`PatientGeneSets` package

Simina M. Boca
Johns Hopkins Bloomberg School of Public Health
email: `sboca@jhsph.edu`,

Giovanni Parmigiani
Dana-Farber Cancer Institute and
Harvard School of Public Health
email: `gp@jimmy.harvard.edu`

October 18, 2010

Contents

1   Overview

Patient-oriented methods are a novel approach to gene-set analysis which calculate gene-set scores for each sample, then combine them across samples. By comparison, gene-oriented methods calculate a score for each gene across all samples, then combine them into gene-set scores. We find that for cancer mutation data, patient-oriented methods often perform better on both real and simulated data.

The `PatientGeneSets` package has functions which perform gene-set analyses for cancer mutation data. It can be used to compute p-values and q-values for 4 patient-oriented methods described in [2], as well as the gene-oriented method described in [10] (see also [8]), which is in the `limma` package and performs a Wilcoxon test. It can also be used to do the simulations described in [2]. The user has a choice between calculating q-values based on the FDR-control method in [1] or [11]. This package is closely related to the `CancerMutationAnalysis` package, available at `http://bcb.dfci.harvard.edu/~gp/software/CancerMutationAnalysis/cma.htm` ([6]), which provides methods for gene-level analysis of cancer mutation data. In particular, the `cma.scores` function and the data format are nearly identical, but the `PatientGeneSets` package does not consider the two-phase design when performing gene-set analysis.

This vignette represents an introduction on the `PatientGeneSets` package. The primary function `do.gene.set.analysis`, which implements the gene-set analysis methods. The function

`sim.data.p.values` performs simulations using either the permutation null or the passenger null (see [2]). The function `cma.data` calculates scores for each gene across all samples, as in [9] and [12]. The functions `extract.sims.method` and `combine.sims` are used to manipulate the objects resulting from `sim.data.p.values`.

2 Glioblastoma dataset

We use the glioblastoma dataset from [7]. When typing `data(Parsons)`, 4 objects are loaded: `CoverageBrain`, `EventsBySampleBrain`, `GeneSizes08` and `MutationsBrain`. For this example, we use KEGG annotations ([3], [5], [4]). from the Bioconductor package `KEGG.db` by using the `KEGGPATHID2EXTID` object.

```
> library(PatientGeneSets)
> data(Parsons)
> ls()

[1] "CoverageBrain"       "EventsBySampleBrain" "GeneSizes08"
[4] "MutationsBrain"

> library(KEGG.db)
> KEGGPATHID2EXTID

PATHID2EXTID map for KEGG (object of class "AnnDbBimap")
```

Since the genes in the glioblastoma dataset are annotated by gene-names, while `KEGGPATHID2EXTID` uses EntrezGene identifiers, we also provide a vector mapping the identifiers to the names, which we obtained by using the `biomaRt` package.

```
> data(ID2name)
> head(ID2name)

     10      100     1000    10000    10007     1001
  "NAT2"    "ADA"   "CDH2"   "AKT3" "GNPDA1"   "CDH3"
```

3 Implementing the gene-set analysis methods

The function `do.gene.set.analysis` returns a data-frame with p-values and q-values for all the methods selected. By default, all the methods are implemented; however this takes several minutes. Setting the `gene.method` parameter to `TRUE` implements the gene-oriented method. The other method parameters refer to the patient-oriented methods: `perm.null.method` refers to the permutation null without heterogeneity, `perm.null.het.method` to the permutation null with heterogeneity, `pass.null.method` to the passenger null without heterogeneity, and `pass.null.het.method` to the passenger null with heterogeneity. See [2] for more details on all these methods.
The gene-sets we use correspond to the KEGG annotations for endometrial cancer, non-small cell lung cancer, and alanine, aspartate and glutamate metabolism i.e. hsa05213, hsa05223, and hsa00250:

```
> as.character(KEGGPATHNAME2ID[c("Endometrial cancer", "Non-small cell lung cancer",
+      "Alanine, aspartate and glutamate metabolism")])
```

```
Alanine, aspartate and glutamate metabolism
                                    "00250"
                       Endometrial cancer
                                    "05213"
              Non-small cell lung cancer
                                    "05223"

> resultsBrain <- do.gene.set.analysis(EventsBySample = EventsBySampleBrain,
+     GeneSizes = GeneSizes08, GeneSets = KEGGPATHID2EXTID[c("hsa05213",
+         "hsa05223", "hsa00250")], Coverage = CoverageBrain, ID2name = ID2name,
+     gene.method = FALSE, perm.null.method = TRUE, perm.null.het.method = FALSE,
+     pass.null.method = TRUE, pass.null.het.method = FALSE)

[1] "Permutation null w/o heterogeneity"
[1] "Mon Oct 18 03:40:09 2010"
[1] "Passenger null w/o heterogeneity"
[1] "Mon Oct 18 03:40:09 2010"
[1] "Mon Oct 18 03:40:12 2010"
```

