The CGDS-R library

Anders Jacobsen

November 1, 2010

Contents

1 Introduction

This package provides a basic set of R functions for querying the Cancer Genomic Data Server (CGDS) hosted by the Computational Biology Center at Memorial-Sloan-Kettering Cancer Center (MSKCC). In summary, the library can issue the following types of queries:

- `getCancerTypes()` : What cancer types are hosted on the server? For example TCGA glioblastoma or TCGA ovarian cancer.
- `getGeneticProfiles()` : What genetic profile types are available for cancer type X? For example mRNA expression or copy number alterations.
- `getCaseLists()` : what case sets are available for cancer type X? For example all samples or only samples corresponding to a given cancer subtype.

- `getProfileData()`: Retrieve slices of genomic data. For example, a client can retrieve all mutation data from PTEN and EGFR in TCGA glioblastoma.

Each of these functions will be briefly described in the following sections. The last part of this document includes some concrete examples of how to access and plot the data.

The purpose of this document is to give the reader a quick overview of the `cgdsr` package. Please refer to the corresponding R manual pages for a more detailed explanation of arguments and output for each function.

2 The CGDS R interface

2.1 `CGDS()` : Create a CGDS connection object

Initially, we will establish a connection to the public CGDS server hosted by Memorial Sloan-Kettering Cancer Center. The function for creating a CGDS connection object requires the URL of the CGDS server service, in this case `http://cbio.mskcc.org/cgds-public/`, as an argument.

```
> library(cgdsr)
> mycgds = CGDS("http://cbio.mskcc.org/cgds-public/")
> test(mycgds)

getCancerTypes...  ok
getCaseLists...  ok
getGeneticProfiles...  ok
```

The variable `mycgds` is now a CGDS connection object pointing at the URL for the public CGDS server. This connection object must be included as an argument to all subsequent interface calls. Optionally, we can now perform a set of simple test of the data returned from the CGDS connection object using the `test` function:

```
> test(mycgds)

getCancerTypes...  ok
getCaseLists...  ok
getGeneticProfiles...  ok
```

2.2 `getCancerTypes()` : Retrieve a set of available cancer types

Having created a CGDS connection object we can now retrieve a data frame with available cancer types using the `getCancerTypes` function:

```
> getCancerTypes(mycgds)[, c(1, 2)]

  cancer_type_id                    name
1            gbm     Glioblastoma (TCGA)
2            pca Prostate Cancer (MSKCC)
3           Sarc   Sarcoma (MSKCC/Broad)
```

Here we are only showing the first two columns, the cancer type ID and short name, of the result data frame. There is also a third column, a longer description of the cancer type. The cancer type ID must be used in subsequent interface calls to retrieve case lists and genetic data profiles (see below).

2.3 getGeneticProfiles() : Retrieve genetic data profiles for a specific cancer type

This function queries the CGDS API and returns the available genetic profiles, e.g. mutation or copy number profiles, stored about a specific cancer type. Below we list the current genetic profiles for the TCGA glioblastoma cancer type:

```
> getGeneticProfiles(mycgds, "gbm")[, c(1:2)]
```

```
  genetic_profile_id                     genetic_profile_name
1      gbm_mutations                                Mutations
2  gbm_cna_consensus    Copy number alterations, GBM Pathways
3        gbm_cna_rae Copy number alterations (RAE algorithm)
4           gbm_mrna                          mRNA Expression
5   gbm_mrna_zscores                mRNA Expression z-Scores
```

Here we are only listing the first two columns, genetic profile ID and short name, of the resulting data frame. Please refer to the R manual pages for a more extended specification of the arguments and output.

2.4 getCaseLists() : Retrieve case lists for a specific cancer type

This function queries the CGDS API and returns available case lists for a specific cancer type. For example, within a particular study, only some cases may have sequence data, and another subset of cases may have been sequenced and treated with a specific therapeutic protocol. Multiple case lists may be associated with each cancer type, and this method enables you to retrieve meta-data regarding all of these case lists. Below we list the current case lists for the TCGA glioblastoma cancer type:

```
> getCaseLists(mycgds, "gbm")[, c(1:2)]
```

```
              case_list_id                       case_list_name
1                 gbm_all                            All Tumors
2    gbm_sequenced_nohyper          Sequenced, No Hypermutators
3 gbm_sequenced_nottreated               Sequenced, Not Treated
4    gbm_sequenced_treated                   Sequenced, Treated
5        gbm_3way_complete Three-way complete (seq, mRNA, CNA)
```

Here we are only listing the first two columns, case list ID and short name, of the resulting data frame. Please refer to the R manual pages for a more extended specification of the arguments and output.

