The Discordant R Package: A Novel Approach to Differential Correlation

Charlotte Siska and Katerina Kechris

May 21, 2016

Contents

1 Introduction

Discordant is an R package that identifies pairs of features that correlate differently between phenotypic groups, with application to -omics datasets. Discordant uses a mixture model that âĂIJbinsâĂİ molecular feature pairs based on their type of coexpression. More information on the algorithm can be found in [1, 2]. The final output are posterior probabilities of differential correlation. This package can be used to determine differential correlation within one âĂŞomics dataset or between two âĂŞomics datasets (provided that both âĂŞomics datasets were taken from the same samples). Also, the type of data can be any type of âĂŞomics with normal or non-normal distributions. Some examples are metabolomics, transcriptomic, proteomics, etc.

The functions in the Discordant package provide a simple pipeline for intermediate R users to determine differentially correlated pairs. The final output is a table of molecular feature pairs and their respective posterior probabilities. Functions have been written to allow flexibility for users in how they interpret results, which will be discussed further. Currently, the package only supports the comparison between two phenotypic groups (e.g., disease vs control, mutant vs wildtype).

2 Discordant Algorithm

Discordant is originally derived from the Concordant algorithm written by [3, 4]. It was used to determine concordance between microarrays. We have applied it to determine differential correlation of features between groups [1, 2].

Using a three component mixture model and the EM algorithm, the model predicts if the correlation coefficients in phenotypic groups 1 and 2 for a molecular feature pair are dissimilar [1]. The correlation coefficients are generated for all possible molecular feature pairs witin an -omics dataset or between two -omics datasets. The correlation coefficients are transformed into z scores using Fisher's tranformation. The three components are -, + and 0 which correspond respectively to a negative, positive or no correlation. Molecular features that have correlation coefficients in *different* components are considered *differentially* correlated, as opposed to when correlation coefficients are in the *same* component then they are *equivalently* correlated.

	0	-	+
0	1	2	3
-	4	5	6
+	7	8	9

Table 1: Class Matrix for Three Component Mixture Model

The class matrix (Table 1) contains the classes that represent all possible paired-correlation scenarios. These scenarios are based off the components in the mixture models. Molecular features that have correlation coefficients in different components are considered differentially correlated, as opposed to when correlation coefficients are in the same component they are equivalently correlated. This can be visualized in the class matrix, where the rows represent the components for group 1 and the columns represent the components for group 2. The classes on the diagonal represent equivalent correlation (1, 5 and 9), and classes in the off-diagonal represent differential correlation (2, 3, 4, 6, 8).

After running the EM algorithm, we have 9 posterior probabilities for each molecular feature pair that correspond to the 9 classes in the class matrix. Since we want to summarize the probability that the molecular feature pair is differentially correlated, we sum the posterior probabilities representing the off-diagonal classes in the class matrix.

3 Example Data

All datasets are originally from the Cancer Genome Atlas (TCGA) and can be found at http://cancergenome.nih.gov/.

`TCGA_GBM_miRNA_microarray` Data is miRNA expression values from 10 control and 20 tumor samples for a Glioblastoma multiforme (GBM) comparison. The feature size was originally 470, but after features with outliers were filtered out feature size reduces to 331. In this sample dataset, random 10 features are present. The dataset was generated with an Agilent miRNA microarray.

`TCGA_GBM_transcript_microarray` Data is transcript (or mRNA) expression values from 10 control and 20 tumor samples in a GBM comparison. The feature size was originally 90797, but after features with outliers were filtered out feature size reduces to 72656. In this sample dataset, 20 random features are present. The dataset was generated with an Agilent 244k micorarray.

`TCGA_Breast_miRNASeq` Data is miRNA counts from 15 control and 45 tumor samples in a Breast Cancer comparison. The feature size was originally 212, but after features with outliers were filtered out feature size reduces to 200. In this sample dataset, 100 random features are present. Dataset was generated using Illumina HiSeq miRNASeq.

`TCGA_Breast_RNASeq` Data is transcript (or mRNA) counts from 15 control and 45 tumor samples in a Breast Cancer comparison. The feature size was originally 19414, but after features with outliers were filtered out feature size reduces to 16656. In this sample dataset, 100 random features are present. The dataset was generated using Illumina HiSeq RNASeq.

`TCGA_Breast_miRNASeq_voom` This dataset is the voom-transformed `TCGA_Breast_miRNASeq`.

`TCGA_Breast_RNASeq_vooim` This dataset is the voom-transformed `TCGA_Breast_RNASeq`.

