The CGDS-R library

Anders Jacobsen

April 20, 2011

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 (cBio) at the Memorial Sloan-Kettering Cancer Center (MSKCC). This service is a part of the cBio Cancer Genomics Portal, `http://www.cbioportal.org/cgx/`.

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 for PTEN and EGFR in TCGA glioblastoma.
- `getClinicalData()`: Retrieve clinical data (e.g. patient survival time and age) for a given cancer type and list of cases.

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/")
```

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 tests of the data returned from the CGDS connection object using the `test` function:

```
> test(mycgds)

getCancerTypes...  OK
getCaseLists (1/2) ...  OK
getCaseLists (2/2) ...  OK
getGeneticProfiles (1/2) ...  OK
getGeneticProfiles (2/2) ...  OK
getClinicalData (1/4) ...  OK
getClinicalData (2/4) ...  OK
getClinicalData (3/4) ...  OK
getClinicalData (4/4) ...  OK
getProfileData (1/7) ...  OK
getProfileData (2/7) ...  OK
getProfileData (3/7) ...  OK
getProfileData (4/7) ...  OK
getProfileData (5/7) ...  OK
getProfileData (6/7) ...  OK
getProfileData (7/7) ...  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            pca      Prostate Cancer (MSKCC)
2            gbm          Glioblastoma (TCGA)
3            ova Serous Ovarian Cancer (TCGA)
4           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 Putative copy-number alterations (GBM Pathways)
3        gbm_cna_rae         Putative copy-number alterations (RAE)
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_3way_complete All Complete Tumors (seq, mRNA, CNA)
2                   gbm_all                              All Tumors
3        gbm_expr_classical          Expression Cluster Classical
4      gbm_expr_mesenchymal        Expression Cluster Mesenchymal
5           gbm_expr_neural             Expression Cluster Neural
6        gbm_expr_proneural          Expression Cluster Proneural
7             gbm_seq_paper     Sequenced Tumors, GBM Manuscript
8     gbm_sequenced_nohyper           Sequenced, No Hypermutators
9  gbm_sequenced_nottreated                Sequenced, Not Treated
10    gbm_sequenced_treated                    Sequenced, Treated
```

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),
+     ]
```

```
             gbm_cna_rae     gbm_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
```

We are here only showing the first five rows of the data frame. 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),
+     ]
```

```
                   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
```

We are again only showing the first five rows of the data frame.

2.6 `getClinicalData()` : Retrieve clinical data for a list of cases

The function queries the CGDS API and returns available clinical data (e.g. patient survival time and age) for a given case list. Results are returned in a data frame with a row for each case and a column for each clinical attribute. The available clinical attributes are:

- `overall_survival_months`: Overall survival, in months.
- `overall_survival_status`: Overall survival status, usually indicated as "LIVING" or "DECEASED".
- `disease_free_survival_months`: Disease free survival, in months.
- `disease_free_survival_status`: Disease free survival status, usually indicated as "DiseaseFree" or "Recurred/Progressed".
- `age_at_diagnosis`: Age at diagnosis.

Below we retrieve clinical data for the TCGA ovarian cancer dataset (only first five cases/rows are shown):

