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

Applications Note

PyMethylProcess - highly parallelized preprocessing for DNA methylation array data

Joshua J. Levy[1,*], Alexander J. Titus[2], Lucas A. Salas[2] and Brock C. Christensen[2,3]

[1]Program in Quantitative Biomedical Sciences, Geisel School of Medicine at Dartmouth, Lebanon, NH, [2]Department of Epidemiology, Geisel School of Medicine at Dartmouth, Lebanon, NH, [3]Department of Epidemiology and Department of Molecular and Systems Biology, Lebanon, NH

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Summary: The ability to perform high-throughput preprocessing of methylation array data is essential in large scale methylation studies. While R is a convenient language for methylation analyses, performing highly parallelized preprocessing using Python can accelerate data preparation for downstream methylation analyses, including large scale production-ready machine learning pipelines. Here, we present a methylation data preprocessing pipeline called PyMethylProcess that is highly reproducible, scalable, and that can be quickly set-up and deployed through Docker and PIP.

Availability and Implementation: Project Name: PyMethylProcess
Project Home Page: https://github.com/Christensen-Lab-Dartmouth/PyMethylProcess. Available on PyPI as *pymethylprocess.* Available on DockerHub via *joshualevy44/pymethylprocess*.
Help Documentation: https://christensen-lab-dartmouth.github.io/PyMethylProcess/
Operating Systems: Linux, MacOS, Windows (Docker)
Programming Language: Python, R
Other Requirements: Python 3.6, R 3.5.1, Docker (optional)
License: MIT
Contact: joshua.j.levy.gr@dartmouth.edu

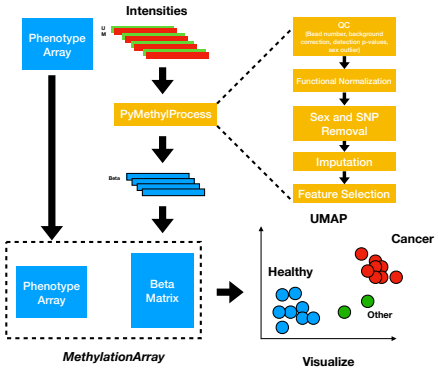
1. Implementation

DNA methylation plays a critical role in cell fate determination during development and regulation of transcription throughout life (Ji *et al.*, 2010; Khavari *et al.*, 2010; Rönnerblad *et al.*, 2014). Studies that measure DNA methylation in large numbers of human biospecimens typically use the Illumina Infinium BeadArray platforms known as 27K, 450K and 850K/EPIC arrays (Bibikova *et al.*, 2011, 2009; Moran *et al.*, 2016). Popular R packages used for DNA methylation data processing and analysis include *minfi* (Aryee *et al.*, 2014), *ENmix* (Xu *et al.*, 2016), or *meffil* (Min *et al.*, 2018). However, a straightforward and tractable approach to perform data quality control and functional normalization in bulk to prepare the data for use in the object-oriented environment is lacking. Here, we introduce a simple and easy-to-use command line interface that makes methylation analyses more object oriented and ready for use in downstream analyses such as immune cell proportion estimation, and python-oriented Epigenome-Wide Association Studies (EWAS) studies. In addition to more traditional methylation-based analyses, popular machine learning libraries such as scikit-learn, keras, and tensorflow (Abadi *et al.*, 2016; Pedregosa *et al.*, 2011) become more accessible to both machine learning researchers without an epigenetic background and epigenetic researchers with a limited machine learning background.

