UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL
PROGRAMA DE PÓS-GRADUAÇÃO EM BIOINFORMÁTICA

DIEGO ARTHUR DE AZEVEDO MORAIS

TRANSCRIPTOGRAMER: PACOTE EM R PARA ANÁLISE TRANSCRICIONAL

NATAL - RN
2018

DIEGO ARTHUR DE AZEVEDO MORAIS

TRANSCRIPTOGRAMER: PACOTE EM R PARA ANÁLISE TRANSCRICIONAL

Defesa de Mestrado apresentada ao Programa de Pós-Graduação em Bioinformática da Universidade Federal do Rio Grande do Norte.

Área de concentração: Bioinformática
Linha de Pesquisa: Biologia de Sistemas
Orientador: Prof. Dr. Rodrigo Juliani Siqueira Dalmolin

NATAL - RN
2018

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Setorial Prof. Leopoldo Nelson - -Centro de Biociências - CB

Morais, Diego Arthur de Azevedo.
Transcriptogramer: pacote em R para análise transcricional / Diego Arthur de Azevedo Morais. - Natal, 2018.
73 f.: il.

Dissertação (Mestrado) - Universidade Federal do Rio Grande do Norte. Instituto Metrópole Digital. Programa de Pós-Graduação em Bioinformática.
Orientador: Prof. Dr. Rodrigo Juliani Siqueira Dalmolin.

1. Software - Dissertação. 2. Análise transcricional - Dissertação. 3. Biologia de sistemas - Dissertação. 4. Interação proteína-proteína - Dissertação. 5. Associação funcional - Dissertação. 6. Transcriptograma - Dissertação. I. Dalmolin, Rodrigo Juliani Siqueira. II. Universidade Federal do Rio Grande do Norte. III. Título.

RN/UF/BSE-CB CDU 004.4

Elaborado por KATIA REJANE DA SILVA - CRB-15/351

DIEGO ARTHUR DE AZEVEDO MORAIS

TRANSCRIPTOGRAMER: PACOTE EM R PARA ANÁLISE TRANSCRICIONAL

Defesa de Mestrado apresentada ao Programa de Pós-Graduação em Bioinformática da Universidade Federal do Rio Grande do Norte.

Área de concentração: Bioinformática
Linha de Pesquisa: Biologia de Sistemas
Orientador: Prof. Dr. Rodrigo Juliani Siqueira Dalmolin

Natal, 29 de junho de 2018.

BANCA EXAMINADORA

Prof. Dr. Rodrigo Juliani Siqueira Dalmolin
Universidade Federal do Rio Grande do Norte
(Presidente)

Prof. Dr. Jorge Estefano Santana de Souza
Universidade Federal do Rio Grande do Norte
(Examinador Interno do Programa)

Prof. Dr. Mauro Antônio Alves Castro
Universidade Federal do Paraná
(Examinador Externo à Instituição)

RESUMO

O transcriptograma, um método utilizado na análise de transcriptomas, utiliza dados de interação proteína-proteína para construir uma lista ordenada de genes. Nesta lista, genes são posicionados de forma que a probabilidade de interação entre seus produtos decaia exponencialmente com o aumento da distância entre suas posições. A lista ordenada de genes é então utilizada para calcular o valor de expressão médio de genes funcionalmente associados numa janela com raio configurável, permitindo a expressão diferencial de grupos gênicos não pré-definidos em estudos caso-controle. O objetivo deste estudo é a implementação de um pacote em R que use transcriptogramas e integre funcionalidades de pacotes já conhecidos pela comunidade científica, capaz de realizar: expressão diferencial, enriquecimento funcional, e visualização de rede. O pacote transcriptogramer foi implementado e encontra-se disponível no Bioconductor, um repositório para *softwares open source* desenvolvidos na linguagem R para utilização em bioinformática. Numa comparação entre o transcriptogramer e um *pipeline* combinando funcionalidades dos pacotes limma e topGO, observou-se que o transcriptogramer identificou aproximadamente 10 vezes mais termos do *Gene Ontology* significativamente enriquecidos, dentre os quais foram encontrados a maioria dos termos identificados pelo *pipeline* convencional.

Palavras-chave: *Software*, análise transcricional, biologia de sistemas, interação proteína-proteína, associação funcional, transcriptograma.

ABSTRACT

The transcriptogram, a method used on transcriptomes analysis, uses protein-protein interaction data to build an ordered gene list. On this list, genes are placed such that the probability of interaction between its products exponentially decreases with the increase of the distance between its positions. The ordered gene list is then used to calculate the average expression value of functionally associated genes in a window with settable radius, allowing the differential expression of non-predefined gene sets in case-control studies. This study aims to implement an R package that uses transcriptograms and integrates features from packages known by the scientific community, able to perform: differential expression, functional enrichment, and network visualization. The transcriptogramer package was implemented and is available at Bioconductor, a repository for open source softwares developed in the R language for use in bioinformatics. In a comparison between the transcriptogramer and a pipeline combining features from limma and topGO packages, was noticed that the transcriptogramer identified nearly 10 times more Gene Ontology terms significantly enriched, among which most of the terms identified by the conventional pipeline were found.

Keywords: Software, transcriptional analysis, systems biology, protein-protein interaction, functional association, transcriptogram.

LISTA DE FIGURAS

Figura 1. Representação visual do ordenamento. (A) Estado inicial de uma matriz de adjacência contendo 4386 proteínas do *Saccharomyces cerevisiae*. (B) Matriz de adjacência resultante do processo de ordenamento. A interação entre duas proteínas é representada por um ponto preto. Na matriz resultante é perceptível o agrupamento das arestas na diagonal secundária.....12
Figura 2. Número de *downloads* por mês durante o primeiro semestre de 2018.....27
Figura 3. Condições levadas em consideração pela janela. A lista ordenada de genes é representada pelo retângulo branco e a janela é representada pela área azul. Min e Max representam as posições extremas da lista ordenada de genes. L1 e L2 representam as posições extremas da janela. (A) Os extremos da janela não ultrapassam os extremos da lista ordenada de genes, assim, uma região sequencial é selecionada. (B e C) Um dos extremos da janela ultrapassa um dos extremos da lista ordenada de genes e precisa ser recalculado, nestes casos são selecionadas duas áreas.....28
Figura 4. Identificação dos grupos gênicos alterados. (A) Posições dos centros das janelas cuja variação da expressão foi significativa entre as condições. (B) Delimitação das extremidades de cada grupo funcionalmente associado para o raio 2. A utilização do valor 3 como raio mesclaria os grupos C1 e C2.....29

LISTA DE ABREVIATURAS

CFM	*Cost Function Method*
GEO	*Gene Expression Omnibus*
GO	*Gene Ontology*
PPI	Interação proteína-proteína (do inglês, *protein-protein interaction*)

SUMÁRIO

1 INTRODUÇÃO 10

1.1 Biologia de sistemas 10

1.2 Tecnologias de alta vazão 11

1.3 Transcriptograma 12

1.4 Linguagem R 13

1.5 Repositórios de pacotes 14

1.6 Justificativa do estudo 15

2 OBJETIVOS 16

2.1 Geral 16

2.2 Específicos 16

CAPÍTULO I 17

3 DISCUSSÃO 27

3.1 Transcriptogramer 27

4 CONCLUSÃO 31

REFERÊNCIAS 32

ANEXOS 35

1 INTRODUÇÃO

1.1 Biologia de sistemas

O reducionismo, método de estudo no qual fenômenos complexos são divididos em componentes mais simples, foi uma estratégia bastante utilizada nas ciências biológicas. A segmentação de sistemas celulares em partes mais básicas, a serem estudadas isoladamente, provou-se uma estratégia essencial para a compreensão da biologia molecular e de sistemas biológicos como um todo. Entretanto, existem detalhes biológicos que não podem ser explicados utilizando uma abordagem reducionista. Como alternativa, a biologia de sistemas propõe uma visão holística, baseada na integração de dados multidisciplinares, com o objetivo de inferir associações e gerar novos conhecimentos sobre processos biológicos (SAURO *et al*., 2006).

Estudos recentes mostram que cada gene interage em média com outros 4 ou 8 genes, e estão envolvidos com 10 funções biológicas (LI *et al*., 2018). Uma das vertentes da biologia de sistemas consiste na utilização da teoria de grafos na análise de dados biológicos. Assim, informações sobre a interação entre genes, ou seus produtos (FRANCESCHINI *et al*., 2013), podem ser utilizadas na montagem de redes densamente conectadas. Redes associadas a processos biológicos com potencial de afetar a saúde humana podem ser integradas a dados de expressão clínicos, auxiliando na identificação de possíveis alvos terapêuticos de uma determinada doença complexa por exemplo.

Avanços tecnológicos possibilitaram o desenvolvimento de tecnologias de alta vazão para o estudo de dados biológicos. Tecnologias como microarranjo e RNA-Seq impulsionam o estudo de ciências ômicas, como genômica e transcriptômica, por meio da produção de vários gigabytes de dados por dia. Alguns dados podem não ser significativos durante a análise em apenas um nível ômico, entretanto, a integração de dados produzidos por níveis ômicos distintos pode formar associações capazes de esclarecer mecanismos moleculares desconhecidos até então (D'ARGENIO, 2018). A integração e análise de dados biológicos requer o uso de métodos computacionais. Portanto, ferramentas que implementem uma abordagem baseada em biologia de sistemas são úteis para que bioinformatas e biólogos convertam dados em novos conhecimentos biológicos.

1.2 Tecnologias de alta vazão

Microarranjo, uma tecnologia criada na década de 1990, é bastante utilizada em estudos de expressão gênica (ZHAO *et al*., 2014). A capacidade desta tecnologia de coletar dados de dezenas de milhares de transcritos contribuiu para a resolução de diversos problemas biológicos. Dados públicos gerados por experimentos que utilizaram plataformas de microarranjos podem ser encontrados no *Gene Expression Omnibus* (GEO) (EDGAR, 2002; BARRETT *et al*., 2013). Estes dados de expressão podem ser utilizados na identificação de genes diferencialmente expressos entre tecidos que se encontram em condições distintas, caso e controle, que podem então ser utilizados em análises de enriquecimento, com o objetivo de identificar processos biológicos. Dados de microarranjo são populares em análises transcricionais devido ao seu custo acessível, entretanto, esta tecnologia possui limitações. A hibridização utilizada por microarranjos possui precisão limitada, principalmente para transcritos que possuem baixa abundância na amostra, o que induz variações na medida da expressão. Apesar da capacidade de identificar diversos transcritos, sondas de microarranjo são projetadas para hibridizarem com um escopo limitado de transcritos conhecidos.

A tecnologia de RNA-Seq surgiu posteriormente como uma alternativa para estas limitações. Esta tecnologia é capaz de identificar novos transcritos, independentemente da existência de anotações de genoma, e é mais precisa na medição da expressão. Entretanto, dados de RNA-Seq requerem muito mais processamento computacional (PERTEA *et al*., 2016; ANDERS *et al*., 2013) do que o requerido por dados de microarranjo. Diversos métodos computacionais podem ser utilizados para o alinhamento de *reads*, quantificação de genes/transcritos, e identificação de genes diferencialmente expressos. Experimentos que utilizaram RNA-Seq e microarranjo na identificação de genes diferencialmente expressos (ZHAO *et al*., 2014) mostram que os resultados apresentam uma boa sobreposição. Por este motivo, alguns estudos utilizam ambas as tecnologias para a análise de dados (XU *et al*., 2013), com o foco na análise dos resultados em comum. Ferramentas computacionais capazes de realizar análises, tanto em dados de microarranjo quanto em dados de RNA-Seq, são necessárias para a exploração da massiva quantidade de dados públicos disponíveis.

1.3 Transcriptograma

O transcriptograma (RYBARCZYK-FILHO *et al*., 2011), um método baseado em biologia de sistemas para análise de transcriptomas, utiliza dados de interação proteína-proteína (PPI) para identificar grupos gênicos alterados em estudos caso-controle. Para este fim, inicialmente é utilizado o algoritmo *Cost Function Method* (CFM) (KUENTZER *et al*., 2014) para organizar informações de PPI em uma dimensão. Este algoritmo troca um par de linhas ou colunas de uma matriz de adjacência, calculando o custo da matriz resultante. Quanto menor o custo da matriz obtida, maior a proximidade entre proteínas interagentes. Caso o custo da matriz resultante seja menor, ou seja aceito como viável, esta passa a ser a matriz de referência e o processo é repetido. Repetições são realizadas por meio de simulações de Monte Carlo, e o critério de aceitação para o custo é baseado no algoritmo de arrefecimento simulado. Um exemplo do resultado deste processo pode ser visto na Figura 1.

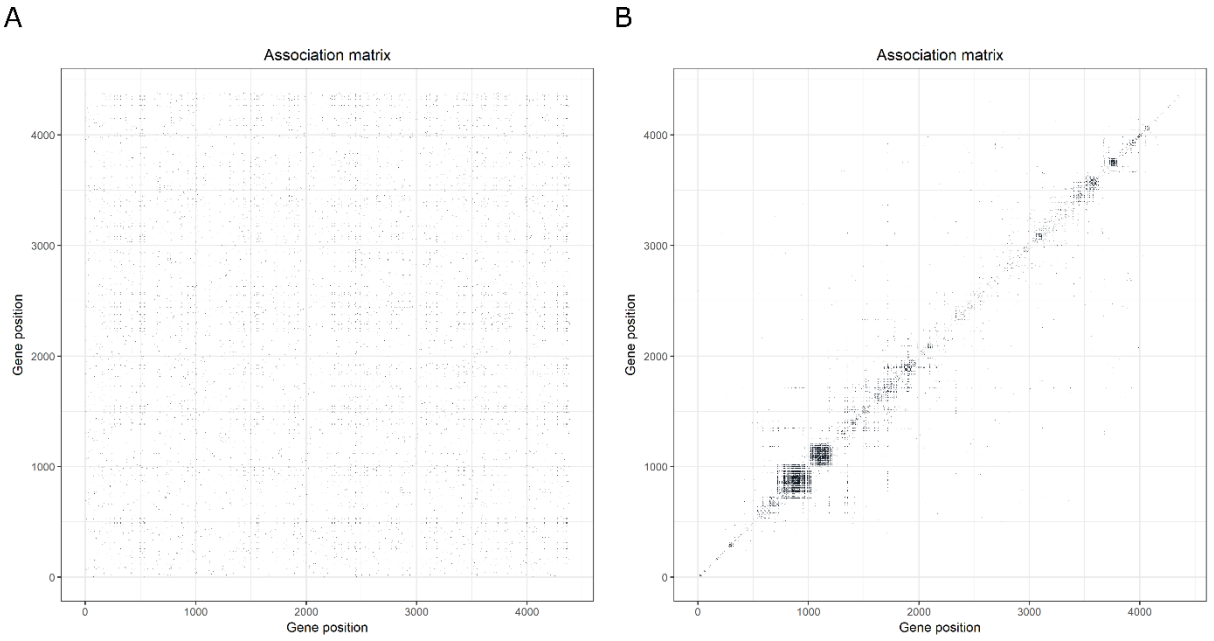


Figura 1. Representação visual do ordenamento. (A) Estado inicial de uma matriz de adjacência contendo 4386 proteínas do *Saccharomyces cerevisiae*. (B) Matriz de adjacência resultante do processo de ordenamento. A interação entre duas proteínas é representada por um ponto preto. Na matriz resultante é perceptível o agrupamento das arestas na diagonal secundária.

