SigRepo Function Suite

Monti Lab

2025-02-05

Introduction

The `SigRepo` package includes a suite of functions for easily storing and managing biological signatures and its constituents. Currently, `Sigrepo` is capable of storing, searching, and retrieving signatures and its signature collections from a MySQL Database of choice. See here for documentation of how set-up the MySQL database with the appropriate schema.

In order to interact with a suite of functions in `SigRepo` package, the input data must be in a format of R6 objects for the representation of signatures and signature collections, and they can be created using our proprietary package, OmicSignature.

For more information click the links below.

- Overview of the object structure
- Create an OmicSignature (OmS)
- Create an OmicSignatureCollection (OmSC)

For demonstrations, we will walk through the steps of how to use `SigRepo` package to store, retrieve, and interact with a list of signatures stored in our MySQL SigRepo Database.

Installation

- Using `devtools` package

```
# Load devtools package
library(devtools)

# Install SigRepo
devtools::install_github(repo = 'montilab/SigRepo')

# Install OmicSignature
devtools::install_github(repo = 'montilab/OmicSignature')
```

Connect to SigRepo Database

We adopted a MySQL Database structure for efficiently storing, searching, and retrieving the biological signatures and its constituents. To access the signatures stored in our database, you MUST register here to create an account or contact our admin to be added.

There are three types of user accounts: - `admin` has READ and WRITE access to all signatures in the database. - `editor` has READ and WRITE access to ONLY their own uploaded signatures in the database. - `viewer` has ONLY READ access to view a list of signatures in the database but DO NOT HAVE WRITE access to the database.

Once you have a valid account, to connection to our SigRepo Database, one can use `newConnHandler()` function to create a handler which will contain appropriate credentials to establish connection to our database.

```
# Create a connection handler
conn_handler <- SigRepo::newConnHandler(
  dbname = Sys.getenv("DBNAME"),
  host = Sys.getenv("HOST"),
  port = as.integer(Sys.getenv("PORT")),
  user = Sys.getenv("USER"),
  password = Sys.getenv("PASSWORD")
)
```

Load Signatures

Here, we provided two signature objects that came with the package for demonstrations:

1. omic_signature_AGS_OmS
2. omic_signature_MDA_CYP

```
# Getting the signature path
signature_path <- base::system.file("inst/data/signatures", package = "SigRepo")

# Read in the signature object
omic_signature_AGS_OmS <- base::readRDS(file.path(signature_path, "omic_signature_AGS_OmS.RDS"))
omic_signature_MDA_CYP <- base::readRDS(file.path(signature_path, "omic_signature_MDA_CYP.RDS"))
```

Upload a signature

The function `addSignature()` allows the user to upload a signature to the database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function.

- **Example 1**: Create an omic signature using **OmicSignature** package and upload to database

```
# Create signature metadata
metadata <- OmicSignature::createMetadata(

  # required attributes:
  signature_name = "Myc_reduce_mice_liver_24m",
  organism = "Mus Musculus",
  direction_type = "bi-directional",
  assay_type = "transcriptomics",
  phenotype = "Myc_reduce",

  # optional and recommended:
```

```
  covariates = "none",
  description = "mice MYC reduced expression",
  platform = "GPL6246", # use GEO platform ID
  sample_type = "liver", # use BRENDA ontology

  # optional cut-off attributes.
  # specifying them can facilitate the extraction of signatures.
  logfc_cutoff = NULL,
  p_value_cutoff = NULL,
  adj_p_cutoff = 0.05,
  score_cutoff = 5,

  # other optional built-in attributes:
  keywords = c("Myc", "KO", "longevity"),
  cutoff_description = NULL,
  author = NULL,
  PMID = 25619689,
  year = 2015,

  # example of customized attributes:
  others = list("animal_strain" = "C57BL/6")
)

# Create difexp object
difexp <- readRDS(file.path(system.file("extdata", package = "OmicSignature"), "difmatrix_Myc_mice_live
colnames(difexp) <- OmicSignature::replaceDifexpCol(colnames(difexp))

# Create signature object
signature <- difexp %>%
  dplyr::filter(abs(score) > metadata$score_cutoff & adj_p < metadata$adj_p_cutoff) %>%
  dplyr::select(probe_id, feature_name, score) %>%
  dplyr::mutate(direction = ifelse(score > 0, "+", "-"))

# Create signature object
omic_signature <- OmicSignature::OmicSignature$new(
  metadata = metadata,
  signature = signature,
  difexp = difexp
)
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m created.
```

```
# Add signature to database
SigRepo::addSignature(
  conn_handler = conn_handler,
  omic_signature = omic_signature
)
#> Uploading signature metadata into the database...
#> Saving signature difexp into the database...
#> Adding user to signature access in the database...
#> Adding signature feature set into the database...
#> Finished uploading.
#> ID of the uploaded signature:
#> [1] 20
```

