

Manual of the **MATLAB** pipeline for the analysis of genomic time course data from the *Gene Expression Omnibus* (Version 1.42)

January 11, 2017

Contents

1	The Pipeline for Dynamic Gene Expression Data	2
2	Theoretical basis and terminology	3
2.1	The <i>Gene Expression Omnibus</i> (<i>GEO</i>)	3
2.2	GEO Series	3
2.3	GEO Samples	4
2.4	GEO Platforms	4
2.5	Experimental conditions	4
2.6	Study	4
2.7	Experimental macroconditions	5
2.8	Replicates	5
2.9	Genes' between-replicate noise	5
3	Usage of Pipeline4DGEData	5
4	Input files	11
4.1	Condition files	11
4.2	Macrocondition files	12
5	Output of results	13
6	Guided creation of condition files	13

7	<code>compare.m</code>	14
8	<code>measure_fit_of_replicates.m</code>	14
9	Output of the pipeline	15
10	Simulation of time course data	15

List of Figures

1	Samples associated to series GSE59015.	6
2	Table of platform GPL17880 showing that the gene names are stored in the 7-th column (titled 'ORF').	8
3	Figure 3a shows the contents of the condition file for condition Wildtype and Figure 3b shows the contents of the condition file for condition D10.	8
4	9
5	9
6	9
7	10
8	10
9	10
10	Calling <code>create_input_files</code> to create condition files.	13
11	Entering the accession number for the creation of the condition files.	14
12	Entering the accession number for the creation of the condition files.	14
13	Folder hierarchy of results.	16

List of Tables

1 The Pipeline for Dynamic Gene Expression Data

The *Pipeline for Dynamic Gene Expression Data* (Pipeline4DGEData) is the implementation of the pipeline proposed by Carey et al. (2017). It is

written in MATLAB (R2016a). The program is designed to analyze time course data from the *Gene Expression Omnibus* (GEO) in order to discover gene regulatory networks from high-throughput data.

The system requirements are listed below.

- MATLAB R2016a or newer.

The latest version of the source files can be downloaded from the official GitHub repository. These are organized in a downloadable folder named by default as `Pipeline4DGEData`. User-defined input for new analyses to be run must be provided in subfolder `Input` (details in Section 4) and the program returns the results in subfolder `Output` (details in Section 5).

Section 2 provides a very brief summary of the theoretical concepts involved in the usage of `Pipeline4DGEData`. However, the reader is strongly encouraged to refer to the GEO documentation and the work of Carey et al. (2017) for full details. Section 3 illustrates the usage of `Pipeline4DGEData`. This example should be used as basis for other analyses to be performed by the reader.

2 Theoretical basis and terminology

2.1 The *Gene Expression Omnibus* (GEO)

The *Gene Expression Omnibus* (GEO)¹ is a public repository administered by the *National Center for Biotechnology Information* (NCBI)² for the storage of high throughput data. The GEO stores microarray data in MIAME compliant formats and makes it publicly available. The data is organized into series, samples and platforms.

2.2 GEO Series

A data series is a set of related samples. It may also provide information of the investigation associated to the contained samples. The GEO identifies each series by an accession number with the prefix ‘GSE’. For example, series GSE59015.

¹<https://www.ncbi.nlm.nih.gov/geo/>

²<https://www.ncbi.nlm.nih.gov>

2.3 GEO Samples

The data obtained from an experimental sample. It also describes how the experiment was conducted and how the data was handled or manipulated. The GEO identifies each sample by an accession number with the prefix ‘GSM’. For example, samples `GSM1424445` and `GSM1424453`, which are associated to series `GSE59015`.

2.4 GEO Platforms

A description of the microarray used to collect a sample. The GEO identifies each platform by an accession number with the prefix ‘GPL’. For example, platform `GPL17880`, which is associated to the samples in series `GSE59015`.

2.5 Experimental conditions

An experimental condition is the set of time-independent characteristics or parameters of a sample and/or the circumstances in which it was taken. Usually the main characteristic differentiating a condition from a related one is a stimulus (or lack of), which normally consists of a treatment of viral infection. Depending on the study, other characteristics may be present. These may include, the subjects gender or cell type used, for instance. The experimental condition excludes the time point, if any.

