Discordant:

Mixture Model for Determining Differentially Coexpressed Pairs

User’s Guide

Charlotte Siska and Katerina Kechris

February 6, 2015

Contents

1. Introduction

2. Citing Discordant.

3. Single –omics analysis

1. Brief Introduction

2. Required Inputs

3. Creating Vectors

4. Running Discordant Algorithm

4. Dual –omics analysis

1. Introduction

Discordant is a package for the analysis of molecular feature pairs derived from –omics to data to determine if they coexpress differently between phenotypic groups. Discordant is based on a mixture model that “bins” pairs based on their type of coexpression. More information on the algorithm can be found in Siska, et. al. Final output is summed up posterior probabilities of differential coexpression bins. This package can be used to determine differential coexpression within one –omics dataset or between two –omics datasets (provided that both –omics datasets were taken from the same samples). Also, the type of data can be any type of –omics, such as metabolomics, transcriptomic, proteomic, etc. as long as the data is continuous (numerical) rather than discrete (categorical, count).

The functions in the Discordant package provide a simple pipeline for moderate R users to determine differentially coexpressed pairs. The final output is a table of molecular feature pairs and their respective posterior probalities. Functions have been written to allow flexibility for users in how they interpret results, which will be discussed further.

2. Citing Discordant

Discordant is originally derived from the Concordant algorithm written by Lai, et. al. When citing Discordant, please also include Lai, et. al in references.

Lai, Y., Adam, B. -l., Podolsky, R., and She, J.-X. (2007). A mixture model approach to the tests of concordance and discordance between two large-scale experiments with two-sample groups. Bioinformatics *23*, 1243–1250.

Siska, Kechris….

3. Workflow

1. Brief Introduction

Single –omics is when Discordant analysis is done within one –omics dataset. This means that all molecular features are analyzed to each other, rather than separating them by molecular type. This is mainly applicable to one –omics dataset, such as a single microarray experiment.

2. Required inputs

All analyses require the following inputs:

**data1/data2**

An m by n matrix of expression/abundance values, where m is number of features and n is number of samples. Values should already be pre-processed and normalized respective to the type of –omics. Dataset should be separated by group, where data1 contains data for group 1 and data2 contains data for group2. If running dual –omics, -omics datasets should be stacked on top of each other.

**featureNames**

List of feature names in same order of m rows.

Dual –omics require the following inputs:

**featureSize1/featureSize2**

number of features in first –omics, and number of features in second –omics respectively.

**featureNames1/featureNames2**

names of features in first –omics, and names of features in second –omics respectively in same order of m rows in dataset1 and dataset2.

3. Creating correlation vectors.

Vectors of correlation coefficients for all possible molecular feature pairs need to be created before running Discordant algorithm. The inputs for this are the data sets. For a single-omics analysis, we must specify that we are not running dual-omics analysis by setting multOmics to FALSE.

> vectors <- createVectors(data1, data2, multOmics = FALSE)

When running for dual –omics, we specific set multOmics to TRUE and also specify the length of the featureSize1.

vectors <- createVectors(data1, data2, multOmics = TRUE, featureSize = 10)

4. Running Discordant Algorithm

Once correlation vectors are obtained, it is possible to run the discordant algorithm. Note that the functions for creating correlation vectors and running the discordant algorithm are separate, making it possible for the user to input their own correlation vectors if they wish to do so.