- **Example 2**: Upload `omic_signature_AGS_OmS` signature

```
SigRepo::addSignature(
  conn_handler = conn_handler,
  omic_signature = omic_signature_AGS_OmS
)
#>  You already uploaded a signature with signature_name = 'LLFS_Aging_Gene_2023' into the SigRepo Data
#>  Use searchSignature() to see more details about the signature.
#>  To re-upload, try to use a different name.
#>  ID of the uploaded signature:
#> [1] 1
```

- **Example 3**: Upload `omic_signature_MDA_CYP` signature

```
SigRepo::addSignature(
  conn_handler = conn_handler,
  omic_signature = omic_signature_MDA_CYP
)
#> Uploading signature metadata into the database...
#> Saving signature difexp into the database...
#> Adding user to signature access in the database...
#> Adding signature feature set into the database...
#> Error in value[[3L]](cond): Error in SigRepo::showTranscriptomicsErrorMessage(db_table_name = ref_ta
#> The following features do not existed in the 'transcriptomics_features' table of the database:
#> 'ENSG00000281508'
#> 'ENSG00000199900'
#> 'ENSG00000247844'
#> 'ENSG00000258777'
#> 'ENSG00000230836'
#> 'ENSG00000204282'
#> 'ENSG00000179979'
#> 'ENSG00000198384'
#> 'ENSG00000277203'
#> 'ENSG00000250889'
#> 'ENSG00000170647'
#> 'ENSG00000276797'
#> 'ENSG00000237975'
#> 'ENSG00000241990'
#> 'ENSG00000155640'
#> 'ENSG00000199404'
#> 'ENSG00000230641'
#> 'ENSG00000227895'
#> 'ENSG00000150526'
#> 'ENSG00000277555'
#> 'ENSG00000274744'
#> 'ENSG00000250588'
#> 'ENSG00000223414'
#> 'ENSG00000184258'
#> 'ENSG00000228265'
#> 'ENSG00000146521'
#> 'ENSG00000232224'
#> 'ENSG00000256045'
#> 'ENSG00000240875'
```

```
#> 'ENSG00000182584'
#> 'ENSG00000239332'
#> 'ENSG00000186354'
#> 'ENSG00000200649'
#> 'ENSG00000225163'
#> 'ENSG00000255145'
#> 'ENSG00000228439'
#> 'ENSG00000201126'
#> 'ENSG00000225986'
#> 'ENSG00000238648'
#> 'ENSG00000228393'
#> 'ENSG00000112096'
#> 'ENSG00000170590'
#> 'ENSG00000269028'
#> 'ENSG00000280524'
#> 'ENSG00000238266'
#> 'ENSG00000235825'
#> 'ENSG00000243587'
#> 'ENSG00000203441'
#> 'ENSG00000207770'
#> 'ENSG00000132832'
#> 'ENSG00000236850'
#> 'ENSG00000235884'
#> 'ENSG00000249860'
#> 'ENSG00000215271'
#> 'ENSG00000256164'
#> 'ENSG00000215067'
#> 'ENSG00000223797'
#> 'ENSG00000244349'
#> 'ENSG00000208035'
#> 'ENSG00000255090'
#> 'ENSG00000242349'
#> You can use 'searchFeature()' to see a list of available features in the database.
#> To add these features into the database, please contact our admin at montilab@bu.edu for support.
```

Search for a signature

The function `searchSignature()` allows the user to search for all or a specific signature that is available in the database.