```
> getClinicalData(mycgds, "ova_all")[c(1:5), ]
```

```
             overall_survival_months overall_survival_status
TCGA.04.1331                   43.80                DECEASED
TCGA.04.1332                   40.89                DECEASED
TCGA.04.1336                   49.02                  LIVING
TCGA.04.1337                    2.03                DECEASED
TCGA.04.1338                   46.49                  LIVING
             disease_free_survival_months disease_free_survival_status
TCGA.04.1331                        15.05          Recurred/Progressed
TCGA.04.1332                        12.95          Recurred/Progressed
TCGA.04.1336                        49.02                  DiseaseFree
TCGA.04.1337                           NA          Recurred/Progressed
TCGA.04.1338                        12.46          Recurred/Progressed
             age_at_diagnosis
TCGA.04.1331            79.04
TCGA.04.1332            70.64
TCGA.04.1336            55.53
TCGA.04.1337            78.42
TCGA.04.1338            78.87
```

3 Examples

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

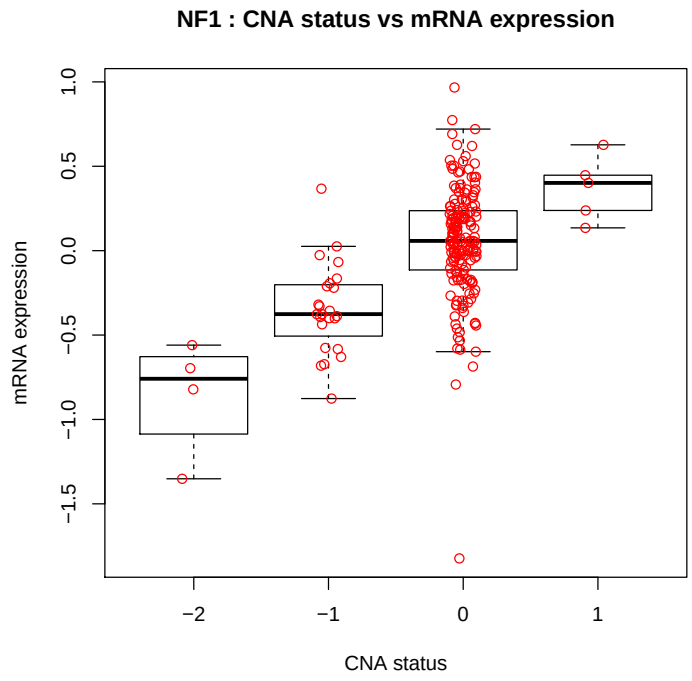
As a simple example, we will generate a plot of the association between copy number alteration (CNA) status and mRNA expression change for the NF1 tumor suprpressor gene in glioblastoma. This plot is very similar to Figure 2b

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).

```
> df = getProfileData(mycgds, "NF1", c("gbm_cna_rae", "gbm_mrna"),
+     "gbm_all")
> head(df)
```

```
             gbm_cna_rae     gbm_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
```

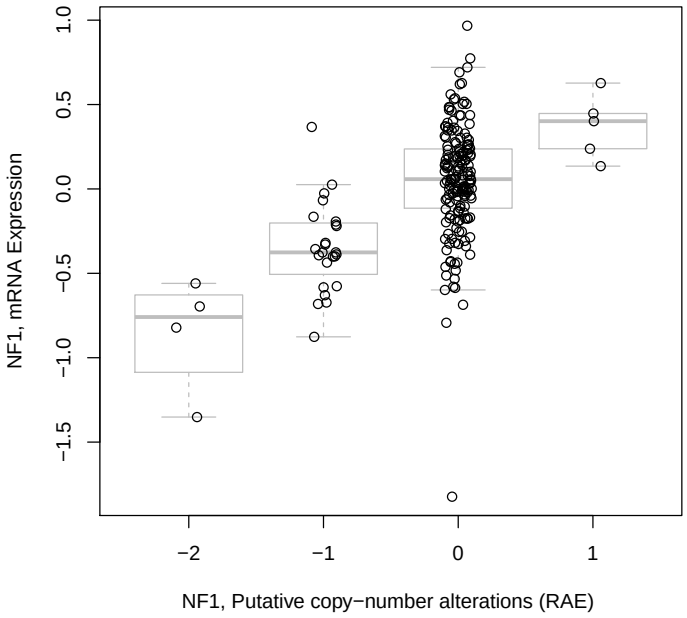
```
> boxplot(df[, 2] ~ df[, 1], main = "NF1 : CNA status vs mRNA expression",
+     xlab = "CNA status", ylab = "mRNA expression", outpch = NA)
> stripchart(df[, 2] ~ df[, 1], vertical = T, add = T, method = "jitter",
+     pch = 1, col = "red")
```



Alternatively, the generic `cgdsr plot()` function can be used to generate a similar plot:

```
> plot(mycgds, "gbm", "NF1", c("gbm_cna_rae", "gbm_mrna"), "gbm_all",
+     skin = "disc_cont")
```