4 Before Starting

4.1 Required Inputs

Within âĂŞomics refers to when the Discordant analysis is performed within one âĂŞomics dataset where all molecular features within a -omics dataset are paired to each other (e.g. transcript-transcript pairs in a microarray transcriptomics experiment).

Between -omics refers to analysis of two -omics datasets. Molecular feature pairs analyzed are between the two -omics, (e.g. transcript-protein, protein-metabolite) are paired.

x m by n matrix where m are features and n are samples. If only this matrix is provided, a within -omics analysis is performed.

y m by n matrix where m are features and n are samples. Optional, will induce between -omics analysis. Samples must be matched with those in x.

groups vector containing 1s and 2s that correspond to the location of samples in the columns of x (and y if provided). For example, the control group is group 1 and the experimental group 2, and the location of samples corresponding to the two groups matches the locations of 1s and 2s in the group vector

4.2 Outliers

In our work, we found that features with outliers would skew correlation and cause false positives. Our approach was to filter out features that had large outliers. With normal data, such as in microarrays, Grubbs' test can be used. The null hypothesis is that there are no outliers in the data, and so features with p-value ≤ 0.05 are kept. A simple R function is found in the `outliers` R package as `grubbs.test`.

Determining outliers in non-normal data is more complicated. We used the median absolute deviation (MAD). Normally, features are filtered if they are outside 2 or 3 MADs from the median [5]. This is not completely applicable to sequencing data, because sequencing data has large variance and a non-symmetrical distribution. We used "split MAD," which has been used before [6]. A left MAD is determined based on data left to the median and a right MAD is determined based on data to the right of the median. If there are any feature outside a factor of the left or right MAD from the median, there are featured out.

A function in Discordant is provided called `split.madOutlier`. The number of MAD outside of the median can be changed with option `threshold`. Another option is `filter0` which if `TRUE` will filter out any feature with at least one 0. Arguments returned are `mat.filtered`, which is the filtered matrix and `index` which is the index of features that are retained in `mat.filtered`.

```
> data(TCGA_Breast_miRNASeq)
> mat.filtered <- splitMADOutlier(TCGA_Breast_miRNASeq,
+       filter0 = TRUE, threshold = 4)
```

5 Create Correlation Vectors

To run the Discordant algorithm correlation vectors respective to each group are necessary for input, which are easy to create using the `createVectors`.

Each correlation coefficient represents the correlation between two molecular features. The type of molecular feature pairs depend if a within -omics or between -omics analysis is performed. Correlation between molecular features in the same -omics dataset is within -omics, and correlation between molecular features in two different -omics datasets is between -omics. Whether or not within -omics or between -omics analysis is performed depends on whether one or two matrices are parameters for this function. `createVectors` has two outputs:

v1 Correlation vector of molecular feature pairs corresponding to samples labeled 1 in group parameter.

v2 Correlation vector of molecular feature pairs corresponding to samples labeled 2 in group parameter.

```
> data(TCGA_GBM_miRNA_microarray)
> data(TCGA_GBM_transcript_microarray)
> groups <- c(rep(1,10), rep(2,10))

#Within -Omics
> vectors <- createVectors(TCGA_GBM_transcript_microarray,
+      groups = groups)
#Between -Omics
> vectors <- createVectors(TCGA_GBM_miRNA_microarray,
+      TCGA_GBM_transcript_microarray, groups = groups)
```

5.1 Correlation Metric

We also have included different options for correlation metrics. This argument is called `cor.method` and its default is `"spearman"`. Other options are `"pearson"`, `"bwmc"` and `"sparcc"`. For information and comparison of Spearman, Pearson and biweight midcorrelation (bwmc) please read this paper by Song et al [7]. We have also investigated correlation metrics in Discordant in relation to sequencing data, and found Spearman's correlation had the best performance [2].

The algorithm for SparCC was introduced by Friedman et al [8]. We use R code written by Huaying Fang [9] .

Example

```
> data(TCGA_GBM_miRNA_microarray)
> data(TCGA_GBM_transcript_microarray)
> groups <- c(rep(1,10), rep(2,10))

#Within -Omics
> vectors <- createVectors(TCGA_GBM_transcript_microarray,
+      groups = groups, cor.method = c("bwmc"))
#Between -Omics
> vectors <- createVectors(TCGA_GBM_miRNA_microarray,
+      TCGA_GBM_transcript_microarray, groups = groups,
+      cor.method = c("bwmc"))
```

6 Run the Discordant Algorithm

The Discordant Algorithm is implemented in the the function `discordantRun` which requires two correlation vectors and the original data. If the user wishes to generate their own correlation vector before inputting into the dataset, they can do so. However, the function will return an error message if the dimensions of the datasets inserted do not match the correlation vector.