Como resultado, é gerada uma lista ordenada de genes na qual a probabilidade de interação entre dois genes quaisquer decai exponencialmente com o aumento da distância entre suas posições. Na representação visual do resultado do ordenamento (Figura 1) pode ser observado o agrupamento hierárquico de redes de interação proteína-proteína.

Dados de expressão de um determinado experimento são então projetados na lista e uma janela deslizante, cujo raio pode ser ajustado, obtém a média de expressão de grupos gênicos funcionalmente associados. A janela atribui o valor obtido ao gene posicionado no seu centro e desloca-se para a próxima posição da lista, levando em consideração condições de contorno periódico para genes próximos das extremidades da lista. A utilização desta janela deslizante reduz o ruído aleatório presente nos dados de expressão (DA SILVA *et al*., 2014) e permite a análise de um perfil de expressão, fornecendo uma visão geral do metabolismo celular. O resultado deste procedimento é um transcriptograma, uma lista ordenada de genes na qual cada posição representa a expressão de um grupo de $2 \times \mathrm{raio} + 1$ genes. Possibilitando assim a utilização de um teste estatístico para a identificação de grupos gênicos alterados quando duas condições são analisadas.

O transcriptograma possui uma implementação escrita em C e Python, já utilizada em alguns casos de uso descritos na literatura (FERRAREZE *et al*., 2017; DE ALMEIDA *et al*., 2016; XAVIER *et al*., 2017). A ferramenta pode ser executada em sistemas operacionais baseados em Windows e encontra-se disponível no repositório público do Instituto de Física da Universidade Federal do Rio Grande do Sul. A ferramenta é capaz de gerar saídas, como imagens e arquivos de texto, de fácil interpretação. Entretanto, sua forma de implementação e distribuição não são facilmente integráveis com *pipelines* existentes de bioinformática.

1.4 Linguagem R

A linguagem R, criada por Robert Gentleman e Ross Ihaka no Departamento de Estatística da Universidade de Auckland, é uma linguagem interpretada e *open source* (PENG, 2018). Códigos escritos em R podem ser executados em praticamente qualquer sistema operacional (Windows, Linux, Mac OS X). Porém, por ser uma linguagem interpretada, a velocidade de execução de algoritmos de complexidade alta demanda mais tempo computacional. Para minimizar

tal desvantagem, a linguagem permite a chamada de códigos escritos com outras linguagens de programação, como C++ por exemplo. A linguagem é atualizada com frequência, com atualizações disponibilizadas ao longo do ano sempre que necessário, sendo a maior atualização anual geralmente disponibilizada em outubro. Esta frequência no lançamento de versões é consequência da atividade de sua comunidade, composta em parte por desenvolvedores. A equipe que mantém a linguagem R, e desenvolvedores da comunidade, frequentemente desenvolvem pacotes. Pacotes agregam funções projetadas com o objetivo de resolver problemas de um determinado tema, como a manipulação de grafos, importação e exportação de arquivos, e até mesmo a criação de pacotes. Existem diversos pacotes disponíveis, vários deles voltados para a execução de testes estatísticos, visualização de dados, e criação de imagens com qualidade em nível de publicação, pontos fortes da linguagem.

A linguagem R possui um protocolo de desenvolvimento de pacotes que combina componentes de *software* e documentação (GENTLEMAN *et al*., 2004), facilitando a integração de pacotes ao instalar automaticamente as versões corretas das dependências necessárias. Esta facilidade de integração reduz o tempo necessário para o desenvolvimento de um novo pacote. A documentação de pacotes geralmente envolve 2 documentos: vinheta e manual de referência. A vinheta descreve brevemente as funcionalidades do *software*, apresentando um tutorial que demonstra os comandos a serem executados para tal. O manual de referência descreve todas as funções disponíveis, com detalhes sobre seu funcionamento, argumentos de entrada e saída, referências do método utilizado. Por ser gratuita, multiplataforma, e possuir excelentes capacidades gráficas e estatísticas, a linguagem R é muito utilizada no desenvolvimento de ferramentas de bioinformática (BLOOMFIELD, 2009).

1.5 Repositórios de pacotes

Repositórios armazenam pacotes com o objetivo de facilitar sua distribuição e instalação. O Bioconductor (GENTLEMAN *et al*., 2004), o principal repositório de pacotes de bioinformática, possui pacotes distribuídos em 3 categorias: *software*, anotação, experimentos. Pacotes de *software* agregam funções voltadas para a análise e compreensão de dados genômicos de alta vazão. Pacotes de anotação

contém bancos de dados descrevendo detalhes relevantes na execução de análises, como informações sobre quais genes hibridizam com sondas de uma determinada plataforma de microarranjo. Pacotes de experimento contém dados, brutos ou pré-processados, coletados para a realização de um determinado experimento. O Bioconductor possui mais de 1500 pacotes de *software* disponíveis, além de pacotes pertencentes às demais categorias, com mais pacotes sendo adicionados geralmente em abril e outubro de cada ano.

Pacotes submetidos ao Bioconductor são avaliados por um sistema automatizado e por um revisor humano. Para que um pacote seja aceito ele precisa utilizar métodos descritos na literatura, ser diferente de métodos já implementados por pacotes disponíveis no repositório, estar implementado de modo a facilitar a integração com pacotes já existentes, e possuir uma documentação capaz de ser executada sem erros e que utilize exemplos reprodutíveis. Uma vez aceito, o pacote é incorporado ao repositório de desenvolvimento e disponibilizado no próximo lançamento anual. Pacotes do Bioconductor são bem aceitos pela comunidade científica. Como exemplo é possível citar o pacote limma (RITCHIE *et al*., 2015), conhecido por realizar expressão diferencial em diversas circunstâncias de forma rápida (CONESA *et al*., 2016), baixado 331952 vezes apenas no ano de 2017.

1.6 Justificativa do estudo

Ferramentas computacionais existentes, como limma, DESeq2 (LOVE *et al*., 2014) e edgeR (ROBINSON *et al*., 2010; MCCARTHY *et al*., 2012), avaliam a expressão gênica de forma individual, embora o fato de que genes não trabalham isoladamente seja conhecido (LI *et al*., 2018). Sistemas biológicos, e consequentemente processos biológicos, são afetados pela interação coordenada de produtos gênicos, que interagem entre si ou são co-regulados. O uso de metodologias que integrem dados para a compreensão de sistemas, como o transcriptograma, possibilita a conversão de dados em novos conhecimentos biológicos. Apesar do transcriptograma já possuir uma ferramenta, uma nova implementação de maneira modular facilitaria a integração com *pipelines* existentes de bioinformática. Assim, uma implementação em R possibilitaria a submissão ao Bioconductor, um meio de distribuição mais apropriado, contribuindo assim na difusão e utilização da metodologia.

2 OBJETIVOS

2.1 Geral

Este trabalho tem como objetivo principal a implementação de um pacote em R que utilize transcriptogramas em análises transcricionais.

2.2 Específicos

- Integrar funcionalidades de pacotes disponíveis no CRAN e Bioconductor;
- Permitir a execução em diferentes sistemas operacionais;
- Permitir que o usuário defina a quantidade de núcleos de processamento a ser utilizada na execução de algumas funções;
- Realizar análise topológica, expressão diferencial, enriquecimento funcional, e visualização de rede;
- Realizar análises transcricionais em dados de microarranjo e RNA-Seq;
- Garantir facilidades na distribuição e instalação;
- Gerar saídas de fácil interpretação.

CAPÍTULO I

Artigo: transcriptogramer: an R/Bioconductor package for transcriptional analysis based on protein-protein interaction

Escrito por: Diego Arthur de Azevedo Morais, Rita Maria Cunha de Almeida, Rodrigo Juliani Siqueira Dalmolin

Artigo submetido no periódico Bioinformatics

Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Applications Note

Systems Biology

transcriptogramer: an R/Bioconductor package for transcriptional analysis based on protein-protein interaction

Diego A. A. Morais[1], Rita M. C. Almeida[2] and Rodrigo J. S. Dalmolin[1,3,*]

[1]Bioinformatics Multidisciplinary Environment, Federal University of Rio Grande do Norte, Natal, RN, Brazil

[2]Institute of Physics and National Institute of Science and Technology: Complex Systems, Federal University of Rio Grande do Sul, Porto Alegre, RS, Brazil

[3]Department of Biochemistry, Federal University of Rio Grande do Norte, Natal, RN, Brazil

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Several freely available tools perform analysis using algorithms developed to identify significant variation of gene expression individually. The transcriptogramer R package uses protein-protein interaction to perform differential expression of functionally associated genes, assessing the expression profile of entire genetic systems and reveling which biological systems are significantly altered in case-control designed transcriptome experiments.

Results: The transcriptogramer is an open source R/Bioconductor package for transcriptional analysis that projects expression values on an ordered gene list to perform topological analysis, differential expression, and Gene Ontology enrichment analysis, independently of the data platform or operating system.

Availability: http://bioconductor.org/packages/transcriptogramer.

Contact: rodrigo.dalmolin@imd.ufrn.br

Supplementary information: Supplementary data are available at *Bioinformatics* online.

The available tools to assess gene expression, such as limma (Ritchie *et al.*, 2015), and DESeq2 (McCarthy *et al.*, 2014), were designed to evaluate individual gene expression. However, gene products are known to interact with each other to collectively perform biological functions and metabolic pathways (Li *et al.*, 2018).

The transcriptogram (Rybarczyk-Filho *et al.*, 2011), a systems biology-based method to analyze transcriptomes, uses protein-protein interaction (PPI) to build an ordered gene list. Briefly, the method order genes in one dimension, clustering the genes by the probability that their products interact to each other. In other words, interacting genes are expected to get closer to each other on the ordered gene list. The ordered gene list is then used to take the average expression of functionally associated genes in a given window with settable radius. This strategy reduces gene expression data noise and improves data measure reproducibility (da Silva *et al.*, 2014).

This Applications Note introduces a new R package for transcriptional analysis based on transcriptograms. The transcriptogramer R package is capable to perform the analysis on RNA-Seq and Microarray expression data by applying to transcriptograms the protocol adopted by the limma package, known to perform well and fast under many circumstances (Conesa *et al.*, 2016).

The transcriptogramer R package performs topological analysis, differential expression, and Gene Ontology (GO) enrichment analysis. The workflow initially requires an ordered gene list and an edge list representing the connection among the genes. The package contains ordered gene lists for 4 species (*Homo sapiens*, *Mus musculus*, *Saccharomyces cerevisiae*, and *Rattus norvegicus*). Edge lists representing the connection among the genes can be created from STRINGdb PPI.

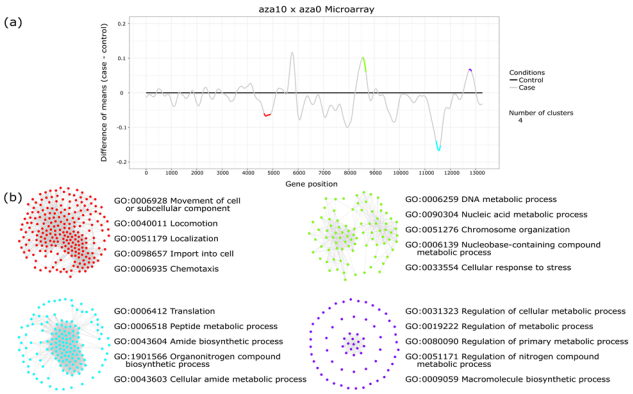


Fig. 1. (a) Differential Expression (DE) plot showing the difference of means between the two conditions. The two horizontal lines represent the average expression level of the conditions, black for the control samples, and gray for case. Each color over the gray line represents one DEGG (FDR < 0.02 for radius = 125). (b) DEGG PPI of the significant genes used as windows centers and its top 5 terms (FDR < 0.05) taking into account the Gene Ontology hierarchy. Network colors match those in DE plot. A reproducible example can be found at https://github.com/arthurvinx/CaseStudy1.

Using the initial inputs, it is possible to measure the degree and the clustering coefficient of individual genes as well as the same properties of functionally associated genes evaluated in a window segment of the ordered gene list. Average graph properties, such as the average assortativity and clustering coefficient, can also be calculated as function of node connectivity.

Transcriptograms are built in two steps that requires expression data (Microarray or RNA-seq), a dictionary (mapping proteins to identifiers used as expression data rownames), and the initial inputs (the ordered gene list and the edge list). The first step assigns the expression values (obtained from the expression data) to each respective gene on the ordered gene list. The second step assigns to each gene the average expression of the neighbor genes on the ordered gene list. The logic is to define a sliding window with a fixed radius (set by the user) and centered on the given gene. The sliding window considers periodic boundary conditions to deal with genes near the ends of the ordered list. The goal here is to measure the average expression of functionally associated genes, represented by neighbor genes on the ordered gene list.

Differential expression analysis evaluates the expression variations of functionally associated genes in case-control experiments. As a result, the package provides a data.frame and plots a figure showing differentially expressed gene groups (DEGG) (Figure 1 a). PPI among inferred DEGG can be displayed as graphs (Figure 1 b) and it is also possible to perform Gene Ontology enrichment analysis to identify the GO terms significantly associated with each DEGG (Figure 1 b). The complete transcriptogramer R package pipeline is shown in Supplementary Figure S1 and the main dependencies used by the package on the analysis are shown in Supplementary table S1.

The transcriptogramer package is implemented in R and is freely available at Bioconductor open source software for bioinformatics. The package is cross-platform, runs on Windows, Linux, and Mac OS X, and is multi-thread compatible, allowing users to easily define the number of cores to be used on some methods to reduce the runtime.

The transcriptogramer R package was applied to Microarray and RNA-Seq datasets of human colorectal adenocarcinoma cells HT-29 treated with two different concentrations of 5-aza-deoxy-cytidine (NCBI-BioProject, PRJNA177602) (Xu *et al.*, 2013). The RNA-Seq sequence read archives (SRAs) were processed following a count-based protocol (Anders *et al.*, 2013) and the log2-counts-per-million values were obtained using the limma voom() function (the complete list of tools used on the protocol is shown in Supplementary table S2 and the pipeline is shown in Supplementary Figure S2).

After the processing, filtering, and normalization (a multidimensional scaling plot of the RNA-Seq samples is shown in Supplementary Figure S3), the data were analyzed using the transcriptogramer package pipeline as well as another pipeline combining functions from limma and topGO (Alexa and Rahnenfuhrer, 2016) packages.

Samples (9 at total) were classified into 3 groups (control, 5-aza-deoxy-cytidine 5 µM, and 10 µM). Two case-control of differential expression and functional enrichment were performed on the Microarray and RNA-Seq data for each pipeline. All methodologies were able to identify similar number of differentially expressed genes. However, the transcriptogramer R package was able to identify nearly ten times more GO terms when comparing with classic differential expression analyses followed by topGO enrichment in both Microarray and RNA-Seq data. A detailed comparison can be found in Supplementary Tables S3-S8.