In series `GSE59015` there are two experimental conditions, namely `Wildtype` and `D10`, each one comprising eight samples.

2.6 Study

A study is a set of samples that refer to a common domain of experimental conditions. A common domain of experimental conditions may consist of, for instance, the presence of a stimulus (treatment or viral infection) and the absence of it. In most cases, the series and study coincide. In these cases, all the samples contained in the series belong to the same study. However, it is possible that the GEO stores series that encompass samples from varying domains. In these cases, the samples contained in the series are grouped into different studies.

2.7 Experimental macroconditions

2.8 Replicates

Replicates can be *biological* or *technical*.³

2.9 Genes' between-replicate noise

Genes may exhibit variation across replicates from the same condition. This variation, referred to as the *gene's between-replicate noise* and denoted by δ , is measured for each gene as follows. Let $x_{i,t}^s$ be the expression level of replicate s 's i -th gene at time t and let S be the total number of replicates. Let $\sigma_{i,t}$ be the standard deviation of the expression level of the i -th gene at time t across the S replicates. The gene i 's between-replicate noise, denoted by δ_i ,⁴ is given by

$$\delta_i = \frac{1}{T} \sum_{t \in t_1}^{t_T} \sigma_{i,t}. \quad (1)$$

Section 8 introduces the **MATLAB** function provided for the calculation of the genes' between-replicate noise from a group of replicates.

3 Usage of Pipeline4DGEData

This section shows how to use **Pipeline4DGEData** for the analysis of a set of experimental conditions by illustrating its usage on GEO series **GSE59015**. Application of **Pipeline4DGEData** to any other GEO series should be analogous.

GEO series **GSE59015** refers to a study about the resistance of *Plasmodium falciparum* (*i.e.*, the malaria parasite) to antimalarial drugs for the design of new medications. Two experimental conditions (Section 2.5 provides

³Biological Replicates vs Technical Replicates.

⁴Another possibility is to define δ_i in terms of the coefficient of variation instead of the standard deviation. In other words, the between-replicate noise could be defined as $\delta_i = \frac{1}{T} \sum_{t \in t_1}^{t_T} \frac{\sigma_{i,t}}{\mu_{i,t}}$, where $\mu_{i,t}$ is the mean of the expression level of the i -th gene at time t across the S replicates. However, this measure is unreliable given that expression levels are interval scale measures and the coefficient of variation is reliable only in ratio scale measures.

GSM1424445 Wildtype 0 hours post invasion
 GSM1424446 Wildtype 6 hours post invasion
 GSM1424447 Wildtype 12 hours post invasion
 GSM1424448 Wildtype 18 hours post invasion
 GSM1424449 Wildtype 24 hours post invasion
 GSM1424450 Wildtype 30 hours post invasion
 GSM1424451 Wildtype 36 hours post invasion
 GSM1424452 Wildtype 42 hours post invasion
 GSM1424453 D10 Δ IDH/ Δ KDH 0 hours post invasion
 GSM1424454 D10 Δ IDH/ Δ KDH 6 hours post invasion
 GSM1424455 D10 Δ IDH/ Δ KDH 12 hours post invasion
 GSM1424456 D10 Δ IDH/ Δ KDH 18 hours post invasion
 GSM1424457 D10 Δ IDH/ Δ KDH 24 hours post invasion
 GSM1424458 D10 Δ IDH/ Δ KDH 30 hours post invasion
 GSM1424459 D10 Δ IDH/ Δ KDH 36 hours post invasion
 GSM1424460 D10 Δ IDH/ Δ KDH 42 hours post invasion

Figure 1: Samples associated to series GSE59015.

an introduction to experimental conditions) are considered: **Wildtype** and **D10**. The latter refers to *P. falciparum* cells that have been exposed to a medication and the former refers to a cells that haven't been exposed to any stimulus and thus serves as the control group. The list of samples is shown in Figure 1.

The following checklist provides the information that is required for running a pipeline analysis. This information should be compiled before starting `Pipeline4DGEData`.

1. The accession number of the GEO series.
 - In this example, this number is **GSE59015**.
2. The experimental conditions on which the analysis is to be performed and their corresponding GEO samples. And identifying name should be chosen in advance for each condition.
 - In this example, the conditions' names and their samples are as follows.
 - Condition **Wildtype** comprising samples **GSM1424445** through **GSM1424452**, as shown in Figure 1.

