STALICLA R&D

Overview of the PBPM protocol

Gabriela Martínez

September 2019

The PBPM protocol

A command line interface program built in Python 3 to generate Patient by Disrupted Pathways Matrices (PBPMs). It generates three types of matrices:

- **Binary PBPM:** unique patient identifiers as rows and pathways as columns. Values are binary (1 or 0) indicators for each column when a mutation was found for any gene in that pathway for a patient.

- **Numerical PBPM:** unique patient identifiers as rows and pathways as columns. Values are the number of mutations found for each pathway and patient.

- **Normalized numerical PBPM:** unique patient identifiers as rows and pathways as columns. Values are the number of mutations found for each pathway and patient, normalized considering the number of genes of each pathway.

How to execute this protocol?

This module is written in Python 3 and uses up-to-date libraries for this version as well.

To run the main module in a Linux environment, simply open a new terminal and call the script together with the arguments it accepts:

```
python3 main.py
positional arguments:
  [inputFile] [pathwaysDirectory]
optional arguments:
  [-p PATHWAY, --pathway PATHWAY, --pathways PATHWAY]
  [-g GENE, --gene GENE, --genes GENE]
  [-id PATIENT, --patient PATIENT, --patients PATIENT]
  [-csq CSQ, --consequence CSQ, --consequences CSQ]
  [-pli PLI, --pli_gt PLI]
  [-pr pRECESSIVE, --pr_g pRECESSIVE]
  [-af MAX_AF, --af_lt MAX_AF]
  [-pph2 PPH2, --polyphen PPH2, --polyphen2 PPH2]
  [-mpc MPC, --mpc_gt MPC]
  [-adj_csq ADJ-CSQ, --adjusted_consequence ADJ-CSQ, --adjusted_consequences ADJ-CSQ]
  [-m PBPM-TYPE, --matrix PBPM-TYPE]
  [-im True/False(default), --intermediate_matrix True/False(default)]
  [-a APPEND-FILE, --append APPEND-FILE]
  [--path PATH-RESULTS]
```

PBPM protocol arguments

Mandatory parameters

- Input file containing patients and mutations.
- Folder containing pathways and their genes.

Optional parameters: act as filters

- Pathways.
- Genes.
- Patients.
- Consequences.
- pLI.
- pRecessive.
- Max control AF.
- PolyPhen2 prediction label.
- MPC.
- Adjusted consequence.

The protocol accepts a explicit subset of them and also external files

Note: help() is available for all the parameters via the command line.

PBPM mandatory parameters: input file

The input file containing patients and mutations has the following columns*. Moreover, only observations with a valid **HGNC_symbol** will be considered.

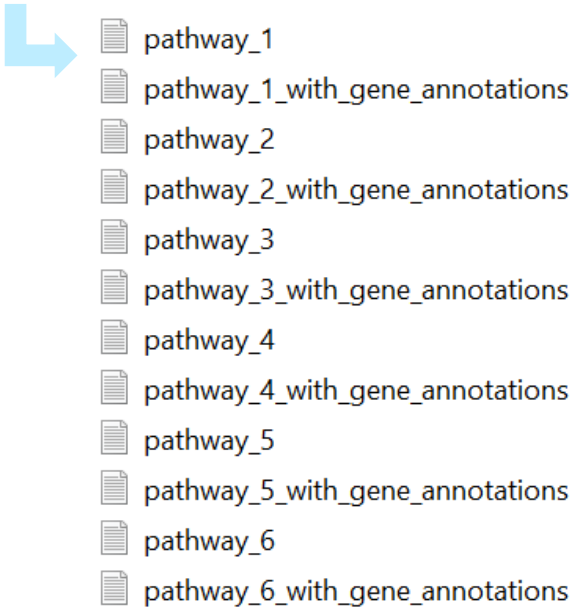
Chr	Position	…	child_id	consequence	HGNC_symbol	constraint_score	…	pLI	MPC
11	881802		Patient_10	missense_variant	V595I	0.018		0.14	1.7
12	686702		Patient_19	Near_3splice	NA	0.651		0.21	NA
3	981711		Patient_21	Intron	NA	0.006		0.35	NA
15	1121834		Patient_31	Silent	R589R	0.124		0.05	NA
20	541255		Patient_98	missense_variant	L239P	0.032		0.12	3.5

*and many more. This is not exhaustive.