The transcriptogramer provides a systems biology-based pipeline for RNA-Seq and Microarray data transcriptional analysis designed to identify differential expression of functionally associated genes. The advantage of the transcriptogramer over existing functional analysis pipelines involves the use of canonical PPI data to define non-predefined gene sets, which enable the identification of gene clusters responsible for the enrichment of a given GO term. Functionally associated genes have inherent hierarchical properties, thus, users can increase the window radius to merge gene sets and explore the enrichment results of bigger systems. The transcriptogramer R/Bioconductor package is easy to install well documented tool, developed to help bioinformaticians and biologists to convert their gene expression data into biological insights.

Acknowledgements

The authors would like to thank the NPAD/UFRN.

Funding

This work was supported by the CNPq [grant 444856/2014–5] and the CAPES [grant 3381/2013].

Conflict of Interest: none declared.

References

Alexa, A. and Rahnenfuhrer, J. (2016). topGO: Enrichment Analysis for Gene Ontology. R package version 2.30.1.

Anders, S. *et al.* (2013). Count-based differential expression analysis of RNA sequencing data using R and Bioconductor. *Nature Protocols*, **8**(9), 1765-1786.

Conesa, A. *et al.* (2016). A survey of best practices for RNA-seq data analysis. *Genome Biology*.

da Silva, S. R. M. *et al.* (2014). Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. *BMC Genomics*.

Li, J. *et al.* (2018). Application of Weighted Gene Co-expression Network Analysis for Data from Paired Design. *Scientific Reports*.

Love, M. I. *et al.* (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*.

Ritchie, M. E. *et al.* (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research*.

Rybarczyk-Filho, J. L. *et al.* (2011). Towards a genome-wide transcriptogram: the *Saccharomyces cerevisiae* case. *Nucleic Acids Research*, **39**(8), 3005-3016.

Xu, X. *et al.* (2013). Parallel comparison of Illumina RNA-Seq and Affymetrix microarray platforms on transcriptomic profiles generated from 5-aza-deoxy-cytidine treated HT-29 colon cancer cells and simulated datasets. *BMC Bioinformatics*.

Supplementary Materials for

Transcriptogramer: an R/bioconductor package for transcriptional analysis based on canonical protein-protein interaction data

Diego A. A. Morais[1], Rita M. C. Almeida[2] and Rodrigo J. S. Dalmolin[1,3]

[1]Bioinformatics Multidisciplinary Environment, Federal University of Rio Grande do Norte, Natal, RN, Brazil

[2]Institute of Physics and National Institute of Science and Technology: Complex Systems, Federal University of Rio Grande do Sul, Porto Alegre, RS, Brazil

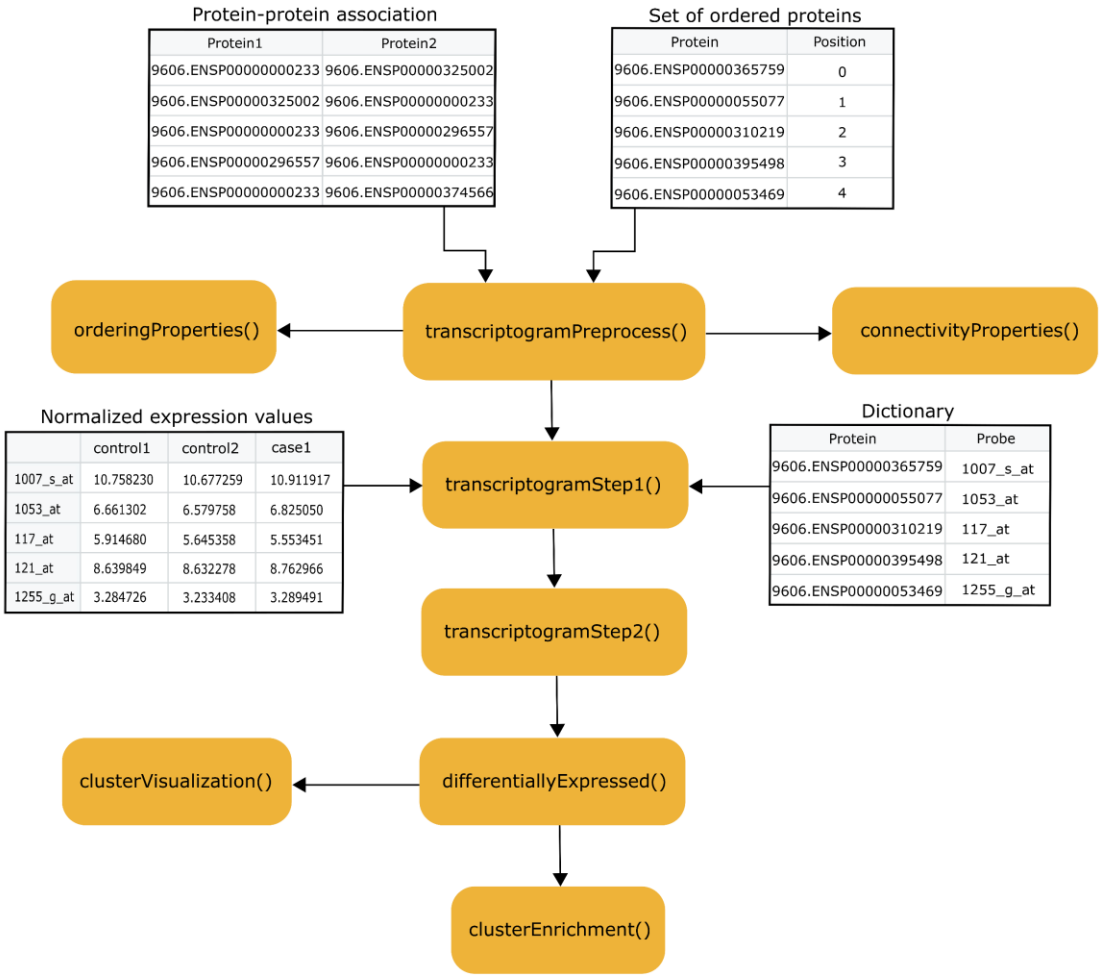
[3]Department of Biochemistry – CB, Federal University of Rio Grande do Norte, Natal, RN, Brazil

Contents

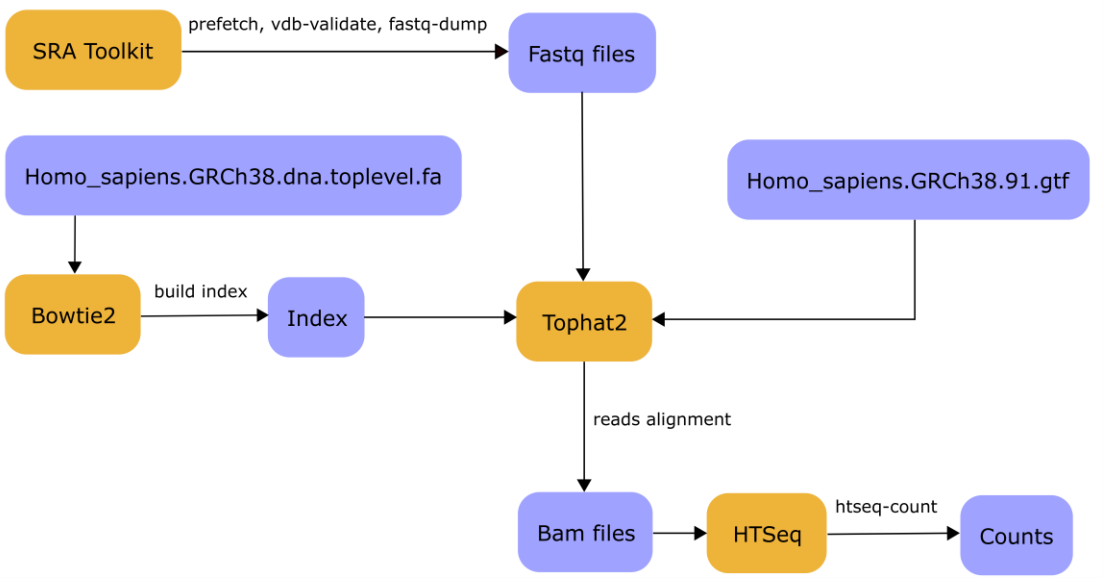
Supplementary Figures S1-S3 …… **1**

Supplementary Tables S1-S6 …… **4**

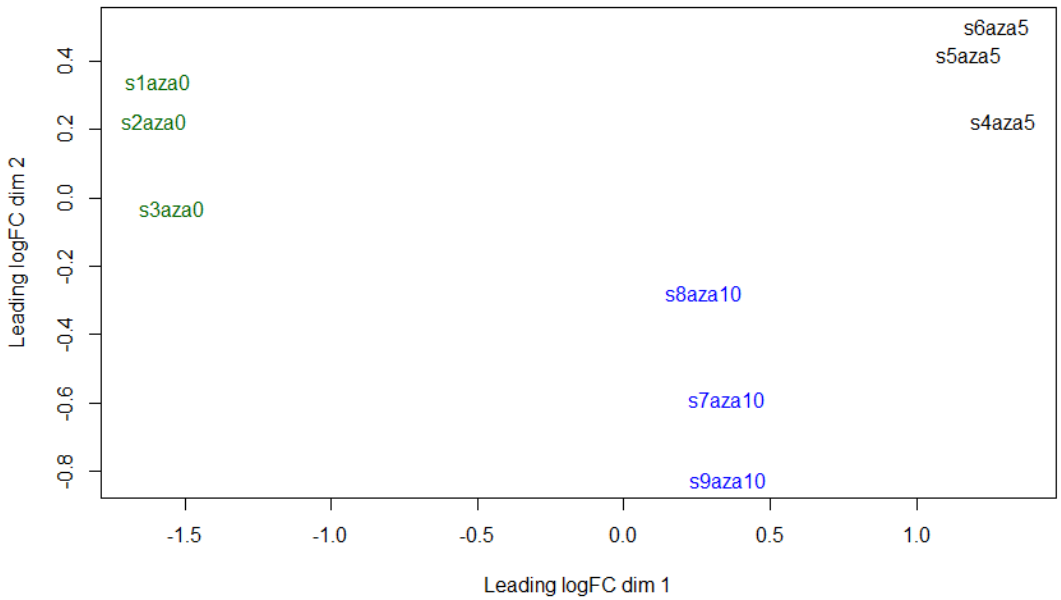
Supplementary Notes #1-2 …… **7**



Supplementary Figure S1. Analysis schematic workflow. Methods are represented as orange rounded squares and inputs are represented as squares. Despite the input data.frame like appearance, other input formats are accepted by the methods. The outputs are provided to the user as Transcriptogram objects, RedPort objects, figures, and data.frames.



Supplementary Figure S2. Sequence read archives processing schematic workflow. The tools are represented as orange rounded squares, and the inputs/outputs are represented as blue rounded squares. The raw count files obtained at the end of this workflow are then filtered and normalized to obtain the $\log_2$-counts-per-million used as expression values.



Supplementary Figure S3. RNA-Seq samples multidimensional scaling plot. Visual representation of the three groups, HT-29 cells treated with 0 µM (aza0), 5 µM (aza5), and 10 µM (aza10) of 5-aza-deoxy-cytidine, after the processing, filtering and trimmed mean of M values (TMM) normalization.

Supplementary Table S1. Package and main dependencies versions.

Tool	Version
transcriptogramer	1.1.21
limma	3.34.9
topGO	2.30.1

Supplementary Table S2. Count-based protocol software versions.

Software	Version
R	3.4.3
tophat2	2.1.1
bowtie2	2.3.4.1
HTSeq	0.9.1
SRA toolkit	2.8.2-1
edgeR	3.20.9

Supplementary Table S3. Differential expression results.

Description	Pipeline	Platform	aza5 x aza0	aza10 x aza0
Number of DEGs	limma-topGO	Microarray	6326	4139
Number of DEGs	limma-topGO	RNA-Seq	6188	4238
Number of DEGs	transcriptogramer	Microarray	6815	2274
Number of DEGs	transcriptogramer	RNA-Seq	6375	3164
Number of clusters	transcriptogramer	Microarray	15	11
Number of clusters	transcriptogramer	RNA-Seq	5	19

Supplementary Table S4. Gene Ontology enrichment analysis.

Description	Pipeline	Platform	aza5 x aza0	aza10 x aza0
Number of unique terms	limma-topGO	Microarray	247	142
Number of unique terms	limma-topGO	RNA-Seq	54	117
Number of unique terms	transcriptogramer	Microarray	3996	2460
Number of unique terms	transcriptogramer	RNA-Seq	2807	2801

Supplementary Table S5. Top 5 terms transcriptogramer Microarray.

	transcriptogramer Microarray			transcriptogramer RNA-Seq		limma-topGO Microarray		limma-topGO RNA-Seq	
Contrast	**Top terms**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**
aza5 x aza0	Chromosome organization	1	0.54	8	0.714	-	-	-	-
	DNA metabolic process	2	0.507	29	0.687	98	0.257	-	-
	Protein modification by small protein conjugation or removal	3	0.472	1	0.76	-	-	-	-
	DNA repair	4	0.65	39	0.761	-	-	-	-
	Cellular response to DNA damage stimulus	5	0.524	28	0.712	-	-	-	-
aza10 x aza0	Transmembrane receptor protein tyrosine kinase signalling pathway	1	0.301	139	0.157	-	-	-	-
	Enzyme linked receptor protein signalling pathway	2	0.22	140	0.123	-	-	-	-
	Endocytosis	3	0.245	141	0.157	-	-	-	-
	Import into cell	4	0.232	143	0.148	-	-	-	-
	Cell surface receptor signalling pathway	5	0.108	105	0.123	-	-	65	0.389

[1]Significant/Annotate

Supplementary Table S6. Top 5 terms transcriptogramer RNA-Seq.

	transcriptogramer RNA-Seq			transcriptogramer Microarray		limma-topGO Microarray		limma-topGO RNA-Seq	
Contrast	**Top terms**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**
aza5 x aza0	Protein modification by small protein conjugation or removal	1	0.76	3	0.472	-	-	-	-
	Cellular protein modification process	2	0.567	68	0.227	-	-	-	-
	Protein modification process	3	0.567	69	0.227	-	-	-	-
	Macromolecule modification	4	0.551	30	0.228	-	-	-	-
	Cellular response to stress	5	0.647	12	0.339	40	0.607	-	-
aza10 x aza0	Chromosome organization	1	0.431	76	0.153	-	-	-	-
	Chromatin organization	2	0.513	106	0.12	-	-	-	-
	DNA metabolic process	3	0.418	74	0.2	45	0.167	-	-
	DNA repair	4	0.55	73	0.31	-	-	-	-
	Nucleic acid metabolic process	5	0.194	78	0.062	63	0.146	-	-

[1]Significant/Annotated

Supplementary Table S7. Top 5 terms limma-topGO Microarray.