- Condition D10 comprising samples GSM1424453 through GSM1424460, as shown in Figure 1.
3. The condition files for the conditions on which the analysis is to be performed (Section 4.1 provides an introduction to condition files). These files must be located in folder **Input** and named as specified in Section 4.
 - In this example, the files for the two conditions must be as shown in Figure 3.
 4. The accession number of the GEO platform associated to the GEO series.
 - In this example, this number is GPL17880.
 5. The index of the column in the GPL record where the gene names are stored.
 - In this example, this index is 7, as shown in Figure 2.

Once the above is ready, the analysis can be started by running **pipeline** from the **MATLAB** console, as shown in Figure 4. The program will then prompt the user to enter the index of the column in the GPL record where the gene names are located (Item 5 in the checklist above). In this example, this index is 7 and must be entered as shown in Figure 5. After this the program will proceed to read and match all the probe ids to the corresponding gene names, informing the user that the process can take some time and notifying the progress as shown in Figure 6. When finished, the program will prompt the user to specify the gene ID type used in the dataset by selecting from a predefined list displayed on the **MATLAB** console. In this example, the type is **AGILENT_ID** and thus 5 must be entered, as shown in Figure 7. After this, the program will start the analysis for ALL the condition files present in folder **Input**. In this example, the program starts with condition D10 as shown in Figure 8. When the analysis has been completed for all the experimental conditions, the program will confirm that the process has been finished successfully, as shown in Figure 9. The results are output to folder **Output** and organized as explained in Section 5.

Data table								
ID	COL	ROW	NAME	SPOT_ID	CONTROL_TYPE	ORF	CHROMOSOMAL_LOCATION	DESCRIPTION
1	96	164	GE_BrightCorner	GE_BrightCorner	pos			
2	96	162	DarkCorner	DarkCorner	pos			
3	96	160	DarkCorner	DarkCorner	pos			
4	96	158	PF14_0264_v7.1_P6/7	PF14_0264_v7.1_P6/7	FALSE	PF14_0264	unmapped	protein kinase, p
5	96	156	PF13_0192_v7.1_P2/2	PF13_0192_v7.1_P2/2	FALSE	PF13_0192	unmapped	conserved Plasm
6	96	154	PFF0150C_v7.1_P2/2	PFF0150C_v7.1_P2/2	FALSE	PFF0150c	unmapped	conserved Plasm
7	96	152	MAL13P1.133_v7.1_P5/16	MAL13P1.133_v7.1_P5/16	FALSE	MAL13P1.133	unmapped	conserved Plasm
8	96	150	PFF0630C-A_v7.1_P1/1	PFF0630C-A_v7.1_P1/1	FALSE	PFF0630c-a	unmapped	conserved Plasm
9	96	148	MAL13P1.264_v7.1_P1/2	MAL13P1.264_v7.1_P1/2	FALSE	MAL13P1.264	unmapped	conserved Plasm
10	96	146	PF11_0465_v7.1_P2/3	PF11_0465_v7.1_P2/3	FALSE	PF11_0465	unmapped	dynamn-like prot
11	96	144	PF14_0264_v7.1_P4/7	PF14_0264_v7.1_P4/7	FALSE	PF14_0264	unmapped	protein kinase, p
12	96	142	PFF0525W_v7.1_P1/1	PFF0525W_v7.1_P1/1	FALSE	PFF0525w	unmapped	conserved Plasm
13	96	140	PFE0360C_v7.1_P2/3	PFE0360C_v7.1_P2/3	FALSE	PFE0360c	unmapped	conserved Plasm
14	96	138	PFF0765C_v7.1_P3/4	PFF0765C_v7.1_P3/4	FALSE	PFF0765c	unmapped	conserved Plasm
15	96	136	PF13_0155_v7.1_P6/8	PF13_0155_v7.1_P6/8	FALSE	PF13_0155	unmapped	conserved Plasm
16	96	134	PFA0160C_v7.1_P2/2	PFA0160C_v7.1_P2/2	FALSE	PFA0160c	unmapped	nucleoside transp
17	96	132	PFC0875W_v7.1_P6/9	PFC0875W_v7.1_P6/9	FALSE	PFC0875w	unmapped	ABC transporter,
18	96	130	PF14_0684_v7.1_P2/2	PF14_0684_v7.1_P2/2	FALSE	PF14_0684	unmapped	conserved Plasm
19	96	128	PF11_0288_v7.1_P1/2	PF11_0288_v7.1_P1/2	FALSE	PF11_0288	unmapped	conserved Plasm

Figure 2: Table of platform GPL17880 showing that the gene names are stored in the 7-th column (titled ‘ORF’).

