NEAT for RNAseq

NGS pipelines made easy

Patrick Schorderet

Patrick.schorderet@molbio.mgh.harvard.edu

Jan 2015

1 Introduction

NExt generation ***A***nalysis ***T***oolbox (NEAT) is a perl/R package that supports users during the analysis of next generation sequencing (NGS)

NEAT provides an easy, rapid and reproducible exploratory data analysis (EDA) tool that allows users to assess their data in less than 24 hours (based on a 200mio read Highseq run). NEAT was developed in two main sections, ChIPpip (RNApip) and ChIPmE (RNAmE). The first section (ChIPpip) helps users with demanding computationally (mapping, filtering, etc). The second section (ChIPmE) consists of two standalone applications that provide users with human readable data.

2 Getting started with UNIX

2.1 Notes

In the following tutorial, all unix/perl/R command lines will be bold, italicized and highlighted in blue. Most will be embedded in tables. The command line is the text following, but not including, the dollar sign ($).

---

*[~]$ **this is a unix command***

This tutorial is intended to run on MacOS/LINUX environments. For MacOS users, we suggest to use the *Terminal* (applications/Terminal) for all following steps. The terminal/shell output will be depicted in black.

Copy pasting the *unix* commands should allow you to follow all steps of this tutorial. Please be aware that *unix* commands are case sensitive, including white spaces.

2.2 Remote servers and SSH

Secure Shell (SSH) is a cryptographic network protocol for secure data communication. In brief, it is a way for users to access remote computers (and their content) using a secure channel (a tunnel) through an insecure network (the internet). To access your computer cluster, you will need to establish an SSH connection. In analogy to an access card to your building, each user should be provided with an SSH username and password. Finally, the last essential parameter to access your computer cluster is the virtual *address* of the server. However, before accessing the remote server, you will need to copy the ChIPpip directory from your local computer to the remote server.

3 NEAT

3.1 Introduction to NEAT

A central feature of NEAT is its ability to perform repetitive tasks on complex sample setups while managing batch submissions and cluster queuing. NEAT can easily be implemented in any institution with limited to no programming experience. The workflow has been designed to efficiently run on a computer cluster using a distributed resource manager such as TORQUE. NEAT has been developed by and for wet-lab scientists as well as bioinformaticiens to ensure user-friendliness, management of complicated experimental setups and reproducibility in the big data era.

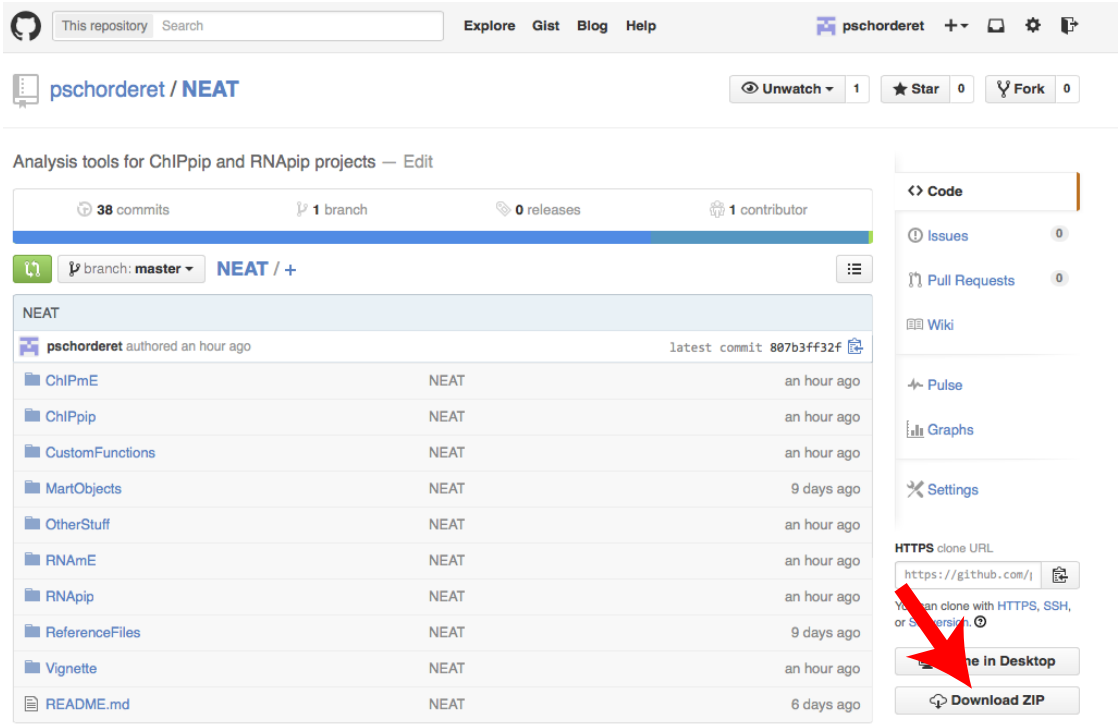
To start using NEAT for RNAseq data analysis, please follow the tutorial. This will walk you through the analysis of a small test dataset (provided as part of RNApip) using your own computer cluster. This will also ensure NEAT and its dependencies are correctly installed before submitting large, memory-savvy analysis.

