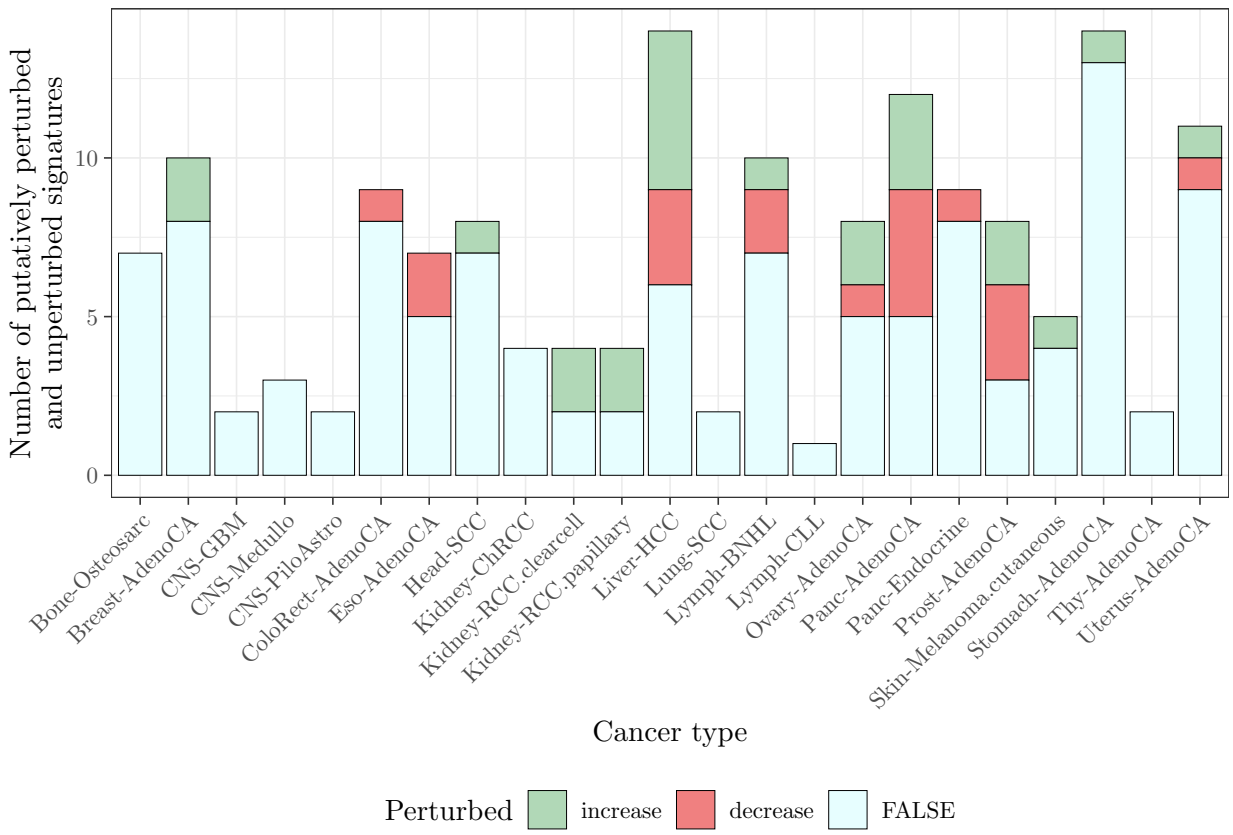


Table of perturbed signatures

```
write("", "../../results/results_TMB/pcawg/minimal_perturbation_sigs.txt", append = F)
for(i in names(perturbed_betas_diagRE_DMDL_nonexo_SP)){
  write(paste0(i, '&', paste0(names(perturbed_betas_diagRE_DMDL_nonexo_SP[[i]]$betas_perturbed)[perturbed
}
```

```
## [1] 27  8
## [1] 27  8
## [1] 136  11
## [1] 136  11
## [1] 34  3
## [1] 34  3
## [1] 106   4
## [1] 106   4
## [1] 41  3
## [1] 41  3
## [1] 37 10
## [1] 37 10
## [1] 65  8
## [1] 65  8
## [1] 32  9
## [1] 32  9
## [1] 38  5
## [1] 38  5
## [1] 86  5
```

```
## [1] 86  5
## [1] 30  5
## [1] 30  5
## [1] 207  15
## [1] 207  15
## [1] 34  3
## [1] 34  3
## [1] 51 11
## [1] 51 11
## [1] 53  2
## [1] 53  2
## [1] 97  9
## [1] 97  9
## [1] 193  13
## [1] 193  13
## [1] 70 10
## [1] 70 10
## [1] 208   9
## [1] 208   9
## [1] 29  6
## [1] 29  6
## [1] 30 15
## [1] 30 15
## [1] 41  3
## [1] 41  3
## [1] 40 12
## [1] 40 12
```