0 hours,GSM1424453
6 hours,GSM1424454
12 hours,GSM1424455
18 hours,GSM1424456
24 hours,GSM1424457
30 hours,GSM1424458
36 hours,GSM1424459
42 hours,GSM1424460

(a) Condition file for condition Wildtype.

0 hours,GSM1424445
6 hours,GSM1424446
12 hours,GSM1424447
18 hours,GSM1424448
24 hours,GSM1424449
30 hours,GSM1424450
36 hours,GSM1424451
42 hours,GSM1424452

(b) Condition file for condition D10.

Figure 3: Figure 3a shows the contents of the condition file for condition Wildtype and Figure 3b shows the contents of the condition file for condition D10.

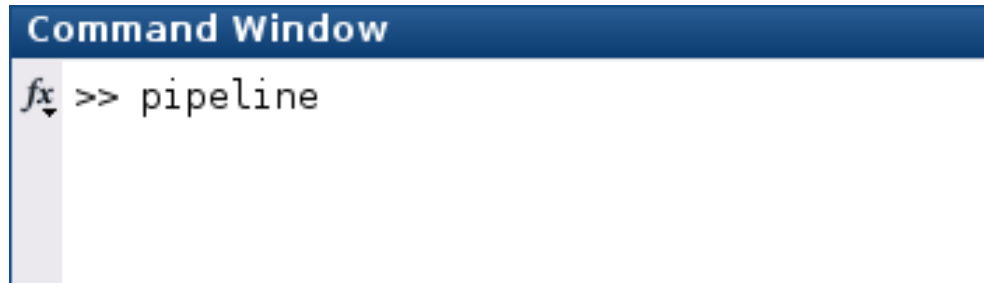


Figure 4

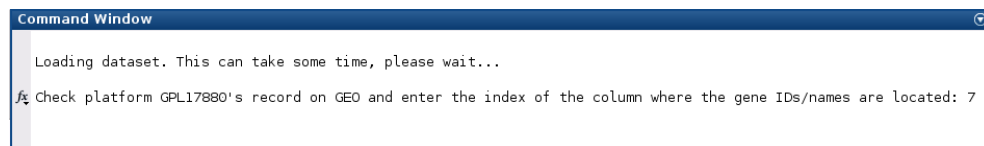


Figure 5

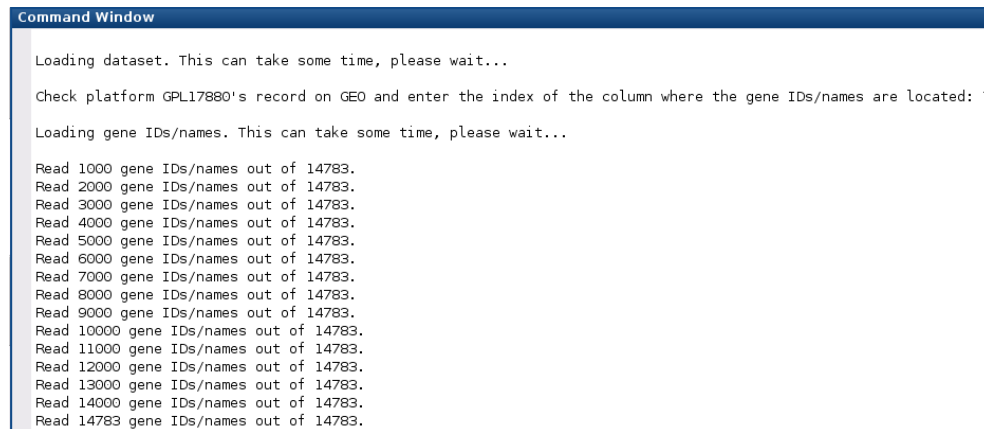


Figure 6

```
Command Window
1: AFFYMETRIX_3PRIME_IVT_ID
2: AFFYMETRIX_EXON_GENE_ID
3: AFFYMETRIX_SNP_ID
4: AGILENT_CHIP_ID
5: AGILENT_ID
6: AGILENT_OLIGO_ID
7: ENSEMBL_GENE_ID
8: ENSEMBL_TRANSCRIPT_ID
9: ENTREZ_GENE_ID
10: FLYBASE_GENE_ID
11: FLYBASE_TRANSCRIPT_ID
12: GENBANK_ACCESSION
13: GENOMIC_GI_ACCESSION
14: GENPEPT_ACCESSION
15: ILLUMINA_ID
16: IPI_ID
17: MGI_ID
18: PFAM_ID
19: PIR_ID
20: PROTEIN_GI_ACCESSION
21: REFSEQ_GENOMIC
22: REFSEQ_MRNA
23: REFSEQ_PROTEIN
24: REFSEQ_RNA
25: RGD_ID
26: SGD_ID
27: TAIR_ID
28: UCSC_GENE_ID
29: UNIGENE
30: UNIPROT_ACCESSION
31: UNIPROT_ID
32: UNIREF100_ID
33: WORMBASE_GENE_ID
34: WORMPEP_ID
35: ZFIN_ID

Enter the type of gene ID used in the study associated to GSE59015 (e.g., enter 9 for ENSEMBL_GENE_ID): 5
```

Figure 7

```
Command Window

Enter the type of gene ID used in the study associated to GSE59015 (e.g., enter 9 for ENSEMBL_GENE_ID): 5

The analysis of condition "D10" is starting.
```

Figure 8

```
Command Window

The analysis of condition "D10" is starting.

The analysis of condition "D10" has been completed.

Results have been output to folder /home/jramirez/Pipeline4DGEData/Results/GSE59015/Conditions/D10/

Loading dataset. This can take some time, please wait...

The analysis of condition "Wildtype" is starting.

The analysis of condition "Wildtype" has been completed.

Results have been output to folder /home/jramirez/Pipeline4DGEData/Results/GSE59015/Conditions/Wildtype/

The analysis is complete for all the subjects/conditions.

All results from dataset GSE59015 have been output to folder /home/jramirez/Pipeline4DGEData/Output/GSE59015/Conditions/