The resulting object is a data-frame, with NAs for the methods which were not implemented:

```
> resultsBrain

        p.values.gene q.values.gene p.values.perm.null q.values.perm.null
hsa05213            NA             1       9.019202e-16       2.705761e-15
hsa05223            NA             1       1.644843e-15       2.467264e-15
hsa00250            NA             1       3.105934e-01       3.105934e-01
        p.values.perm.null.het q.values.perm.null.het p.values.pass.null
hsa05213                     NA                      1       2.108377e-10
hsa05223                     NA                      1       2.735935e-11
hsa00250                     NA                      1       6.340137e-01
        q.values.pass.null p.values.pass.null.het q.values.pass.null.het
hsa05213       3.162565e-10                     NA                      1
hsa05223       8.207804e-11                     NA                      1
hsa00250       6.340137e-01                     NA                      1
```

3.1 Calculating gene scores

In order to implement the gene-oriented method, we need to have gene-level scores. We can obtain a variety of scores using the `cma.scores` function. The functions in this package are designed to use the `logLRT` scores.

```
> GeneScores <- cma.scores(cma.data = MutationsBrain, number.genes = nrow(GeneSizes08))
> head(GeneScores)

                   CaMP neglogPg   logLRT logitBinomialPosteriorDriver
A2M@@408      0.7113996 2.279349 1.845074                           NA
A4GALT@@21470 1.2286366 3.232244 2.798146                           NA
ABCA10@@388   0.9279687 2.597908 2.163654                           NA
ABCA4@@140    0.5650629 2.097952 1.663669                           NA
```

```
ABCA7@@194     0.9745022 2.673914 2.239663                          NA
ABCB6@@2071    1.2044322 3.146882 2.712739                          NA
               PoissonlogBF PoissonPosterior Poissonlmlik0 Poissonlmlik1
A2M@@408                 NA               NA            NA            NA
A4GALT@@21470            NA               NA            NA            NA
ABCA10@@388             NA               NA            NA            NA
ABCA4@@140              NA               NA            NA            NA
ABCA7@@194              NA               NA            NA            NA
ABCB6@@2071             NA               NA            NA            NA
```

4 Simulating data

We now demonstrate the use of the `sim.data.p.values` function, which simulates datasets under either the permutation or passenger null (see [2]), depending on whether `pass.null` is set to `TRUE` or `FALSE`, and calculates the p-values and q-values for those datasets for the selected methods. The simulations may also include spiked-in gene-sets, by using the `perc.samples` and `spiked.set.sizes` options. For example, if one desires to have two spiked-in gene-sets, both having 50 genes, but one having the probability of being altered in any given sample of 0.75 and the other of 0.95, then these parameters should be set to `perc.samples = c(75, 90)` and `spiked.set.sizes = 50`. The spiked-in gene-sets are artificially created with hypothetical genes (for more details on how they are generated, see [2]. To simulate the data without spiked-in sets, under the permutation or passenger null hypotheses, the parameters should be set as following: `perc.samples = NULL`, `spiked.set.sizes = NULL`. The object outputted by `sim.data.p.values` is of the class `SetMethodsSims`. Note that this code takes several minutes to run:

```
> set.seed(831984)
> resultsSim <- sim.data.p.values(EventsBySample = EventsBySampleBrain,
+     Mutations = MutationsBrain, GeneSizes = GeneSizes08, Coverage = CoverageBrain,
+     GeneSets = KEGGPATHID2EXTID[c("hsa05213", "hsa05223", "hsa00250")],
+     ID2name = ID2name, nr.iter = 2, pass.null = TRUE, perc.samples = c(75,
+         95), spiked.set.sizes = c(50), show.iter = TRUE, gene.method = FALSE,
+     perm.null.method = TRUE, perm.null.het.method = FALSE, pass.null.method = TRUE,
+     pass.null.het.method = FALSE)

[1] "Currently simulating data: Iteration #1"
[1] "Mon Oct 18 03:40:13 2010"
[1] "Currently simulating data: Iteration #2"
[1] "Mon Oct 18 03:40:46 2010"
[1] ""
[1] "Implement methods: Iteration #1"
[1] "Permutation null w/o heterogeneity"
[1] "Mon Oct 18 03:41:17 2010"
[1] "Passenger null w/o heterogeneity"
[1] "Mon Oct 18 03:41:17 2010"
[1] "Mon Oct 18 03:41:19 2010"
[1] ""
[1] "Implement methods: Iteration #2"
[1] "Permutation null w/o heterogeneity"
```

```
[1] "Mon Oct 18 03:41:20 2010"
[1] "Passenger null w/o heterogeneity"
[1] "Mon Oct 18 03:41:20 2010"
[1] "Mon Oct 18 03:41:21 2010"

> resultsSim

Simulation results for gene-set analysis of mutations
  Data-generating mechanism : Passenger null
  Number of simulations      : 2
  Number of gene-sets        : 5
     Original  : 3
     Spiked-in : 2
  Spiked-in sets             :
     Probability of being mutated in a set : 0.75 0.95
     Length (as number of genes)           : 50

> slotNames(resultsSim)

 [1] "null.dist"        "perc.samples"    "spiked.set.sizes" "GeneSizes"
 [5] "GeneSets"         "Coverage"        "EventsBySample"   "Mutations"
 [9] "Scores"           "results"

> resultsSim@null.dist

[1] "Passenger null"
```