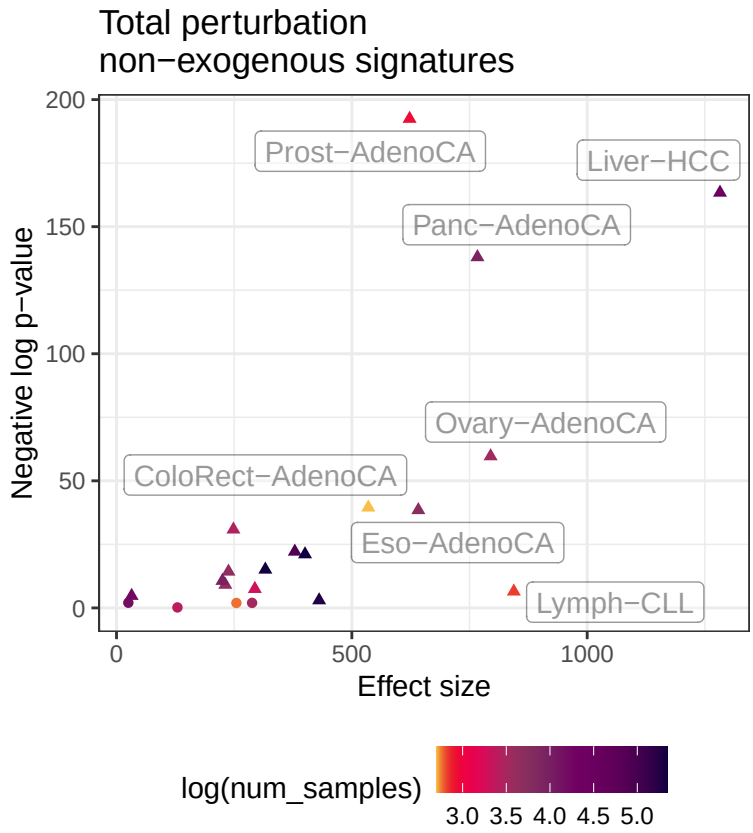
```
##          Bone-Osteosarc          Breast-AdenoCA                 CNS-GBM
##                       8                      11                       3
##             CNS-Medullo           CNS-PiloAstro        ColoRect-AdenoCA
##                       4                       3                      10
##             Eso-AdenoCA               Head-SCC            Kidney-ChRCC
##                       8                       9                       5
##    Kidney-RCC.clearcell    Kidney-RCC.papillary               Liver-HCC
##                       5                       5                      15
##                Lung-SCC             Lymph-BNHL              Lymph-CLL
##                       3                      11                       2
##           Ovary-AdenoCA           Panc-AdenoCA         Panc-Endocrine
##                       9                      13                      10
##           Prost-AdenoCA Skin-Melanoma.cutaneous         Stomach-AdenoCA
##                       9                       6                      15
##             Thy-AdenoCA          Uterus-AdenoCA
##                       3                      12
```

```
## Error in apply(pert, 1, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
## Error in apply(pert, 1, function(i) sqrt(sum((i - 1/(ncol(exposures_cancertype_obj$Y)))^2))) :
##   dim(X) must have a positive length
```

```
## Warning: NAs introduced by coercion
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
```

```
## "none")` instead.

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_label_repel).

## Warning: ggrepel: 14 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
##          Bone-Osteosarc          Breast-AdenoCA                 CNS-GBM
##                320.02704                469.29125              524.08517
##              CNS-Medullo           CNS-PiloAstro       ColoRect-AdenoCA
##                217.31161                 27.02067             1031.46320
##              Eso-AdenoCA                 Head-SCC          Kidney-ChRCC
##                522.66804                221.26052              103.16727
##    Kidney-RCC.clearcell    Kidney-RCC.papillary                Liver-HCC
##                247.53963                156.30958             1332.22216
##                 Lung-SCC              Lymph-BNHL                Lymph-CLL
##                 86.07143                345.88535              442.78303
##           Ovary-AdenoCA             Panc-AdenoCA          Panc-Endocrine
##                755.21919                829.19711              220.55650
##           Prost-AdenoCA Skin-Melanoma.cutaneous          Stomach-AdenoCA
##                459.55271              1017.96421              540.50633
##              Thy-AdenoCA           Uterus-AdenoCA
##                 82.57481                458.24059

##          Bone-Osteosarc          Breast-AdenoCA                 CNS-GBM
##            1.080828e-04            2.239756e-28            3.390137e-03
```

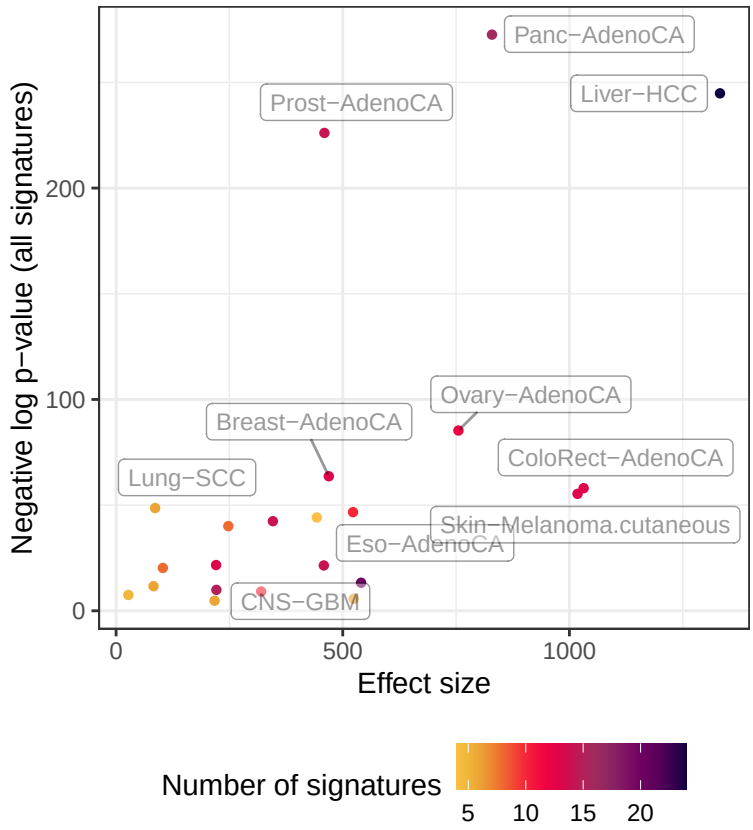
```
##             CNS-Medullo           CNS-PiloAstro        ColoRect-AdenoCA
##            8.431463e-03            5.615238e-04            6.356131e-26
##             Eso-AdenoCA                Head-SCC            Kidney-ChRCC
##            5.329093e-21            4.975610e-05            1.562125e-09
##    Kidney-RCC.clearcell    Kidney-RCC.papillary               Liver-HCC
##            4.027485e-18                      NA           4.747822e-107
##               Lung-SCC             Lymph-BNHL               Lymph-CLL
##            7.747310e-22            3.908637e-19            6.611927e-20
##           Ovary-AdenoCA            Panc-AdenoCA          Panc-Endocrine
##            8.965185e-38           4.096402e-119            3.987099e-10
##           Prost-AdenoCA Skin-Melanoma.cutaneous         Stomach-AdenoCA
##            6.474116e-99            9.272113e-25            1.715150e-06
##             Thy-AdenoCA          Uterus-AdenoCA
##            8.821583e-06            4.819867e-10
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

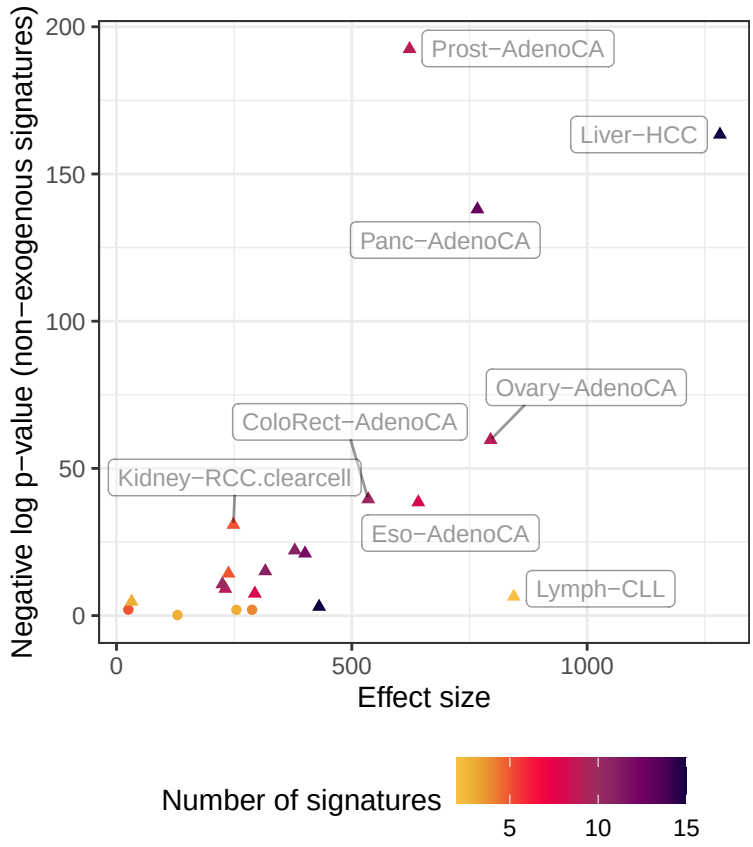
```
## Warning: Removed 1 rows containing missing values (geom_label_repel).
```

