**Software paper for submission to MEDSCI 736 University of Auckland**

**(1) Overview**

Title

“ePygenetics: An extraction tool for epigenetic data”

Paper Authors

1. Chalmers, Callum

Paper Author Roles and Affiliations

1. Student, Faculty of Medical and Health Sciences UoA

Abstract

ePygenetics is a python module that extracts epigenetic data from .wig files. It allows the user to input the data they require from the file, finds the data within the file and outputs the data to a database. This allows geneticists to easily extract the epigenetic status of single nucleotide polymorphisms (SNPs) in particular cell line so this can be considered in genetic analyses. It is licensed under the MIT license and the source code is freely available for use and modification at: <https://github.com/UOA-MEDSCI-736/CallumChalmers29-crispy-disco>

Keywords

Software, Python, Genetics, Epigenetics, SNPs

Introduction

There are a huge number of complex diseases that are known to be caused by a combination of genetic and environmental factors including both type I and II diabetes, Alzheimer’s disease and autism spectrum disorder [1]. Understanding the genetic component these diseases could lead to the development of preventative or therapeutic interventions which could reduce the burden they pose to our society.

This genetic discovery process has rapidly accelerated in the digital age with the invention of high throughput sequencing technologies which can quickly and cheaply generate large volumes of genetic data [2]. There has however been a lag in the development of software to analyse the huge volumes of data being produced [2]. One area of genetics which is beginning to benefit from analytical softwares is genome wide association studies or GWAS.

A GWAS compares the genetic code or ‘genome’ of healthy individuals to individuals with a disease to look for regions of DNA associated with disease [3]. These regions are found by comparing points in the DNA sequence called single nucleotide polymorphisms (SNPs) which are known to display variation between humans [4]. If humans with a disease all share a particular SNP and healthy individuals have a different SNP, then that section of DNA will be detected by GWAS and can be further investigated to see if it plays a role in the disease [3].

One problem with this is that there are around 10 million SNPs in the genome (source) and so GWAS generate huge volumes of data. These large numbers also increase the probability of false associations occuring so the threshold for detection causative SNPs must be high to ensure correlations are genuine [3]. Another way to achieve this would be increasing the sample size [3], but that is costly and not always feasible. As a result lots of potentially causative SNPs are missed by GWAS resulting in a lack of understanding of the genetic causes of disease known as ‘hidden heritability’ [4].

However, one thing these studies tend to ignore is that different cells express different genes. We have around 400 different cell types in the body and each has its own unique combination of active genes which allow it to function [5]. All the other genes are turned off or ‘silenced’. The study of which genes are on and which genes are off in different cells is called epigenetics. If a SNP was in an area of DNA that is turned off, then it is not affecting cell function and so cannot be causing disease [6]. By removing these SNPs from GWAS, the probability of false positives is lower, allowing a lower detection threshold and increasing the chance of locating important causative SNPs [3]. Therefore, gene detection by GWAS can be improved by integrating epigentic data analysis.

Currently, there is limited software available for viewing epigenetic data. The most commonly used programme is the UCSC Genome Browser but this is more focused on viewing individual SNPs rather than comparing SNPs within and between cell lines [7]. ePygenetics is a free, open-source software which allows the user to extract the epigenetic status of SNPs from epigenetic data files and to store this information in a database to allow for easy comparisions within and between cell lines. This information can then be used to improve the power of GWAS and uncover the hidden heritability of complex diseases like diabetes, Alzheimer’s and ASD.

**Implementation and architecture**

ePygenetics was implemented using Python version 3.5.2 in a Linux environment. Python was chosen because it is a simple, easy-to-learn, well-supported language that is commonplace in the scientific community and thus likely to have higher reuse potential.