- Search for all signatures

```
knitr::kable(
  SigRepo::searchSignature(conn_handler = conn_handler),
  row.names = FALSE
)
```

signature_id	signature_name	organism	direction_type	assay_type	phenotype	platform	sample_type	covariates	description	score_cutoff	logfc_cutoff	p_value_cutoff	adj_p_cutoff	cutoff_description	keywords	PMID	year	others	has_difexp	num_of_difexp	num_up_regulated	num_down_regulated	user_name	date_created	signature_hashkey
1	LLFS_Aging_Gene_2023	Homo sapiens	bi-directional	transcriptomics	Aging	GPLXXXXX	[illegible]	sex,fx,education,percent_intergenic,PC1-4,GRM	NA	6	NA	NA	0.01	NA	human,PBMC,aging	NA	2023	LLFS	1	1159	82	957	root	2025-02-05 09:39:49	7fdecac66691beed17
2	Myc_reduce_mice_v1	Mus musculus	bi-directional	transcriptomics	Myc_reduce	GPL6246	[illegible]	none	mice MYC reduced expression	5	NA	NA	0.05	NA	Myc,KO,longevity	25619689	2015	strain: <C57BL/6>	1	884	[illegible]	10	root	2025-02-05 09:40:21	eacd0935d97847d56
3	Myc_reduce_mice_v2	Mus musculus	bi-directional	transcriptomics	Myc_reduce	GPL6246	[illegible]	none	mice MYC reduced expression	5	NA	NA	0.05	NA	Myc,KO,longevity	25619689	2015	strain: <C57BL/6>	1	884	[illegible]	10	root	2025-02-05 09:40:22	ec356cabc33fcc22f8
20	Myc_reduce_mice	Mus musculus	bi-directional	transcriptomics	Myc_reduce	GPL6246	[illegible]	none	mice MYC reduced expression	5	NA	NA	0.05	NA	Myc,KO,longevity	25619689	2015	strain: <C57BL/6>	1	884	[illegible]	10	root	2025-02-05 11:13:16	a97e48241eae7cff27

- Search for a specific signature, e.g., **"LLFS_Aging_Gene_2023"**, in the database

```r
knitr::kable(
  SigRepo::searchSignature(
    conn_handler = conn_handler,
    signature_name = "LLFS_Aging_Gene_2023"
  ),
  row.names = FALSE
)
```

signature_id	signature_name	organism	direction_type	assay_type	phenotype	platform	sample_type	covariates	description	score_cutoff	logfc_cutoff	p_value_cutoff	adj_p_cutoff	cutoff_description	keywords	PMID	year	others	has_difexp	num_of_difexp	num_up_regulated	num_down_regulated	user_name	date_created	signature_hashkey
1	LLFS_Aging_Gene_2023	Homo sapiens	bi-directional	transcriptomics	Aging	GPLXXXXX	[illegible]	sex,fx,education,percent_intergenic,PC1-4,GRM	NA	6	NA	NA	0.01	NA	human,PBMC,aging	NA	2023	LLFS	1	1159	82	957	root	2025-02-05 09:39:49	7fdecac66691beed17

Retrieve an omic signature

The function `getSignature()` allows the user to retrieve a list of omic signature objects that belongs to themselves from the database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function. Furthermore, the user can **ONLY RETRIEVE** their own uploaded signatures or was given the `editor` permission from other users to do so.

- Retrieve all signatures

```
signature_list <- SigRepo::getSignature(conn_handler = conn_handler)
#>   [Success] OmicSignature object LLFS_Aging_Gene_2023 created.
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m_v1 created.
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m_v2 created.
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m created.
```

- Retrieve a specific signature

```
LLFS_signature <- SigRepo::getSignature(
  conn_handler = conn_handler,
  signature_name = "LLFS_Aging_Gene_2023"
)
#>   [Success] OmicSignature object LLFS_Aging_Gene_2023 created.
```

Update a signature

The function `updateSignature()` allows the user to update a signature in the sigrepo database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function. Furthermore, the user can **ONLY UPDATE** their own uploaded signatures or was given the `editor` permission from other users to do so.