```
## Warning: ggrepel: 12 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
## Warning: Removed 2 rows containing missing values (geom_label_repel).
## Warning: ggrepel: 13 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



HMP

Simple HMP tests to see what is differentially abundant

```
p.adjust(pvals_diagRE_DMDL_SP, method = "BH")
```

```
##           Bone-Osteosarc          Breast-AdenoCA                 CNS-GBM
##             1.251484e-04            9.854928e-28            3.551572e-03
##              CNS-Medullo           CNS-PiloAstro        ColoRect-AdenoCA
##             8.431463e-03            6.176762e-04            2.330581e-25
##              Eso-AdenoCA                Head-SCC             Kidney-ChRCC
##             1.302667e-20            6.081301e-05            2.291117e-09
##    Kidney-RCC.clearcell    Kidney-RCC.papillary               Liver-HCC
##             7.383723e-18                      NA           5.222605e-106
##                Lung-SCC              Lymph-BNHL                 Lymph-CLL
##             2.130510e-21            7.817274e-19            1.454624e-19
##           Ovary-AdenoCA           Panc-AdenoCA          Panc-Endocrine
##             4.930852e-37           9.012085e-118            6.747398e-10
##           Prost-AdenoCA Skin-Melanoma.cutaneous          Stomach-AdenoCA
```

```
##            4.747685e-98            2.914093e-24            2.358331e-06
##             Thy-AdenoCA          Uterus-AdenoCA
##            1.141617e-05            7.574077e-10
```

```
p.adjust(pvals_diagRE_DMDL_nonexo_SP, method = "BH")
```

```
##          Bone-Osteosarc          Breast-AdenoCA                 CNS-GBM
##            8.712593e-04            6.123721e-10            1.557247e-01
##             CNS-Medullo           CNS-PiloAstro        ColoRect-AdenoCA
##            1.557247e-01            5.270878e-01            3.141298e-17
##             Eso-AdenoCA                Head-SCC            Kidney-ChRCC
##            7.328650e-17            1.788732e-04            1.557247e-01
##    Kidney-RCC.clearcell    Kidney-RCC.papillary               Liver-HCC
##            1.137607e-13            1.190455e-06            1.263498e-70
##                Lung-SCC              Lymph-BNHL               Lymph-CLL
##            8.301558e-01            5.881884e-07            2.256505e-03
##           Ovary-AdenoCA            Panc-AdenoCA          Panc-Endocrine
##            6.973340e-26            8.893577e-60            4.169125e-05
##           Prost-AdenoCA Skin-Melanoma.cutaneous         Stomach-AdenoCA
##            6.044387e-83            9.919659e-17            6.320633e-02
##             Thy-AdenoCA          Uterus-AdenoCA
##            1.170750e-02            1.568961e-09
```