The posterior probability output of the Discordant algorithm are the differential correlation posterior probabilities (the sum of the off-diagonal of the class matrix described in Table 1). If the user wishes to observe more detailed information, alternative outputs are available. `discordantRun` has five outputs:

discordPPMatrix Matrix of differential correlation posterior probabilities where rows and columns reflect features. If only x was inputted, then the rows and column names are the feature names, and the upper diagonal of the matrix are NAs to avoid repeating results. If x and y are inputted, the row names are features from x and the column names are features from y.

discordPPVector Vector of differential correlation posterior probabilities. This has the same information as discordPPMatrix, but in vector form.

classMatrix Matrix of classes with the highest posterior probability. Row and column names are the same as in discordPPMatrix depending if only x is inputted or both x and y.

classVector Vector of class with the highest posterior probability for each pair. This has the same information as classMatrix, but in vector form.

probMatrix Matrix of all posterior probabilities, where each row represents a feature pair and nine columns that represent the class within the class matrix. The values across each row add up to 1. Posterior probabilities in discordPPMatrix and discordPPVector are summed up from columns 2, 3, 4, 6, 7 and 8 which correspond to differential correlation classes (Table 1).

loglik The log likelihood.

```
> data(TCGA_GBM_miRNA_microarray)
> data(TCGA_GBM_transcript_microarray)
> groups <- c(rep(1,10), rep(2,10))

#Within -omics
> vectors <- createVectors(TCGA_GBM_transcript_microarray,
+      groups = groups)
> result <- discordantRun(vectors$v1, vectors$v2,
+      TCGA_GBM_transcript_microarray}
> result$discordPPMatrix[1:4,1:4]
             A_23_P138644 A_23_P24296 A_24_P345312 A_24_P571870
A_23_P138644           NA          NA           NA           NA
A_23_P24296    0.33050233          NA           NA           NA
A_24_P345312   0.24910061  0.47858580           NA           NA
A_24_P571870   0.07052135  0.23310301   0.07028103           NA
> head(result$discordPPVector)
A_23_P24296_A_23_P138644 A_24_P345312_A_23_P138644
              0.33050233                0.24910061
A_24_P345312_A_23_P24296 A_24_P571870_A_23_P138644
              0.44839316                0.07052135
A_24_P571870_A_23_P24296 A_24_P571870_A_24_P345312
              0.56846815                0.91395405
> result$classMatrix[1:4,1:4]
```

```
             A_23_P138644 A_23_P24296 A_24_P345312 A_24_P571870
A_23_P138644           NA          NA           NA           NA
A_23_P24296             1          NA           NA           NA
A_24_P345312            1           3           NA           NA
A_24_P571870            1           1            1           NA
> head(result$classVector)
A_23_P24296_A_23_P138644 A_24_P345312_A_23_P138644
                       1                         1
A_24_P345312_A_23_P24296 A_24_P571870_A_23_P138644
                       1                         1
A_24_P571870_A_23_P24296 A_24_P571870_A_24_P345312
                       4                         2
> head(result$probMatrix)
                                   1            2           3
A_23_P24296_A_23_P138644  0.63574038 5.223996e-07 0.289699897
A_24_P345312_A_23_P138644 0.74859348 2.151027e-02 0.012949903
A_24_P345312_A_23_P24296  0.92584634 4.666010e-03 0.022843630
A_24_P571870_A_23_P138644 0.55002415 1.359646e-03 0.016168496
A_24_P571870_A_23_P24296  0.43013074 5.668452e-04 0.014985341
A_24_P571870_A_24_P345312 0.05392322 8.877314e-01 0.001065886
                                   4            5           6
A_23_P24296_A_23_P138644  0.01293366 2.138702e-09 7.37399e-03
A_24_P345312_A_23_P138644 0.19191496 1.124351e-03 4.15700e-03
A_24_P345312_A_23_P24296  0.00260480 2.636222e-06 8.20242e-05