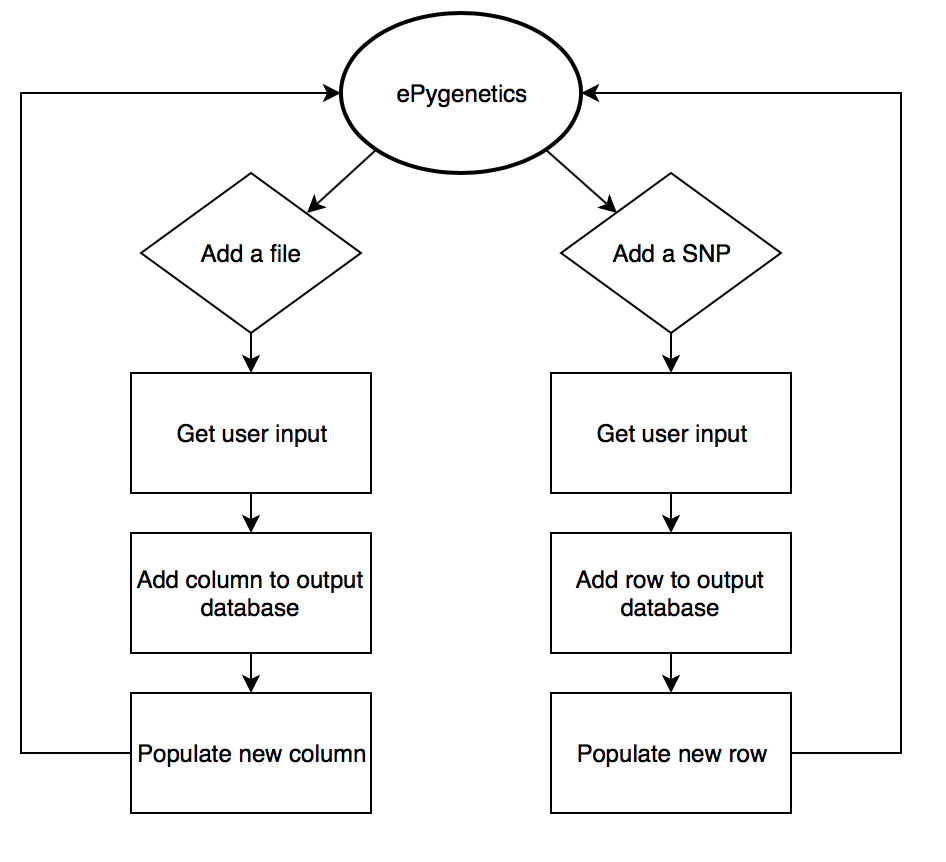


Figure 1: ePygenetics workflow

The programme is designed as a command line application as this is a simple interface for getting user input. The programme has been implemented with a simple bipartite functionality (Figure 1). Once the user opens the programme they input whether they would like to add a file or a SNP which is the equivalent of adding a file search parameter. If they choose to add a file, they input the file name and this automatically adds a column to the database. The programme then searches through the file for all the SNPs in the database and returns the data values which it loads into the database.

The large size of .wig files increases the time and memory required to open these files and search these files, slowing down the processing speed of the programme. To circumvent this, the programme uses the python readlines method so the whole file is not loaded into memory. It also uses the islice method from the itertools package to skip between blocks of data until the relevant data block is found, reducing the processing time almost 50 fold. As a consequence of this, the ends of each chromosome (final <1000 bases) cannot be read by the programme.

If they choose to add a SNP, they input the details of the SNP and this automatically adds a row to the database. The programme then searchs for this SNP in all the files loaded into the database and returns the data values which are loaded into the database. The programme selects files using the glob method from the glob package which selects all the files in the current working directory. An optional argument is added to ensure only .wig files are selected. This does however mean the files need to be in the same directory as the programme script.

The format chosen for the output database is a comma-separated values (CSV) file. This format was chosen because it is an easy to create, open file format which can be opened in a wide variety of software maximising the reproducibility of the software.