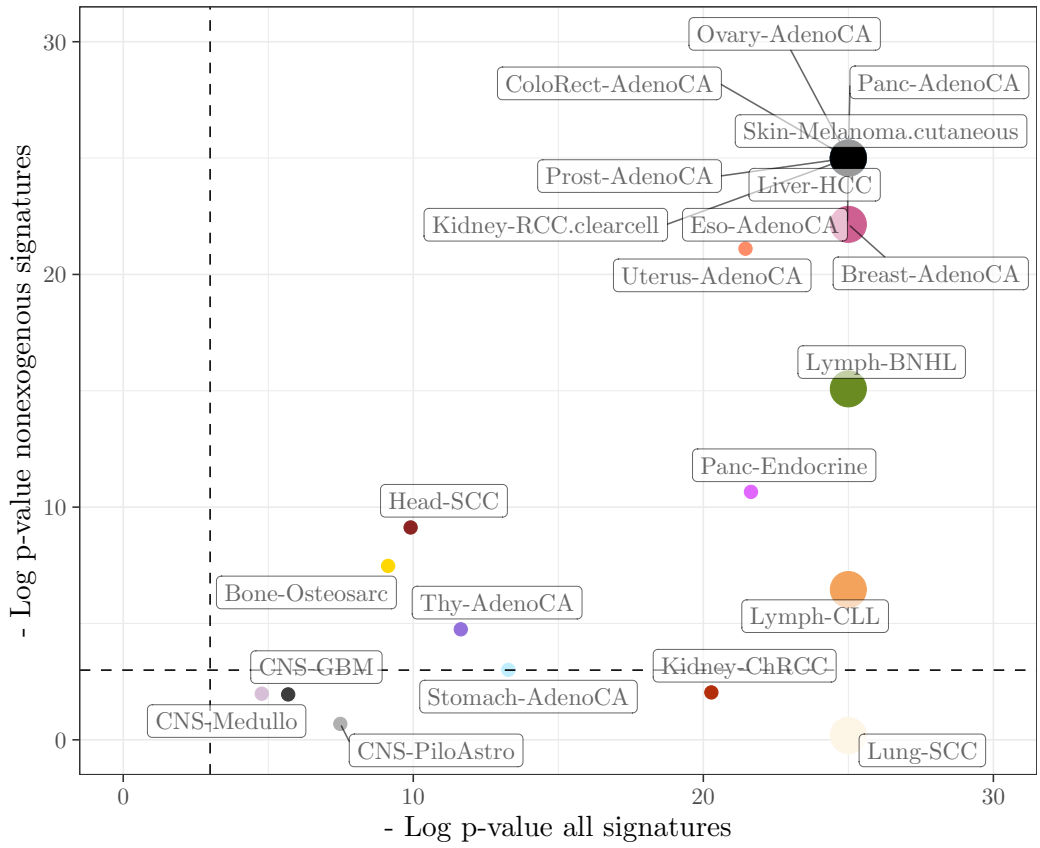
```
pcawg_palette <- pcawg.colour.palette(gsub("\\..*", "", enough_samples), scheme = "tumour.subtype")
names(pcawg_palette) <- enough_samples

df_pvals_DMDL_SP <- cbind.data.frame(pvals_DM=pvals_diagRE_DMDL_SP,
                                     pvals_DMnonexo=pvals_diagRE_DMDL_nonexo_SP,
                         num_samples=as.numeric(num_samples_all_SP),
                         num_sigs_nonexo=as.numeric(num_sigs_nonexo_SP),
                         ct=enough_samples,
                         pvals_DM_censored=sapply(-log(pvals_diagRE_DMDL_SP),
                                                  function(i) min(i, 25)),
                         pvals_DMnonexo_censored=sapply(-log(pvals_diagRE_DMDL_nonexo_SP),
                                                        function(i) min(i, 25)),
                         bool_censored=((-log(pvals_diagRE_DMDL_nonexo_SP) > 25 )| (-log(pvals_diagRE_DMDL
ggplot(df_pvals_DMDL_SP,
       aes(x=pvals_DM_censored, y=pvals_DMnonexo_censored,
           # size=num_samples,
           label=ct, size=bool_censored))+geom_point(aes (col=ct))+
  geom_hline(yintercept = -log(0.05), lty='dashed')+geom_vline(xintercept = -log(0.05), lty='dashed')+
  geom_label_repel(size=3.2, alpha=0.6, max.overlaps = 30)+  theme_bw()+
  theme(legend.position = "bottom", legend.text=element_text(size=8))+
  labs(x='- Log p-value all signatures', y='- Log p-value nonexogenous signatures')+
  guides(size=FALSE, col=FALSE)+ #, col=guide_legend(ncol=4),
  scale_color_manual(values = pcawg_palette)+theme(legend.position = "bottom")+
  lims(x=c(0, 30), y=c(0,30))
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Using size for a discrete variable is not advised.
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_label_repel).
```




```
t.test(df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo <= 0.05,'num_samples'],
       df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo > 0.05,'num_samples'])
```

```
##
##  Welch Two Sample t-test
##
## data:  df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo <= 0.05, "num_samples"] and df_pvals_DMDL_SP[d
## t = 1.4133, df = 14.155, p-value = 0.1792
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -14.86701  72.48923
## sample estimates:
## mean of x mean of y
##  79.61111  50.80000
```

```
t.test(df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo <= 0.05,'num_sigs_nonexo'],
       df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo > 0.05,'num_sigs_nonexo'])
```

```
##
##  Welch Two Sample t-test
##
## data:  df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo <= 0.05, "num_sigs_nonexo"] and df_pvals_DMDL_
## t = 5.5485, df = 20.956, p-value = 1.674e-05
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
##  3.341062 7.347827
## sample estimates:
## mean of x mean of y
##  8.944444  3.600000
```

```
mean(df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo <= 0.05,'num_sigs_nonexo'])
```

```
## [1] 8.944444
```

```
mean(df_pvals_DMDL_SP[df_pvals_DMDL_SP$pvals_DMnonexo > 0.05,'num_sigs_nonexo'])
```

```
## [1] 3.6
```