A_24_P571870_A_23_P138644 0.40351609 2.017512e-04 1.47538e-02
A_24_P571870_A_23_P24296  0.52075349 1.377220e-04 2.24260e-02
A_24_P571870_A_24_P345312 0.00961640 3.201954e-02 2.38872e-04
                                   7            8           9
A_23_P24296_A_23_P138644  0.02049424 1.062232e-08 0.033757287
A_24_P345312_A_23_P138644 0.01824136 3.271036e-04 0.001181551
A_24_P345312_A_23_P24296  0.04019691 1.279636e-04 0.003629672
A_24_P571870_A_23_P138644 0.01257563 1.944655e-05 0.001380934
A_24_P571870_A_23_P24296  0.00972846 7.995730e-06 0.001263389
A_24_P571870_A_24_P345312 0.00138369 1.391785e-02 0.000103189
> result$loglik
[1] 1129.558
> # Between -omics
> vectors <- createVectors(TCGA_GBM_miRNA_microarray,
+      TCGA_GBM_transcript_microarray, groups = groups)
> result <- discordantRun(vectors$v1, vectors$v2,
+       TCGA_GBM_miRNA_microarray,
+       TCGA_GBM_transcript_microarray)
> result$discordPPMatrix[1:3,1:3]
               A_23_P138644 A_23_P24296 A_24_P345312
hsa-miR-19b-5p    0.9734173  0.19568366   0.70425484
hsa-miR-206-5p    0.8422418  0.03809339   0.91989561
hsa-miR-369-5p    0.9509883  0.05088263   0.99194855
> head(result$discordPPVector)
hsa-miR-19b-5p_A_23_P138644  hsa-miR-19b-5p_A_23_P24296
                  0.9734173                   0.8422418
hsa-miR-19b-5p_A_24_P345312 hsa-miR-19b-5p_A_24_P571870
                  0.9509883                   0.8983236
 hsa-miR-19b-5p_A_32_P71885  hsa-miR-19b-5p_A_32_P82889
                  0.9154871                   0.6706972
> result$classMatrix[1:3,1:3]
               A_23_P138644 A_23_P24296 A_24_P345312
hsa-miR-19b-5p            7           1            4
hsa-miR-206-5p            2           1            3
hsa-miR-369-5p            4           1            2
> head(result$classVector)
hsa-miR-19b-5p_A_23_P138644  hsa-miR-19b-5p_A_23_P24296
                          7                           2
hsa-miR-19b-5p_A_24_P345312 hsa-miR-19b-5p_A_24_P571870
                          4                           7
 hsa-miR-19b-5p_A_32_P71885  hsa-miR-19b-5p_A_32_P82889
                          4                           4
> head(result$probMatrix)
                                    1          2          3
hsa-miR-19b-5p_A_23_P138644 0.0265823 1.6948e-02 7.4477e-09
hsa-miR-19b-5p_A_23_P24296  0.1575141 8.3361e-01 2.2858e-10
hsa-miR-19b-5p_A_24_P345312 0.0488204 2.8058e-05 1.1427e-03
hsa-miR-19b-5p_A_24_P571870 0.1016756 4.2631e-02 7.0747e-08
hsa-miR-19b-5p_A_32_P71885  0.0844775 3.4488e-06 2.7536e-02
hsa-miR-19b-5p_A_32_P82889  0.2333993 1.1262e-01 1.1956e-07
                                     4           5        6
hsa-miR-19b-5p_A_23_P138644 1.2909e-11 2.82374e-12 2.26e-19
```

```
hsa-miR-19b-5p_A_23_P24296  1.3327e-04 2.44029e-04 1.21e-14
hsa-miR-19b-5p_A_24_P345312 9.4829e-01 1.90403e-04 1.50e-03
hsa-miR-19b-5p_A_24_P571870 7.6664e-10 1.10858e-10 3.34e-17
hsa-miR-19b-5p_A_32_P71885  8.6805e-01 1.23439e-05 1.98e-02
hsa-miR-19b-5p_A_32_P82889  5.5782e-01 9.59033e-02 1.80e-08
                                     7           8        9
hsa-miR-19b-5p_A_23_P138644 8.4389e-01 1.12569e-01 3.33e-07
hsa-miR-19b-5p_A_23_P24296  4.1233e-03 4.37311e-03 8.73e-12
hsa-miR-19b-5p_A_24_P345312 2.3763e-05 2.67832e-09 8.07e-07
hsa-miR-19b-5p_A_24_P571870 7.8738e-01 6.83115e-02 7.76e-07
hsa-miR-19b-5p_A_32_P71885  4.9158e-05 3.95251e-10 2.30e-05
hsa-miR-19b-5p_A_32_P82889  2.1812e-04 2.06579e-05 1.65e-10
> result$loglik
[1] 1169.986
```

6.1 Subsampling

Subsampling is an option to run the EM algorithm with a random sample of independent feature pairs are drawn. This is repeated for a number of samplings, and then the average of these parameters are used to maximize posterior probabilities for all feature pairs. This option was introduced to speed up Discordant method and to also solve the independence assumption. There are some implementation issues which are explained in Siska et al, submitted [2].