PyMethylProcess is a pip-installable command line interface built using Python 3.6 that interfaces with *minfi, ENmix,* and *meffil* in R via rpy2 (Gautier, 2010) to allow users to preprocess and set-up their methylation array data for downstream analyses included machine learning, presenting unique methylation datatypes that are built for the use of python classification, clustering, dimensionality reduction and regression algorithms such as UMAP, random forest, neural networks, k-nearest neighbors, and HDBSCAN (Campello *et al.*, 2013; Ho, 1995; McInnes *et al.*, 2018). Eight Python classes have been introduced to handle the following tasks: package installation (*PackageInstaller* installs any R/bioconductor package via python), data acquisition from TCGA and GEO (*TCGADownloader*) and formatting (*PreProcessPhenoData*), parallelized quality control (QC), principal component selection via kneedle (Satopaa *et al.*, 2011), and raw, quantile, noob, and functional normalization using *minfi, meffil, ENmix*

(*PreprocessIDAT*), imputation (*ImputerObject)*, feature selection, final preparation and storage for machine learning applications (*MethylationArray[s]* store beta and phenotype data), and a basic machine learning API class *MachineLearning* that trains any scikit-learn-like model on *MethylationArray* objects. These datatypes are used to handle complex preprocessing calculations, while being abstracted away via a convenient command-line interface. Additional commands, such as the removal of non-autosomal sites, SNP removal (either via QC methods or post-QC by subsetting CpGs that are not in a list of CpGs supplied by *meffil* for the respective array platform), and reference-based cell-type estimation (constrained projection/quadratic programming) (Houseman *et al.*, 2012; Jaffe and Irizarry, 2014; Salas *et al.*, 2018), and class methods are available in the help documentation. In addition, a visualization module generates interactive 3-D representations of the data using UMAP and Plotly (Modern Analytic Apps for the Enterprise) for further inspection. The beta values and phenotype data from the *MethylationArray* objects can easily be written to csv format using *write_csv*, and the command line interface can speed up the time it takes for researchers to set-up their standardized methylation data for downstream analyses.

Figure 1. Flow Diagram for *PyMethylProcess*



The pipeline differs from other python frameworks such as pyMAP (Mahpour, 2016) and GLINT (Rahmani *et al.*, 2017) in functionality. pyMAP only operates on the 450k framework, relying on user specific csv annotation files and preprocessed Genome Studio txt files as its input. pyMAP only performs graphical exploration for candidate CpGs, CpG Island feature subsetting, and export to BED files for downstream analyses. Similarly, GLINT requires a txt phenotype file and either a preprocessed beta values txt files or a R data.frame methylation object (in a RData file) as its inputs. GLINT stores beta values and covariate information as a binary "glint" file. GLINT was designed for EWAS analysis, including reference-based and reference-free estimations, imputed genetic structure, and statistical models (linear, logistic and linear mixed effects models). However, it relies on preprocessed data, with some limited quality control options and as so it could benefit from preprocessed data generated in our pipeline. In addition, GLINT is not designed to export this information to perform user customized downstream machine learning analyses.

2. Results

Here we demonstrate some of *PyMethylProcess*'s capabilities, performing preprocessing of seven different datasets (GSE87571, GSE81961, GSE69138, GSE42861, GSE112179, GSE109381, TCGA Pancancer)(Capper *et al.*, 2018; Johansson *et al.*, 2013; Li Yim *et al.*, 2016; Pai *et al.*, 2018; Pidsley *et al.*, 2013, 2013; Salas *et al.*, 2017; Soriano-Tárraga *et al.*, 2018) from the HumanMethylation450 BeadArray (450K array), and the HumanMethylationEPIC BeadArray (850K array), platform, derived from GEO, most of which benchmarked for loading, QC and normalization time. After preprocessing, each of the below data sets were split into 70% training, 10% validation, and 20% test sets for downstream prediction.

