IMPCdata: data retrieval from IMPC database

Jeremy Mason, Natalja Kurbatova

Modified: 08 September, 2014. Compiled: November 24, 2026

IMPCdata is an R package that allows easy access to the phenotyping data produced by the International Mouse Phenotyping Consortium (IMPC). For more information about the IMPC project, please visit `http://www.mousephenotype.org`.

The IMPC implements a standardized set of phenotyping protocols to generate data. These standardized protocols are defined in the International Mouse Phenotyping Resource of Standardised Screens, IMPReSS (`http://www.mousephenotype.org/impress`). Programmatically navigating the IMPC raw data APIs and the IMPReSS SOAP APIs can be technically challenging –
IMPCdata package makes the process easier for R users.

The intended users of the IMPCdata package are familiar with R and would like easy access to the IMPC data. The data retrieved from IMPCdata can be directly used by PhenStat – an R package that encapsulates the IMPC statistical pipeline, available at
`http://www.bioconductor.org/packages/release/bioc/html/PhenStat.html`.

IMPCdata functions

The idea of IMPCdata is to systematically explore the IMPC dataset's multiple dimensions until the correct combination of filters has been selected and then the data of interest downloaded.

The IMPCdata functions can be divided into two logical groups:

- "List" functions to retrieve lists of IMPC database objects and
- "Dataset" functions to obtain datasets from IMPC database.

"List" functions

Each function in this group has "print" and "get" version. All "get" functions return lists of characters containing IMPC object IDs. It is possible to obtain the name of the IMPC object by using function *getName*. For instance, for pipeline name:

```
> library(IMPCdata)
> getName("pipeline_stable_id","pipeline_name","MGP_001")
```

```
$pipeline_name1
[1] "MGP Select Pipeline"
```

For the gene name:

```
> library(IMPCdata)
> getName("gene_accession_id","gene_symbol","MGI:1931466")
```

```
$gene_symbol1
[1] "Selenok"
```

From the examples above you can see that in order to get the name of the IMPC object you have to provide the database fields where IMPC object ID and name are stored together with the IMPC object ID you are interested in. Assuming that it is complicated for the users not familiar with the IMPC database structure we've implemented "print" function for each IMPC object.
All "print" functions print out pairs containing ID and name of the IMPC object hiding from the end user *getName* function complexity. By default all objects defined by combination of "print" function's arguments will be printed out. However, it is possible to restrict the number of printed out objects by defining "print" function's argument *n*.

There are the following objects in IMPC database that can be obtained through appropriate functions:

- IMPC **phenotyping centers**: *getPhenCenters*, *printPhenCenters*. IMPC phenotyping center has the same ID and name, that is why *printPhenCenters* function prints out not the pairs of ID and name as in all other cases but just the names of phenotyping centers.

```
> library(IMPCdata)
> getPhenCenters()

[[1]]
[1] "BCM"

[[2]]
[1] "CCP-IMG"

[[3]]
[1] "HMGU"

[[4]]
[1] "ICS"

[[5]]
[1] "JAX"

[[6]]
[1] "KMPC"

[[7]]
[1] "MARC"

[[8]]
[1] "MRC Harwell"

[[9]]
[1] "RBRC"

[[10]]
[1] "TCP"

[[11]]
[1] "UC Davis"

[[12]]
[1] "WTSI"

> printPhenCenters()
```

```
 [1] "BCM"          "CCP-IMG"     "HMGU"         "ICS"          "JAX"
 [6] "KMPC"         "MARC"        "MRC Harwell" "RBRC"         "TCP"
[11] "UC Davis"     "WTSI"
```

- IMPC **pipelines**: *getPipelines*, *printPipelines*. Both functions have two arguments: *PhenCenterName* and *excludeLegacyPipelines* to exclude legacy pipelines from the list with default value set to TRUE. For instance, to get and to print the pipeplines of Wellcome Trust Sanger Institute (WTSI):

```
> library(IMPCdata)
> getPipelines("WTSI")

[1] "DSS_001"  "IMPC_001" "MGP_001"  "SLM_001"  "TRC_001"

> getPipelines("WTSI",excludeLegacyPipelines=FALSE)

[1] "DSS_001"   "ESLIM_001" "ESLIM_002" "ESLIM_003" "IMPC_001"  "M-G-P_001"
[7] "MGP_001"   "SLM_001"   "TRC_001"

> printPipelines("WTSI")

[1] "DSS_001 - DSS challenge"
[1] "IMPC_001 - IMPC Pipeline"
[1] "MGP_001 - MGP Select Pipeline"
[1] "SLM_001 - Salmonella Challenge"
[1] "TRC_001 - Trichuris challenge"
```