```
[1] TRUE
```



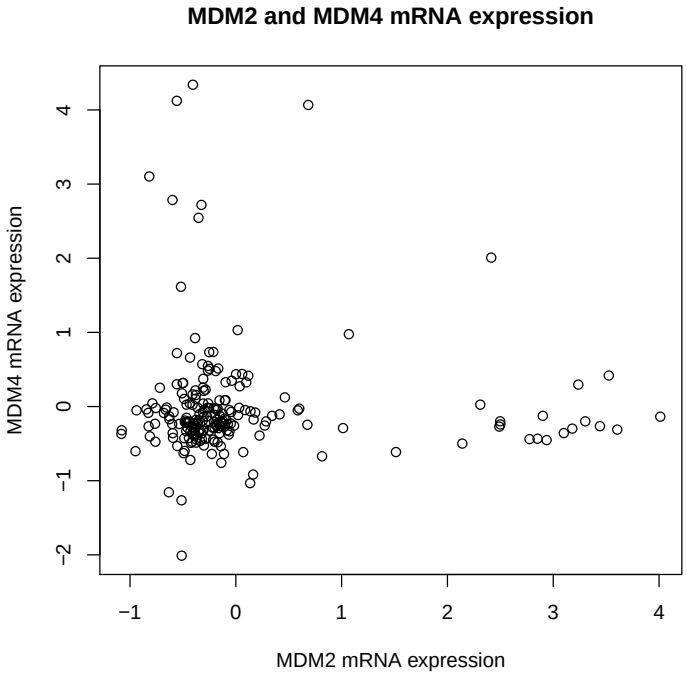
3.2 Example 2: MDM2 and MDM4 mRNA expression levels in glioblastoma

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).

```
> df = getProfileData(mycgds, c("MDM2", "MDM4"), "gbm_mrna", "gbm_all")
> head(df)
```

```
                   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
```

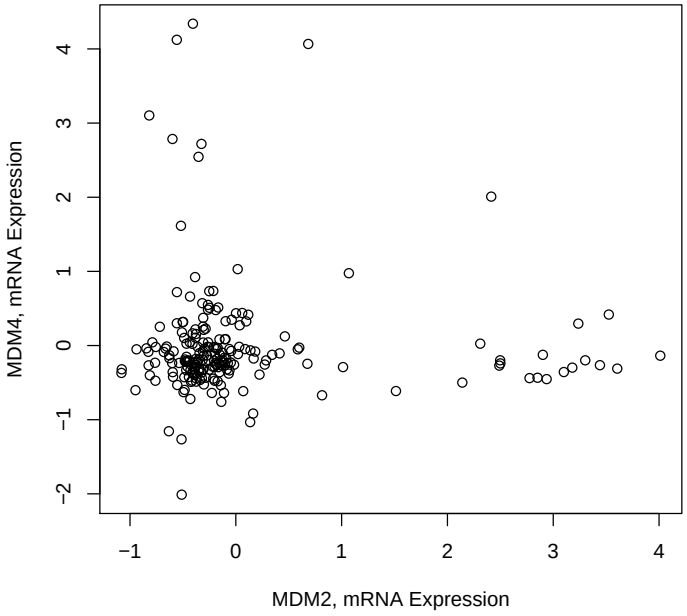
```
> plot(df, main = "MDM2 and MDM4 mRNA expression", xlab = "MDM2 mRNA expression",
+     ylab = "MDM4 mRNA expression")
```



Alternatively, the generic `cgdsr plot()` function can be used to generate a similar plot:

```
> plot(mycgds, "gbm", c("MDM2", "MDM4"), "gbm_mrna", "gbm_all")

[1] TRUE
```



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.

```
> df.pri = getProfileData(mycgds, "PTEN", "pca_mrna", "pca_primary")
> head(df.pri)
```

```
            PTEN
PCA0001 9.467183
PCA0002 9.041528
PCA0003 8.511305
PCA0004      NaN
PCA0005 9.413217
PCA0006      NaN
```

```
> df.met = getProfileData(mycgds, "PTEN", "pca_mrna", "pca_mets")
> head(df.met)
```

```
            PTEN
PCA0182 7.486938
PCA0183      NaN
PCA0184 7.578755
PCA0185      NaN
PCA0186      NaN
PCA0187 8.756132
```

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