	limma-topGO Microarray			limma-topGO RNA-Seq		transcriptogramer Microarray		transcriptogramer RNA-Seq	
Contrast	**Top terms**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**
aza5 x aza0	Oxidation-reduction process	1	0.292	-	-	338	0.156	925	0.346
	Cellular metabolic process	2	0.235	-	-	26	0.169	866	0.174
	Small molecule metabolic process	3	0.26	-	-	341	0.108	384	0.348
	Small molecule biosynthetic process	4	0.297	-	-	1406	0.117	1760	0.339
	Metabolic process	5	0.233	-	-	62	0.162	1008	0.172
aza10 x aza0	Cell cycle	1	0.173	-	-	95	0.085	193	0.184
	Regulation of cell cycle	2	0.181	-	-	412	0.074	754	0.162
	Cellular metabolic process	3	0.144	-	-	268	0.032	86	0.117
	Cell cycle process	4	0.176	-	-	99	0.093	274	0.19
	Metabolic process	5	0.143	-	-	314	0.031	168	0.114

[1]Significant/Annotated

Supplementary Table S8. Top 5 terms limma-topGO RNA-Seq.

	limma-topGO RNA-Seq			limma-topGO Microarray		transcriptogramer Microarray		transcriptogramer RNA-Seq	
Contrast	**Top terms**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**	**Position**	**Rate[1]**
aza5 x aza0	SRP-dependent cotranslational protein targeting to membrane	1	0.876	-	-	229	0.977	187	0.977
	Cotranslational protei targeting to membrane	2	0.851	-	-	223	0.978	184	0.978
	Protein targeting to ER	3	0.814	-	-	228	0.948	188	0.946
	Establishment of protein localization to endoplasmic reticulum	4	0.8	-	-	233	0.91	191	0.917
	Nuclear-transcribed mRNA catabolic process, nonsense-mediated decay	5	0.778	-	-	234	0.857	192	0.845
aza10 x aza0	Cotranslational protein targeting to membrane	1	0.755	67	0.229	130	0.505	-	-
	SRP-dependent cotranslational protein targeting to membrane	2	0.764	-	-	133	0.489	-	-
	Protein targeting to ER	3	0.711	122	0.215	132	0.469	-	-
	Protein targeting to membrane	4	0.645	48	0.215	137	0.324	-	-
	Establishment of protein localization to endoplasmic reticulum	5	0.7	-	-	134	0.45	-	-

[1]Significant/Annotated

Supplementary Note #1: transcriptogramer and limma-topGO pipelines

Transcriptograms are produced by assigning to each gene the average expression level of its neighbors, delimited according to a window of a given radius. Therefore, the radius = 125 was chosen for all experiments using the transcriptogramer package. On the transcriptogramer package pipeline, were used only the STRINGdb 10.5 protein-protein interaction (PPI) of combined score greater than or equal to 800.

The number of differentially expressed genes (FDR < 0.05) and unique Gene Ontology biological processes terms (FDR < 0.05) obtained on each experiment were annotated. A detailed comparison can be found in Supplementary Tables S3 and S4. Each DEG counted on the transcriptogramer pipeline represents one window central gene, whose expression level denotes the average expression level of its neighbors within the window. A group of these DEGs, separated by less gene positions than the radius value, represents a differentially expressed gene group (DEGG).

The Gene Ontology enrichment analysis for both pipelines were performed using the topGO classic algorithm, not taking into account Gene Ontology hierarchy. The transcriptogramer pipeline tested the enrichment of each DEGG and the limma-topGO pipeline tested all the DEGs.

Supplementary Note #2: Pipeline results comparison

The number of differentially expressed genes are close when comparing limma-topGO and transcriptogramer, including Microarray and RNA-Seq results (Supplementary table S3). For the Gene Ontology enrichment analysis, the number of unique terms using the transcriptogramer pipeline was more than 10 times higher than the results obtained by the limma-topGO pipeline, indicating that transcriptogramer is more sensitive in finding alterations in cellular systems. Supplementary table S4 to S8 shows that the transcriptogramer is able to identify almost all GO terms identified when using limma followed by topGO for both Microarray and RNA-Seq. The supplementary tables shows that the transcriptogramer pipeline GO enrichment analysis identifies more terms. Some of these terms appears in the limma-topGO results, but many are unique to the transcriptogramer pipeline GO enrichment results.

3 DISCUSSÃO

3.1 Transcriptogramer

Na análise topológica, a implementação em R calcula as mesmas métricas que a ferramenta desenvolvida em C e Python. Entretanto, as estatísticas utilizadas na expressão diferencial dos grupos gênicos e no enriquecimento funcional das duas implementações diferem. Assim, estas análises apresentam resultados distintos. O pacote em R foi desenvolvido de forma modular, viabilizando assim a sua utilização em *pipelines* já existentes, e possui uma interface de linha de comandos, facilitando assim a parametrização e automatização de *scripts*.

A implementação em R foi submetida ao repositório Bioconductor, sendo aceita em 27 de outubro de 2017. O pacote transcriptogramer foi disponibilizado ao público no lançamento de outubro da versão 3.6 do Bionconductor, ocorrido no dia 31. Como pode ser observado na Figura 2, no primeiro semestre de 2018 o pacote foi baixado 661 vezes por 376 endereços distintos. Não existem dados disponíveis da quantidade de vezes que a versão em C e Python foi baixada, assim, não é possível realizar uma comparação a fim de avaliar o impacto da nova forma de distribuição. Porém, o fato de uma ferramenta, ainda não descrita em publicações, ter sido baixada mais de 500 vezes em 5 meses representa a visibilidade proporcionada pela nova forma de distribuição.

Month	Nb of distinct IPs	Nb of downloads
Jan/2018	49	68
Feb/2018	58	77
Mar/2018	54	78
Apr/2018	113	196
May/2018	64	111
Jun/2018	87	131
Jul/2018	0	0
Aug/2018	0	0
Sep/2018	0	0
Oct/2018	0	0
Nov/2018	0	0
Dec/2018	0	0
2018	**376**	**661**

Figura 2. Número de *downloads* por mês durante o primeiro semestre de 2018.

O pacote transcriptogramer provê listas ordenadas de genes para 4 organismos (*Homo sapiens, Mus musculus, Saccharomyces cerevisiae and Rattus norvegicus*), criadas a partir de dados de PPI do STRING versão 10.5. Uma lista de genes personalizada também pode ser fornecida pelo usuário, desde que uma lista de adjacência entre estes genes também seja fornecida. A possibilidade da utilização de listas personalizadas viabiliza análises de organismos com pouca anotação disponível.

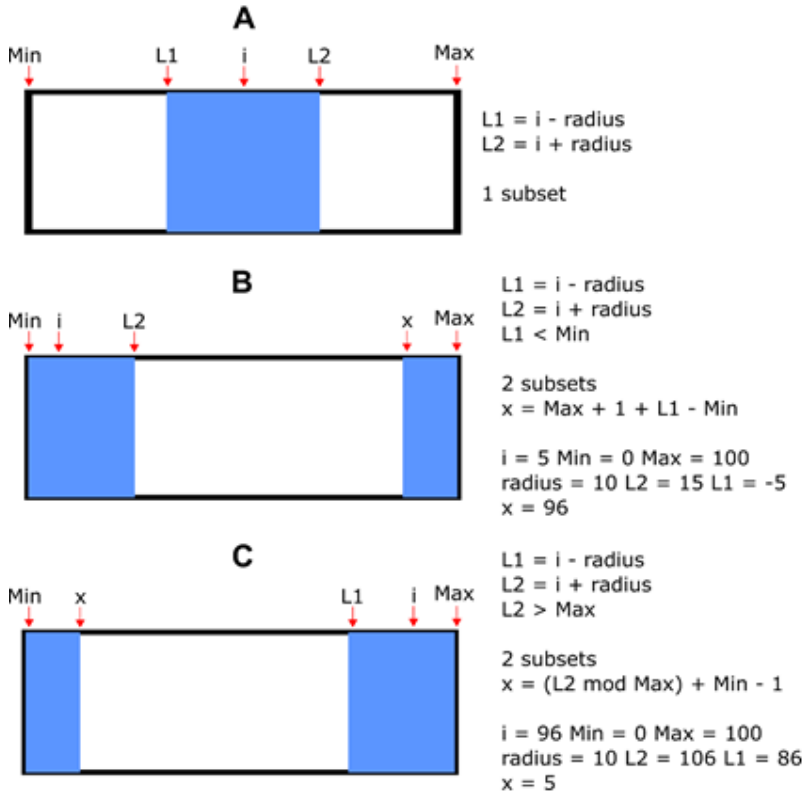


Figura 3. Condições levadas em consideração pela janela. A lista ordenada de genes é representada pelo retângulo branco e a janela é representada pela área azul. Min e Max representam as posições extremas da lista ordenada de genes. L1 e L2 representam as posições extremas da janela. (A) Os extremos da janela não ultrapassam os extremos da lista ordenada de genes, assim, uma região sequencial é selecionada. (B e C) Um dos extremos da janela ultrapassa um dos extremos da lista ordenada de genes e precisa ser recalculado, nestes casos são selecionadas duas áreas.

O uso da janela deslizante é recorrente em certas funcionalidades do pacote, sendo assim, um algoritmo paralelizável capaz de utilizar aritmética modular, para levar em conta condições de contorno periódico, foi desenvolvido (Figura 3). A linguagem R provê paralelismo por meio de diferentes métodos, entretanto, sistemas operacionais baseados em Windows suportam apenas o uso de *sockets*. Assim, o algoritmo desenvolvido utiliza um *parallel socket cluster* para garantir que o

paralelismo funcione na maioria dos sistemas operacionais existentes. Todas as funcionalidades que suportam paralelismo possuem um argumento para a definição do número de núcleos de processamento a serem utilizados.

Funções que realizam análise topológica utilizam o pacote igraph (CSÁRDI e NEPUSZ, 2006), implementado em R, C e C++, e fórmulas matemáticas para calcular propriedades de grafos. A utilização de linguagens como C e C++ reduz o tempo computacional necessário para a realização de cálculos. O de fórmulas matemáticas, como a do cálculo do coeficiente de clusterização de um nó, evita que redes densamente conectadas sejam percorridas, contribuindo também para a redução do tempo necessário na conclusão de algumas tarefas.

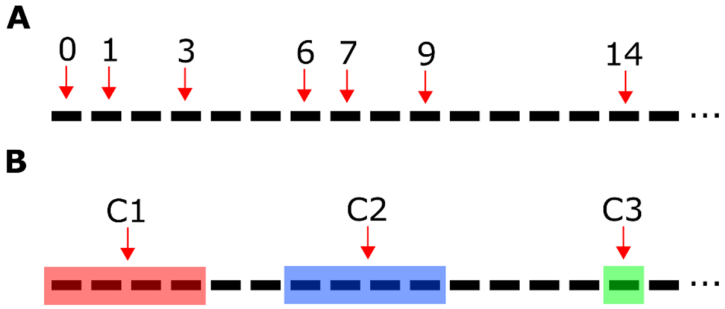


Figura 4. Identificação dos grupos gênicos alterados. (A) Posições dos centros das janelas cuja variação da expressão foi significativa entre as condições. (B) Delimitação das extremidades de cada grupo funcionalmente associado para o raio 2. A utilização do valor 3 como raio mesclaria os grupos C1 e C2.

A expressão diferencial é realizada segundo o protocolo do pacote limma, capaz de aceitar tanto dados provenientes de microarranjos quanto de RNA-Seq. O pacote limma, bastante utilizado pela comunidade científica, já foi comparado com outros pacotes que utilizam outros métodos para o mesmo fim. Resultados demonstram que o limma demanda menos tempo para a realização de tarefas, além de executar bem sob diversas circunstâncias (CONESA *et al.*, 2016). O transcriptogramer utiliza funções do pacote limma na avaliação de valores de expressão. O pacote limma foi projetado para avaliar a expressão individual de genes, porém, em conjunto com a metodologia do transcriptograma, que atribui a média de expressão de uma janela de genes funcionalmente associados a uma posição da lista ordenada de genes, a avaliação passa a ser sobre a expressão de grupos gênicos. Com a identificação destas posições, pontos centrais das janelas, é executado um algoritmo que leva em consideração a distância entre estes pontos para identificar as posições extremas que delimitam cada grupo. Caso a distância entre dois pontos seja

menor do que o raio utilizado para obter as médias de expressão, estes genes são agrupados no mesmo grupo. Este método de expressão diferencial identifica grupos gênicos, de tamanho variável, que se encontram alterados em duas condições distintas. Um exemplo deste processo pode ser visto na Figura 4.

O enriquecimento de termos relacionados a cada um dos grupos identificados é realizado por meio de chamadas a funções do pacote topGO (ALEXA e RAHNENFUHRER, 2016), que utiliza dados do *Gene Ontology* (GO) (ASHBURNER *et al*., 2000), possibilitando o enriquecimento de processos biológicos, componente celular, e função molecular. Na comparação de termos identificados com o transcriptogramer e termos identificados por um *pipeline* que combinava funções do limma e topGO, foi observado que o número de termos identificados pelo transcriptogramer foi aproximadamente 10 vezes maior. Além disto, a grande maioria dos termos identificados pelo *pipeline* limma-topGO também foi detectada pelo transcriptogramer. Análises de enriquecimento comparam um determinado grupo de genes com um grupo maior, universo de genes, sendo o grupo de genes alvo da análise composto por genes diferencialmente expressos, que devem estar contidos no universo de genes. Ferramentas que avaliam a expressão de genes individualmente não distinguem grupos funcionalmente associados, assim, todos os genes diferencialmente expressos são analisados contra o universo de genes. A capacidade do transcriptogramer de definir grupos gênicos funcionalmente associados, a serem analisados isoladamente contra o universo de genes, é capaz de identificar termos negligenciados em análises que utilizam outras metodologias.

Novas funções de visualização podem ser implementadas em trabalhos futuros. O pacote transcriptogramer é especialmente útil quando aplicado a dados produzidos por experimentos de série temporal, permitindo a visualização da resposta sistêmica de uma determinada perturbação. Portanto, um novo método de visualização poderia explorar a hierarquia de termos do GO relacionada a perturbações no metabolismo celular. Novos métodos também podem ser implementados com o objetivo de automatizar a escolha do raio ideal a ser utilizado na janela, melhorando o agrupamento dos grupos gênicos. Futuras versões do pacote serão disponibilizadas nos lançamentos semestrais do Bioconductor.

4 CONCLUSÃO

A implementação em R encontra-se disponível no Bioconductor, um repositório que proporciona visibilidade, e facilidades na distribuição e instalação da ferramenta. O pacote transcriptogramer provê um *pipeline* completo baseado em biologia de sistemas para análise transcricional, envolvendo análise topológica, expressão diferencial e análise de enriquecimento de termos do GO. A utilização de PPI possibilita a definição de grupos gênicos funcionalmente associados não pré-definidos, relacionados a termos do GO. Termos do GO possuem propriedades hierárquicas, assim, usuários podem aumentar o tamanho do raio da janela para explorar o resultado do enriquecimento de sistemas maiores. O pacote transcriptogramer é uma ferramenta fácil de usar, com o potencial de ajudar biólogos e bioinformatas a converter dados em novos conhecimentos biológicos.

REFERÊNCIAS

ALEXA, A.; RAHNENFUHRER, J. topGO. Pacote R/Bioconductor, 2016.