- IMPC **procedures** (sometimes called **screens** or **assays**) that are run for a specified phenotyping center and pipeline: *getProcedures*, *printProcedures*. There may be multiple versions of a procedure. The version is defined by the number, i.e. 001 means the first version of the procedure. Both "get" and "print" functions for procedures have two arguments: *PhenCenterName* and *PipelineID*. For instance, to get the first two procedures that are run in WTSI within the MGP Select Pipeline:

```
> library(IMPCdata)
> head(getProcedures("WTSI","MGP_001"),n=2)

[[1]]
[1] "IMPC_ABR_001"

[[2]]
[1] "IMPC_BWT_001"

> printProcedures("WTSI","MGP_001",n=2)

[1] "IMPC_ABR_001 - Auditory Brain Stem Response"
[1] "IMPC_BWT_001 - Body Weight"
```

- IMPC **parameters** that are measured within specified procedure for a pipeline run by phenotyping center: *getParameters*, *printParameters*. Functions have three arguments: *PhenCenterName*, *PipelineID* and *ProcedureID*.

```
> library(IMPCdata)
> head(getParameters("WTSI","MGP_001","IMPC_CBC_001"),n=2)

[[1]]
[1] "IMPC_CBC_001_001"

[[2]]
[1] "IMPC_CBC_002_001"
```

```
> printParameters("WTSI","MGP_001","IMPC_CBC_001",n=2)

[1] "IMPC_CBC_001_001 - Sodium"
[1] "IMPC_CBC_002_001 - Potassium"
```

- Genetic backgrounds (called **strains** in IMPC database) from which the knockout mice were derived for a specific combination of pipeline, procedure and parameter for a phenotyping center: *getStrains*, *printStrains*. Strain ID is MGI ID (`http://www.informatics.jax.org/`) or temporary ID if the MGI is not assigned yet. There are the following arguments for both functions: *PhenCenterName*, *PipelineID*, *ProcedureID* and *ParameterID*.

```
> library(IMPCdata)
> head(getStrains("WTSI","MGP_001","IMPC_CBC_001","IMPC_CBC_003_001"),n=2)

[[1]]
[1] "MGI:2159965"

[[2]]
[1] "MGI:5446362"

> printStrains("WTSI","MGP_001","IMPC_CBC_001","IMPC_CBC_003_001",n=2)

[1] "MGI:2159965 - C57BL/6N"
[1] "MGI:5446362 - B6Brd;B6Dnk;B6N-Tyr<c-Brd>"
```

- **Genes** that are reported for a specified combination of parameter, procedure, pipeline and phenotyping center: *getGenes*, *printGenes*. Gene ID is MGI ID (`http://www.informatics.jax.org/`) or temporary ID if the MGI is not assigned yet. There are following arguments: *PhenCenterName*, *PipelineID*, *ProcedureID*, *ParameterID*, *StrainID* is an optional argument.

```
> library(IMPCdata)
> head(getGenes("WTSI","MGP_001","IMPC_CBC_001","IMPC_CBC_003_001"),n=2)

[[1]]
[1] "MGI:101835"

[[2]]
[1] "MGI:101857"

> printGenes("WTSI","MGP_001","IMPC_CBC_001",
+ "IMPC_CBC_003_001","MGI:2159965",n=2)

[1] "MGI:101835 - Traf2"
[1] "MGI:101857 - Usp15"
```