The programme is specifically designed to have maximum aesthetic functionality including insertion of blank lines where appropriate, use of the clear method from the os package and use of the colorama package to highlight important messages to the user. This adds additional steps to the install, and requires one modification to be made for execution on a Windows platform but overall improves the users experience and makes the programme easier to use.

**Quality control**

The software was designed using test-driven development (Figure 2).

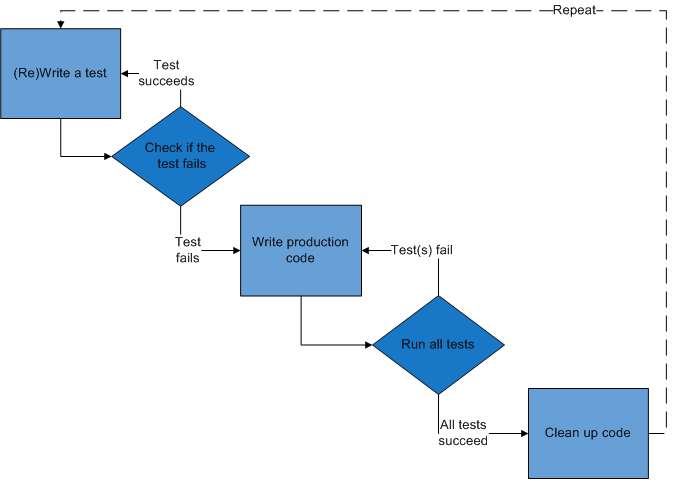


Figure 2: Test-driven Development Cycle

Attribution: “[A quick overview of the Test-driven development lifecycle](https://commons.wikimedia.org/wiki/File:Test-driven_development.PNG)” by [Excirial](https://en.wikipedia.org/wiki/User:Excirial) at [English Wikipedia](https://en.wikipedia.org/wiki/Main_Page) is licensed under [CC-BY SA 3.0](https://creativecommons.org/licenses/by-sa/3.0/deed.en)

This is an iterative process which combines building and testing to ensure the programme is functional. It ensures only the necessary elements are included, it means the code that is developed is simple and allows the programme to adapt to changing requirements. The testing process combined both functional and unit testing. Unit testing was done for functions with a clear input and output using pytest version 2.9.2. Functional testing was done for functions dealing with user input and for functions involved in data output using the Linux platform. Both sets of tests can be run by the end user to verify that their downloaded version is running correctly. Unit testing can be run by downloading the test-ePygenetics script and running pytest. For functional testing, an example set of data and expected output is provided in the test-data folder. Instructions on how to run these tests can be found in the readme file.

**(2) Availability**

***Operating systems***

Ubuntu 16.04 LTS, macOS Sierra 10.12, Windows 10 (requires one line modification, noted in the documentation)

***Programming language***

Python version 3.5.2

***Additional system requirements***

None

***Dependencies***

Colorama 0.3.7

***List of contributors***

1. Kreshnik Pireva, Staff, Faculy of Science, University of Auckland

2. Bibiana Lee, Student, Faculty of Medical and Health Sciences, University of Auckland

3. Luis Miguel, Student, Faculty of Medical and Health Sciences, University of Auckland

***Software location:***

***Archive*** To be decided

***Name:*** The name of the archive

***Persistent identifier:*** e.g. DOI, handle, PURL, etc.

***Licence:*** MIT

***Publisher:*** Callum Chalmers

***Version published:***The version number of the software archived

***Date published:*** dd/mm/yy

**Code repository**

***Name:*** UOA-MEDSCI-736/CallumChalmers29-crispy-disco

***Identifier:*** <https://github.com/UOA-MEDSCI-736/CallumChalmers29-crispy-disco>

***Licence:*** MIT

***Date published:*** 27/10/2016

***Language***

New Zealand English

**(3) Reuse potential**

The reuse potential of this software is high because it is scripted in Python, a platform that is relatively easy to use and understand and is widely used within the scientific community. In addition to this, the software comes with extensive documentation and a step by step walkthrough for sample data and user data to ensure it can be utilised without extensive knowledge of Python.The software does utilise the command line, however, and therefore does require some technical knowledge to run.