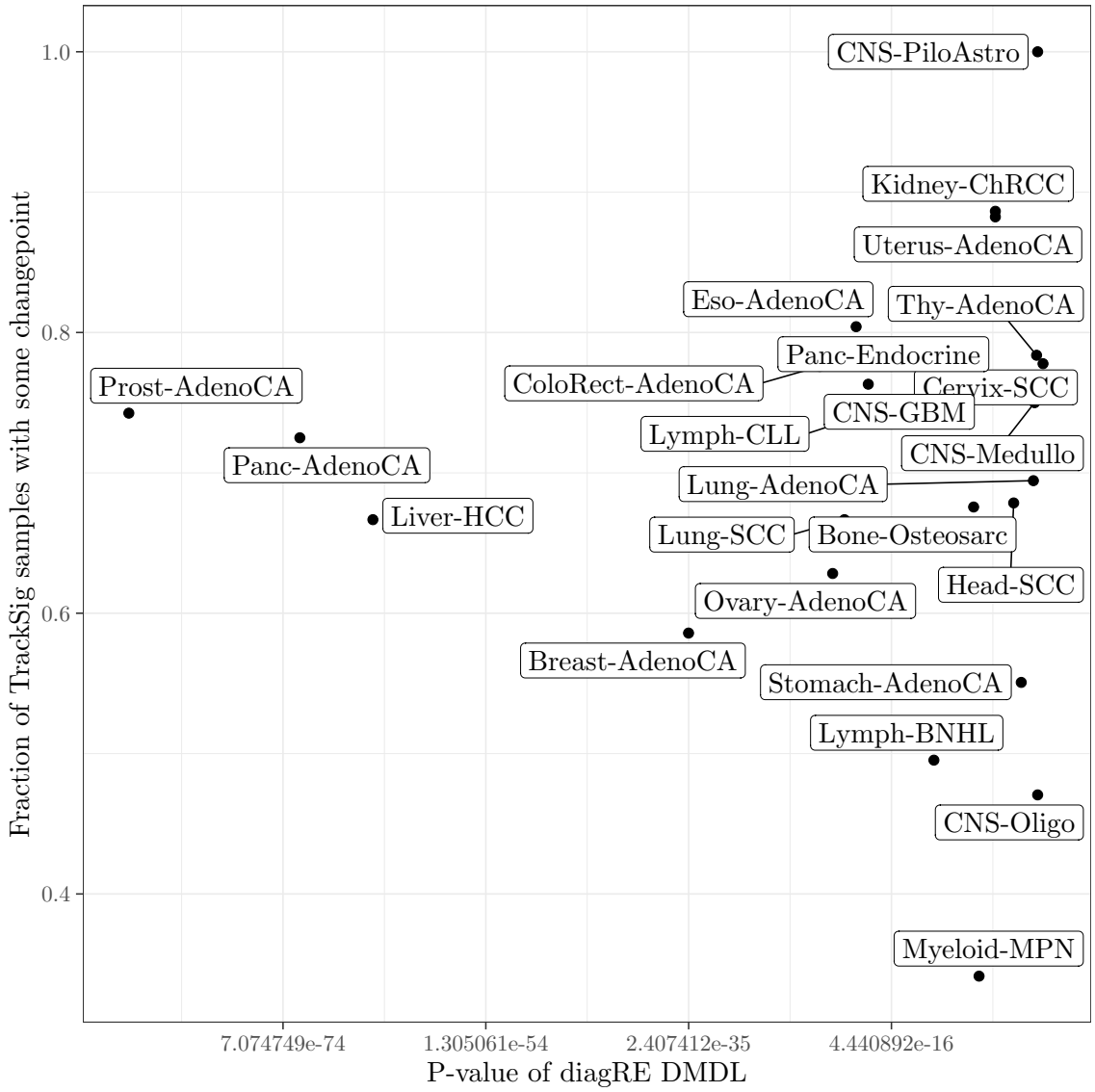
See script `PCAWG_HMP_and_alternative_methods.R` for the analyses of PCAWG data using alternative models.

Tracksig

```
tracksig = read.csv("../../data/restricted/tracksig/changepoints_stats_tracksig.csv", stringsAsFactors =
tracksig = tracksig %>% group_by(type) %>% dplyr::summarize(count = n(), bool_changepoints=sum(n_changepo
  mutate(tracksig_frac= bool_changepoints/count  )
tracksig = cbind.data.frame(pvals_diagRE_DM,
                tracksig[match(names(pvals_diagRE_DM), tracksig$type),],
                effect_size3_SP=effect_size3_SP[match(names(pvals_diagRE_DM), names(effect_size3_SP))])
tracksig$ct = rownames(tracksig)
tracksig$minpvals = -log2(tracksig$pvals_diagRE_DM)

ggplot(tracksig, aes(x=pvals_diagRE_DM, y=tracksig_frac, label=ct))+geom_point()+geom_label_repel()+
  labs(x='P-value of diagRE DMDL', y='Fraction of TrackSig samples with some changepoint')+scale_x_contin
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```

```
## Warning: Removed 4 rows containing missing values (geom_label_repel).
```




```
pcawg_palette <- pcawg.colour.palette(gsub("\\..*", "", tracksig$ct), scheme = "tumour.subtype")
names(pcawg_palette) <- tracksig$ct

