

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.

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.
- Mutations.

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

- pLI.
- Max control AF.
- PolyPhen2 prediction label.
- MPC.
- Adjusted consequence.

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*. Only observations with a valid **HGNC_symbol** will be considered.

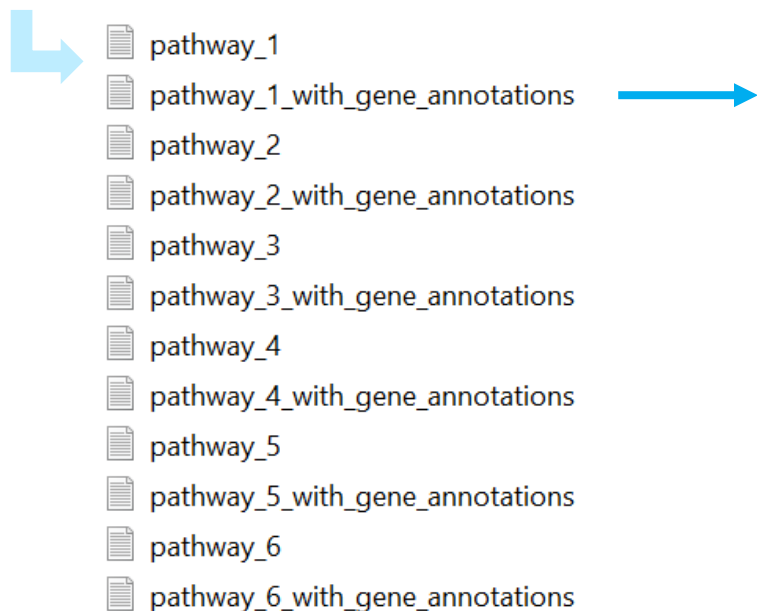
Chr	Position	...	child_id	consequence	HGNC_symbol	constraint_score	...	pLI	MPC
11	881802		Patient_10	missense_variant	V595I	0.018		0.14	1.7
12	686702		Patient_19	Near_3splice	NA	0.651		0.21	NA
3	981711		Patient_21	Intron	NA	0.006		0.35	NA
15	1121834		Patient_31	Silent	R589R	0.124		0.05	NA
20	541255	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

PBPM optional parameters

Final PBPMs will be filtered according to the optional arguments included in the analysis. Examples will illustrate this as follows:

- **Pathways**

Optional argument to extract only specific pathways. If no setting is provided, all available pathways will be extracted by default.

- Example 1: it will run only for pathway R-HSA-69620

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

- Example 2: it will run for all possible pathways

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

- Pathways


For more than one pathway, you can specify one of the following:

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

```
-pathway R-HSA-69620,0051705
```

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

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

 my-file-containing-pathways - Notepad
File Edit Format View Help
pathway_1
pathway_2
pathway_3
pathway_4
pathway_5
pathway_6
pathway_7
pathway_8
pathway_9
pathway_10

- Genes

Optional argument to extract only specific genes. If no setting is provided, all available genes will be extracted by default.

- Example: it will run only for gene CTR9

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

For more than one gene, you can specify one of the following:

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

```
-gene CTR9,NOCL2
```

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

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



my-file-containing-genes - Notepad

File Edit Format View Help

CTR9
NOCL2

- Patients

Optional argument to extract only specific child IDs. If no setting is provided, all available patients will be extracted by default.

- Example: it will run only for patient with ID 1

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


For more than one patient, you can specify one of the following:

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

```
-patient Patient_X, Patient_Y
```

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

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

 my-file-containing-patients - Notepad

File Edit Format View Help

Patient_1
Patient_2
Patient_3

- Mutations

Optional argument to extract only specific mutations. If no setting is provided, all available consequences will be extracted by default.

- Example: it will run only for mutations of type 'missense_variant'

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

For more than one mutation, you can specify one of the following:

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

```
-mutation missense_variant,Intron
```

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

- pLI

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

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

- Max control AF

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

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

- PolyPhen predictions

Optional argument to filter records for qualifiers of pph2 predictions. Available options for this parameter are: **benign**, **possibly damaging**, and **probably damaging**. Note that qualifiers must be enclosed with quotes as they hold blank spaces. Example:

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

To filter 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/original-mutations-file.txt ~/PBPM/data/raw/pathways/ -mpc_gt 0.7
```

- Adjusted consequence

Optional argument to filter records by specific value(s) of adjusted consequence. Available options for this parameter include: PTV, Missense3, Missense, etc. Example:

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

To filter more than one pph2 qualifier, you must separate their labels with comma:

```
-adj Missense3, Missense
```

PBPM: other useful parameters

- Download base matrix

Besides filtering, this protocol allows to extract a base matrix used to compute the final PBPMs. This base 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 NO ('no'), case in which only a final PBPM will be generated. To download this data as well as the final PBPM, a yes ('y') must be explicitly passed to the argument **-r**, as follows:

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

PBPM: other useful parameters

- Download base matrix

The base matrix looks like:

Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/staliciaProjectTemplate/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, pat
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 base matrix

The base matrix will not only be downloaded, but also appended to an existing file, when the parameter **–append** is set. Thus, it will not be necessary to explicitly set the download (**-r**) parameter to yes ('y'), unless a separate copy of the base matrix is desired. Example:

```
$ python3 main.py ~/PBPM/data/raw/original-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 base matrix.

PBPM: other useful parameters

- Append base matrix

An appended base matrix looks like:

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

child_id	Chr	Position	Ref	Alt	consequence	HGNC_sym	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, pathway_70
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 base matrix coming from three different runs

PBPM: choosing the final PBPM

The argument **–matrix** will allow choosing the type of matrix that is desired: binary (b), numerical (n) or normalized (nn). By default, this parameter will extract a binary PBPM.

- Example to explicitly generate a binary matrix :

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

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

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

- Example to generate a normalized numerical matrix:

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

How do PBPMs look like?

- Overview of a binary PBPM

```
# Script generated with the following parameters --- inputFile: /home/gabi/Documents/CookieCutter/stalidlaProjectTemplate/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/stalidlaProjectTemplate/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/stalidlaProjectTemplate/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 Stalicta Cookiecutter template for GitHub projects:



How to execute this protocol?

The scripts PBPM.py, and main.py are written in Python 3, which 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 needs:

```
python3 main.py inputFile pathwaysDirectory -pathway -gene -patient -mutation -pli_gt -af_lt -pph2 -mpc_gt adj -  
matrix -r -append
```

Where is this protocol stored?

Up to now, the PBPM repository is at my private GitHub:

<https://github.com/mgmartinezl/Stalicia-PBPM>

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