A consolidated report on all conditions from dataset GSE59015 can be found in /home/jramirez/Pipeline4DGEData/Output/GSE59015/paper.pdf

>> |
```

Figure 9

4 Input files

4.1 Condition files

Condition files are the input files for the pipeline analysis of one or more experimental conditions (See Section 2.5 regarding experimental conditions). The information of each experimental condition must be provided in a text file. One file for each condition. The condition files must be stored in folder **Input**. Each condition file must be named with the following format: [GEO SERIES NUMBER]__[NAME OF EXPERIMENTAL CONDITION]__[NUMBER OF TOP DRGs FOR CLUSTERING].txt.

All the samples associated with the experimental condition must be provided along with the time points using the format described as follows. Each line of the file must consist of the time point, including the time unit (e.g., 2 hours), followed by a comma (,) and then followed by the accession number of one sample. For example, the file for condition “D10” of *GEO* series GSE59015 must be named GSE59015_-D10-_3000.csv and the contents must be as follows.

```
0 hours,GSM1424453
6 hours,GSM1424454
12 hours,GSM1424455
18 hours,GSM1424456
24 hours,GSM1424457
30 hours,GSM1424458
36 hours,GSM1424459
42 hours,GSM1424460
```

The condition file can group different replicates that represent the same experimental condition. In this case the pipeline will run one single analysis with the average of all the replicates specified in the file. For example, the file for all the replicates in *GEO* series GSE41067 must be named GSE41067_-ALL-_10000.csv and the contents must be as follows.

```

0 hours,GSM1008154,GSM1008162,GSM1008170
1 hours,GSM1008155,GSM1008163,GSM1008171
2 hours,GSM1008156,GSM1008164,GSM1008172
4 hours,GSM1008157,GSM1008165,GSM1008173
6 hours,GSM1008158,GSM1008166,GSM1008174
8 hours,GSM1008159,GSM1008167,GSM1008175
10 hours,GSM1008160,GSM1008168,GSM1008176
12 hours,GSM1008161,GSM1008169,GSM1008177

```

The condition files can be written manually, following strictly the format described above. Optionally, this task can be carried out more easily by using script `create_input_files.m`. In order to do this, `create_input_files.m` must be run and the instructions provided thereafter by the program must be followed. More details in Section 6.