ggplot(tracksig, aes(x=effect_size3_SP, y=tracksig_frac, label=ct, col=ct))+
  geom_point(aes(size=minpvals))+geom_label_repel(max.overlaps = 5)+
  labs(x='Effect size', y='Fraction of TrackSig samples with some changepoint', col="")+theme_bw()+
  scale_color_manual(values = pcawg_palette)+theme(legend.position = "bottom")+
  guides(col=guide_legend(ncol=4), size=FALSE)
```

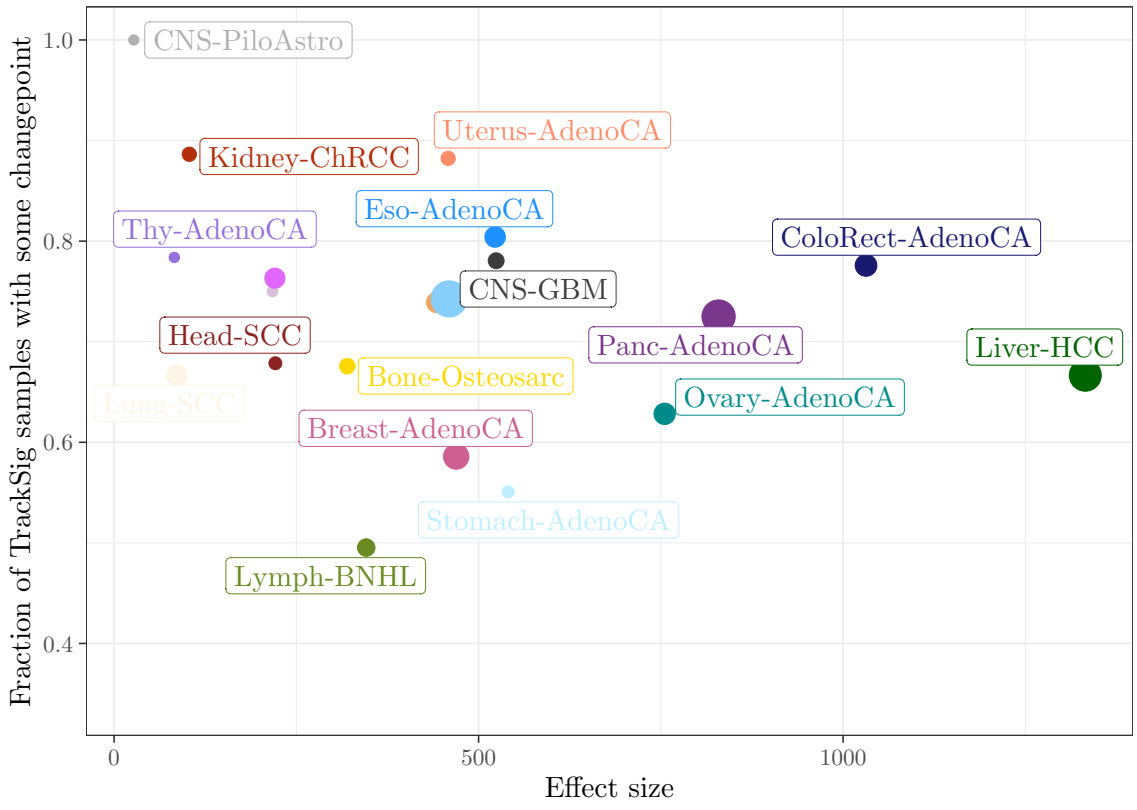
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: Removed 8 rows containing missing values (geom_point).
```

```
## Warning: Removed 8 rows containing missing values (geom_label_repel).
```

```
## Warning: ggrepel: 4 unlabeled data points (too many overlaps). Consider
```

```
## increasing max.overlaps
```



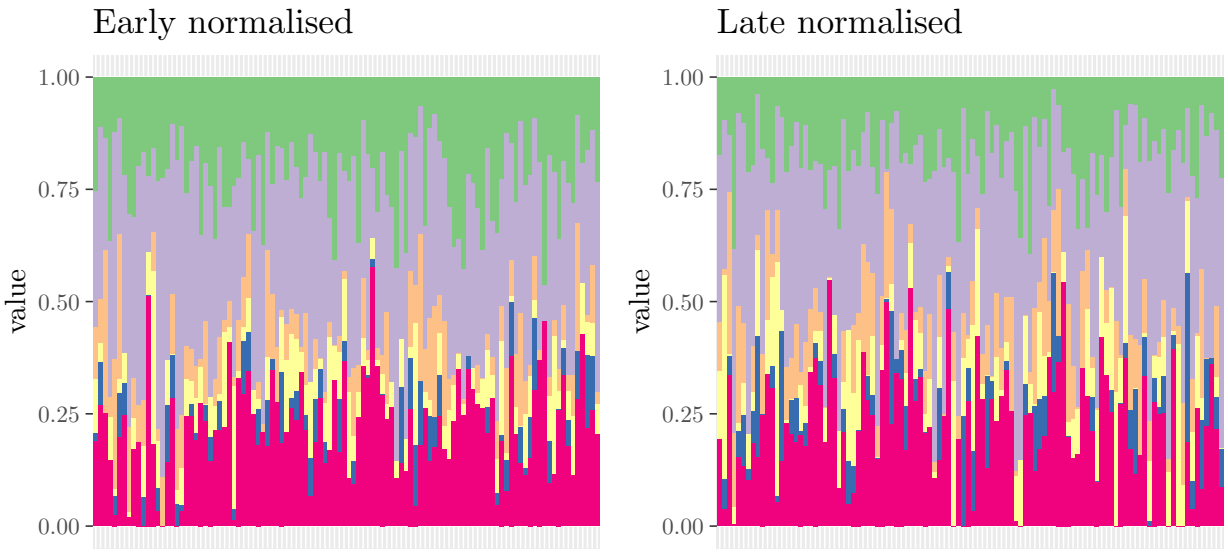
a Bone-Osteosarc	a ColoRect-AdenoCA	a Lung-AdenoCA	a Panc-Endocrine
a Breast-AdenoCA	a Eso-AdenoCA	a Lung-SCC	a Prost-AdenoCA
a Cervix-SCC	a Head-SCC	a Lymph-BNHL	a Skin-Melanoma.acral
a CNS-GBM	a Kidney-ChRCC	a Lymph-CLL	a Skin-Melanoma.cutaneou
a CNS-Medullo	a Kidney-RCC.clearcell	a Myeloid-MPN	a Stomach-AdenoCA
a CNS-Oligo	a Kidney-RCC.papillary	a Ovary-AdenoCA	a Thy-AdenoCA
a CNS-PiloAstro	a Liver-HCC	a Panc-AdenoCA	a Uterus-AdenoCA

Barplots of cancer types with and without differential abundance

```
give_barplot_from_obj(obj = signatures_PCAWG[['CNS-Medullo']], legend_on = F,
                      nrow=1, verbose=F,
                      only_normalised=T)
```