All fastq files from the test data have been subsetted to ca. 15'000 reads. This data comes from an unpublished 50bp single end (SE) sequencing experiment although NEAT can deal with paired-end (PE) sequencing as well. For more information on the test data provided in this tutorial, please read below.

Although NEAT can be run by scientists with limited to no programming experience, this tutorial requires access to a remote server. Users are thereby required to have SSH accessibility with a username and a password. Please refer to your system administrator to obtain such credentials. For more information on how to access a remote server through SSH, please read below.

3.2 Install NEAT

To install NEAT, download the NEAT repository from GitHub to any directory on your computer cluster.



As an example for the following tutorial, we will suppose the *NEAT/* directory was saved to the user's desktop on a local computer (*~/Desktop*) and that it will be saved to their home directory (*~/*) on the remote server. In general, home directories can be accessed using the ~ sign. Make sure the folder is named *NEAT* and not *NEAT-master*. Start the transfer to the remote folder by typing the following command:

```
[~Desktop]$ rsync -avz ~/Desktop/NEAT username@serveradress.edu:~/
```

Enter your password and wait for the folders to transfer.

An alternative for copying files and folders from your local computer to a remote server are free softwares such as *FileZilla* or *Cyberduck*.
Once it has finished, the NEAT directory should be saved to your remote server. Check this by accessing your remote server. In a *Terminal* window, type the following:

```
[MY_COMPUTE~]$ ssh username@serveradress.edu
password
[username@setrveradress ~]$
```

These commands bring you to your <HOME> directory on the remote server. If you are not sure of your current working directory, type ***pwd*** to *print* your current *working directory*. Navigate to the section 1 (RNApip) directory of NEAT using the ***cd*** and list files using the ***ls*** command. If NEAT was properly copied to your <HOME> directory, you should see the following:

```
[username@setrveradress ~]$ cd ~/NEAT/RNApip
[username@setrveradress RNApip]$ ls -l
1911    README.md
3371    RNApipCreateNewProject.pl
9932    RNApipRunPipeline.pl
4096    scripts
```

Note here that *./* is a short cut to your working directory. In this tutorial, because we have set our working directory to *~/NEAT/RNApip/* by using the command ***cd ~/NEAT/RNApip/***, '*./*' will be a short cut for *~/NEAT/RNApip/* and these can and will be used interchangeably.

3.3 Test data

3.3.1 Structure and significance of test data

The unpublished test data provided for this tutorial is part of the RNApip folder. If you have followed the previous steps, the compressed fastq files required for the tutorial can be found in the test data folder *~/NEAT/RNApip/scripts/testdata/*.

```
[RNApip]:$ ls -l ./scripts/testdata
751359    PSa29-5_R1.fastq.gz
751215    PSa29-6_R1.fastq.gz
```

The data consists of a 50 base pair, single end RNAseq experiment on cells under two growing conditions (noDox, Dox). No replicates were generated.