For example, if the platform in the previous uploaded **"Myc_reduce_mice_liver_24m"** object is incorrect, and you wish to update the signature with the correct value, e.g., **GPLXXXXX**. Then you can use this function as follows:

```
# Search for Myc_reduce_mice_liver_24m in the database
# in which we would like to revise the value of platform with GPLXXXXX
signature_tbl <- SigRepo::searchSignature(
  conn_handler = conn_handler,
  signature_name = "Myc_reduce_mice_liver_24m"
)

# Revise the metadata object with new platform = GPLXXXXX
metadata_revised <- OmicSignature::createMetadata(

  # required attributes:
  signature_name = "Myc_reduce_mice_liver_24m",
  organism = "Mus Musculus",
  direction_type = "bi-directional",
  assay_type = "transcriptomics",
  phenotype = "Myc_reduce",

  # optional and recommended:
  covariates = "none",
  description = "mice MYC reduced expression",
  platform = "GPLXXXXX", # use GEO platform ID
  sample_type = "liver", # use BRENDA ontology

  # optional cut-off attributes.
```

```r
  # specifying them can facilitate the extraction of signatures.
  logfc_cutoff = NULL,
  p_value_cutoff = NULL,
  adj_p_cutoff = 0.05,
  score_cutoff = 5,

  # other optional built-in attributes:
  keywords = c("Myc", "KO", "longevity"),
  cutoff_description = NULL,
  author = NULL,
  PMID = 25619689,
  year = 2015,

  # example of customized attributes:
  others = list("animal_strain" = "C57BL/6")
)

# create the updated OmicSignature object
updated_omic_signature <- OmicSignature::OmicSignature$new(
  signature = signature,
  metadata = metadata_revised,
  difexp = difexp
)
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m created.
```

```r
# Updating the signature
SigRepo::updateSignature(
  conn_handler = conn_handler,
  signature_id = signature_tbl$signature_id,
  omic_signature = updated_omic_signature
)
#>   [Success] OmicSignature object Myc_reduce_mice_liver_24m created.
#> signature_id = '20' has been updated.
#> [1] TRUE
```

```r
knitr::kable(
  SigRepo::searchSignature(
    conn_handler = conn_handler,
    signature_name = "Myc_reduce_mice_liver_24m"
  ),
  row.names = FALSE
)
```

signature_id	signature_name	organism	direction_type	assay_type	phenotype	platform	sample_type	description	score_cutoff	logfc_cutoff	p_value_cutoff	adj_p_cutoff	cutoff_description	keywords	PMID	year	others	has_difexp	num_of_difexp	num_up_regulated	user_name	date_created	signature_hashkey
20	Myc_reduce_mice_liver_24m	Mus musculus	bi-directional	transcriptomics	Myc_reduce	GPLXXXXX	liver	mice MYC reduced expression	5	NA	NA	0.05	NA	Myc, KO, longevity	25619689	2015	animal_strain: <C57BL/6>	1	884	5	10	root	2025-02-05 11:13:20	a97e48241eae7cff270

Delete a signature

The function `deleteSignature()` allows the user to delete a signature from the database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function. Furthermore, the user can **ONLY DELETE** their own uploaded signatures or was given the `editor` permission from other users to do so.

```
# Search for Myc_reduce_mice_liver_24m in the database and remove it
signature_tbl <- SigRepo::searchSignature(
  conn_handler = conn_handler,
  signature_name = "Myc_reduce_mice_liver_24m"
)

# Remove signature from the database
SigRepo::deleteSignature(
  conn_handler = conn_handler,
  signature_id = signature_tbl$signature_id
)
#> Remove signature_id = '20' from 'signatures' table of the database.
#> Remove features belongs to signature_id = '20' from 'signature_feature_set' table of the database.
#> Remove user access to signature_id = '20' from 'signature_access' table of the database.
#> Remove signature_id = '20' from 'signature_collection_access' table of the database.
#> signature_id = '20' has been removed.
#> [1] TRUE
```