Table 1. Benchmark and Preprocessing Results

PyMethylProcess Pipeline Benchmarks

DataSets	Brief Description	Sample Size	Preprocessing Pipeline	# CpGs After Normalization	Principal Components	# Outlier Samples	# CPUs	Memory (Gb)	Runtime (Minutes)	# Sites Removed	Percentage Imputed (%)	Imputation Method
GSE87571	Johannson Aging	732	Minfi: Noob Normalization	482669	NA	13	1	NA	150.0	11233	0.706	K-NN: 15 neighbors
GSE81961	Crohn's Disease	40	Meffil: Functional Normalization	480329	4.0	0	35	10	4.1	11587	0.096	K-NN: 5 neighbors
GSE69138	Stroke	185	Meffil: Functional Normalization	474021	13.0	2	35	NA	11.5	11305	0.166	K-NN: 5 neighbors
GSE42861	Smoking and Arthritis	689	Meffil: Functional Normalization	477482	13.0	8	30[1]	110[2]	38.5[3]	11448	0.190	K-NN: 5 neighbors
GSE112179	Schizophrenia and Bipolar	100	Meffil: Functional Normalization	853772	10.0	2	30	NA	40.0	19278	0.240	K-NN: 10 neighbors
GSE109381	Brain Cancer Subclasses	3897	Meffil: Functional Normalization[4]	320023	10.4[5]	135	30	60	135.0	6791	0.054	Mean
TCGA Pancancer	33 Pan-Cancer subtypes	8891	Meffil: Functional Normalization[4]	378588	14.6[6]	515	30	60	255.0	7869	0.112	Mean

[1] Quality control used 30 CPUs, Normalization Used 14 CPUs
[2] Only for normalization step
[3] 13.5 Minutes for QC, 25 minutes for Normalization
[4] Subclasses Processed in Parallel
[5] Averaged Across Disease Subtypes

3. Benefits and Future Direction

The *PyMethylProcess* streamlines the process of preprocessing Methylation Array data while making Methylation data highly accessible and standardized for the Python machine learning community. *PyMethylProcess* is open source software, and additional development based on the community needs may be added. The authors of this paper would like to solicit issues and pull requests from the greater bioinformatics community on GitHub.

Future development will expand functionality to other preprocessing pipelines such as Watermelon and BigMelon (Gorrie-Stone *et al.*) and feature importance evaluations using measures such as the Gini index will be included. In addition, this command-line tool is available for use via Docker (*joshualevy44/pymethylprocess*) (Boettiger, 2015), making this analysis easily sharable, standardized, and operating system agnostic. The entire workflow will be wrapped using Common Workflow Language (CWL) (Amstutz *et al.*, 2016), making the entire analysis executable at the click of a button, highly reproducible, and easily sharable.

PyMethylProcess is installable from PyPI via the name *pymethylprocess*, and source code ***can be found on GitHub at: https://github.com/Christensen-Lab-Dartmouth/PyMethylProcess***

Acknowledgements

JJL conceived of idea and implementation, programmed and tested pipeline on datasets, wrote text; AJT tested pipeline; LAS provided technical support to streamline and debug important aspects of the pipeline; BCC provided research direction support; all authors participated in editing the text.

Funding

This work was supported by NIH grants R01CA216265, R01DE022772, and P20GM104416 to BCC, a Dartmouth College Neukom Institute for Computational Science CompX award to BCC, and training fellowship support for AJT from T32LM012204.

Conflicts of Interest: none declared.

List of Abbreviations

EWAS - Epigenome-Wide Association Studies
QC - Quality Control
CWL - Common Workflow Language
450K - HumanMethylation450 BeadArray
850K - HumanMethylationEPIC BeadArray

References

Abadi,M. *et al.* (2016) TensorFlow: A System for Large-Scale Machine Learning., pp. 265–283.

Amstutz,P. *et al.* (2016) Common Workflow Language, v1.0.

Aryee,M.J. *et al.* (2014) Minfi: a flexible and comprehensive Bioconductor package for the analysis of Infinium DNA methylation microarrays. *Bioinforma. Oxf. Engl.*, **30**, 1363–1369.

Bibikova,M. *et al.* (2009) Genome-wide DNA methylation profiling using Infinium® assay. *Epigenomics*, **1**, 177–200.

Bibikova,M. *et al.* (2011) High density DNA methylation array with single CpG site resolution. *Genomics*, **98**, 288–295.

Boettiger,C. (2015) An Introduction to Docker for Reproducible Research. *SIGOPS Oper Syst Rev*, **49**, 71–79.

Campello,R.J.G.B. *et al.* (2013) Density-Based Clustering Based on Hierarchical Density Estimates. In, Pei,J. *et al.* (eds), *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 160–172.