4 NEAT Part 1 : RNApip

4.1 Running NEAT part 1.1 (RNApip)

4.1.1 Creating a new RNApip project

The first step to run RNApip is to create a new RNApip project. Navigate to the RNApip directory and run the *RNApipCreateNewProject.pl* script. We need to add ***perl*** in front of this command to tell the computer it should deal with this file as a perl script. This script requires the user to specify the path where the new RNApip

project will be created including the name of the new project. In this example, we will create a project in the RNApip directory named EXAMPLE.

```
[RNApip]:$ perl RNApipCreateNewProject.pl ~NEAT/RNApip/EXAMPLE

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-

Creating a new RNApip project

-------------------------------------------------------

New RNApip project has been built as follows:

        ~/NEAT/RNApip/EXAMPLE
                └─ DataStructure
                        └─ Targets.txt
                └─ fastq
                └─ scripts

-------------------------------------------------------------------------------------------------

IMPORTANT:  Fill in the Targets.txt file before running RNApipRunPipeline.pl
            To modify Targets.txt, copy paste the following command in your terminal:

            nano ~/NEAT/RNApip/EXAMPLE/DataStructure/Targets.txt

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
```

Running the *RNApipCreateNewProject.pl* script should prompt the message above. If not, please troubleshoot before proceeding to the next step.

4.1.2 The Targets.txt file structure

4.1.2.1 Filling in the Targets.txt file

The Targets.txt file found in the *~/EXAMPLE/DataStructure/* directory is the backbone of RNApip. It contains all the information specific to your experiment and your computer cluster, including the names of files, the paths to the reference genomes, the steps to execute, the name of your samples, their relationships, etc.

IMPORTANT: Sample names including inputs and fastq files cannot start with an 'n' (small or capital) as this is the universal perl symbol for carriage return.

The *Targets.txt* file is the most important piece of RNApip and users are expected to invest the time to ensure all paths and parameters exist and are correctly set. However, once set, most of these parameters will not be changed on a specific computer cluster (users from a same institute will use the same paths). Therefor, we suggest modifying the *original* Targets.txt template file (see below).

All parameters of the Targets file should be self-explanatory. Here is a brief summary:

My_email	:	If users would like to be notified by emailed when the cluster has finished. This will only work if your computer cluster has activated the emailing feature (please check with system administrator). To ensure servers are not overwhelmed by email services, RNApip is configured in such a way as to notify users only if the pipeline has terminated properly (with no error). Users may change this parameter by modifying the QSUB_header.sh template file found in *./RNApip/scripts/*.
My_project_title	:	This is the name of the folder of your project on the remote server [automatically generated by RNApipCreateNewProject].
Reference_genome	:	The genome your data will be aligned to. Make sure your core faility has this genome reference installed on your cluster and that the extensions of the files are '.fa'.
Path_to_proj_folder	:	Full path to your project folder (without the project name) [automatically generated by RNApipCreateNewProject].
Path_to_RNApip	:	Full path to your RNApip folder. Note that in our example, we have created our project within the RNApip folder itself, but users can freely decide to create a dedicated folder for all of their RNApip projects.
Path_to_orifastq.gz	:	Full path to where your .fastq.gz files are. Usually, your sequencing core facility will let you know where they store these files. Note that all .fastq.gz files can be kept in a single location, they do not need to be copied to your folder.
Path_to_chrLens.dat	:	Path to a .dat file containing chromosome information for your reference genome. Refer to your computer core facility.
Path_to_RefGen.fa	:	Path to folder containing your reference genome files. Refer to your computer core facility.
Aligner_algorithm	:	"Tophat" for standard alignment. If other algorithms are used, modify the *AdvancedSettings.txt* and the *RNApip.pl* files

		accordingly.
Paired_end_run	:	"0" for single end sequencing. "1" for paired end sequencing.
Path_to_gtfFile	:	Path to the gtf file corresponding to your genome. Refer to your computer core facility.
Steps_to_execute	:	Users can choose from the following tasks: *unzip*, *qc*, *map* and *filter*. If you do not want to run all of these, simply delete them for the Targets.txt or rename them. Once ran, RNApip will change the value of these from '*unzip*' to '*unzip_DONE*'. Obviously, a certain hierarchy has to be followed, e.g. attempting to filter reads without having previously mapped them (in the same run or in a previous run) will not work. Note that '*qc*' requires Thomas Girke's systemPipeR package; map requires TopHat and *filtering* requires samtools. Refer below for exact requirements.
# Local	:	These parameters are only necessary for users who go on to use the NEAT tookit for metagene analysis, etc on their local computer. If you will not use this package, please disregard (leave as is).
TaxonDatabaseKG	:	Database of preferred feature such as known gene for RNAseq.
TaxonDatabaseDict	:	Idem

Please modify the Targets.txt file to your needs. The paths to the reference genomes should be obtained from your computer core facility (system administrator), as they are the ones usually maintaining these up to date. Moreover, the reference genome files should have a '*.fa*' extension (e.g. mm9.fa). Please check that your core has named these files accordingly as any other extension will lead the pipeline to abort prematurely. To modify the Targets.txt file, we suggest users get accustomed to using a terminal text editor such as ***vi*** or ***nano*** as it will avoid including spaces and special characters.