4.2 Macrocondition files

The conditions comprising the macrocondition must be provided in a text file that must be located in folder `Input`. This file must be named using the following format: `[GEO SERIES NUMBER]_--[NAME OF MACRO CONDITION].txt`. For example, the following are the contents of a macrocondition of five H3N1 subjects from GSE52428.

```

H3N1_001
H3N1_002
H3N1_003
H3N1_004
H3N1_005

```

This file can be constructed manually, or with BASH script `prepare_input.sh`. This option requires running the script with the following syntax.

```
./prepare_input.sh [GEO SERIES] [NAME OF MACRO CONDITION].txt
```

Example

```
./prepare_input.sh GSE52428 H1N1.txt
```

The above will read ALL the conditions whose analyses have been completed for the GEO series indicated and write the file with the format described earlier.

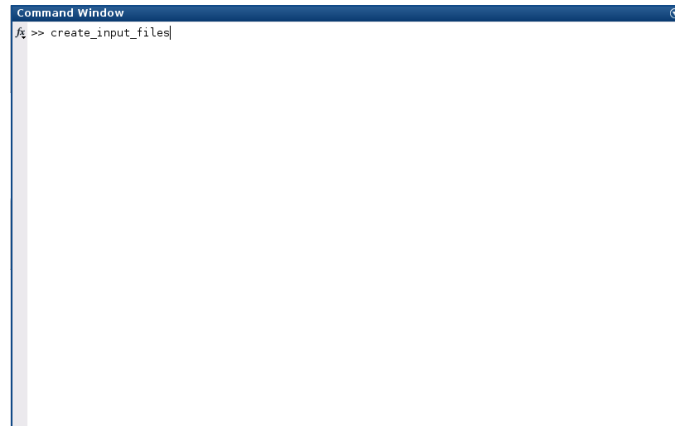


Figure 10: Calling `create_input_files` to create condition files.

5 Output of results

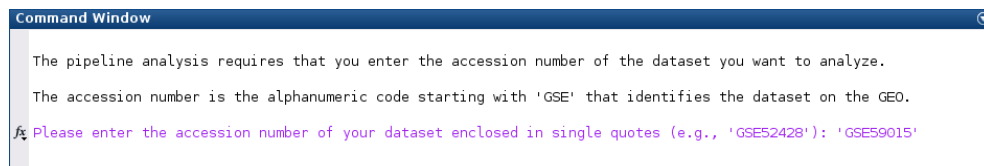
`Pipeline4DGEData` outputs all of its results to subfolder `Output`. Each pipeline analysis is carried out on an experimental condition and results from several conditions are grouped in folder by the accession number of the associated GEO series.

6 Guided creation of condition files

The (experimental) condition files for the pipeline analysis can be created manually with the format described in Section 4.1 on page 11 or with the help of script `create_input_files.m`. In order to do the latter, the command `create_input_files` must be executed on the `MATLAB` console from the pipeline directory.

Prior to running the pipeline analysis for the two conditions (Subsection 4.1 provides an introduction to conditions files), the corresponding condition files must be created. In order to do this, `MATLAB` must be started inside folder `Pipeline4DGEData/` and in the console the command `create_input_files` must be called, as shown in Figure 10 and then pressing **Enter**.

The program will then request the accession number of the GEO series on which the analysis is to be performed. This information must be entered as single quotes (`'`), as shown in Figure 12. In this example, the accession number entered is `GSE59015`.