Capper,D. *et al.* (2018) DNA methylation-based classification of central nervous system tumours. *Nature*, **555**, 469–474.

Gautier,L. (2010) An intuitive Python interface for Bioconductor libraries demonstrates the utility of language translators. *BMC Bioinformatics*, **11**, S11.

Gorrie-Stone,T.J. *et al.* Bigmelon: tools for analysing large DNA methylation datasets. *Bioinformatics*.

Ho,T.K. (1995) Random Decision Forests. In, *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95. IEEE Computer Society, Washington, DC, USA, pp. 278–.

Houseman,E.A. *et al.* (2012) DNA methylation arrays as surrogate measures of cell mixture distribution. *BMC Bioinformatics*, **13**, 86.

Jaffe,A.E. and Irizarry,R.A. (2014) Accounting for cellular heterogeneity is critical in epigenome-wide association studies. *Genome Biol.*, **15**, R31.

Ji,H. *et al.* (2010) A comprehensive methylome map of lineage commitment from hematopoietic progenitors. *Nature*, **467**, 338–342.

Johansson,Å. *et al.* (2013) Continuous Aging of the Human DNA Methylome Throughout the Human Lifespan. *PLOS ONE*, **8**, e67378.

Khavari,D.A. *et al.* (2010) DNA methylation and epigenetic control of cellular differentiation. *Cell Cycle Georget. Tex*, **9**, 3880–3883.

Li Yim,A.Y.F. *et al.* (2016) Peripheral blood methylation profiling of female Crohn's disease patients. *Clin. Epigenetics*, **8**, 65.

Mahpour,A. (2016) pyMAP: a Python package for small and large scale analysis of Illumina 450k methylation platform. *bioRxiv*, 078048.

McInnes,L. *et al.* (2018) UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv180203426 Cs Stat*.

Min,J.L. *et al.* (2018) Meffil: efficient normalization and analysis of very large DNA methylation datasets. *Bioinforma. Oxf. Engl.*, **34**, 3983–3989.

Modern Analytic Apps for the Enterprise *Plotly*.

Moran,S. *et al.* (2016) Validation of a DNA methylation microarray for 850,000 CpG sites of the human genome enriched in enhancer sequences. *Epigenomics*, **8**, 389–399.

Pai,S. *et al.* (2018) Differential DNA modification of an enhancer at the IGF2 locus affects dopamine synthesis in patients with major psychosis. *bioRxiv*, 296756.

Pedregosa,F. *et al.* (2011) Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.*, **12**, 2825–2830.

Pidsley,R. *et al.* (2013) A data-driven approach to preprocessing Illumina 450K methylation array data. *BMC Genomics*, **14**, 293.

Rahmani,E. *et al.* (2017) GLINT: a user-friendly toolset for the analysis of high-throughput DNA-methylation array data. *Bioinformatics*, **33**, 1870–1872.

Rönnerblad,M. *et al.* (2014) Analysis of the DNA methylome and transcriptome in granulopoiesis reveals timed changes and dynamic enhancer methylation. *Blood*, **123**, e79-89.

Salas,L.A. *et al.* (2018) An optimized library for reference-based deconvolution of whole-blood biospecimens assayed using the Illumina HumanMethylationEPIC BeadArray. *Genome Biol.*, **19**.

Salas,L.A. *et al.* (2017) Integrative epigenetic and genetic pan-cancer somatic alteration portraits. *Epigenetics*, **12**, 561–574.

Satopaa,V. *et al.* (2011) Finding a 'Kneedle' in a Haystack: Detecting Knee Points in System Behavior. In, *2011 31st International Conference on Distributed Computing Systems Workshops*., pp. 166–171.

Soriano-Tárraga,C. *et al.* (2018) Biological Age is a predictor of mortality in Ischemic Stroke. *Sci. Rep.*, **8**, 4148.

Xu,Z. *et al.* (2016) ENmix: a novel background correction method for Illumina HumanMethylation450 BeadChip. *Nucleic Acids Res.*, **44**, e20.