Fill in your Targets.txt fill using the following command:

[RNApip]:$ ***nano ./EXAMPLE/DataStructure/Targets.txt***

```
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
#
# Project ID: "EXAMPLE"
#
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
#
# Remote Server
#
# My_email	=	"your.email@harvard.edu"
#
# My_project_title	=	"EXAMPLE"
# Reference_genome	=	"mm9"
# Path_to_proj_folder	=	"~/NEAT/RNApip"
# Path_to_RNApip	=	"~/NEAT/RNApip"
# Path_to_orifastq.gz	=	"~/NEAT/RNApip/fastq"
# Path_to_chrLens.dat	=	"~/.../reference_files/mm9/chr_lens.dat"
# Path_to_RefGen.fa	=	"~/.../bowtie/"
# Path_to_gtfFile	=	"~/.../Mus_musculus.NCBIM37.67.allchr.gtf
# Aligner_algorithm	=	"Tophat"
# Paired_end_run	=	"0"
#
# Steps_to_execute	=	" unzip + qc + map + filter "
#
# Remove_from_bigwig	=	" random, chrM"
#
#
# Local
#
# TaxonDatabaseKG	=	"TxDb.Mmusculus.UCSC.mm9.knownGene"
# TaxonDatabaseDict	=	"org.Mm.eg.db"
#
#--------------------------------------------------------------------------------------------------------
#
#
OriFileName	FileName	OriInpName	InpName	Factor	Replicate	FileShort	Experiment	Date
PSa36-1_R1	PSa36-1_noDox_K4me3	PSa37-3_R1	PSa37-3_noDox_Inp	K4me3	1	noDox	1	2015-01-01
PSa36-2_R1	PSa36-2_Dox_K4me3	PSa37-4_R1	PSa37-4_Dox_Inp	K4me3	2	Dox	1	2015-01-01
```

Note here that *# Path_to_RefGen.fa* refers to the path to the folder in which '*.fa*' files are located, not to the file itself. This is different than in ChIPpip. *#_Remove_from_bigwig* can usually be disregarded.

Once all information has been modified, hit ***cmd x*** to save the file. Confirm by hitting the ***y*** and ***enter***. This will save all changes to the file.

To avoid repeating these steps at each new RNApip project creation, we suggest you modify the *original* Targets.txt file that is used as template when creating a new RNApip project. Modify it using the same approach as above:

```
[RNApip]:$ nano ./scripts/NewRNApipProject/DataStructure/Targets.txt
```

4.2 Running NEAT part 2

4.2.1 Run the analysis

Once the Targets.txt file is correctly set up, the *RNApipRunPipeline.pl* script can be run. This script will execute the tasks specified in the *Targets.txt* file. Users can choose to perform the following tasks: *unzip, qc, map* and *filter*. If one or several tasks should not be run, simply discard them from the *Targets.txt* file under *# Steps to execute*.

As does the *RNApipCreateNewProject.pl*, the *RNApipRunPipeline.pl* script requires the user to specify the path to the RNApip project folder (users will obviously feed the same path to both scripts). In our example, the path is *./EXAMPLE*.