PBPM mandatory parameters: pathways folder

The sample folder to build the PBPM protocol contained 80 pathways, for which the files ended with **"_gene_annotations"** were considered

Folder structure



Pathway_1	HGNC_symbol	…
SCN4A	SCN4A	
ANK1	ANK1	
SPTB	SPTB	
KCNQ2	KCNQ2	
ACTB	ACTB	

How to specify mandatory parameters?

Run the protocol by specifying both positional arguments:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt ~/PBPM/data/raw/pathways/
```

Input file

Pathways directory

Important notes:

- Be aware of respecting the order of the positional arguments.
- Also, keep the names of the patients and mutations file as follows:

Chr, Position, Ref, Alt, Type, Gene, consequence, child_id, HGNC_symbol, HGNC_mapping, constraint_score, pLI, pRecessive, max_control_AF, PolyPhen_pred, MPC, Adj_Consequence.

- The directory containing the pathways can contain multiple files. However, only those ending with *'_with_gene_annotations'* will be considered for processing.

PBPM optional parameters

Final PBPMs will be filtered according to the optional arguments included in the analysis. Most of the parameters have both short and long names to be called.

- ## Pathways

Optional argument to filter specific pathways. If no setting is provided, all available pathways will be considered.

- Example 1: protocol will filter data for pathway R-HSA-69620

```
python3 main.py ~/PBPM/data/raw/mutations-file.txt
                ~/PBPM/data/raw/pathways/
                -p R-HSA-69620
```

- Example 2: protocol will not filter data by any pathway.

```
python3 main.py ~/PBPM/data/raw/mutations-file.txt
                ~/PBPM/data/raw/pathways/
```

- ## Pathways

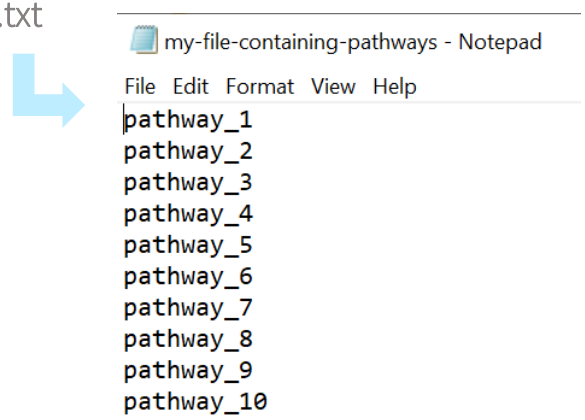
To filter by more than one pathway, you can specify one of the following:

- A subset of pathways separated by comma (without spaces):

  -p R-HSA-69620,0051705

- A **txt tab delimited file** with no headers and the desired pathways to filter written in the first column:

  -p ~/PBPM/data/raw/filters/my-file-containing-pathways.txt



- Genes

Optional argument to filter specific genes. If no setting is provided, all available genes will be considered.

- Example: protocol will filter data for gene CTR9

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -g CTR9
```

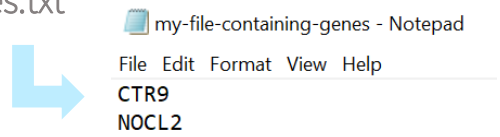
To filter by more than one gene, you can specify one of the following:

- Subset of genes separated by comma (without spaces):

  -g CTR9,NOCL2

- A **txt tab delimited file** with no headers and the desired genes to filter:

  -g ~/PBPM/data/raw/filters/my-file-containing-genes.txt



- ## Patients

Optional argument to filter specific patient IDs. If no setting is provided, all available patients will be considered.

- Example: protocol will filter data for patient with ID 1

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -id Patient_1
```

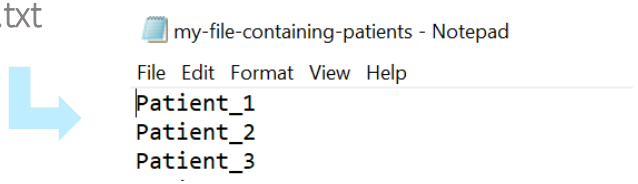
To filter by more than one patient, you can specify one of the following:

- A subset of patients IDs separated by comma (without spaces):

```
-id Patient_X,Patient_Y
```

- A **txt tab delimited file** with no headers and the desired patients to filter:

```
-id ~/PBPM/data/raw/filters/my-file-containing-patients.txt
```



- ## Consequences

Optional argument to filter specific consequences. If no setting is provided, all available consequences will be considered.

- Example: protocol will filter data for consequences of type 'missense_variant'

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -csq missense_variant
```

To filter by more than one consequence, you can specify one of the following:

- A subset of mutations separated by comma (without spaces):

```
-csq missense_variant,Intron
```

- A **txt tab delimited file** with no headers and the desired consequences to filter.

```
-csq ~/PBPM/data/raw/filters/my-file-containing-mutations.txt
```

- pLI

Optional argument to filter records with values greater than or equal to a specified pLI threshold. Example:

```
$ python3 main.py  ~/PBPM/data/raw/mutations-file.txt
                   ~/PBPM/data/raw/pathways/
                   -pli 0.6
```

- pRecessive

Optional argument to filter records with values greater than a specified pRecessive threshold. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -pr 0.6
```

- ## Max control AF

Optional argument to filter records with values less or equal than a max_control_AF threshold. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -af 0.3
```

- ## PolyPhen predictions

Optional argument to filter records with specific PolyPhen predictions. Available options for this parameter are: benign, possibly damaging, and probably damaging. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -pph2 "possibly damaging"
```

Note that qualifiers must be enclosed with quotes as they hold blank spaces.

- ## PolyPhen predictions

To filter by more than one qualifier, you must separate their labels with comma and without spaces while keeping the quotes:

-pph2 "probably damaging,possibly damaging"

- ## MPC

Optional argument to filter records with values greater than or equal to a specified MPC threshold. Example:

```
$ python3 main.py  ~/PBPM/data/raw/mutations-file.txt
                   ~/PBPM/data/raw/pathways/
                   -mpc 0.7
```

- ## Adjusted consequence

Optional argument to filter records by specific adjusted consequence labels. Available options for this parameter include: PTV, Missense3, Missense, etc. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -adj_csq PTV
```

To filter by more than one adjusted consequence, you must separate their labels with comma:

```
-adj_csq Missense3,Missense
```

PBPM: other useful parameters

- ## Download intermediate matrix

Besides filtering, this protocol allows to extract an intermediate matrix used to compute the final PBPMs. This matrix serves as raw data for individual exploratory analysis, and can be also appended to an existing dataset of the same nature.

By default, this boolean parameter is set to **False**, case in which only a final PBPM will be downloaded and a copy of the intermediate matrix will not be saved. To download this dataset, as well as the final PBPM, a **True** must be explicit by typing the parameter **–im** or **--intermediate_matrix.**

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -im

$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  --intermediate_matrix
```

PBPM: other useful parameters

- Download intermediate matrix

The base matrix looks like:

Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/dummy-mutations-file.txt

child_id	Chr	Position	Ref	Alt	consequence	HGNC_symbol	Pathway
Patient_1	11	62677944	C	A	missense_variant	CHRM1	pathway_27
Patient_100	12	65639657	A	C	missense_variant	LEMD3	pathway_55, pathway_54
Patient_1000	6	36653488	C	T	Intron	CDKN1A	pathway_78, pathway_69
Patient_1005	12	14019043	T	TG	frameshift_variant	GRIN2B	pathway_4, pathway_8, pathway_28, pathway_39, pathway_38, path
Patient_1009	3	133896766	T	A	Intron	RYK	pathway_67, pathway_46
Patient_101	12	31540736	G	A	Intron	DENND5B	pathway_67, pathway_70
Patient_1019	16	4165336	CT	C	frameshift_variant	ADCY9	pathway_63
Patient_102	10	88817407	G	C	Intron	GLUD1	pathway_59
Patient_102	10	104633032	T	TTTA	Intron	AS3MT	pathway_61
Patient_1020	6	168431425	A	G	Intron	KIF25	pathway_41
Patient_1021	12	108925059	T	C	splice_acceptor_variant	SART3	pathway_49

Patient-level data for more in-depth information

PBPM: other useful parameters

• Append intermediate matrix

Optional argument to append the intermediate matrix of the current session to the file provided, which is expected to be another intermediate matrix. Appended information will be used to compute the final PBPM. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways//
                  -a path-to-my-historic-file.txt

$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways//
                  --append path-to-my-historic-file.txt
```

The appended file will be overwritten and it will contain the commands used to generate all the information of the intermediate matrix.