ANDERS, S. *et al*. Count-based differential expression analysis of RNA sequencing data using R and Bioconductor. **Nature Protocols**, 2013.

ASHBURNER, M. *et al*. Gene ontology: tool for the unification of biology. **Nature Genetics**, 2000.

BARRETT, T. *et al*. NCBI GEO: Archive for functional genomics data sets - Update. **Nucleic Acids Research**, 2013.

BLOOMFIELD V. A. Using R for Computer Simulation and Data Analysis in Biochemistry, Molecular Biology, and Biophysics. **Computational Science – ICCS**, 2009.

CONESA, A. *et al*. A survey of best practices for RNA-seq data analysis. **Genome Biology**, 2016.

CSÁRDI, G.; NEPUSZ, T. The igraph software package for complex network research. **InterJournal Complex Systems**, 2006.

DA SILVA, S. R. M. *et al*. Reproducibility enhancement and differential ex-pression of non predefined functional gene sets in human genome. **BMC Genomics**, 2014.

DE ALMEIDA, R. M. C. *et al*. Transcriptome analysis reveals manifold mechanisms of cyst development in ADPKD. **Human Genomics**, 2016.

D'ARGENIO, V. The High-Throughput Analyses Era: Are We Ready for the Data Struggle? **High-Throughput**, 2018.

EDGAR, R. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. **Nucleic Acids Research**, 2002.

FERRAREZE, P. A. G. *et al*. Transcriptional Analysis Allows Genome Reannotation and Reveals that Cryptococcus gattii VGII Undergoes Nutrient Restriction during Infection. **Microorganisms**, 2017.

FRANCESCHINI, A. *et al*. STRING v9.1: Protein-protein interaction networks, with increased coverage and integration. **Nucleic Acids Research**, 2013.

GENTLEMAN, R. *et al*. Bioconductor: open software development for computational biology and bioinformatics. **Genome Biology**, 2004.

KUENTZER, FELIPE A. *et al*. Optimization and analysis of seriation algorithm for ordering protein networks. **IEEE International Conference on Bioinformatics and Bioengineering (BIBE)**, 2014.

LI, J. *et al*. Application of Weighted Gene Co-expression Network Analysis for Data from Paired Design. **Scientific Reports**, 2018.

LOVE, M. I.; HUBER, W.; ANDERS, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. **Genome Biology**, 2014.

MCCARTHY, D. J.; CHEN, Y.; SMYTH, G. K. Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. **Nucleic Acids Research**, 2012.

PENG, R. D. R Programming for Data Science. Livro. 2018.

PERTEA, M. *et al*. Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown. **Nature Protocols**, 2016.

RITCHIE, M. E. *et al*. limma powers differential expression analyses for RNA-sequencing and microarray studies. **Nucleic Acids Research**, 2015.

RYBARCZYK-FILHO, J. L. *et al*. Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. **Nucleic Acids Research**, 2011.

ROBINSON, M. D.; Mccarthy, D. J.; SMYTH, G. K. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. **Bioinformatics**, 2010.

SAURO, H. M. *et al*. Challenges for modeling and simulation methods in systems biology. **Winter Simulation Conference**, 2006.

XAVIER, L. A. DA C. *et al*. Analysis of genome instability biomarkers in children with non-syndromic orofacial clefts. **Mutagenesis**, 2017.

XU, X. *et al*. Parallel comparison of Illumina RNA-Seq and Affymetrix microarray platforms on transcriptomic profiles generated from 5-aza-deoxy-cytidine treated HT-29 colon cancer cells and simulated datasets. **BMC Bioinformatics**, 2013.

ZHAO, S. *et al*. Comparison of RNA-Seq and microarray in transcriptome profiling of activated T cells. **PLoS ONE**, 2014.

ANEXOS

The transcriptogramer user's guide

Version: 1.3.5

Overview

The transcriptogramer package is designed for transcriptional analysis based on transcriptograms, a method to analyze transcriptomes that projects expression values on a set of ordered proteins, arranged such that the probability that gene products participate in the same metabolic pathway exponentially decreases with the increase of the distance between two proteins of the ordering. Transcriptograms are, hence, genome wide gene expression profiles that provide a global view for the cellular metabolism, while indicating gene sets whose expression are altered (da Silva (2014); Rybarczyk-Filho (2011); de Almeida (2016); Ferrareze (2017)).

Methods are provided to analyze topological properties of an interactome, to generate transcriptograms, to detect and to display differentially expressed gene clusters, and to perform a Gene Ontology enrichment analysis on these clusters.

As a set of ordered proteins is required in order to run the methods, datasets are available for four species (*Homo sapiens*, *Mus musculus*, *Saccharomyces cerevisiae* and *Rattus norvegicus*). Each species has three datasets, originated from STRINGdb release 10.5 protein network data, with combined scores greater than or equal to 700, 800 and 900 (see **Hs900**, **Hs800**, **Hs700**, **Mm900**, **Mm800**, **Mm700**, **Sc900**, **Sc800**, **Sc700**, **Rn900**, **Rn800** and **Rn700** datasets). Custom sets of ordered proteins can be generated from protein network data using The transcriptogramer on Windows.

Quick start

The first step is to create a Transcriptogram object by running the `transcriptogramPreprocess()` function. This example uses a subset of the *Homo sapiens* protein network data, from STRINGdb release 10.5, containing only associations of proteins of combined score greater than or equal to 900 (see **Hs900** and **association** datasets).

```
library(transcriptogramer)
t <- transcriptogramPreprocess(association = association, ordering = Hs900)
```

Topological analysis

There are two methods to perform topological analysis, `connectivityProperties()` calculates average graph properties as function of node connectivity, and `orderingProperties()` calculates graph properties projected on the ordered proteins. Some methods, such as orderingProperties(), uses a window, region of n (radius * 2 + 1) proteins centered at a protein, whose radius changes the output. The Transcriptogram object has a **radius** slot that can be setted during, or after, its preprocessing (see **Transcriptogram-class** documentation).

```
## during the preprocessing

## creating the object and setting the radius as 0
t <- transcriptogramPreprocess(association = association, ordering = Hs900)

## creating the object and setting the radius as 50
t <- transcriptogramPreprocess(association = association, ordering = Hs900,
                               radius = 50)
```

```
## after the preprocessing

## modifying the radius of an existing Transcriptogram object
radius(object = t) <- 25

## getting the radius of an existing Transcriptogram object
r <- radius(object = t)
```

The output of the orderingProperties() method is partially affected by the radius slot.

```
oPropertiesR25 <- orderingProperties(object = t, nCores = 1)

## slight change of radius
radius(object = t) <- 30

## this output is partially different comparing to oPropertiesR25
oPropertiesR30 <- orderingProperties(object = t, nCores = 1)
```

As the connectivityProperties() method does not uses a window, its output is not affected by the radius slot.

```
cProperties <- connectivityProperties(object = t)
```

Transcriptogram

A transcriptogram is generated in two steps and requires expression values, from microarray or RNA-Seq assays (log2-counts-per-million), and a dictionary. This example uses the datasets **GSE9988**, which contains normalized expression values of 3 cases and 3 controls (GSM252443, GSM252444, GSM252445, GSM252465, GSM252466 and GSM252467 respectively), and **GPL570**, a mapping between ENSEMBL Peptide ID and Affymetrix Human Genome U133 Plus 2.0 Array probe identifier.

The methods to generate a transcriptogram are `transcriptogramStep1()` and `transcriptogramStep2()`. The transcriptogramStep1() assigns to each protein, of each transcriptome sample, the average of the expression values of all the identifiers related to it.

```
t <- transcriptogramStep1(object = t, expression = GSE9988,
                          dictionary = GPL570, nCores = 1)
```

To each position of the ordering, the transcriptogramStep2() method assigns a value equal to the average of the expression values inside a window, which considers periodic boundary conditions to deal with proteins near the ends of the ordering, in order to reduce random noise.

```
t <- transcriptogramStep2(object = t, nCores = 1)
```

The Transcriptogram object has slots to store the outputs of the transcriptogramStep1() and transcriptogramStep2() methods, called transcriptogramS1 and transcriptogramS2 respectively. As the output of some methods are affected by the content of the transcriptogramS2 slot, it can be recalculated using the content of the transcriptogramS1 slot.

```
radius(object = t) <- 80
t <- transcriptogramStep2(object = t, nCores = 1)
```

Gene Ontology enrichment analysis

As nearby genes of a transcriptogram have a high probability to interact with each other, gene sets whose expression are altered can be identified using the *limma* package. The `differentiallyExpressed()` method

uses the limma package to identify differentially expressed genes (the approaches voom and trend are supported for RNA-Seq), for the contrast "case-control", grouping as a cluster a set of genes which positions are within a radius range specified by the content of the radius slot.

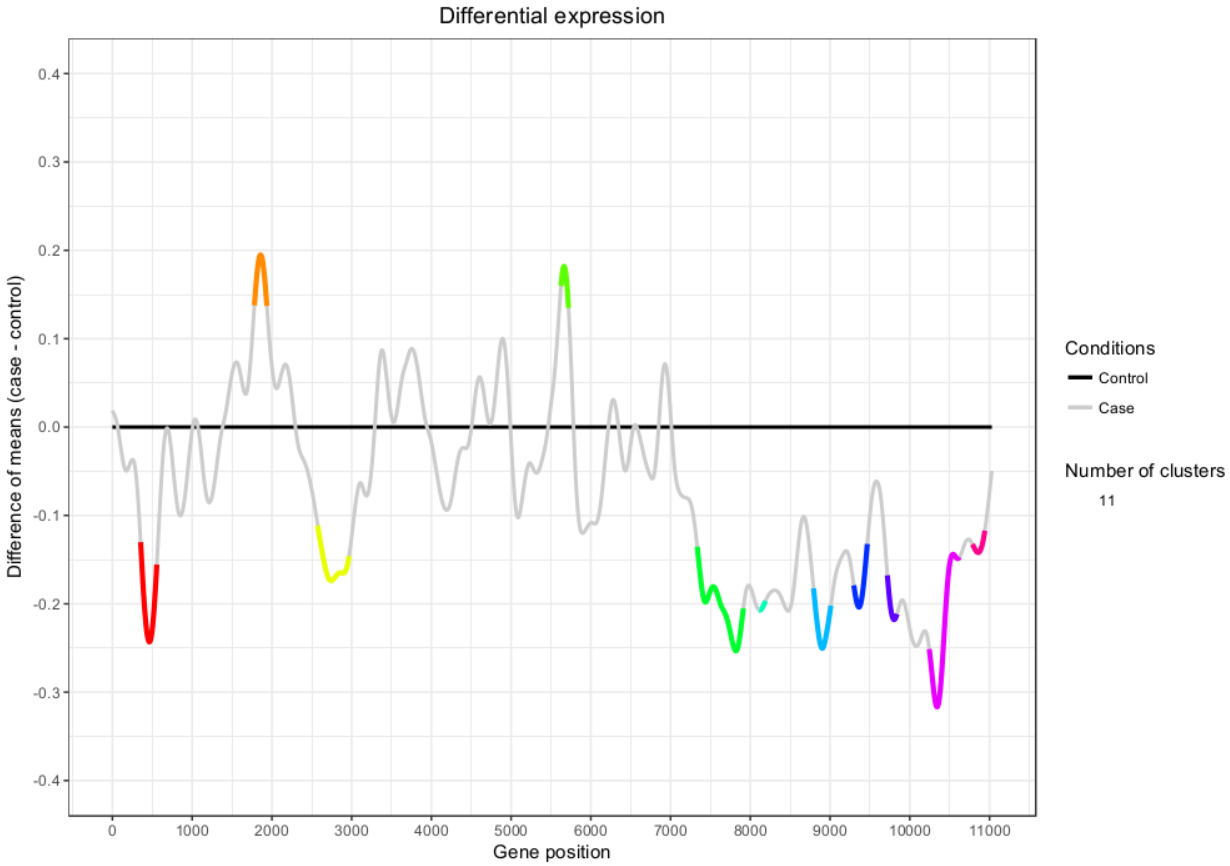
For this example, the p-value threshold for false discovery rate will be set as 0.01. If a species name is provided, the *biomaRt* package is used to translate the ENSEMBL Peptide ID to Symbol (Gene Name), alternatively, a data.frame with such content can be provided. The levels argument classify the columns of the transcriptogramS2 slot referring to samples, as there are 6 columns (see dataset **GSE9988**), is created a logical vector that uses TRUE to label the columns referring to controls samples, and FALSE to label the columns referring to case samples.

```
## trend = FALSE for microarray data or voom log2-counts-per-million
## the default value for trend is FALSE
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
t <- differentiallyExpressed(object = t, levels = levels, pValue = 0.01,
                             trend = FALSE)
```

```
## translating ENSEMBL Peptide IDs to Symbols using the biomaRt package
## Internet connection is required for this command
t <- differentiallyExpressed(object = t, levels = levels, pValue = 0.01,
                             species = "Homo sapiens")

## translating ENSEMBL Peptide IDs to Symbols using the DEsymbols dataset
t <- differentiallyExpressed(object = t, levels = levels, pValue = 0.01,
                             species = DEsymbols)
```

This method also produces a plot referring to its output. In this case, eleven clusters were detected, and each one is represented by a color. The genes that are above the horizontal black line are upregulated, and the genes that are below are downregulated.