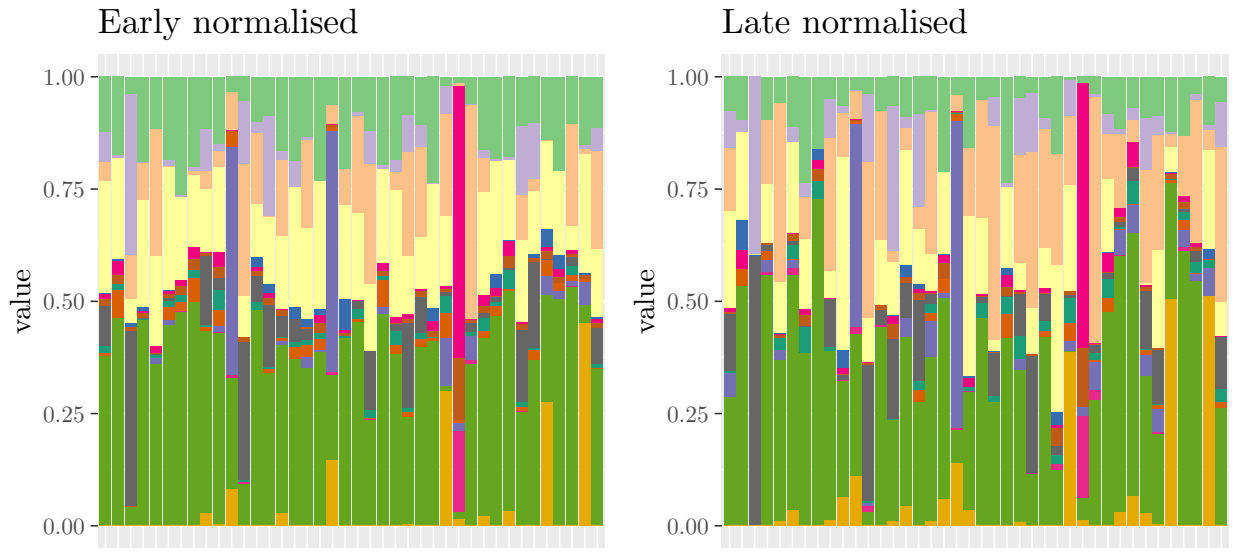
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```



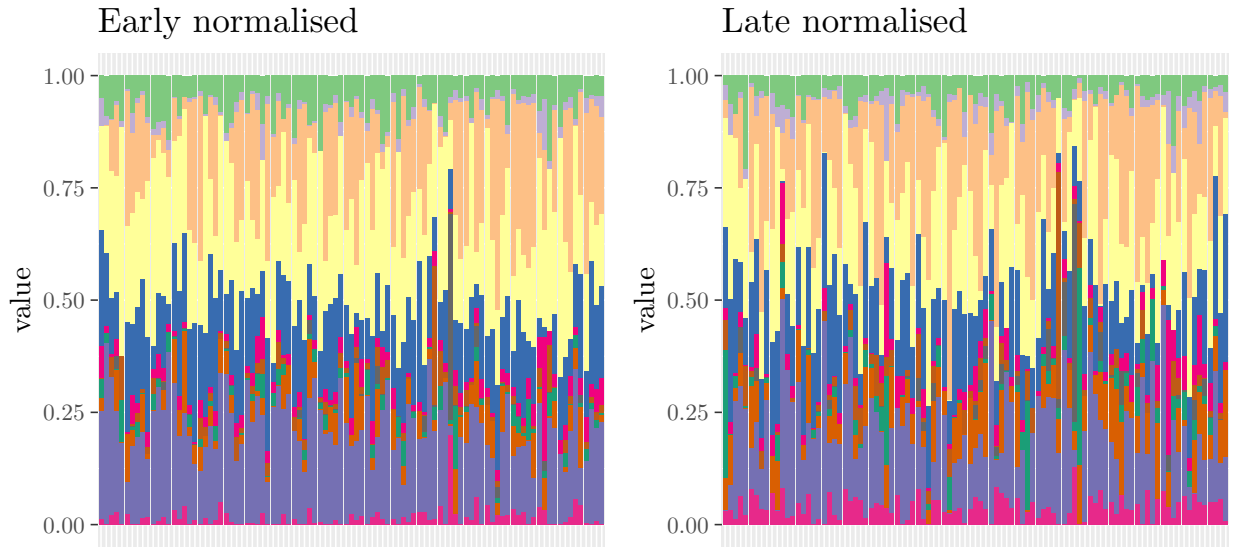

```
give_barplot_from_obj(obj = signatures_PCAWG[['Uterus-AdenoCA']], legend_on = F,
                      nrow=1, verbose=F,
                      only_normalised=T)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

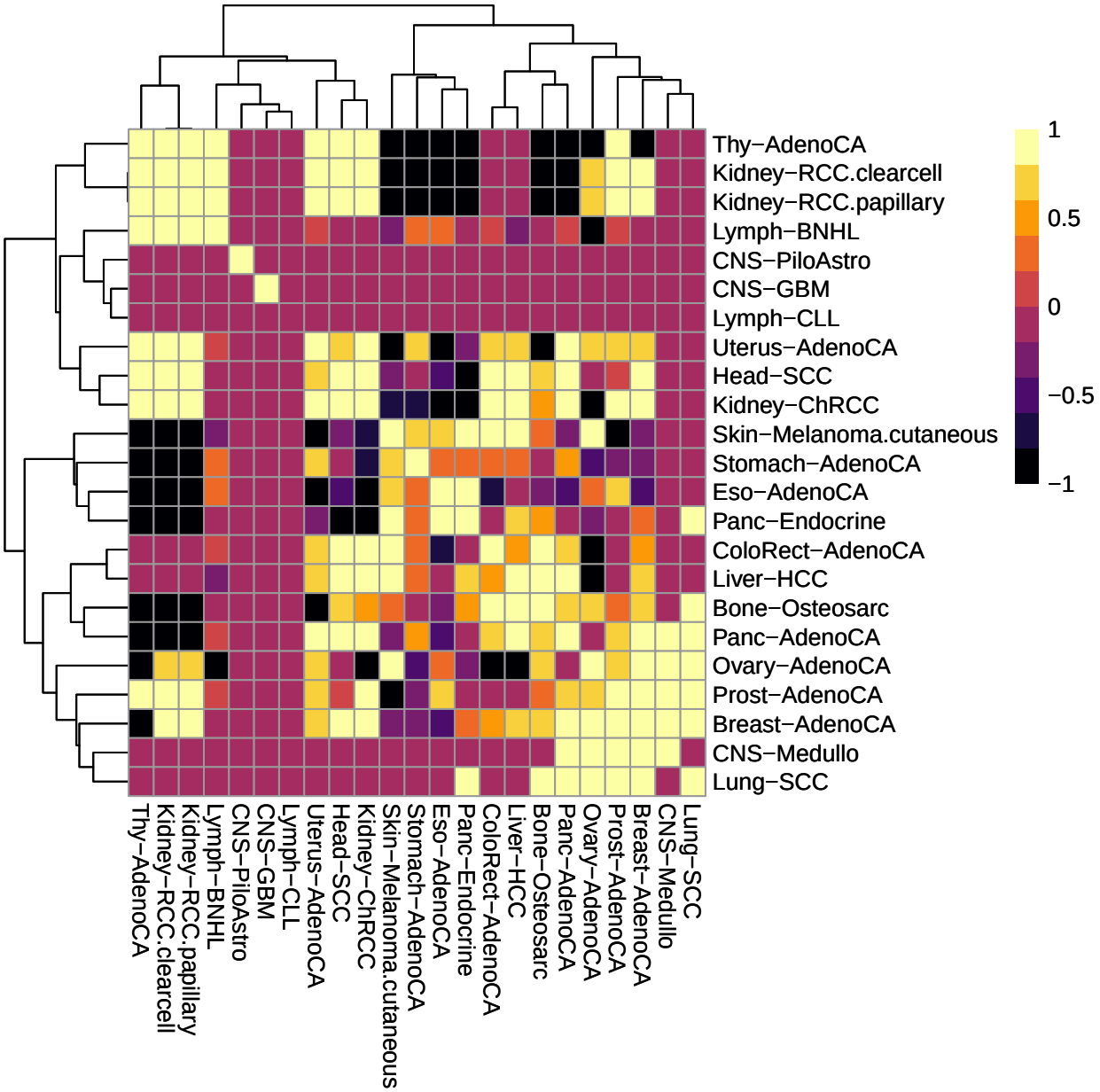