2.5 getProfileData() : Retrieve genomic profile data for genes and genetic profiles

The function queries the CGDS API and returns data based on gene(s), genetic profile(s), and a case list. The function only allows specifying a list of genes and a single genetic profile, or oppositely a single gene and a list of genetic profiles. Importantly, the format of the output data frame depends on if a single or a list of genes was specified in the arguments. Below we are retrieving mRNA expression and copy number alteration genetic profiles for the NF1 gene in all samples of the TCGA glioblastoma cancer type:

```
> getProfileData(mycgds, "NF1", c("gbm_cna_rae", "gbm_mrna"), "gbm_all")[,
+     c(1:5)]

  GENETIC_PROFILE_ID        ALTERATION_TYPE GENE_ID COMMON TCGA.02.0001
1        gbm_cna_rae COPY_NUMBER_ALTERATION    4763    NF1    0.0000000
2           gbm_mrna        MRNA_EXPRESSION    4763    NF1   -0.2966920
```

We are here only showing the first five columns, the remaining columns are data for the remaining samples. In the next example we are retrieving mRNA expression data for the MDM2 and MDM4 genes:

```
> getProfileData(mycgds, c("MDM2", "MDM4"), "gbm_mrna", "gbm_all")[,
+     c(1:5)]

  GENE_ID COMMON TCGA.02.0001 TCGA.02.0003 TCGA.02.0004
1    4193   MDM2   -0.4232196   -0.3428422   -0.7600202
2    4194   MDM4   -0.4896734   -0.1022788   -0.4743049
```

We are again only showing the first five columns, the remaining columns are data for the remaining samples.

3 Examples

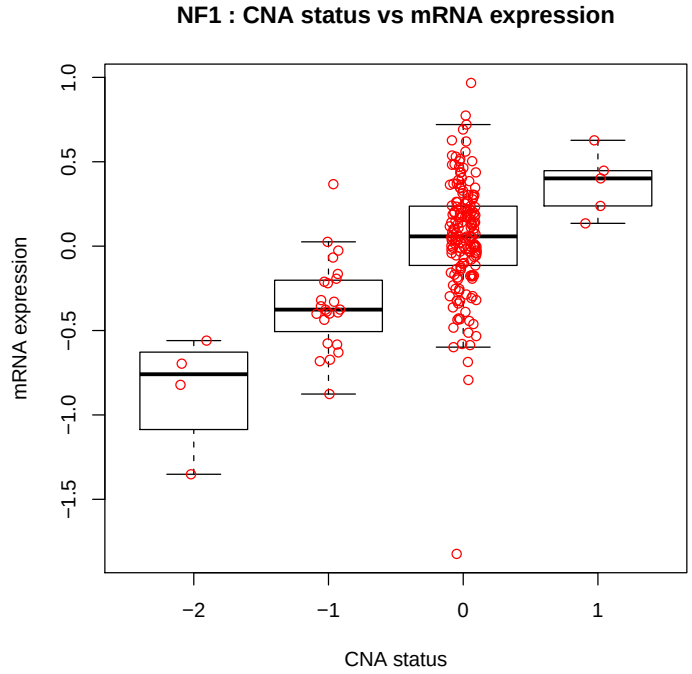
3.1 Example 1: Association of NF1 copy number alteration and mRNA expression in glioblastoma

As a simple example we will generate a plot of the association between copy number alteration (CNA) status and mRNA expression change for the NF1 tumor suprpressor gene in glioblastoma. This plot is very similar to Figure 2b in the TCGA research network paper on glioblastoma (McLendon et al. 2008). The mRNA expression of NF1 has been median adjusted on the gene level (by globally subtracting the median expression level of NF1 across all samples).

```
> df1 = getProfileData(mycgds, "NF1", c("gbm_cna_rae", "gbm_mrna"),
+     "gbm_all")
> df2 = data.frame(t(df1[, -c(1:4)]))
> names(df2) = c("CNA", "MRNA")
> head(df2)
```

```
             CNA         MRNA
TCGA.02.0001   0 -0.296691953
TCGA.02.0003   0 -0.001066810
TCGA.02.0004 NaN -0.236265119
TCGA.02.0006   0 -0.169150677
TCGA.02.0007   0 -0.009593496
TCGA.02.0009   0  0.537073170
```

```
> boxplot(df2$MRNA ~ df2$CNA, main = "NF1 : CNA status vs mRNA expression",
+     xlab = "CNA status", ylab = "mRNA expression", outpch = NA)
> stripchart(df2$MRNA ~ df2$CNA, vertical = T, add = T, method = "jitter",
+     pch = 1, col = "red")
```