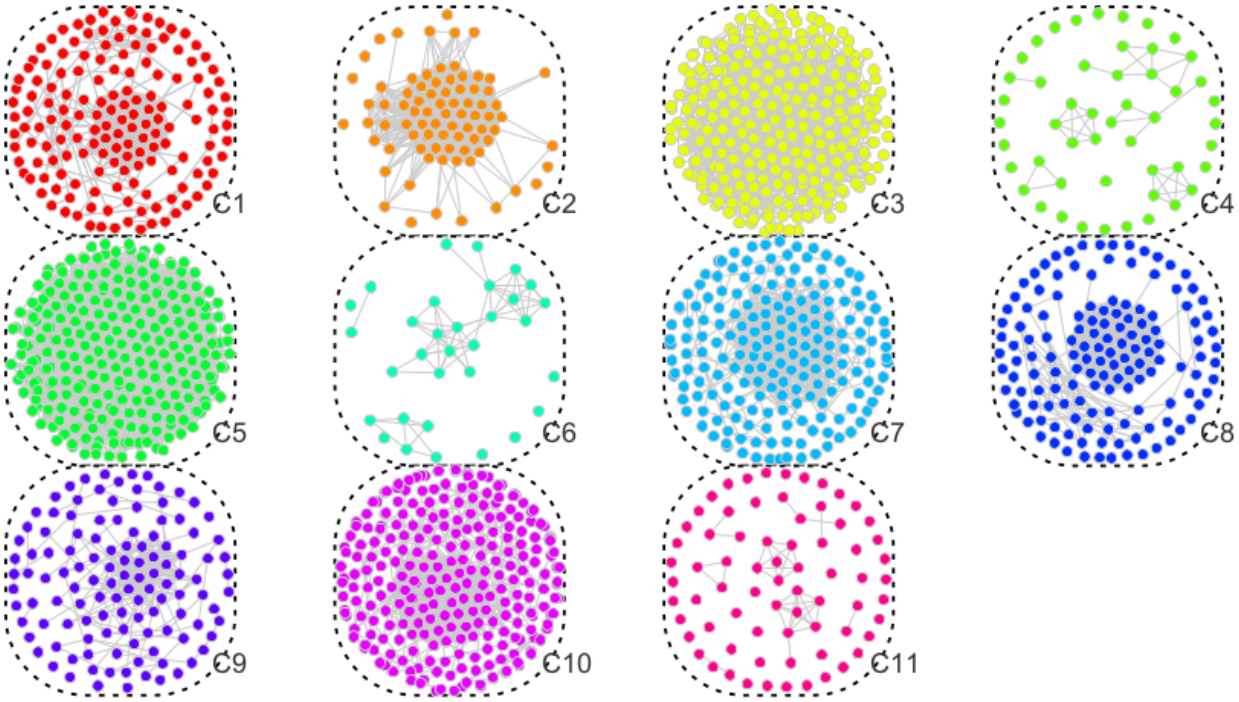


The differentially expressed genes identified by this method are stored in the DE slot of the Transcriptogram object, its content can be obtained using the DE method. By default, the p-values are adjusted by the Benjamini-Hochberg procedure.

```
DE <- DE(object = t)
```

The `clusterVisualization()` method uses the *RedeR* package to display graphs of the differentially expressed clusters and returns an object of the RedPort Class, allowing interactions through functions of the RedeR package. This method requires the Java Runtime Environment (>= 6).

```
rdp <- clusterVisualization(object = t)
```



The `clusterEnrichment()` method performs a Gene Ontology enrichment analysis using the *topGO* package. By default, the universe is composed by all the proteins present in the ordering slot, the ontology is setted to biological process, the algorithm is setted to classic, the statistic is setted to fisher, and the p-values are adjusted by the Benjamini-Hochberg procedure. For this example, the p-value threshold for false discovery rate will be set as 0.005. This method uses the biomaRt package to build a Protein2GO data.frame using the given species name, or data.frame.

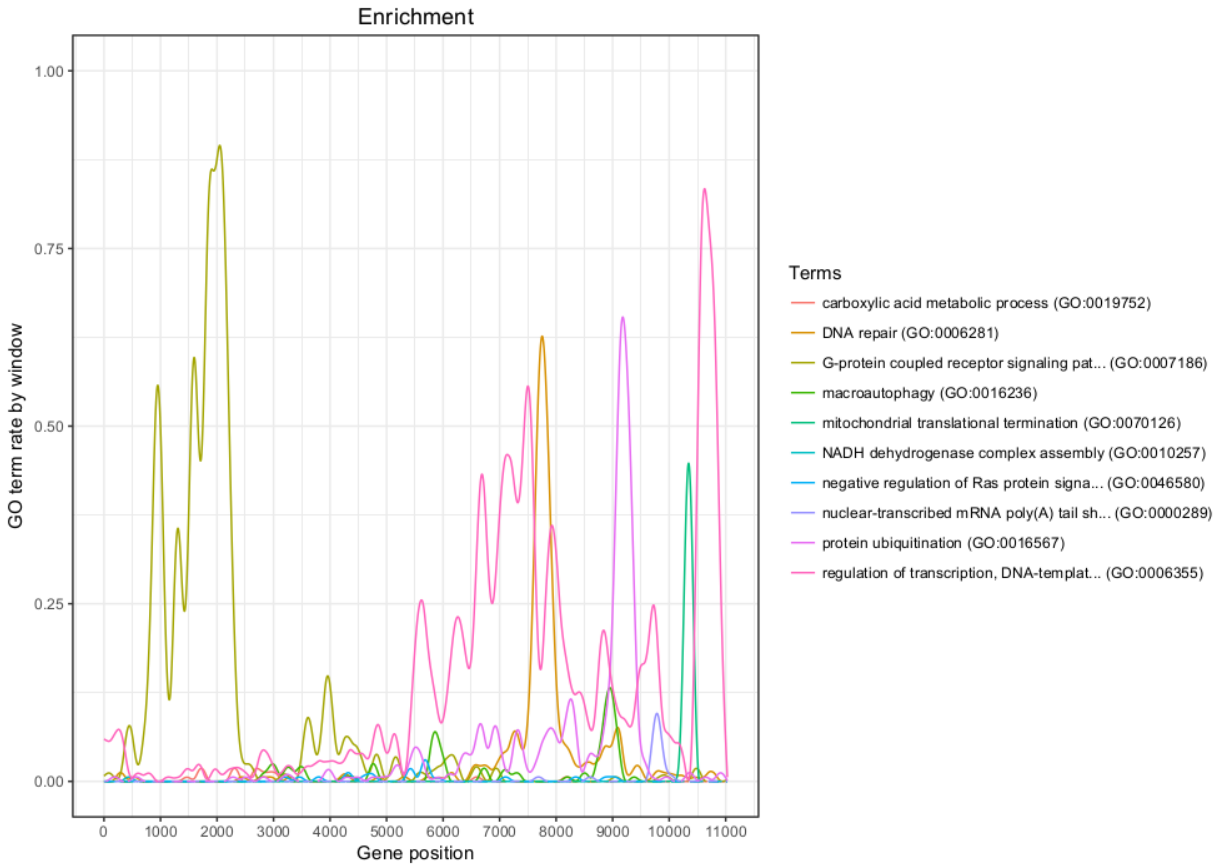
```
## using the HsBPTerms dataset to create the Protein2GO data.frame
t <- clusterEnrichment(object = t, species = HsBPTerms,
                           pValue = 0.005, nCores = 1)
```

```
## using the biomaRt package to create the Protein2GO data.frame
## Internet connection is required for this command
t <- clusterEnrichment(object = t, species = "Homo sapiens",
                           pValue = 0.005, nCores = 1)
```

```
head(Terms(t))
```

```
##        GO.ID                                          Term Annotated
## 1 GO:0010257         NADH dehydrogenase complex assembly        57
## 2 GO:0032981 mitochondrial respiratory chain complex ...        57
## 3 GO:0033108 mitochondrial respiratory chain complex ...        72
## 4 GO:0006120 mitochondrial electron transport, NADH t...        51
## 5 GO:0042775 mitochondrial ATP synthesis coupled elec...        89
## 6 GO:0022904        respiratory electron transport chain       104
##   Significant Expected       pValue ClusterNumber
## 1          52     0.95 6.652273e-28             1
## 2          52     0.95 6.652273e-28             1
## 3          52     1.20 6.652273e-28             1
## 4          42     0.85 6.652273e-28             1
## 5          43     1.48 6.652273e-28             1
## 6          45     1.73 6.652273e-28             1
```

```
enrichmentPlot(t)
```



References

da Silva, S. R. M. et al. (2014). "Reproducibility Enhancement and Differential Expression of Non Predefined Functional Gene Sets in Human Genome." *BMC Genomics*. doi:10.1186/1471-2164-15-1181.

de Almeida, R. M.C. et al. (2016). "Transcriptome Analysis Reveals Manifold Mechanisms of Cyst Development in Adpkd." *Human Genomics* 10 (1): 1–24. doi:10.1186/s40246-016-0095-x.

Ferrareze, P. A. G. et al. (2017). "Transcriptional Analysis Allows Genome Reannotation and Reveals That Cryptococcus Gattii Vgii Undergoes Nutrient Restriction During Infection." *Microorganisms* 5 (3). doi:10.3390/microorganisms5030049.

Rybarczyk-Filho, J. L. et al. (2011). "Towards a Genome-Wide Transcriptogram: The Saccharomyces Cerevisiae Case." *Nucleic Acids Research* 39 (8): 3005–16. doi:10.1093/nar/gkq1269.

Package 'transcriptogramer'

June 1, 2018

Type Package

Title Transcriptional analysis based on transcriptograms

Version 1.3.5

Date 2018-05-31

Description R package for transcriptional analysis based on transcriptograms, a method to analyze transcriptomes that projects expression values on a set of ordered proteins, arranged such that the probability that gene products participate in the same metabolic pathway exponentially decreases with the increase of the distance between two proteins of the ordering. Transcriptograms are, hence, genome wide gene expression profiles that provide a global view for the cellular metabolism, while indicating gene sets whose expression are altered.

Depends R (>= 3.4), methods

License GPL (>= 2)

Encoding UTF-8

LazyData true

biocViews Software, Network, Visualization, SystemsBiology, GeneExpression, GeneSetEnrichment, GraphAndNetwork, Clustering, DifferentialExpression, Microarray, RNASeq, Transcription

Imports biomaRt, data.table, doSNOW, foreach, ggplot2, graphics, grDevices, igraph, limma, parallel, progress, RedeR, snow, stats, tidyr, topGO

RoxygenNote 6.0.1

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, RUnit, BiocGenerics

SystemRequirements Java Runtime Environment (>= 6)

URL https://github.com/arthurvinx/transcriptogramer

BugReports https://github.com/arthurvinx/transcriptogramer/issues

Author Diego Morais [aut, cre], Rodrigo Dalmolin [aut]

Maintainer Diego Morais <vinx@ufrn.edu.br>

R topics documented:

transcriptogramer-package . . . 2
association . . . 3
clusterEnrichment . . . 4
clusterVisualization . . . 5
connectivityProperties . . . 7
DE . . . 8
DEsymbols . . . 9
differentiallyExpressed . . . 9
enrichmentPlot . . . 11
GPL570 . . . 12
GSE9988 . . . 13
Hs700 . . . 14
Hs800 . . . 14
Hs900 . . . 15
HsBPTerms . . . 16
Mm700 . . . 16
Mm800 . . . 17
Mm900 . . . 18
orderingProperties . . . 18
radius<- . . . 20
Rn700 . . . 21
Rn800 . . . 21
Rn900 . . . 22
Sc700 . . . 23
Sc800 . . . 23
Sc900 . . . 24
Terms . . . 25
Transcriptogram-class . . . 26
transcriptogramPreprocess . . . 26
transcriptogramStep1 . . . 27
transcriptogramStep2 . . . 29

Index **31**

`transcriptogramer-package`

Transcriptional analysis based on transcriptograms

Description

R package for transcriptional analysis based on transcriptograms, a method to analyze transcriptomes that projects expression values on a set of ordered proteins, arranged such that the probability that gene products participate in the same metabolic pathway exponentially decreases with the increase of the distance between two proteins of the ordering. Transcriptograms are, hence, genome wide gene expression profiles that provide a global view for the cellular metabolism, while indicating gene sets whose expression are altered.

Author(s)

Diego Morais <vinx@ufrn.edu.br> [maintainer]

Rodrigo Dalmolin <rodrigo.dalmolin@imd.ufrn.br>

References

da Silva, S. R. M., Perrone, G. C., Dinis, J. M., and de Almeida, R. M. C. (2014). Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. BMC Genomics.

de Almeida, R. M. C., Clendenon, S. G., Richards, W. G., Boedigheimer, M., Damore, M., Rossetti, S., Harris, P. C., Herbert, B. S., Xu, W. M., Wandinger-Ness, A., Ward, H. H., Glazier, J. A. and Bacallao, R. L. (2016). Transcriptome analysis reveals manifold mechanisms of cyst development in ADPKD. Human Genomics, 10(1), 1–24.

Ferrareze, P. A. G., Streit, R. S. A., Santos, P. R. dos, Santos, F. M. dos, Almeida, R. M. C. de, Schrank, A., Kmetzsch, L., Vainstein, M. H. and Staats, C. C. (2017). Transcriptional Analysis Allows Genome Reannotation and Reveals that Cryptococcus gattii VGII Undergoes Nutrient Restriction during Infection. Microorganisms.

Rybarczyk-Filho, J. L., Castro, M. A. A., Dalmolin, R. J. S., Moreira, J. C. F., Brunnet, L. G., and de Almeida, R. M. C. (2011). Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. Nucleic Acids Research, 39(8), 3005-3016.

See Also

Bioconductor: release, devel

Github: source code, bug reports

References: (da Silva et al., 2014; de Almeida et al., 2016; Ferrareze et al., 2017; Rybarczyk-Filho et al., 2011)

`association` *Association*

Description

A subset of the Homo sapiens protein network data from STRINGdb, release 10.5. This subset contains only associations of proteins of combined score greater than or equal to 900.

Usage

```
association
```

Format

Each row of the data.frame contains two variables:

V1 The ENSEMBL Peptide ID of the first protein

V2 The ENSEMBL Peptide ID of the second protein

Author(s)

Diego Morais

See Also

Hs900

Examples

```
association
```

`clusterEnrichment` *Term enrichment*

Description

If `species` is a character, this method uses the **biomaRt** package to build a Protein2GO list, if `species` is a data.frame, it will be used instead. The Protein2GO list will be used with the **topGO** package to detect the most significant terms of each cluster present in the DE slot of the `object`.

Usage

```
clusterEnrichment(object, universe = NULL, species,
  ontology = "biological process", algorithm = "classic",
  statistic = "fisher", pValue = 0.05, adjustMethod = "BH", nCores = 1L,
  onlyGenesInDE = TRUE)

## S4 method for signature 'Transcriptogram'
clusterEnrichment(object, universe = NULL,
  species, ontology = "biological process", algorithm = "classic",
  statistic = "fisher", pValue = 0.05, adjustMethod = "BH", nCores = 1L,
  onlyGenesInDE = TRUE)
```

Arguments

`object`	An object of class Transcriptogram.
`universe`	A character vector containing ENSEMBL Peptide IDs, or NULL, if the universe is composed by all the proteins present in the ordering slot of `object`.
`species`	A character string specifying the species; or a data.frame containing two columns, the first one with ENSEMBL Peptide IDs (character), which may, or not, to contain the taxonomy ID of the species as prefix, and the second containing its respective Gene Ontology term (character).
`ontology`	A character string specifying the Gene Ontology domain, ignoring case sensitivity, the possible values are 'biological process', 'cellular component' and 'molecular function'. The default value of this argument is 'biological process'.
`algorithm`	Character string specifying which algorithm to use, the possible values are 'classic', 'elim', 'weight', 'weight01', 'lea' and 'parentchild'. The default value of this argument is 'classic'.
`statistic`	Character string specifying which test to use, the possible values are 'fisher', 'ks', 't', 'sum' and 'globaltest'. The default value of this argument is 'fisher'.
`pValue`	A numeric value between 0 and 1 giving the required family-wise error rate or false discovery rate. The default value of this argument is 0.05.

adjustMethod	Character string specifying p-value adjustment method, the possible values are 'none', 'BH', 'fdr' (equivalent to 'BH'), 'BY', 'hochberg', 'hommel', 'bonferroni', and 'holm'. The default value of this argument is 'BH'.
nCores	An integer number, referring to the number of processing cores to be used; or a logical value, TRUE indicating that all processing cores should be used, and FALSE indicating the use of just one processing core. The default value of this argument is 1.
onlyGenesInDE	Logical value, set as TRUE to use only the genes in the DE slot. Set as FALSE to use all the genes referring to the positions in the clusters slot. The default value of this argument is TRUE.

Value

This method creates a data.frame, containing the most significant terms of each cluster, to feed the Terms slot of an object of class Transcriptogram.