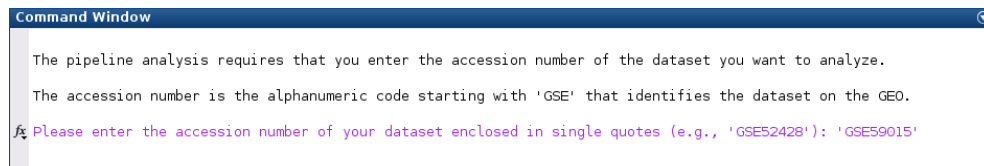
```
Command Window

The pipeline analysis requires that you enter the accession number of the dataset you want to analyze.

The accession number is the alphanumeric code starting with 'GSE' that identifies the dataset on the GEO.

Please enter the accession number of your dataset enclosed in single quotes (e.g., 'GSE52428'): 'GSE59015'
```

Figure 11: Entering the accession number for the creation of the condition files.



```
Command Window

The pipeline analysis requires that you enter the accession number of the dataset you want to analyze.

The accession number is the alphanumeric code starting with 'GSE' that identifies the dataset on the GEO.

Please enter the accession number of your dataset enclosed in single quotes (e.g., 'GSE52428'): 'GSE59015'
```

Figure 12: Entering the accession number for the creation of the condition files.

After hitting **Enter**, the program will read the series data from the GEO and will request the index of the column where the gene names are located in the platform record.

7 `compare.m`

`compare.m`: input is a macrocondition file. See Subsection 4.2.

8 `measure_fit_of_replicates.m`

This is a script that calculates the between-replicate noise for all genes across a number of replicates, as shown in Subsection 2.9. Its input is a macrocondition file, as defined in Subsection 4.2. Results will be output in folder `Pipeline_HOME/Output/[GSE Series]/Comparison_of_replicates/[Macrocondition]/`.

This folder contains a file named `[Macrocondition]_noise_per_gene_ALL_GENES.csv`. Each line of this file corresponds to a probe in the GSE matrix and contains three values separated by commas (,). The first field is the probe ID, the second is the gene name, and the third is the gene's between-replicate noise. The probes are listed in this file in the same order they appear in the GSE matrix. A secondary file, named `[Macrocondition]_noise_per_gene_ALL_GENES_SORTED_BY_NOISE.csv`,

provides the same information with the probes listed by their between-replicate noise in decreasing order. That is to say, probes that appear on top are the ones whose expression is most consistent across the probes provided in the macrocondition file whereas those at the bottom are the ones exhibiting the most discrepancy in expression across these replicates.

9 Output of the pipeline

All results are output to folder `Output`. Results are grouped by the *GEO* series' accession number and by the name of the experimental condition, as illustrated in Figure 13.

10 Simulation of time course data

The robustness of the pipeline process can be validated by simulating gene expression data and verifying if the pipeline returns the same model the simulated data originate from. More specifically, expression data can be simulated from given a GRN and set of GRMs and given as input to the pipeline in order to verify that the algorithm approximates the original GRN and GRMs. In order to achieve this, two types of the gene expression curves need to be generated: those from DRGs and those from non-DRGs. First, the expression curves of the DRGs are simulated from the mean curves of the GRMs as follows. Let K be the matrix representing the expression of one gene response module composed of n genes over t time points, given by

$$K = \begin{pmatrix} X_{1,1} & X_{1,2} & \cdots & X_{1,t} \\ X_{2,1} & X_{2,2} & \cdots & X_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n,1} & X_{n,2} & \cdots & X_{n,t} \end{pmatrix}.$$