- **Alleles** that are processed for a specified combination of parameter, procedure, pipeline and phenotyping center: *getAlleles*, *printAlleles*. Allele ID is MGI ID (`http://www.informatics.jax.org/`) or temporary ID if the MGI is not assigned yet. There are following arguments: *PhenCenterName*, *PipelineID*, *ProcedureID*, *ParameterID* and *StrainID*, which is an optional argument.

```
> library(IMPCdata)
> head(getAlleles("WTSI","MGP_001","IMPC_CBC_001",
+ "IMPC_CBC_003_001","MGI:5446362"),n=2)
```

```
[1] "https://www.ebi.ac.uk/mi/impc/solr/experiment/select?q=phenotyping_center:%2
[[1]]
[1] "MGI:4363077"

[[2]]
[1] "MGI:4431605"

> printAlleles("WTSI","MGP_001","IMPC_CBC_001","IMPC_CBC_003_001",n=2)

[1] "https://www.ebi.ac.uk/mi/impc/solr/experiment/select?q=phenotyping_center:%2
[1] "MGI:3808228 - Rsad1<tm1(KOMP)Vlcg>"
[1] "MGI:4362139 - Rbm14<tm1a(KOMP)Wtsi>"
```

- **Zygosities** (homozygous, heterozygous and hemizygous) for mice that were measured for a gene/allele for a specified combination of parameter, procedure, pipeline and phenotyping center: *getZygosities*. There are following arguments of the *getZygosities* function: *PhenCenterName*, *PipelineID*, *ProcedureID*, *ParameterID*, *StrainID*, which is an optional argument, *GeneID*, which is an optional argument and finally *AlleleID*, which is also an optional argument. There is no "print" function version for zygosities.

```
> library(IMPCdata)
> getZygosities("WTSI","MGP_001","IMPC_CBC_001","IMPC_CBC_003_001",
+ StrainID="MGI:5446362",AlleleID="EUROALL:64")

[1] "https://www.ebi.ac.uk/mi/impc/solr/experiment/select?q=phenotyping_center:%2
list()
```

"Dataset" functions

The IMPCdata package "dataset" functions allow extraction of IMPC datasets where potential phenodeviants data and control data are matched together according to the internal IMPC rules. There are two functions in this group: *getIMPCTable* and *getIMPCDataset*.

Function *getIMPCTable* does not return any values, it creates a table using a combination of objects to define IMPC datasets according to the parameters that have been passed to the function. Table is stored in comma separated format in the file specified by user. Function's arguments are:

- *fileName* – defines name of the file where to save resulting table with IMPC objects, mandatory argument with default value set to "IMPCdata";
- *PhenCenterName* – IMPC phenotyping center;
- *PipelineID* – IMPC pipeline ID;
- *ProcedureID* – IMPC procedure ID;
- *ParameterID* – IMPC parameter ID;
- *AlleleID* – allele ID;
- *StrainID* – strain ID;
- *multipleFiles* – flag: "FALSE" value to get all records into one specified file; "TRUE" value (default) to split records across multiple files named starting with "fileName";
- *recordsPerFile* – number that specifies how many records to write into one file with default value set to 10000.

Example of usage:

```
> library(IMPCdata)
> getIMPCTable("./IMPCData.csv","WTSI","MGP_001","IMPC_CBC_001")
```

All possible combinations are stored now into the file "IMPCData.csv" in comma separated format. There are six columns in the saved file: "Phenotyping Center", "Pipeline", "Screen/Procedure", "Parameter", "Allele" and "Function to get IMPC Dataset". The last one column "Function to get IMPC Dataset" contains the prepared R code to call the function *getIMPCDataset* with appropriate parameters and to obtain the dataset.

Function *getIMPCDataset* returns a traditional *data.frame*. For example, take the value from the first row and the last column of the file created by *getIMPCTable* function where dataset function call is prepared for you:

```
> library(IMPCdata)
> IMPC_dataset1 <- getIMPCDataset("WTSI","MGP_001","IMPC_CBC_001",
+ "IMPC_CBC_003_001","MGI:4431644")
```

Now IMPC dataset is obtained and ready to use with PhenStat, for example:

```
> library(PhenStat)
> testIMPC1 <- PhenList(dataset=IMPC_dataset1,
+ testGenotype="MDTZ",
+ refGenotype="+/+",
+ dataset.colname.genotype="Colony")
```

For more information about the PhenStat package, please read User Guide available in Bioconductor and here `https://github.com/mpi2/stats_working_group/blob/master/PhenStatUserGuide/PhenStatUsersGuide.pdf`. Bugs reports and suggestions concerning IMPCdata package new versions can be submited by using github repository: `https://github.com/mpi2/stats_working_group`.