Author(s)

Diego Morais

See Also

differentiallyExpressed, transcriptogramPreprocess, GSE9988, GPL570, Hs900, HsBPTerms, association, transcriptogramStep1, transcriptogramStep2

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01)
transcriptogram <- clusterEnrichment(transcriptogram, species = "Homo sapiens",
pValue = 0.005)

## this call also works
transcriptogram <- clusterEnrichment(transcriptogram, species = HsBPTerms,
pValue = 0.005)

## End(Not run)
```

---

clusterVisualization *Displays graphs of the differentially expressed clusters*

Description

This method uses the **RedeR** package to display graphs of the differentially expressed clusters.

Usage

```
clusterVisualization(object, maincomp = FALSE, connected = FALSE,
  host = "127.0.0.1", port = 9091, clusters = NULL,
  onlyGenesInDE = TRUE)

## S4 method for signature 'Transcriptogram'
clusterVisualization(object, maincomp = FALSE,
  connected = FALSE, host = "127.0.0.1", port = 9091, clusters = NULL,
  onlyGenesInDE = TRUE)
```

Arguments

object	An object of class Transcriptogram.
maincomp	Logical value, set as TRUE if you want to display only the main component of each cluster. The default value of this argument is FALSE.
connected	Logical value, set as TRUE if you want to display only connected nodes. The default value of this argument is FALSE.
host	The domain name of the machine that is running the RedeR XML-RPC server.
port	An integer specifying the port on which the XML-RPC server should listen.
clusters	An integer vector specifying the clusters to be displayed, if NULL, all clusters will be displayed.
onlyGenesInDE	Logical value, set as TRUE to use only the genes in the DE slot. Set as FALSE to use all the genes referring to the positions in the clusters slot. The default value of this argument is TRUE.

Details

RedeR package requirements: Java Runtime Environment (>= 6).

Value

This function returns an object of the RedPort Class.

Author(s)

Diego Morais

See Also

differentiallyExpressed, transcriptogramPreprocess, GSE9988, GPL570, Hs900, association, transcriptogramStep1, transcriptogramStep2, RedPort

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01,
DEsymbols)
rdp <- clusterVisualization(transcriptogram)
```

```
## End(Not run)
```

`connectivityProperties` *Calculates average graph properties as function of node connectivity*

Description

Calculates network properties as function of node connectivity/degree (k), such as: probability of a protein of the graph has degree k, average assortativity of the nodes of degree k, and the average clustering coefficient of the nodes of degree k.

Usage

```
connectivityProperties(object)

## S4 method for signature 'Transcriptogram'
connectivityProperties(object)
```

Arguments

`object`	An object of class Transcriptogram.

Details

The assortativity of a node can be measured by the average degree of its neighbors.

Value

This method returns a data.frame containing: unique degrees (k) of the nodes of the graph, probability (pk) of a node of the graph has degree k, average assortativity (ak) of the nodes of degree k, and the average clustering coefficient (ck) of the nodes of degree k.

Author(s)

Diego Morais

See Also

transcriptogramPreprocess, Hs900, association

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900)
## Not run:
cProperties <- connectivityProperties(transcriptogram)

## End(Not run)
```

DE *Get DE*

Description

Gets the content of the DE slot of an object of class Transcriptogram.

Usage

```
DE(object)

## S4 method for signature 'Transcriptogram'
DE(object)
```

Arguments

object	An object of class Transcriptogram.

Value

This method returns the content of the DE slot of an object of class Transcriptogram.

Author(s)

Diego Morais

See Also

Hs900, association, transcriptogramPreprocess

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01)
DE(transcriptogram)

## End(Not run)
```

DEsymbols *Dictionary Protein2Symbol*

Description

A mapping between ENSEMBL Peptide ID and Symbol (Gene Name). This dataset was created to map the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
DEsymbols
```

Format

Each row of the data.frame contains two variables:

ensembl_peptide_id The ENSEMBL Peptide ID

external_gene_name The Gene Name

Details

This dataset was created to map the differentially expressed proteins of the vignette example.

Author(s)

Diego Morais

Examples

```
DEsymbols
```

differentiallyExpressed *Identify which genes are differentially expressed*

Description

This method uses the **limma** package to identify which genes are differentially expressed, meeting the `pValue` requirement, for the contrast "case-control". The `levels` lenght must be equal to the number of samples present in the transcriptogramS2 slot of the `object`, and its contents is related to the order that the samples appear. FALSE must be used to indicate case samples, and TRUE to indicate control samples. If `species` is NULL, no translation will be done, if `species` is a character, the **biomaRt** package will be used to translate the ENSEMBL Peptide ID to Symbol (Gene Name), and if `species` is a data.frame, it will be used instead. If the translation fail for some protein, its ENSEMBL Peptide ID will be present into the Symbol column. This method also groups the proteins detected as differentially expressed in clusters, and plots a graphical representation of the groupings.

Usage

```
differentiallyExpressed(object, levels, pValue = 0.05,
  species = object@Protein2Symbol, adjustMethod = "BH", trend = FALSE,
  title = "Differential expression", boundaryConditions = FALSE)

## S4 method for signature 'Transcriptogram'
differentiallyExpressed(object, levels,
  pValue = 0.05, species = object@Protein2Symbol, adjustMethod = "BH",
  trend = FALSE, title = "Differential expression",
  boundaryConditions = FALSE)
```

Arguments

`object`	An object of class Transcriptogram.
`levels`	A logical vector that classify the columns, referring to samples, of the transcriptogramS2 slot of the `object`. FALSE must be used to indicate case samples, and TRUE to indicate control samples.
`pValue`	A numeric value between 0 and 1 giving the required family-wise error rate or false discovery rate. The default value of this argument is 0.05.
`species`	A character string that will be used, ignoring case sensitivity, to translate the ENSEMBL Peptide ID to Symbol (Gene Name); or a data.frame containing two columns, the first one with ENSEMBL Peptide IDs (character), which may, or not, to contain the taxonomy ID of the species as prefix, and the second containing its respective Symbol (character). The default value of this argument is the content of the object Protein2Symbol slot.
`adjustMethod`	Character string specifying p-value adjustment method, the possible values are 'none', 'BH', 'fdr' (equivalent to 'BH'), 'BY' and 'holm'. The default value for this argument is 'BH'.
`trend`	Logical value, set as TRUE to use the limma-trend approach for RNA-Seq. The default value of this argument is FALSE.
`title`	An overall title for the plot. The default value of this argument is "Differential expression"
`boundaryConditions`	Logical value, set as TRUE to check if nearby clusters could be merged. The default value of this argument is FALSE.

Value

This method creates a data.frame to feed the DE slot of an object of class Transcriptogram. This data.frame of differentially expressed proteins contains the log2-fold-change, the p-values and an integer number that indicates if the protein is downregulated or upregulated.

Author(s)

Diego Morais

See Also

transcriptogramPreprocess, GSE9988, GPL570, Hs900, association, DEsymbols, transcriptogramStep1, transcriptogramStep2

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01)

## translating ENSEMBL Peptide IDs to Symbols
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01,
"Homo sapiens")

## these calls also works
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01,
"H sapiens")

transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01,
DEsymbols)

## End(Not run)
```

---

enrichmentPlot *Projects Gene Ontology terms on the ordering*

Description

Plots the rate (number of genes related to a term inside the window/total number of genes in the window) of given terms.

Usage

```
enrichmentPlot(object, nCores = 1L, nTerms = 1L, GOIDs = NULL,
  title = "Enrichment")

## S4 method for signature 'Transcriptogram'
enrichmentPlot(object, nCores = 1L, nTerms = 1L,
  GOIDs = NULL, title = "Enrichment")
```

Arguments

object	An object of class Transcriptogram.
nCores	An integer number, referring to the number of processing cores to be used; or a logical value, TRUE indicating that all processing cores should be used, and FALSE indicating the use of just one processing core. The default value of this argument is 1.
nTerms	An integer number referring to the number of top terms from each cluster. The default value of this argument is 1.
GOIDs	A character vector containing the Gene Ontology accessions to be plotted. If NULL, the top nTerms of each cluster will be used.
title	An overall title for the plot. The default value of this argument is "Enrichment"

Value

This method returns an ggplot2 object.

Author(s)

Diego Morais

See Also

differentiallyExpressed, transcriptogramPreprocess, GSE9988, GPL570, Hs900, HsBPTerms, association, transcriptogramStep1, transcriptogramStep2, clusterEnrichment

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01)
transcriptogram <- clusterEnrichment(transcriptogram, species = "Homo sapiens",
pValue = 0.005)
enrichmentPlot(transcriptogram)

## End(Not run)
```

GPL570 *Dictionary Protein2Probe*

Description

A mapping between ENSEMBL Peptide ID and probe identifier, for the Homo sapiens and the platform GPL570, [HG-U133_Plus_2] Affymetrix Human Genome U133 Plus 2.0 Array.

Usage

```
GPL570
```

Format

Each row of the data.frame contains two variables:

Protein The ENSEMBL Peptide ID

Probe The probe identifier

Details

This dataset was created to map the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 700.

Author(s)

Diego Morais

See Also

GSE9988

Examples

```
GPL570
```

`GSE9988` *Dataset containing expression values*

Description

Expression values, obtained by microarray, of 3 cases and 3 controls referring to the Gene Expression Omnibus accession number GSE9988. The data.frame has 6 columns, each one contains expression values of a sample, the first 3 columns are case samples, and the last 3 are control samples. Each row contains expression values obtained by the probe mentioned in its respective rowname. The expression values were normalized using the **affy** package and, to reduce the storage space required for the data, this data.frame is a subset from the original samples (GSM252443, GSM252444, GSM252445, GSM252465, GSM252466, GSM252467), containing only the rows on which the probes are mapped by the GPL570 dictionary to proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
GSE9988
```

Format

An object of class `data.frame` with 24804 rows and 6 columns.

Author(s)

Diego Morais

Source

GSE9988

See Also

GPL570

Examples

```
GSE9988
```

Hs700 *Ordered Homo sapiens proteins of combined score greater than or equal to 700*

Description

A character vector containing the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 700.

Usage

```
Hs700
```

Format

An object of class `character` of length 15154.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 975354074.

Author(s)

Diego Morais

Examples

```
Hs700
```

Hs800 *Ordered Homo sapiens proteins of combined score greater than or equal to 800*

Description

A character vector containing the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 800.

Usage

```
Hs800
```

Format

An object of class `character` of length 13273.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 551245262.

Author(s)

Diego Morais

Examples

```
Hs800
```

Hs900 — *Ordered Homo sapiens proteins of combined score greater than or equal to 900*

Description

A character vector containing the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
Hs900
```

Format

An object of class `character` of length 11030.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 342029174.

Author(s)

Diego Morais

Examples

```
Hs900
```

HsBPTerms *Dictionary Protein2GO*

Description

A mapping between ENSEMBL Peptide ID and Gene Ontology, biological processes, terms. This dataset was created to map the Homo sapiens proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
HsBPTerms
```

Format

Each row of the data.frame contains two variables:

ensembl_peptide_id The ENSEMBL Peptide ID

go_id The Gene Ontology ID

Details

This dataset was created to map the Homo sapiens proteins that appear in the slot transcriptogramS2 of the vignette example.

Author(s)

Diego Morais

Examples

```
HsBPTerms
```

Mm700 *Ordered Mus musculus proteins of combined score greater than or equal to 700*

Description

A character vector containing the Mus musculus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 700.

Usage

```
Mm700
```

Format

An object of class `character` of length 13921.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 544163840.

Author(s)

Diego Morais

Examples

```
Mm700
```

`Mm800` *Ordered Mus musculus proteins of combined score greater than or equal to 800*

Description

A character vector containing the Mus musculus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 800.

Usage

```
Mm800
```

Format

An object of class `character` of length 12166.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 337846752.

Author(s)

Diego Morais

Examples

```
Mm800
```

Mm900 — *Ordered Mus musculus proteins of combined score greater than or equal to 900*

Description

A character vector containing the Mus musculus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
Mm900
```

Format

An object of class `character` of length 9648.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 147064928.

Author(s)

Diego Morais

Examples

```
Mm900
```

orderingProperties — *Calculates graph properties projected on the ordered proteins*

Description

Calculates protein (node) properties, such as: degree/connectivity, number of triangles and clustering coefficient; and properties of the window, region of n (radius * 2 + 1) proteins centered at a protein, such as: connectivity, clustering coefficient and modularity.

Usage

```
orderingProperties(object, nCores = 1L)

## S4 method for signature 'Transcriptogram'
orderingProperties(object, nCores = 1L)
```

Arguments

object	An object of class Transcriptogram.
nCores	An integer number, referring to the number of processing cores to be used; or a logical value, TRUE indicating that all processing cores should be used, and FALSE indicating the use of just one processing core. The default value of this argument is 1.

Details

Connectivity/degree of a node is the number of edges it presents. A triangle of a node represents a pair of connected neighbors, the number of triangles on the adjacency list of a node is required to calculate its clustering coefficient. The clustering coefficient of a node measures, in the interval [0, 1], the likelihood that any two of its neighbors are themselves connected, this is calculated by the ratio between the number of triangles that the node has, and the maximum possible number of edges on its cluster (nodeTriangles / (nodeDegree * (nodeDegree - 1) / 2)). The window connectivity is the average connectivity calculated over the window. The window clustering coefficient, a value in the interval [0, 1], is the average clustering coefficient calculated over the window. The window modularity, a value in the interval [0, 1], is defined as the ratio between the total number of edges between any two nodes of the window, and the sum of the degrees of the nodes presents in the window. The window considers periodic boundary conditions to deal with proteins near the ends of the ordering.

Value

This method returns a data.frame containing: ENSEMBL Peptide ID, its position on the ordering, node degree, number of triangles and clustering coefficient, and window connectivity, clustering coefficient and modularity.

Author(s)

Diego Morais

References

da Silva, S. R. M., Perrone, G. C., Dinis, J. M., and de Almeida, R. M. C. (2014). Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. BMC Genomics.

de Almeida, R. M. C., Clendenon, S. G., Richards, W. G., Boedigheimer, M., Damore, M., Rossetti, S., Harris, P. C., Herbert, B. S., Xu, W. M., Wandinger-Ness, A., Ward, H. H., Glazier, J. A. and Bacallao, R. L. (2016). Transcriptome analysis reveals manifold mechanisms of cyst development in ADPKD. Human Genomics, 10(1), 1–24.

Ferrareze, P. A. G., Streit, R. S. A., Santos, P. R. dos, Santos, F. M. dos, Almeida, R. M. C. de, Schrank, A., Kmetzsch, L., Vainstein, M. H. and Staats, C. C. (2017). Transcriptional Analysis Allows Genome Reannotation and Reveals that Cryptococcus gattii VGII Undergoes Nutrient Restriction during Infection. Microorganisms.

Rybarczyk-Filho, J. L., Castro, M. A. A., Dalmolin, R. J. S., Moreira, J. C. F., Brunnet, L. G., and de Almeida, R. M. C. (2011). Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. Nucleic Acids Research, 39(8), 3005-3016.

See Also

transcriptogramPreprocess, Hs900, association

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 2)
## Not run:
oProperties <- orderingProperties(transcriptogram)

## End(Not run)
```

`radius<-` *Radius*

Description

Retrieve or set the content of the radius slot of an object of class Transcriptogram.