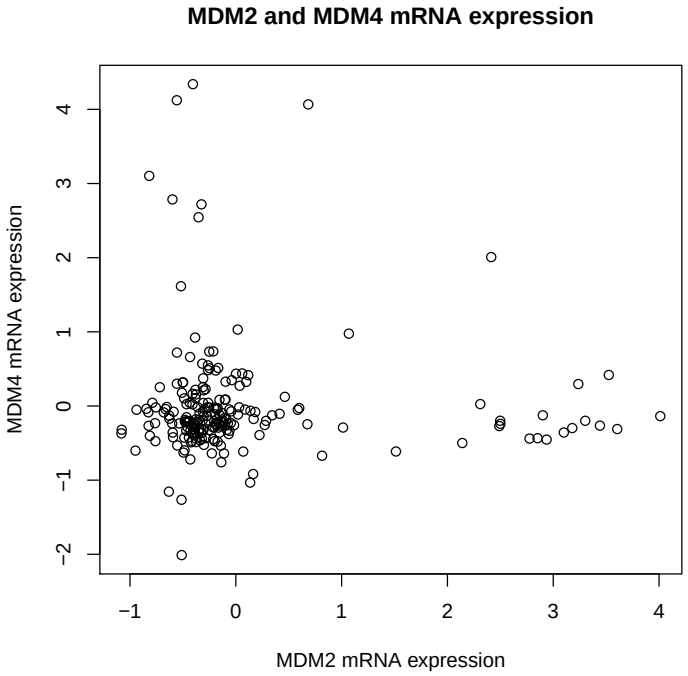
3.2 Example 2: MDM2 and MDM4 mRNA expression levels in glioblastoma

In this example we evaluate the relationship of MDM2 and MDM4 expression levels in glioblastoma. mRNA expression levels of MDM2 and MDM4 have been median adjusted on the gene level (by globally subtracting the median expression level of the individual gene across all samples).

```
> df1 = getProfileData(mycgds, c("MDM2", "MDM4"), "gbm_mrna", "gbm_all")
> df2 = data.frame(t(df1[, -c(1:2)]))
> names(df2) = c("MDM2", "MDM4")
> head(df2)
```

```
                    MDM2        MDM4
TCGA.02.0001 -0.4232196 -0.48967339
TCGA.02.0003 -0.3428422 -0.10227876
TCGA.02.0004 -0.7600202 -0.47430488
TCGA.02.0006  2.3102925  0.02569512
TCGA.02.0007 -0.5988326  2.78569512
TCGA.02.0009 -0.4095651  0.02249049
```

```
> plot(df2, main = "MDM2 and MDM4 mRNA expression", xlab = "MDM2 mRNA expression",
+     ylab = "MDM4 mRNA expression")
```



3.3 Example 3: Comparing expression of PTEN in primary and metastatic prostate cancer tumors

In this example we plot the mRNA expression levels of PTEN in primary and metastatic prostate cancer tumors.

```
> df1.pri = getProfileData(mycgds, "PTEN", "pca_mrna", "pca_primary")
> df2.pri = as.numeric(df1.pri[, -c(1, 2)])
> df1.met = getProfileData(mycgds, "PTEN", "pca_mrna", "pca_mets")
> df2.met = as.numeric(df1.met[, -c(1, 2)])
> head(df2.pri)
```

```
[1] 9.467183 9.041528 8.511305      NaN 9.413217      NaN
```

```
> head(df2.met)
```

```
[1] 7.486938      NaN 7.578755      NaN      NaN 8.756132
```

```
> boxplot(list(df2.pri, df2.met), main = "PTEN expression in primary and metastatic tumors
+     xlab = "Tumor type", ylab = "PTEN mRNA expression", names = c("primary",
+         "metastatic"), outpch = NA)
> stripchart(list(df2.pri, df2.met), vertical = T, add = T, method = "jitter",
+     pch = 1, col = "red")
```