[RNApip]:$ ***perl RNApipRunPipeline.pl ~/NEAT/RNApip/EXAMPLE***

```
########################################################################
#                                                                      #
#                        RNAseq pipeline v1.0.1 (Jan 2015)             #
#                                                                      #
########################################################################

 My email:              your.email@harvard.edu

 expFolder:             EXAMPLE
 genome:                mm9
 userFolder:            ~/
 path2RNApip:           ~/NEAT/RNApip
 path2expFolder:        ~/NEAT/RNApip/EXAMPLE
 path2fastq.gz:         ~/NEAT/RNApip/testdata/
 Targets:               ~/NEAT/RNApip/EXAMPLE/DataStructure/Targets.txt
 chrlens:               ~/…/reference_files/mm9/chr_lens.dat
 refGenome:             ~/…/mm9.fa

 Paired end sequencing:      0
 Align.command:              /usr/local/bin/tophat2.4 --bowtie1 --no-mixed --no-discordant -g 1 -p 4 -G
 Remove pcr dupl:            1
 Make unique reads:          1

 ..........
 Performing following tasks:
 ..........
 unzip:                 TRUE
 map:                   TRUE
 qc:                    TRUE
 filter:                TRUE
 ..........

Samples:
        PSa29-5_noDox_K4me3
        PSa29-6_Dox_K4me3

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 Exiting INITIAL section with no known error

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

This will launch the pipeline and will prompt a summary of the user's parameters. RNApip automatically manages all creations and batch submissions of jobs, dependencies, ordering of files, queuing, etc. If the cluster is using TORQUE, the processes can be followed by the ***qstat*** command (type ***qstat*** in your terminal). Briefly, *Q* stands for queuing, *R* for running, *E* for exiting and *H* for holding.

From this step on, the user will NOT need to intervene further. The pipeline is completely automated.

Once the pipeline has finished, it will notify the user of its status by email. The first step is to check whether everything ran smoothly. To this end, please open the Targets.txt file and check whether all jobs are marked as *DONE* under the *# Steps to execute* tag. If not, please follow the 'Troubleshooting' section below.

The mock data provided as a test example should take no more than one hour to run, usually a lot less.

4.2.2 RNApip workflow

4.2.2.1 Unzipping and renaming fastq files

Using the *Targets.txt* file, RNApip will unzip fastq.gz files found under '*OriFileName*' /'*OriInpName*' and will rename them to names found under '*FileName*'/'*InpName*'. RNAseq experiments usually do not require inputs, so these entries will be left empty ('-'). RNApip will use the virtual path to the compressed files and will save the unzipped fastq files in the project folder. This minimizes file transfer and ensures all original fastq files can be kept in a central directory.

4.2.2.2 Quality control (QC)

Quality control of fastq files uses the elegant systemPipeR package developed by Thomas Girke (Girke, 2014).

4.2.2.3 Mapping and filtering reads

RNApip utilizes TopHat (REFERENCE), a well-established, splice-aware and commonly accepted algorithm for RNAseq data. The most common parameters can be changed in the *AdvancedSettings.txt* file (*/EXAMPLE/DataStructure/AdvancedSettings.txt*).

4.3 Outputs

4.3.1 Architecture

RNApip will generate many files of which the majority will not be used in further analysis. We should note that the aligned, filter .bam files are stored in */EXAMPLE/Tophat/<sample>/<sample>.bam*. Quality reports (if applicable) are found in */EXAMPLE/QC/*.

4.4 Next steps: RNAmE

Although some users may prefer to take over the analysis from this step, we suggest using RNAmE. RNAmE is part of the NEAT toolkit and has been developed as a downstream module for RNApip. RNAmE accompanies users from an RNApip output to differential gene expression (DEG) analysis in as few as two double clicks. It automatically transfers files from remote server to local computer to create wet-lab scientist readable data including pdf smear graphs, Venn diagrams (overlap of DEG), count tables and RPKM tables. Users interested in such analysis can download the NEAT package from GitHUB and follow the tutorial.

4.5 Troubleshooting

4.5.1 Output and error files

RNApip is broken down into distinct job sections. For example, all files corresponding to the 'map' section can be found in the scripts folder (*/<HOME>/RNApip/EXAMPLE/scripts/map/*). In each job folder, the qsub directory

contains all output (.o.jobID) and error (.e.jobID) files for individual jobs, which makes it easy to troubleshoot any possible errors. Files are named as follows:

```
[RNApip]:$ ls -l ./EXAMPLE/scripts/map/

1018   map.sh
 759   PSa29-5_noDox_RNA_map.sh
 749   PSa29-6_noDox_RNA_map.sh
 ...
4096   qsub

[RNApip]:$ ls –l ./EXAMPLE/scripts/map/qsub

   0   Iterate_map.o<jobID>
   0   Iterate_map.e<jobID>
 354   PSa36-1_noDox_K4me3_map.sh.e<jobID>
   0   PSa36-1_noDox_K4me3_map.sh.o<jobID>
```

Most .o.jobID and .e.jobID qsub files should be empty. Exceptions to this are the map.e.jobID and the filter.e.jobID files, which contain terminal outputs. Most of these can be disregarded.

In the scenario where RNApip cannot proceed to all jobs, it will stop. Users can modify the email settings in the QSUB_header.sh to be notified in case of an error (refer to the Advanced settings section below). Users can follow up which jobs induced the stop by looking at the Targets.txt file. The last <job>_DONE is the <job> that induced the premature stop.

4.5.2 Example

As an example, lets suppose RNApip crashed while analyzing the test data. Troubleshoot the error by looking at the Targets.txt file:

[RNApip]:$ ***less* ./EXAMPLE/DataStructure/Targets.txt**