The argument `subsampling` must be set to `TRUE` for subsampling to be used. The number of independent feature pairs to be subsampled is determined by the argument `subSize` whose default value is the number of rows in `x`. The number of independent feature pairs must be less or equal to the number of features in `x` and `y`. The number of random samplings to be run is set by the argument `iter`, whose default value is 100.

```
> data(TCGA_GBM_miRNA_microarray)
> data(TCGA_GBM_transcript_microarray)
> groups <- c(rep(1,10), rep(2,10))

> vectors <- createVectors(TCGA_GBM_miRNA_microarray,
+      TCGA_GBM_transcript_microarray, groups = groups)
> result <- discordantRun(vectors$v1, vectors$v2,
+       TCGA_Breast_miRNASeq,
+       TCGA_Breast_RNASeq, subsampling = TRUE,
+       iter = 200, subSize = 20)
```

	0	-	−	+	++
0	1	2	3	4	5
-	6	7	8	9	10
−	11	12	13	14	15
+	16	17	18	19	20
++	21	22	23	24	25

Table 2: Class Matrix for Five Component Mixture Model

6.2 Increase Component Size

We also provide the option to increase component size from three to five in the mixture model. The number of classes in the class matrix increases, as seen in Table 2. Incorporating the extra components means that it is possible to identify elevated differential correlation, which is when there are associations in both groups in the same direction but one is more extreme. Using this options introduces more parameters, which does have an effect on run-time. We also found that using the five mixture component mixture model reduces performance compared to the three component mixture model [2]. However, the option is available if users wish to explore more types of differential correlation.

The default is to run the three component mixture model and can be changed with option `components`.

Example:

```
> data(TCGA_GBM_miRNA_microarray)
> data(TCGA_GBM_transcript_microarray)
> groups <- c(rep(1,10), rep(2,10))

> vectors <- createVectors(TCGA_GBM_miRNA_microarray,
+       TCGA_GBM_transcript_microarray, groups = groups)
> result <- discordantRun(vectors$v1, vectors$v2,
+       TCGA_GBM_transcript_microarray,
+       TCGA_GBM_miRNA_microarray, components = 5)
```

7 Example Run with Microarrays

```
data(TCGA_GBM_miRNA_microarray)
data(TCGA_GBM_transcript_microarray)
groups <- c(rep(1,10), rep(2,10))

#Within -Omics

vectors <- createVectors(TCGA_GBM_transcript_microarray,
      groups = groups)
result <- discordantRun(vectors$v1, vectors$v2,
      TCGA_GBM_transcript_microarray)

#Between -Omics

vectors <- createVectors(TCGA_GBM_miRNA_microarray,
      TCGA_GBM_transcript_microarray, groups = groups)
result <- discordantRun(vectors$v1, vectors$v2,
      TCGA_GBM_miRNA_microarray,
      TCGA_GBM_transcript_microarray)
```

References

[1] C. Siska, R. Bowler, and K. Kechris, “The discordant method: a novel approach for differential correlation,” *Bioinformatics*, vol. 32, no. 5, pp. 690–696, 2015.

[2] C. Siska. and K. Kechris, “Differential correlation for sequencing data.” Submitted, 2016.

[3] Y. Lai, B. ling Adam, R. Podolsky, and J.-X. She, “A mixture model approach to the tests of concordance and discordance between two large-scale experiments with two-sample groups,” *Bioinformatics*, vol. 23, no. 10, pp. 1243–1250, 2007.

[4] Y. Lai, F. Zhang, T. K. Naya, R. Modarres, N. H. Lee, and T. A. McCaffrey, “Concordant integrative gene set enrichment analysis of multiple large-scale two-sample expression data sets,” *BMC Genomics*, vol. 15, 2014.

[5] C. Leys, O. Klein, P. Bernard, and L. Licata, “Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median,” *Journal of Experimental Social Psychology*, vol. 49, no. 4, 2013.

[6] P. M. Magwene, J. H. Willis, J. K. Kelley, and A. Siepel, “The statistics of bulk segregant analysis using next generation sequencing,” *PLoS Computational Biology*, vol. 7, no. 11, 2011.

[7] L. Song, P. Langfelder, and S. Horvath, “Comparison of co-expression measures: mutual information, correlation, and model based indices,” *BMC Bioinformatics*, vol. 13, no. 328, 2012.

[8] J. Friedman and E. J. Alm, “Inferring correlation networks from genomic survey data,” *PLoS Computational Biology*, 2012.

[9] H. Fang, C. Huang, H. Zhao, and M. Deng, “Cclasso: correlation inference for compositional data through lasso,” *Bioinformatics*, vol. 31, no. 19, pp. 3172–3180, 2015.