PBPM: other useful parameters

- ## Append intermediate matrix

An appended intermediate matrix looks like:

```
# Script generated with the following parameters ---  inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/dummy-mutations-file.txt
# Script generated with the following parameters ---  inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/dummy-mutations-file.txt
# Script generated with the following parameters ---  inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/dummy-mutations-file.txt
```

child_id	Chr	Position	Ref	Alt	consequence	HGNC_syml	Pathway
Patient_1	11	62677944	C	A	missense_variant	CHRM1	pathway_27
Patient_100	12	65639657	A	C	missense_variant	LEMD3	pathway_54, pathway_55
Patient_1000	6	36653488	C	T	Intron	CDKN1A	pathway_69, pathway_78
Patient_1005	12	14019043	T	TG	frameshift_variant	GRIN2B	pathway_39, pathway_38, pathway_4, pathway_77, pathway_25, pathway_7,
Patient_1009	3	133896766	T	A	Intron	RYK	pathway_67, pathway_46
Patient_101	12	31540736	G	A	Intron	DENND5B	pathway_67, pathway_70
Patient_1019	16	4165336	CT	C	frameshift_variant	ADCY9	pathway_63
Patient_102	10	88817407	G	C	Intron	GLUD1	pathway_59
Patient_102	10	104633032	T	TTTA	Intron	AS3MT	pathway_61

This is an appended-like intermediate matrix coming from three different runs

PBPM: other useful parameters

- ## Path to store final PBPM

Optional argument to specify the path where the user wants to store the final PBPM generated. Example:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways//
                  --path location-where-I-want-to-save-PBPM
```

When not specified, the final PBPMs will be stored in the **analysis** folder of the project.

PBPM: choosing the final PBPM

The argument **–m** (also **--matrix**) will allow choosing the type of matrix that is desired. When not specified, by default, this parameter will extract a binary PBPM.

- Example to explicitly generate a binary matrix :

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -m b
```

- Example to generate a **numerical** matrix (not normalized):

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -m n
```

- Example to generate a normalized numerical matrix:

```
$ python3 main.py ~/PBPM/data/raw/mutations-file.txt
                  ~/PBPM/data/raw/pathways/
                  -m nn
```

How do PBPMs look like?

- Overview of a binary PBPM

Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/c

child_id	pathway_1	pathway_10	pathway_11	pathway_12	pathway_13	pathway_14	pathway_15	pathway_16
Patient_1215	0	0	1	1	0	0	0	0
Patient_1218	0	0	1	1	0	0	0	0
Patient_1393	0	0	1	1	0	0	0	0
Patient_2130	0	0	1	1	0	0	0	0
Patient_2139	0	0	0	1	0	0	0	0
Patient_271	0	0	1	1	0	0	0	0
Patient_3594	0	0	1	1	0	0	0	0
Patient_3606	1	0	1	1	0	0	0	0
Patient_3617	0	1	1	1	0	0	0	0
Patient_3623	1	0	1	1	0	0	0	0
Patient_3643	1	0	1	1	0	0	0	0
Patient_3660	0	0	1	1	0	0	0	0
Patient_667	0	0	0	1	0	0	0	0

How do PBPMs look like?

- Overview of a numerical PBPM

Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/

child_id	pathway_1	pathway_17	pathway_18	pathway_19	pathway_2	pathway_20	pathway_21	pathway_22
Patient_275	0	0	0	0	2	0	0	2
Patient_3	0	0	0	0	0	0	0	4
Patient_3594	0	0	0	0	1	1	0	4
Patient_364	0	0	0	0	2	0	0	2
Patient_3640	0	0	0	0	0	0	0	3
Patient_383	0	0	0	0	0	0	0	4

- Overview of a normalized PBPM

Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/staliclaProjectTemplate/PBPM/data/processed/

child_id	pathway_1	pathway_17	pathway_18	pathway_19	pathway_2	pathway_20	pathway_21	pathway_22
Patient_275	0.0000	0.0000	0.0000	0.0000	0.0339	0.0000	0.0000	0.0139
Patient_3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0278
Patient_3594	0.0000	0.0000	0.0000	0.0000	0.0169	0.3333	0.0000	0.0278
Patient_364	0.0000	0.0000	0.0000	0.0000	0.0339	0.0000	0.0000	0.0139
Patient_3640	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0208
Patient_383	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0278

What is the project organization?

It has the Stalicla Cookiecutter template for GitHub projects:

Available for the entire project settings

Execution logs are stored at this level and also PBPMs when an alternative path is not specified

Intermediate matrices are stored at this level

Specific libraries for the project can be found here

Where this presentation is

Source code for the project can be found here: PBPM.py and main.py

Where is this protocol stored?

Up to now, the PBPM repository is at my private GitHub:
https://github.com/mgmartinezl/Stalicla-PBPM

PENDING task: upload to the cluster and schedule some tests with final users.