```
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
#
# Project ID: "EXAMPLE"
#
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
#
# Remote Server
#
# My_email        =     "your.email@harvard.edu"
#
# My_project_title      =     "EXAMPLE"
# Reference_genome      =     "mm9"
# Path_to_proj_folder   =     "~/NEAT/RNApip"
# Path_to_RNApip        =     "~/NEAT/RNApip"
# Path_to_orifastq.gz   =     "~/NEAT/RNApip/fastq"
# Path_to_chrLens.dat   =     "~/.../reference_files/mm9/chr_lens.dat"
# Path_to_RefGen.fa     =     "~/.../"
# Paired_end_run        =     "0"
#
# Steps_to_execute      =     " unzip_DONE+ qc_DONE + map_DONE + filter "
#
# Remove_from_bigwig    =     "random, chrM"
#
#
# Local
#
# TaxonDatabaseKG       =     "TxDb.Mmusculus.UCSC.mm9.knownGene"
# TaxonDatabaseDict     =     "org.Mm.eg.db"
#
#-----------------------------------------------------------------------------------------------------$
#
#
OriFileName  FileName  OriInpName  InpName  Factor  Replicate  FileShort  Experiment  Date
PSa29-5_R1  PSa29-5_noDox_RNA  -  -  RNA  1  noDox  1  2015-01-01
PSa29-6_R1  PSa29-6_Dox_RNA    -  -  RNA  2  Dox    1  2015-01-01
```

The last <job>_DONE is the map_DONE, which indicates the source of the error. Troubleshoot the origin by looking at the qsub error files corresponding to the 'map' section.

[RNApip]:$ ***ls -l ./scripts/map/qsub***

```
790    PSa29-5_noDox_RNA_map.sh.e<jobID>
  0    PSa29-5_noDox_RNA_map.sh.o<jobID>
```

The .e.<jobID> file is a lot bigger than usual. Use the unix ***less*** command to open it in a read mode. Error messages should be self-explanatory. To exit, hit ***Enter***. In our example, we have the following error message:

```
[RNApip]:$ less ./EXAMPLE/scripts/map/qsub/
[bwt_restore_bwt] fail to open file '/data/ref/mm9/bwa/mm9.fa.bwt'. Abort!
```

RNApip tells you it could not open the '/data/ref/mm9/bwa/mm9.fa.bwt' file. Check the existence of the file in this path. Once the source of the error is determined, modify the Targets.txt file accordingly and move on.

4.6 Advanced settings

4.6.1 AdvancedSettings.txt

Users can modify advanced settings (map, filter, etc) in the AdvanceSettings.txt file found in the DataStructure directory.

Would users decide to modify the Tophat aligner command (# *Tophat.command.line*), … .

4.6.2 QSUB parameters

The qub header can be modified to meet the requirements of specific clusters, including queuing times, nodes, number of CPUs, etc. If this is of interest, please modify the QSUB_header.sh template file in *./RNApip/scripts/QSUB_header.sh* such as to apply personalized settings to all jobs (this needs to be only once). Please refer to your computer core facility systems administrator for further details.

5 NEAT part 2 : RNAmE

5.1 Introduction

RNAmE is a package of two applications. Its main goal is to accompany users during the analysis of next generation sequencing (NGS) data as part of the NEAT toolkit (NGS easy analysis toolkit). RNApip, in conjuncture with the NEAT package, provides an easy, rapid and reproducible exploratory data analysis (EDA) tool that allows users to assess their data in as few as a couple. The primary goal of RNAmE is to provide differential gene expression analysis, RPKM and coun tables. RNAmE has been developed as a downstream analysis tool for NGS data that has been analyzed using the NEAT (ChIPpip) package.

RNAmE runs through the MacOS automator software. This software should be installed by default on most modern Apple computers. Please ensure this is installed on your computer in the *applications* directory. In addition, ChIPmE is an R script that requires a few widely recognized R packages. For details on these, please refer to the *Version information and required packages* section below.

5.2 Before running RNAmE

Please make sure all R packages required for RNAmE are installed on your computer (refer to the R manual) prior to running RNAmE. Refer to the *Version information and required packages* section below for more information.

Finally, make sure your *Terminal* software is closed before launching RNAmE.

5.3 Running NEAT Part 2.1 (RNAmE)

5.3.1 Step 1: Download a RNApip project

To transfer an RNApip project from a remote server to a local computer, double click the RNAmE *Transfer* app.

This app is can be found in the NEAT directory *~/Desktop/NEAT/RNAmE/*. Users will be asked to locate the NEAT directory and where they want to save their RNApip project. In this example, the NEAT directory is on our desktop and we will save our RNApip project on the desktop as well.

The next step is to provide the path to the RNApip directory on the remote server. In the RNApip tutorial, we had saved our project in the RNApip directory (*/HOME/RNApip/EXAMPLE*), so this is the path we will enter.

Finally, RNAmE will prompt you to enter your SSH information. Once again, following the RNApip tutorial, we will enter: *username@server-address.edu*.

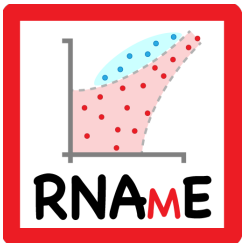
Starting the download will launch the *Terminal* to open and start running R. It will require users to enter their password several times (for each call to the remote server). Please follow this process as failing to enter your password will break the connection after some time.

Downloading an entire project should not take more than a few minutes.

5.4 Running NEAT Part 2.2 (RNAmE)

5.4.1 Step 2: Run a RNAmE analysis

Once the project is downloaded, users can run the proper RNAmE analysis. To this end, double click the RNAmE *Analyze* app.



Users will be asked to locate the RNApip folder (the one you just downloaded). In our case, the RNApip project was downloaded to the desktop.

Users will then have to locate the NEAT folder (also on desktop in this example). RNAmE will use the databases specified in the Targets.txt file. In our example, we are using the TxDb.Mmusculus.UCSC.mm9.knownGene database. This comprises all known transcripts from UCSC and is a good starting point for initial EDA. Please note here that care should be brought to match the database objects with the reference genome initially used. In our case, our data was mapped to the mouse mm9 genome, hence the TxDb.Mmusculus.UCSC.mm9.knownGene. Several parameters can be set before running the analysis including *topGenes* and *toHighlight*, which correspond to the number of top DEG users want ot highlight in