Usage

```
radius(object) <- value

radius(object)

## S4 replacement method for signature 'Transcriptogram'
radius(object) <- value

## S4 method for signature 'Transcriptogram'
radius(object)
```

Arguments

`object`	An object of class Transcriptogram.
`value`	An non-negative integer referring to the window radius required for some methods.

Value

This method returns the content of the radius slot of an object of class Transcriptogram.

Author(s)

Diego Morais

See Also

Hs900, association, transcriptogramPreprocess, transcriptogramStep2, orderingProperties

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
radius(transcriptogram) <- 80
radius(transcriptogram)
```

`Rn700` *Ordered Rattus norvegicus proteins of combined score greater than or equal to 700*

Description

A character vector containing the Rattus norvegicus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 700.

Usage

```
Rn700
```

Format

An object of class `character` of length 14285.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 515790574.

Author(s)

Diego Morais

Examples

```
Rn700
```

`Rn800` *Ordered Rattus norvegicus proteins of combined score greater than or equal to 800*

Description

A character vector containing the Rattus norvegicus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 800.

Usage

```
Rn800
```

Format

An object of class `character` of length 12437.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 280242352.

Author(s)

Diego Morais

Examples

```
Rn800
```

Rn900 *Ordered Rattus norvegicus proteins of combined score greater than or equal to 900*

Description

A character vector containing the Rattus norvegicus proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
Rn900
```

Format

An object of class `character` of length 9747.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 123574716.

Author(s)

Diego Morais

Examples

```
Rn900
```

Sc700 — *Ordered Saccharomyces cerevisiae proteins of combined score greater than or equal to 700*

Description

A character vector containing the Saccharomyces cerevisiae proteins, from STRINGdb release 10.5, of combined score greater than or equal to 700.

Usage

```
Sc700
```

Format

An object of class `character` of length 5586.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 141868972.

Author(s)

Diego Morais

Examples

```
Sc700
```

Sc800 — *Ordered Saccharomyces cerevisiae proteins of combined score greater than or equal to 800*

Description

A character vector containing the Saccharomyces cerevisiae proteins, from STRINGdb release 10.5, of combined score greater than or equal to 800.

Usage

```
Sc800
```

Format

An object of class `character` of length 5090.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 66127712.

Author(s)

Diego Morais

Examples

```
Sc800
```

Sc900 — *Ordered Saccharomyces cerevisiae proteins of combined score greater than or equal to 900*

Description

A character vector containing the Saccharomyces cerevisiae proteins, from STRINGdb release 10.5, of combined score greater than or equal to 900.

Usage

```
Sc900
```

Format

An object of class `character` of length 4386.

Details

Generated by The Transcriptogramer V.1.0 for Windows. Input arguments: isothermal steps - 100; Monte Carlo steps - 20000; cooling factor - 0.5; alpha value - 1.0; percentual energy for initial temperature - 0.0001. Final energy: 27272296.

Author(s)

Diego Morais

Examples

```
Sc900
```

`Terms` *Get terms*

Description

Gets the content of the Terms slot of an object of class Transcriptogram.

Usage

```
Terms(object)

## S4 method for signature 'Transcriptogram'
Terms(object)
```

Arguments

`object`	An object of class Transcriptogram.

Value

This method returns the content of the Terms slot of an object of class Transcriptogram.

Author(s)

Diego Morais

See Also

differentiallyExpressed, transcriptogramPreprocess, GSE9988, GPL570, Hs900, HsBPTerms, association, transcriptogramStep1, transcriptogramStep2, clusterEnrichment

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)
levels <- c(rep(FALSE, 3), rep(TRUE, 3))
transcriptogram <- differentiallyExpressed(transcriptogram, levels, 0.01)
transcriptogram <- clusterEnrichment(transcriptogram, species = "Homo sapiens",
pValue = 0.005)
Terms(transcriptogram)

## End(Not run)
```

Transcriptogram-class *Class Transcriptogram*

Description

This S4 class includes methods to use expression values with ordered proteins.

Slots

`association` A data.frame containing two columns, with rows containing ENSEMBL Peptide IDs that are connected.

`ordering` A data.frame containing two columns, the first one with ENSEMBL Peptide IDs, and the second containing its respective position.

`transcriptogramS1` A data.frame produced as the result of averaging over all identifiers related to the same protein.

`transcriptogramS2` A data.frame produced as the result of averaging over the window.

`radius` An non-negative integer referring to the window radius.

`status` An integer used internally to check the status of the object.

`DE` A data.frame of differentially expressed proteins.

`clusters` A list indicating the first and the last position belonging to each cluster.

`pbc` Logical value used internally to indicate the overlapping of the first and the last cluster.

`Protein2Symbol` A data.frame containing two columns, the first one with ENSEMBL Peptide IDs, and the second containing its respective Symbol.

`Protein2GO` A data.frame containing two columns, the first one with ENSEMBL Peptide IDs, and the second containing its respective Gene Ontology accession.

`Terms` A data.frame containing the enriched Gene Ontology terms.

Author(s)

Diego Morais

See Also

transcriptogramPreprocess, DE, radius, orderingProperties, connectivityProperties, transcriptogramStep1, transcriptogramStep2, differentiallyExpressed, clusterVisualization, clusterEnrichment

transcriptogramPreprocess *Creates an object of class Transcriptogram*

Description

Constructor for the Transcriptogram object.

Usage

```
transcriptogramPreprocess(association, ordering, radius = 0L)
```

Arguments

association	A matrix, or data.frame, containing two columns of ENSEMBL Peptide IDs (character); or the path for a file containing two columns, no header, with rows composed by the ENSEMBL Peptide IDs of two proteins that are connected.
ordering	A character vector containing ordered ENSEMBL Peptide IDs; a data.frame containing two columns, the first one with ENSEMBL Peptide IDs (character), and the second containing its respective position (non-negative integer); or the path for a file containing two columns, a row for the headers, with rows composed respectively by a ENSEMBL Peptide ID and its respective position.
radius	An non-negative integer referring to the window radius required for some methods.

Value

A preprocessed object of class Transcriptogram.

Author(s)

Diego Morais

See Also

Transcriptogram-class, association, Hs900

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900)
```

---

transcriptogramStep1 *Calculates the average of the expression values related to the same protein*

Description

For each transcriptome sample, this method assigns to each protein the average of the expression values of all the identifiers related to it. It is necessary a `dictionary` to map the identifiers to proteins.

Usage

```
transcriptogramStep1(object, expression, dictionary, nCores = 1L)

## S4 method for signature 'Transcriptogram'
transcriptogramStep1(object, expression, dictionary,
  nCores = 1L)
```

Arguments

`object`	An object of class Transcriptogram.
`expression`	A matrix, or data.frame, containing normalized expression values from samples of microarrays or RNA-Seq (log2-counts-per-million).
`dictionary`	A matrix, or data.frame, containing two columns, the first column must contains the ENSEMBL Peptide ID, and the second column must contains values that appear as rownames in `expression`, in order to recognize the ENSEMBL Peptide ID of the other column.
`nCores`	An integer number, referring to the number of processing cores to be used; or a logical value, TRUE indicating that all processing cores should be used, and FALSE indicating the use of just one processing core. The default value of this argument is 1.

Value

This method creates a data.frame to feed the transcriptogramS1 slot of an object of class Transcriptogram. Each row of the data.frame contains: an ENSEMBL Peptide ID, its respective position in the ordering and the mean of the expression values of the identifiers related to the same protein.

Author(s)

Diego Morais

References

da Silva, S. R. M., Perrone, G. C., Dinis, J. M., and de Almeida, R. M. C. (2014). Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. BMC Genomics.

de Almeida, R. M. C., Clendenon, S. G., Richards, W. G., Boedigheimer, M., Damore, M., Rossetti, S., Harris, P. C., Herbert, B. S., Xu, W. M., Wandinger-Ness, A., Ward, H. H., Glazier, J. A. and Bacallao, R. L. (2016). Transcriptome analysis reveals manifold mechanisms of cyst development in ADPKD. Human Genomics, 10(1), 1–24.

Ferrareze, P. A. G., Streit, R. S. A., Santos, P. R. dos, Santos, F. M. dos, Almeida, R. M. C. de, Schrank, A., Kmetzsch, L., Vainstein, M. H. and Staats, C. C. (2017). Transcriptional Analysis Allows Genome Reannotation and Reveals that Cryptococcus gattii VGII Undergoes Nutrient Restriction during Infection. Microorganisms.

Rybarczyk-Filho, J. L., Castro, M. A. A., Dalmolin, R. J. S., Moreira, J. C. F., Brunnet, L. G., and de Almeida, R. M. C. (2011). Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. Nucleic Acids Research, 39(8), 3005-3016.

See Also

transcriptogramPreprocess, GSE9988, GPL570, Hs900, association

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)

## End(Not run)
```

`transcriptogramStep2` *Calculates the average of the expression values using a sliding window*

Description

To each position of the ordering, this method assigns a value equal to the average of the expression values inside a window, region of n (radius * 2 + 1) proteins centered at a protein. The window considers periodic boundary conditions to deal with proteins near the ends of the ordering.

Usage

```
transcriptogramStep2(object, nCores = 1L)

## S4 method for signature 'Transcriptogram'
transcriptogramStep2(object, nCores = 1L)
```

Arguments

`object`	An object of class Transcriptogram.
`nCores`	An integer number, referring to the number of processing cores to be used; or a logical value, TRUE indicating that all processing cores should be used, and FALSE indicating the use of just one processing core. The default value of this argument is 1.

Value

This method creates a data.frame to feed the transcriptogramS2 slot of an object of class Transcriptogram. Each row of the data.frame contains: the ENSEMBL Peptide ID used as center of the window, its position on the ordering, and the mean of the expression values of the window.

Author(s)

Diego Morais

References

da Silva, S. R. M., Perrone, G. C., Dinis, J. M., and de Almeida, R. M. C. (2014). Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. BMC Genomics.

de Almeida, R. M. C., Clendenon, S. G., Richards, W. G., Boedigheimer, M., Damore, M., Rossetti, S., Harris, P. C., Herbert, B. S., Xu, W. M., Wandinger-Ness, A., Ward, H. H., Glazier, J. A. and Bacallao, R. L. (2016). Transcriptome analysis reveals manifold mechanisms of cyst development in ADPKD. Human Genomics, 10(1), 1–24.

Ferrareze, P. A. G., Streit, R. S. A., Santos, P. R. dos, Santos, F. M. dos, Almeida, R. M. C. de, Schrank, A., Kmetzsch, L., Vainstein, M. H. and Staats, C. C. (2017). Transcriptional Analysis Allows Genome Reannotation and Reveals that Cryptococcus gattii VGII Undergoes Nutrient Restriction during Infection. Microorganisms.

Rybarczyk-Filho, J. L., Castro, M. A. A., Dalmolin, R. J. S., Moreira, J. C. F., Brunnet, L. G., and de Almeida, R. M. C. (2011). Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. Nucleic Acids Research, 39(8), 3005-3016.

See Also

transcriptogramPreprocess, GSE9988, GPL570, Hs900, association, transcriptogramStep1

Examples

```
transcriptogram <- transcriptogramPreprocess(association, Hs900, 50)
## Not run:
transcriptogram <- transcriptogramStep1(transcriptogram, GSE9988, GPL570)
transcriptogram <- transcriptogramStep2(transcriptogram)

## End(Not run)
```

Index

∗Topic **datasets**
association, 3
DEsymbols, 9
GPL570, 12
GSE9988, 13
Hs700, 14
Hs800, 14
Hs900, 15
HsBPTerms, 16
Mm700, 16
Mm800, 17
Mm900, 18
Rn700, 21
Rn800, 21
Rn900, 22
Sc700, 23
Sc800, 23
Sc900, 24

association, 3, *5–8*, *10*, *12*, *19*, *20*, *25*, *27*, *28*, *30*

clusterEnrichment, 4, *12*, *25*, *26*
clusterEnrichment,Transcriptogram-method *(*clusterEnrichment*)*, 4
clusterEnrichment-method *(*clusterEnrichment*)*, 4
clusterVisualization, 5, *26*
clusterVisualization,Transcriptogram-method *(*clusterVisualization*)*, 5
clusterVisualization-method *(*clusterVisualization*)*, 5
connectivityProperties, 7, *26*
connectivityProperties,Transcriptogram-method *(*connectivityProperties*)*, 7
connectivityProperties-method *(*connectivityProperties*)*, 7

DE, 8, *26*
DE,Transcriptogram-method *(*DE*)*, 8
DE-method *(*DE*)*, 8
DEsymbols, 9, *10*
differentiallyExpressed, *5*, *6*, 9, *12*, *25*, *26*
differentiallyExpressed,Transcriptogram-method *(*differentiallyExpressed*)*, 9
differentiallyExpressed-method *(*differentiallyExpressed*)*, 9

enrichmentPlot, 11
enrichmentPlot,Transcriptogram-method *(*enrichmentPlot*)*, 11
enrichmentPlot-method *(*enrichmentPlot*)*, 11

GPL570, *5*, *6*, *10*, *12*, 12, *13*, *25*, *28*, *30*
GSE9988, *5*, *6*, *10*, *12*, *13*, 13, *25*, *28*, *30*

Hs700, 14
Hs800, 14
Hs900, *4–8*, *10*, *12*, 15, *19*, *20*, *25*, *27*, *28*, *30*
HsBPTerms, *5*, *12*, 16, *25*

Mm700, 16
Mm800, 17
Mm900, 18

orderingProperties, 18, *20*, *26*
orderingProperties,Transcriptogram-method *(*orderingProperties*)*, 18
orderingProperties-method *(*orderingProperties*)*, 18

radius, *26*
radius *(*radius<-*)*, 20
radius,Transcriptogram-method *(*radius<-*)*, 20
radius-method *(*radius<-*)*, 20
radius<-, 20
radius<-,Transcriptogram-method *(*radius<-*)*, 20
RedPort, *6*
Rn700, 21
Rn800, 21
Rn900, 22

Sc700, 23
Sc800, 23
Sc900, 24

Terms, 25
Terms,Transcriptogram-method *(*Terms*)*, 25
Terms-method *(*Terms*)*, 25
Transcriptogram-class, 26, *27*
transcriptogramer
*(*transcriptogramer-package*)*, 2
transcriptogramer-package, 2
transcriptogramPreprocess, *5–8*, *10*, *12*, *19, 20*, *25, 26*, 26, *28*, *30*
transcriptogramStep1, *5, 6*, *10*, *12*, *25, 26*, 27, *30*
transcriptogramStep1,Transcriptogram-method
*(*transcriptogramStep1*)*, 27
transcriptogramStep1-method
*(*transcriptogramStep1*)*, 27
transcriptogramStep2, *5, 6*, *10*, *12*, *20*, *25, 26*, 29
transcriptogramStep2,Transcriptogram-method
*(*transcriptogramStep2*)*, 29
transcriptogramStep2-method
*(*transcriptogramStep2*)*, 29