4.1 Manipulating the SetMethodsSims objects

We provide 2 functions to help manipulate the **SetMethodsSims** objects which result from the **sim.data.p.values** function: **extract.sims.method** and **combine.sims**. The **extract.sims.method** function is used to obtain a single data frame with the p-values or q-values from one of the specific methods. For instance, the command to get the p-values for the permutation null with no heterogeneity method is:

```
> extract.sims.method(resultsSim, "p.values.perm.null")

                         [,1]          [,2]
hsa05213        1.628379e-02 3.150681e-01
hsa05223        1.951091e-02 5.847419e-01
hsa00250        5.545376e-01 2.693547e-01
gene.set.50.75 6.645712e-10 2.550107e-14
gene.set.50.95 7.502657e-18 4.783445e-16
```

The function **combine.sims** may be used to combine 2 simulations:

```
> combine.sims(resultsSim, resultsSim)
```

```
Simulation results for gene-set analysis of mutations
  Data-generating mechanism : Passenger null
  Number of simulations     : 4
  Number of gene-sets       : 5
     Original  : 3
     Spiked-in : 2
  Spiked-in sets            :
     Probability of being mutated in a set : 0.75 0.95
     Length (as number of genes)           : 50
```

References

[1] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society B*, 57:289–300, 1995.

[2] S.M. Boca, K. Kinzler, Velculescu V.E., Vogelstein B., and Parmigiani G. Patient-oriented gene-set analysis for cancer mutation data. *Submitted*, 2010.

[3] M. Kanehisa and S. Goto. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27, 2000.

[4] M. Kanehisa, S. Goto, M. Furumichi, M. Tanabe, and M. Hirakawa. KEGG for representation and analysis of molecular networks involving diseases and drugs. *Nucleic Acids Research*, 38(Database issue):D355, 2010.

[5] M. Kanehisa, S. Goto, M. Hattori, K.F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa. From genomics to chemical genomics: new developments in KEGG. *Nucleic acids research*, 34(Database Issue):D354, 2006.

[6] G. Parmigiani, J. Lin, S. Boca, T. Sjoblom, KW Kinzler, VE Velculescu, and B. Vogelstein. Statistical methods for the analysis of cancer genome sequencing data. *Johns Hopkins University, Dept. of Biostatistics Working Papers*, page 126, 2007.

[7] D.W. Parsons, S. Jones, X. Zhang, J.C.H. Lin, R.J. Leary, P. Angenendt, P. Mankoo, H. Carter, I. Siu, et al. An Integrated Genomic Analysis of Human Glioblastoma Multiforme. *Science*, 321(5897):1807, 2008.

[8] EM Schaeffer, L. Marchionni, Z. Huang, B. Simons, A. Blackman, W. Yu, G. Parmigiani, and DM Berman. Androgen-induced programs for prostate epithelial growth and invasion arise in embryogenesis and are reactivated in cancer. *Oncogene*, 27(57):7180–7191, 2008.

[9] T. Sjoeblom, S. Jones, L.D. Wood, D.W. Parsons, J. Lin, T.D. Barber, D. Mandelker, R.J. Leary, J. Ptak, N. Silliman, et al. The Consensus Coding Sequences of Human Breast and Colorectal Cancers. *Science*, 314(5797):268–274, 2006.

[10] G.K. Smyth. Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments. *Statistical Applications in Genetics and Molecular Biology*, 3(1):1027, 2004.

[11] J.D. Storey and R. Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences of the United States of America*, 100(16):9440, 2003.

[12] L.D. Wood, D.W. Parsons, S. Jones, J. Lin, T. Sjoblom, R.J. Leary, D. Shen, S.M. Boca, T. Barber, J. Ptak, et al. The Genomic Landscapes of Human Breast and Colorectal Cancers. *Science*, 318(5853):1108, 2007.