the tables and on the graphs. Values are set as a reference, but we suggest users experiment to find the best values for their own need.

Running the analysis using the test data should take less than a minute.

5.5 Outputs

5.5.1 Logs

Each time RNAmE is run, a log file is created and named using the date and time. This file is save in the *./EXAMPLE/logs/* directory. We strongly encourage users to initially look at these files, as any error that might have occurred will be saved there. Usually, if no error is prompted from the terminal, RNAmE has terminated correctly. Also, please note that if there are unrecognized chromosome names such as random chromosomes, warning messages will appear. In the test data, there are 50 or more warnings. These can usually be disregarded.

5.5.2 Count tables and RPKM tables

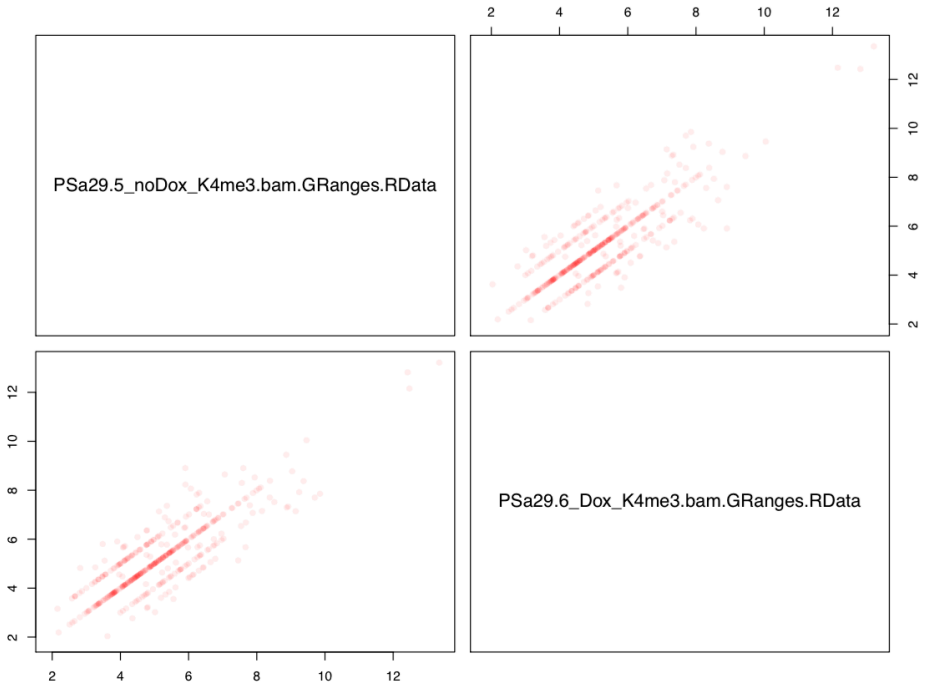
RNAmE will generate count tables that should be self-explanatory. In brief, rows correspond to genes and columns correspond to samples.

5.5.3 Differentially expressed gene plots

RNAmE generates pdf plots that are saved in */EXAMPLE/plots/*. General smear plots are created as a quick EDA between samples and replicates. In additional to smear plots, RNAmE generates all possible combinations of QQplots, with highlighted up and downregulated genes. More information on these genes can be found in the

corresponding count tables. Finally, RNAmE generates gene lists meant to be fed to Gorilla, a well-established GO term EDA software. Venn diagrams are produced if selected. Users should note that computational time increases exponentially with the number of datasets as all combinations of Venn plots are created.

The smear plots below are an example of the results obtained using the test data. Obviously, the limited number of reads in the test data makes this plot relatively uninformative, but analyzing your own data will give much better resolution.



5.6 Advanced settings

5.6.1 Bam files and GRanges

Once Granges objects have been created, bam files are no longer required locally. Users are thus free to delete these files as they are often two orders of magnitude larger than GRanges objects (respectively several Gb vs tens of Mb). We do suggest that users backup their .bam files on the remote server.

5.6.2 Consolidating projects

Consolidating projects is very easy. Users who intend to do so will simply need to change the *Targets.txt* file and copy-paste the Granges objects (or bam files) from one folder to the other. Other files and folder can be left as is.

6 Version information and required packages

Program: bwa (alignment via Burrows-Wheeler transformation)
Version: 0.5.9-r16

Program: samtools (Tools for alignments in the SAM format)
Version: 0.1.18 (r982:295)

R version 3.1.0 (2014-04-10)
Platform: x86_64-redhat-linux-gnu (64-bit)

R packages (with dependencies) required to run RNApip:

- systemPipeR (systemPipeR_0.99.0)
- …

R packages (with dependencies) required to run RNAmE:

- Rsamtools
- GenomicRanges
- GenomicAlignments
- caTools
- VennDiagram

sessionInfo()

R version 3.1.2 (2014-10-31)

Platform: x86_64-apple-darwin10.8.0 (64-bit)

locale:

[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:

[1] grid parallel stats4 stats graphics grDevices utils

[8] datasets methods base

other attached packages:

[1] VennDiagram_1.6.9 caTools_1.17.1 GenomicAlignments_1.2.1

[4] Rsamtools_1.18.2 Biostrings_2.34.1 XVector_0.6.0

[7] GenomicRanges_1.18.3 GenomeInfoDb_1.2.3 IRanges_2.0.1
[10] S4Vectors_0.4.0 BiocGenerics_0.12.1

loaded via a namespace (and not attached):
[1] base64enc_0.1-2 BatchJobs_1.5 BBmisc_1.8 BiocParallel_1.0.0
[5] bitops_1.0-6 brew_1.0-6 checkmate_1.5.0 codetools_0.2-9
[9] DBI_0.3.1 digest_0.6.4 fail_1.2 foreach_1.4.2
[13] iterators_1.0.7 RSQLite_1.0.0 sendmailR_1.2-1 stringr_0.6.2
[17] tools_3.1.2 zlibbioc_1.12.0

7 Reported bugs

No bugs reported so far.

8 Funding

This pipeline was developed with funding from the Swiss National Science Foundation.

9 References

- ...