```
give_barplot_from_obj(obj = signatures_PCAWG[['Ovary-AdenoCA']], legend_on = F,
                      nrow=1, verbose=F,
                      only_normalised=T)
```

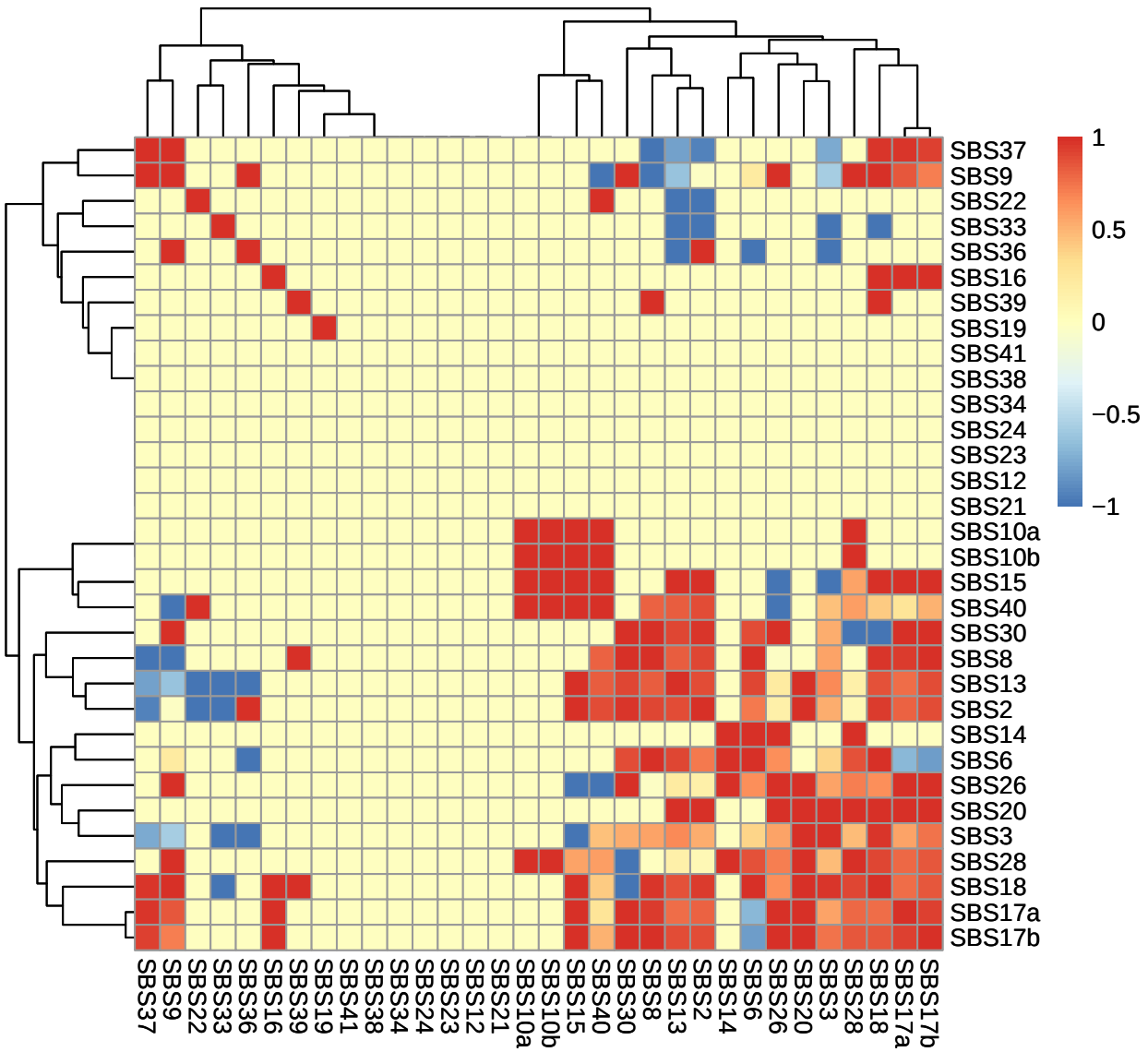
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```



Correlations of cancer types and of signatures based on betas




```
## null device
##           1

## null device
##           1
```




```
## null device
##           1
```

```
## null device
##           1
```