Currently this application is limited to reading fixed step .wig files with a step of 20. However, the end user can easily edit the code to adapt it to their own file type. For example, the code on line 202 can be altered from “20” to the step value of the end user’s file allowing different .wig files to be read. The code is under the MIT license so can be freely adapted to meet the end users needs. While this software has applications within the fields of genetics and epigenetics, the .wig file format has a highly specialised structure only utilised by these fields. Thus, the reuse potential outside these areas is very limited.

The software is available at <https://github.com/UOA-MEDSCI-736/CallumChalmers29-crispy-disco> and is supported on a voluntary basis. Any problems can be reported using the GitHub issue tracker. Any contributions are welcome and can be made by forking the repository, making the changes and then submitting a pull request.

**Acknowledgements**

The whole Medsci 736 teaching staff, especially Kreshnik Pireva, for helping CC learn the skills necessary to complete this project

**Funding statement**

Vice Chancellors Strategic Development Fund 2015, University of Auckland

**Competing interests**

The authors declare that they they have no competing interests

**References**

[1]Risch, N. and Merikangas, K., 1996. The future of genetic studies of complex human diseases. *Science*, *273*(5281), pp.1516-1517. DOI: [10.1126/science.273.5281.1516](https://dx.doi.org/10.1126/science.273.5281.1516)

[2]Excoffier, L. and Heckel, G., 2006. Computer programs for population genetics data analysis: a survival guide. *Nature Reviews Genetics*, *7*(10), pp.745-758. DOI: [10.1038/nrg1904](https://dx.doi.org/10.1038/nrg1904)

[3] Hayes, B., 2013. Overview of statistical methods for genome-wide association studies (GWAS). *Genome-wide association studies and genomic prediction*, pp.149-169. DOI: [10.1007/978-1-62703-447-0\_6](https://dx.doi.org/10.1007/978-1-62703-447-0_6)

[4] Manolio, T.A., Collins, F.S., Cox, N.J., Goldstein, D.B., Hindorff, L.A., Hunter, D.J., McCarthy, M.I., Ramos, E.M., Cardon, L.R., Chakravarti, A. and Cho, J.H., 2009. Finding the missing heritability of complex diseases. *Nature*, *461*(7265), pp.747-753. DOI: [10.1038/nature08494](https://dx.doi.org/10.1038/nature08494)

[5]Arendt, D., 2008. The evolution of cell types in animals: emerging principles from molecular studies. *Nature Reviews Genetics*, *9*(11), pp.868-882. DOI: [10.1038/nrg2416](http://dx.doi.org/10.1038/nrg2416)

[6]Hardison, R.C., 2012. Genome-wide epigenetic data facilitate understanding of disease susceptibility association studies. *Journal of Biological Chemistry*, *287*(37), pp.30932-30940.

DOI: [10.1074/jbc.R112.352427](https://dx.doi.org/10.1074%2Fjbc.R112.352427)

[7]Karolchik, D., Hinrichs, A.S. and Kent, W.J., 2009. The UCSC genome browser. *Current protocols in bioinformatics*, pp.1-4.

DOI: [10.1002/0471250953.bi0104s28](https://dx.doi.org/10.1002%2F0471250953.bi0104s28)

**Copyright Notice**

Authors who publish with this journal agree to the following terms:

Authors retain copyright with the work simultaneously licensed under a [Creative Commons Attribution License](http://creativecommons.org/licenses/by/3.0/) that allows others to share the work with an acknowledgement of the work's authorship and initial publication in this journal.

Authors are able to enter into separate, additional contractual arrangements for the non-exclusive distribution of the journal's published version of the work (e.g., post it to an institutional repository or publish it in a book), with an acknowledgement of its initial publication in this journal.

By submitting this paper you agree to the terms of this Copyright Notice, which will apply to this submission if and when it is published by this journal.