Also let M be the tuple of per-column means of K , given by

$$M = (\mu_1 \quad \mu_2 \quad \cdots \quad \mu_t),$$

and let S be the per-column standard deviation of K , given by

$$S = (\sigma_1 \quad \sigma_2 \quad \cdots \quad \sigma_t).$$

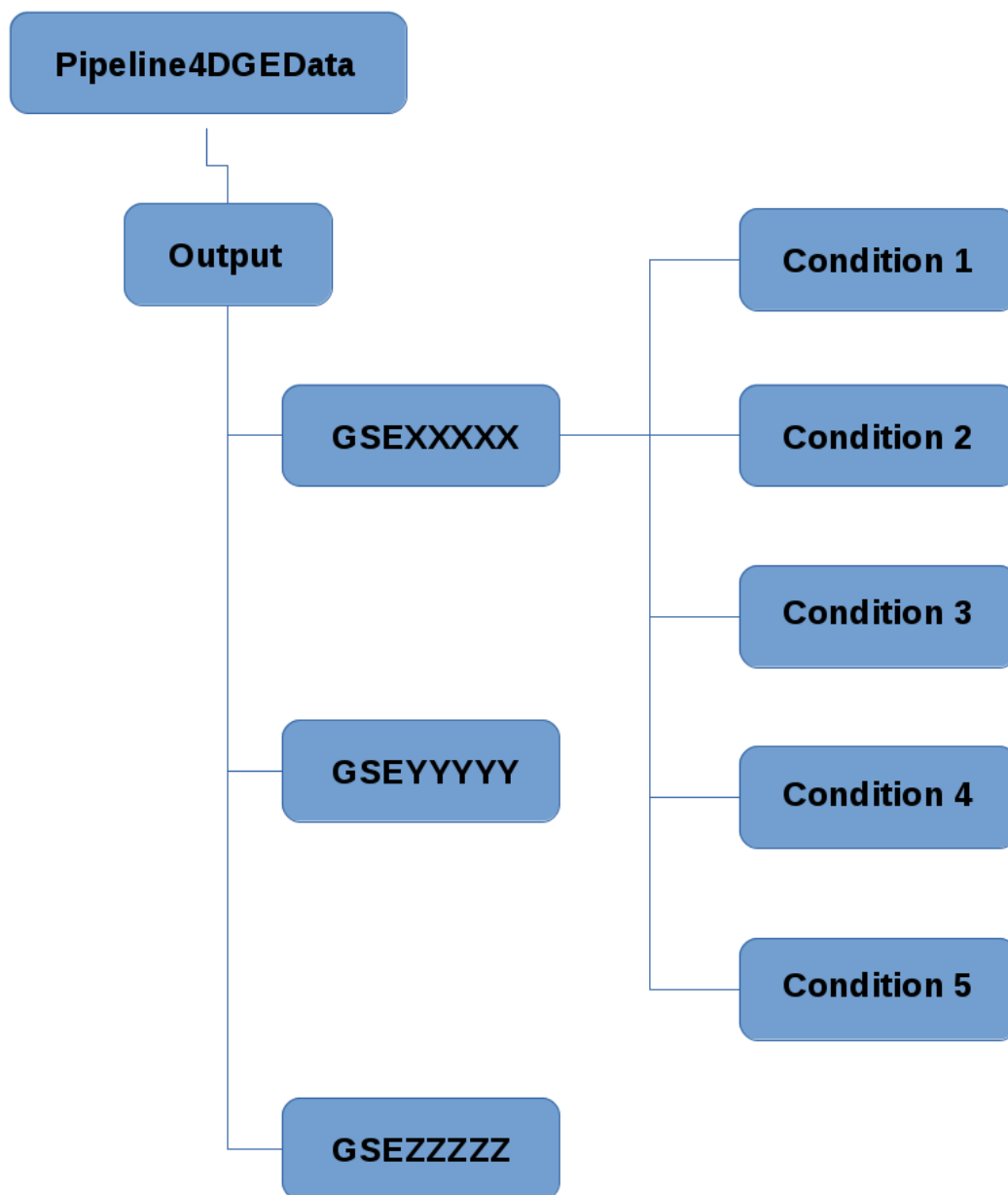


Figure 13: Folder hierarchy of results.

Then n simulated curves are obtained where the i -th element is a normal deviate of μ_i with standard deviation σ_i . This process is repeated for all the GRMs until obtaining the simulated expression curves of all the DRGs. Thereafter, the non-DRG curves are simulated as normal deviates of the per-time point means of all the DRG curves, with standard deviation 1.0. The simulated DRG and non-DRG expression curves can then be passed as arguments to the pipeline and see if the GRMs and GRN obtained approximate the originals.

The procedure described above is performed by script `run_simulation.m`, which reads the results of a previously-completed analysis from folder `Output`, constructs the simulated data and runs the pipeline in folder `Output/[Condition]_simulated`, where `[Condition]` is the name of the original experimental condition.

References

- Carey, Michelle, Ramirez, Juan Camilo, Wu, Shuang, and Wu, Hulin, 2017. A Big Data Pipeline: Identifying Dynamic Gene Regulatory Networks from Time Course GEO Data with Applications to Influenza Infection. *Statistical Methods in Medical Research*, 1–14.